Original Research Paper

Volume-8 | Issue-3 | March-2018 | PRINT ISSN No 2249-555X

Computer Science

BIG DATA: GRAPHS, WEB AND DATA SECURITY

Youssef Amlal BAJJA University Hassan 2 Casablanca

Amina EL KEBBAJ University Hassan 2 Casablanca

Abdelwahed NAMIR University Hassan 2 Casablanca

ABSTRACT With the explosion of data's volumes, it has become essential for companies to put in place new tools to enable real-time detection of changes in order to make the best decisions. The main objective of this work is to offer an algorithm and a web application based on graphs which its ultimate goal is to secure the data and to detect the frauds within the company. It is in this spirit where is situated this work which consists in giving a clear vision on the utility of Big Data, and its contribution to data security by proposing an algorithm and a web application to detect the frauds and allow the decision maker to react. We will use a new IT tools such as the graph-oriented database Neo4j, the scripting language PHP 5 and JavaScript.

KEYWORDS : Algorithm; Big Data, fraud, graph, Neo4j, security, volume, real time, PHP5

I. INTRODUCTION

Security threats are becoming more serious and complex. Administrations and professional organizations are strengthening compliance standards. Due to the exponential complexity, security management is an increasingly difficult challenge. Big Data has been developed to make analytical technics more advanced, such as predictive analysis and advanced statistical technics, also allows to analyze large masses of data in real time and to establish correlations that human beings would put months to be found [1][2].

In this paper, we are going to handle accesses domain within the company (reading, writing) and detect in real time what was added in a malicious way. We will propose an algorithm and a web application using the graph-oriented database Neo4j, the scripting language PHP 5 and JavaScript which will add a user and give him access to such a directory.

If a hacker has tried to give access to someone directly from the database, we will have a sound signal which reveals malicious manipulation and generates the user name and the access provided in a real time. Any access rights added from the application will be modeled in a graph which presents all the nodes (users and folders) with their corresponding links.

The introduction defined the framework of reflection where this work is situated.

The following consists of 4 paragraphs:

- In the second paragraph, we approach the problem to be handled,
- In the third paragraph, we propose a fraud detection algorithm,
- In the fourth paragraph, we give a practical solution to our problem, a web application, for fraud detection.
- In the fifth paragraph, we propose examples of application.

The conclusion resumes the main lines of this study and our contribution. It shows also the various extensions and possible perspectives of this work.

II. APPROACHED PROBLEM

The data security's stakes are today very important within the company. Especially with the rise of big data which data volumes are increasingly enormous, it has become difficult to control the company's data, including data security.

Let us take the case of a company of services divided into two departments:

- IT department
- HR department

Each department has access to separate directories, for example the HR department has files concerning the salaries of employees which are confidential and an employee's access to these files is considered as FRAUD. Therefore, the main objective of our research is to provide an algorithm and a web application that allows to detect in real time any changes in the database whether it is modifies from the application or directly from the database.

The real-time observation will be modeled by a graph composed of nodes and links, allowing to visualize the access rights of each user with its corresponding files, in two modes (reading or writing).

If someone added an access directly from the database without using the application, this is considered as fraud and illegal, and the application will generate a sound signal indicating the properties of the illegally added relation.

III. ALGORITHMIC SOLUTION: FRAUD DETECTION ALGORITHM

To display the graph which contains the added or modified elements, the following method will be used:

(1) Each added element will be stored in:
(i) The database using a Cypher command
(ii) In a « grapheApp.json » file
(iii) In a json file which a part of its name contains the current date (grapheAppChangeDD-MM-YYYY.json). Therefore, any data added during the day will be stored in the same file.

(2) Comparison of the current date with the date of the last modification (« grapheAppChangeDD-MM-YYYY.json » contains the most recent date) :

(i) if « today's date » - « date of last modification » < 2 days :
  i. The oldest file « grapheAppChangeDD-MM-YYYY.json » will be added to the end of the file « old.json » which contains all the elements of the database minus 2 days.
  ii. Deleting the oldest file « grapheAppChangeDD-MM-YYYY.json »
  iii. Creating the new file « The oldest grapheAppChangeDD-MM-YYYY.json » With the current date containing the daily elements to be added.
(ii) if « today's date » - « date of last modification » >= 2 days :
  i. Concatenation of all files « grapheAppChangeDD-MM-YYYY.json » into « old.json » file.
  ii. Deleting files « grapheAppChangeDD-MM-YYYY.json »
  iii. Creation the new file « grapheAppChangeDD-MM-

YYYY.json » to store the added item.

(3) Creating a new file (diff.json) in order to detect frauds and all changes done.

Diff.json = { The difference between « grapheApp.json » and « old.json » to see what was added from the application
+
The difference between « grapheApp.json » and the json file exported from the DB to detect items added directly from the DB

TABLE I : APPLICATION SEQUENCES

Date	Element added	Old.json	grapheApp.json	Diff.json
Monday	Celine➔folder1	--	Celine➔folder1	Celine➔folder1
Tuesday	Celine➔folder2	--	Celine➔folder1 Celine➔folder2	Celine➔folder1 Celine➔folder2
Wednesday	Joseph➔folder1	Celine➔folder1	Celine➔folder1 Celine➔folder2 Joseph➔folder1	Celine➔folder2 Joseph➔folder1
Thursday	François➔folder2	Celine➔folder1 Celine➔folder2	Celine➔folder1 Celine➔folder2 Joseph➔folder1 François➔folder2	Joseph➔folder1 François➔folder2
Friday	--	--	--	--
Saturday	Joseph➔folder2	Celine➔folder1 Celine➔folder2 Joseph➔folder1 François➔folder2	Celine➔folder1 Celine➔folder2 Joseph➔folder1 François➔folder2 Joseph➔folder2	Joseph➔folder2

IV. WEB SOLUTION: REAL-TIME FRAUD DETECTION APPLICATION

To answer the approached problem, we created a web application which detects fraud in real-time, using the graph-oriented database Neo4j, PHP5 Scripting language, JavaScript and Alchemy framework.

The application has 4 components:

- User management, processing of all possible functions, including the addition, modification and deletion of users.
- The management of files
- Access management (relation between users and folders).

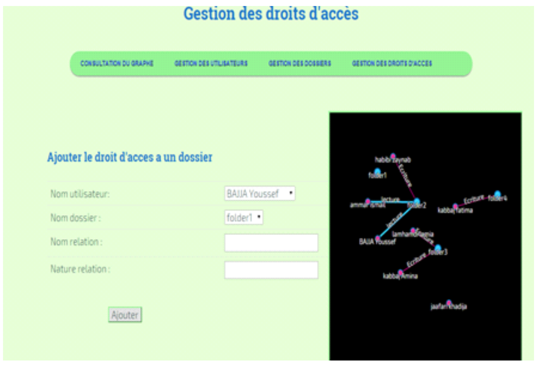
The consultation of the graph including all the interconnected nodes (All access rights given)

A. Management of user accounts and access right

Managing user accounts and groups represents an essential part of the system administration within the company.

The main reason for which user accounts exist, they are used to verify the identity of each person using a computer system. The second reason for their existence, they allow you to assign resources and access privileges according to the needs of each user.

Each new employee within the company has access to the corresponding files and directories. In order not to fall into illegal access, we will propose an application that allows to detect in real-time any access and any modifications made.



Ajouter le droit d'acces a un dossier

Nom utilisateur:	BAJJA Youssef
Nom dossier :	folder1
Nom relation :	
Nature relation :	

Fig. 1 a user's access to a folder

B. Proposed method / execution of the application

The problem will be modeled by a graph representing all nodes and interconnections. For every addition of a node we will have two graphs:

- The first graph will display the DB data as nodes and links, and
- The second contains only the added or modified data in order to know in real time all changes that are legal or illegal.

An illegal change means adding user access to a folder directly from the database without going through the application. In this case, an acoustic signal will be triggered.

The graphs are displayed via a json file (JavaScript Object Notation), it is a lightweight data-interchange format, it is easy for humans to read and write, it is easy for machines to parse and generate. It allows to display the result in the form of an interconnected graph as shown in the following figure:

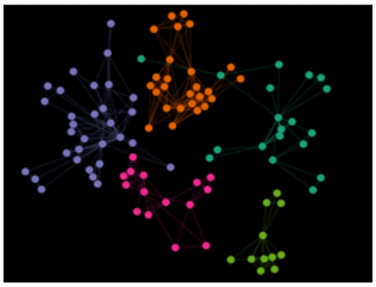
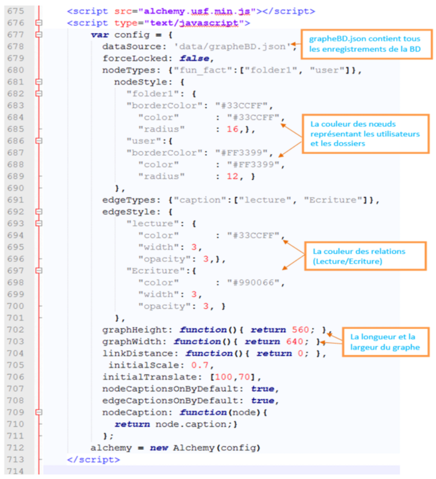


Fig. 2 graph using Alchemy

The graph is displayed using "Alchemy.js" [4]. It's a graphic design application built almost entirely in d3 (Data-Driven Documents), it is a JavaScript graphical library that allows the digital display in a graphic and dynamic form[6]. A minimum of the code is actually required to generate graphs as shown in the following figure:



```
    <script src="alchemy.usf.min.js"></script>
    <script type="text/javascript">
        var config = {
          dataSource: 'data/grapheBD.json',
          forceLocked: false,
          nodeTypes: {"fun_fact":["folder1", "user"]},
            nodeStyle: {
              "folder1": {
              "borderColor": "#33CCFF",
                "color"       : "#33CCFF",
                "radius"      : 16,},
              "user":{
              "borderColor": "#FF3399",
                "color"       : "#FF3399",
                "radius"      : 12, }
            },
          edgeTypes: {"caption":["lecture", "Ecriture"]},
          edgeStyle: {
              "lecture": {
                "color"       : "#33CCFF",
                "width": 3,
                "opacity": 3,},
              "Ecriture":{
                "color"       : "#990066",
                "width": 3,
                "opacity": 3, }
            },
          graphHeight: function(){ return 560; },
          graphWidth: function(){ return 640; },
          linkDistance: function(){ return 0; },
           initialScale: 0.7,
          initialTranslate: [100,70],
          nodeCaptionsOnByDefault: true,
          edgeCaptionsOnByDefault: true,
          nodeCaption: function(node){
            return node.caption;}
          };
        alchemy = new Alchemy(config)
    </script>
```

grapheBD.json contient tous les enregistrements de la BD
La couleur des nœuds représentant les utilisateurs et les dossiers
La couleur des relations (Lecture/Ecriture)
La longueur et la largeur du graphe

Fig. 3 source code of the graph

The result of this script is modeled below by a graph with nodes and interconnections:

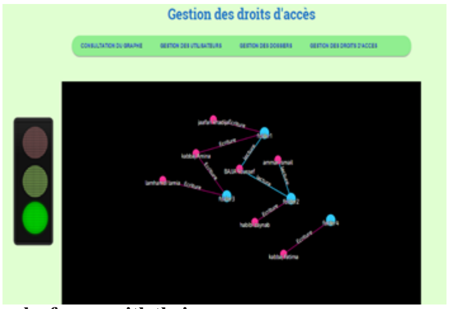


Fig. 4: graph of users with their access

C. The key commands of the application

To add an access between a user and a folder, we use the following Cypher command:

```
<?php

require_once 'vendor/autoload.php';

use Neoxygen\NeoClient\ClientBuilder;

$client = ClientBuilder::create()->addConnection('default','http','localhost',7474,true,'neo4j','123456')->setAutoFormatResponse(true)->
setDefaultTimeout(20)->build();

  $result = $client->sendCypherQuery("
  START user=node(*), folder=node(*)
  where has(user.firstname) and has(folder.name) and user.firstname = '$firstuser' and user.lastname = '$lastuser'
  and folder.name = '$namefolder'
  create (user)-[:link{name:'$nomR',nature:'$natureR'}]->(folder)
  ")->getResult();
```

To remove an access, we use the command below:

```
<?php
  $result = $client->sendCypherQuery("
  match (n:user)-[r]-(m:folder1) where n.firstname = '$firstuser' and  n.lastname = '$lastuser' and m.name = '$namefolder'  delete r
  ")->getResult();
```

To modify an access, we use the following command:

```
<?php
$result = $client->sendCypherQuery("
match (n:$label1)-[r]-(m:$label2) where n.firstname = '$firstuser' and  n.lastname = '$lastuser' and m.name = '$namefolder' set r.nature='
$natureR'
")->getResult();
```

V. APPLICATION EXAMPLES

When an access is added from the application, the GUI that will be displayed will contain 2 graphs:

- The first, represents a graphical modeling of all the users and their accesses.
- The second one, it just corresponds to the modifications made during 2 days.

If a change has occurred, we get a signal in the form of:

- A yellow light indicating that a modification has been made from the application.
- A red light indicating that a modification has been made illegally (directly from the DB).

A. Changing a user's access from the application

The example below shows the GUI that will be displayed once an access will be added from our application. We will get:

- A first graph that contains all the data of the DB
- A yellow light indicating that a change has been made from the application
- A second graph which shows just the modifications made

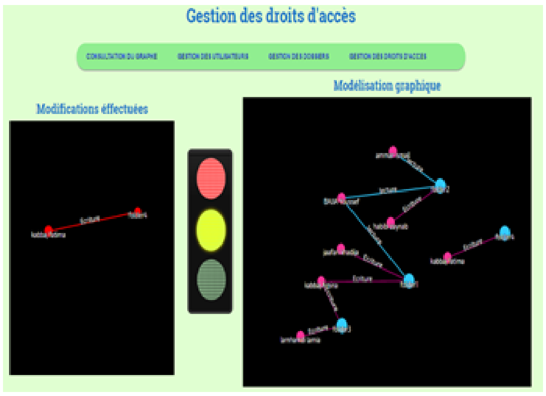
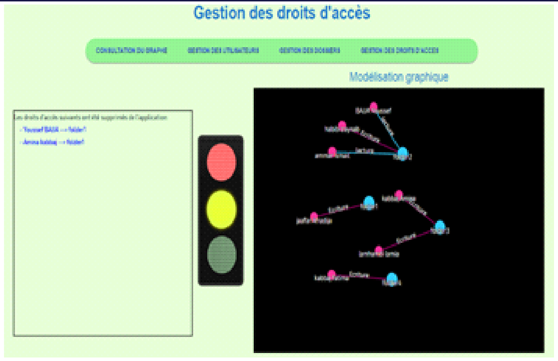


Fig. 5 Graphical interface of the modification

If an access was deleted (eliminated) from the application, we will get a text field with information about the user and the folder:

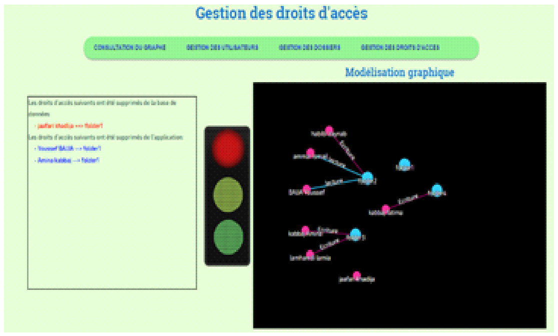


Les droits d'accès suivants ont été supprimés de l'application:
- Youssef BAIJA --> folder1
- Amina kabbaj --> folder1

Fig. 6 Graphic Interface of Deletion

B. Changing a user's access from the DB

If an access has been added, deleted or modified from the database directly, there will be a sound signal and a red light indicating the fraud:



Les droits d'accès suivants ont été supprimés de la base de données
- jaafari khadija --> folder1
Les droits d'accès suivants ont été supprimés de l'application:
- Youssef BAIJA --> folder1
- Amina kabbaj --> folder1

Fig. 7 Graphical interface of a modification directly from the DB (Fraud)

VI. CONCLUSIONS

This work aims to propose a web application based on graphs which the ultimate goal is to secure data and detect fraud within the company. It opens up our senses to various research opportunities that are on two levels: A plan for deepening the research carried out and an expansion plan for the research field.

To deepen the work proposed, it would be interesting initially to:

- Link the fraud detection application with the Active Directory to provide centralized identification and authentication services.

To broaden the scope of research, it would be interesting to:

- To pilot a Hadoop system which consists of performing processing on massive data volumes, of the order of several petabytes.

REFERENCES

[1] David Floyer « Enterprise Big-data »
[2] « Big Data : Security Opportunities and Issues»- Forrester studies
[3] «The 3V Big Data: Volume, Velocity and Variety» - Gérard Clech
[4] « Hadoop : A ten year old story » - Vincent Berdot
[5] « NoSQL: A movement that is gaining momentum » - Nicolas Martignole
[6] Alchemy graphs: https://github.com/GraphAlchemist
[7] Graph theory: http://blog.christophelebot.fr/wpcontent/uploads/2007/03/theorie_graphes.pdf
[8] http://graphalchemist.github.io/Alchemy/#/examples
[9] « Presentation of Big Data » - Xavier Dalloz
[10] « The phenomenon of big data affects all companies» - Claude Bernard