DIMAQs – Dynamic Identification of Malicious Query Sequences

Master Thesis Presentation

Michael Jobst

Advised by: Prof. Dr.-Ing. Alexandra Dmitrienko
1 Introduction
2 Attack & Requirement Analysis
3 Background
4 Proposed Solution
5 Evaluation
6 Conclusion & Future Works
1. Introduction
2. Attack & Requirement Analysis
3. Background
4. Proposed Solution
5. Evaluation
6. Conclusion & Future Works
What is Database Ransomware?

Two known Ransomware types:
- Crypto Ransomware
- Locker Ransomware

imposed 5 billion USD loss in 2017 predicted to hit 11.5 billion in 2019

What about Database Ransomware?
- first appearance in 2016
- connect to DBMS and deleting (dropping) databases/tables
- attacks against MySQL, MongoDB, ElasticSearch, Cassandra, Hadoop, and CouchDB
Prerequisites & Goals

Database Ransomware
- Dropping of Databases and Tables
- Demanding ransom to get database dump (copy of the data) back
- No evidence for such dumps

Goals
- Protect from data loss
- Detect malicious sequences, not only single malicious queries (SQLi)
- Determine database ransomware attacks
Outline

1. Introduction
2. Attack & Requirement Analysis
3. Background
4. Proposed Solution
5. Evaluation
6. Conclusion & Future Works
More than 45,000 servers compromised in total since 2016
BinaryEdge: 124 companies and institutions were victims between 3rd January 2017 and 15th January 2017

a) 11.3% of victims had recent database backups, 88.7% had not
b) 114 victims did not pay ransom, 10 did: no response
Attack Analysis

Threat Model

1. Brute Force Password / Connect
2. Execute SQL Statements (Malicious Query Sequence, varying)
   - List Databases
   - Drop Databases
   - Create Database (e.g. 'PLEASE_READ')
   - Create Table (e.g. 'WARNING')
   - Insert Ransom Message
3. Disconnect
1. Brute Force Password / Connect
2. Execute SQL Statements (Malicious Query Sequence, varying)
   - List Databases
   - Drop Databases
   - Create Database (e.g. ‘PLEASE_READ’) \(\Leftarrow\) not indicating
   - Create Table (e.g. ‘WARNING’)
   - Insert Ransom Message
3. Disconnect
Requirements

- System, that tracks the executed queries
- Backup dropped (permanent) Databases/Tables
- Hide backed up database tables and DIMAQS information from unprivileged users
- Allow authentication for privileged mode
- Restore backed up database tables by privileged users
- Notify Administrator about incidents
Outline

1. Introduction
2. Attack & Requirement Analysis
3. Background
4. Proposed Solution
5. Evaluation
6. Conclusion & Future Works
Background
Colored Petri net (CPN)

- Query Sequence Analysis
- Colored Petri net (CPN) consists of
  - Places (Circles)
  - Transitions (Bars)
  - Tokens (Dots)
  - Arcs
- ShieldFS
  - Backup data at the right time
  - Copy data to a safe storage space
Single Query Analysis (SQLi)

**Benign Query:** SELECT * FROM A WHERE id = 3
**Malicious Query:** SELECT * FROM A WHERE id = 3 OR 1 = 1

Query Sequence Analysis

1: SELECT * FROM information_schema.tables
2: CREATE TABLE A ...
3: INSERT INTO A ...
Background

Colored Petri net (CPN)

Token at $p_1$; Transition $t_1$ active
Transition $t_1$ fires and adds information to token from $p_1$
Token at $p_2$; Transition $t_1$ is disabled; Transition $t_2$ is enabled
Transition $t_2$ fires and adds information to token from $p_2$
Background

Colored Petri net (CPN)

one Token at $p_3$; Transition $t_2$ is disabled
- Dynamic Color Creation based on added information
- Token Duplication
- Transition Action & Condition (additional query type and value checks)
- Transition Always Action (fires immediately when active)
- Place Action (e.g. Query Rewriting, Backup Databases)
- Token Merging
- Token Expiration (remove Tokens after a certain period of time)
ShieldFS

- Backup strategy inspired by ShieldFS
- Developed in July 2016 to cope with crypto ransomware
- Copy files on the fly when process is suspicious
- Acts on file system level
**Background**

ShieldFS

**Figure**: On the right ShieldFS shadowing a file offended by ransomware malicious write, in comparison to standard file systems (on the left)
MySQL auditing plugin
not limited to users and connections (global observation)
CPN enhancements to reduce complexity and improve performance
act on certain queries
- move tables instead of dropping
- notify administrator when attack detected
- hide sensitive information (backed up data)
- create triggers for newly created table
Proposed Solution

System Architecture

MySQL

Monitoring

DIMAQS

Server

(1)

Parser

(7)

(10)

Resolution

(8)

(9)

Notifier

(3)

(5)

Classifier

Policy

(6)

Controller

(2)

(11)

(4)

(3)
### Proposed Solution

#### Classifier (Policy)

<table>
<thead>
<tr>
<th>Place</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$–$I_3$</td>
<td>Initial places</td>
</tr>
<tr>
<td>$OL_1$</td>
<td>Object “Database” listed</td>
</tr>
<tr>
<td>$OL_2$</td>
<td>Object “Table” listed</td>
</tr>
<tr>
<td>$OL_3$</td>
<td>Object “Column” listed</td>
</tr>
<tr>
<td>$TC$</td>
<td>Table created</td>
</tr>
<tr>
<td>$OD$</td>
<td>Object “Database” or “Table” deleted</td>
</tr>
<tr>
<td>$RI$</td>
<td>Ransom message inserted</td>
</tr>
<tr>
<td>$N$</td>
<td>Admin notification to be sent</td>
</tr>
</tbody>
</table>
**Proposed Solution**

**Classifier (Policy)**

<table>
<thead>
<tr>
<th>Transition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_D$</td>
<td>List Databases</td>
</tr>
<tr>
<td>$L_T$</td>
<td>List Tables</td>
</tr>
<tr>
<td>$L_C$</td>
<td>List Columns</td>
</tr>
<tr>
<td>$C_T$</td>
<td>Create Table</td>
</tr>
<tr>
<td>$D_D$</td>
<td>Drop Database</td>
</tr>
<tr>
<td>$D_T$</td>
<td>Drop Table</td>
</tr>
<tr>
<td>$M_T$</td>
<td>Modify table</td>
</tr>
<tr>
<td>$I$</td>
<td>Insert ransom message</td>
</tr>
<tr>
<td>$A$</td>
<td>Always</td>
</tr>
</tbody>
</table>
Initial places $I_{1-3}$ contain one empty token. Other places do not contain tokens. Transitions $L_D$, $L_T$, and $L_C$ are active and will be triggered on matching queries. All other transitions are disabled.
After transition $L_T$ was triggered. Initial states $I_{1-3}$ still contain tokens. Token from $I_2$ is transferred to the places $OL_1$ and $OL_2$. Tokens contain transition information which tables and databases were listed. The transitions $CT$, $DD$, and $DT$ become active.
After firing $C_T$. Tokens from $OL_1$ are copied to $TC$.

$C_T$ adds information about the created table to the transferred tokens.

Transition $I$ becomes active.
After firing $D_T$. Tokens from $OL_2$ are copied to $OD$.
Transition $D_T$ adds information about the dropped Table to the transferred tokens. $A$ does not become active because of $R_I$. 
After firing \( I \). Tokens from \( T_C \) that match the Table name are transferred to \( R_I \). \( I \) adds information about inserted message. \( A \) becomes active and fires immediately until \( O_D \) does not contain tokens anymore. Token values from \( R_I \) are merged with the token values from \( O_D \).
After token expiration.
Evaluation

Data Sets

Self generated: False Negatives
- 13485 tests

False Positives
- Bibspace
  - Query logs from 13th of April 2018 to 22nd of May 2018
  - contains 52085 queries
- MediaWiki
  - Query logs from 3rd of April 2018 to 22nd of May 2018
  - contains 2514764 queries
**False Negatives**

- no false negatives occurred
- 100% detection rate
- expected, since policy is designed to capture attacks from the malicious data set

**False Positives**

<table>
<thead>
<tr>
<th>Query set</th>
<th>$I_1$</th>
<th>$I_2$</th>
<th>$I_3$</th>
<th>$OL_1$</th>
<th>$OL_2$</th>
<th>$OL_3$</th>
<th>$T_C$</th>
<th>$O_D$</th>
<th>RI</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bibspace</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MediaWiki</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

No false positives occurred
Figure: Performance influence of DIMAQS for sysbench and MediaWiki. Values are normalized to the respective value for the disabled plugin.
Limitations

- Variations of BitCoin values
- Renaming of tables during classification
- Capturing of new attack forms requires adjustments of the policy
- Possibility to fill up secure storage space
- Ransomware is an emerging threat
- DB Ransomware attacks have severe consequences as attackers do not always create dumps.
- DIMAQS uses a colored Petri net–based classifier
- DIMAQS implemented as MySQL plugin
- allows to reduce complexity of system representation
- performance overhead below 5%
- Token Merging functionality needs enhancements to increase performance and reduce Notifications
- Trace table renaming during classification
- Detect other malicious query sequences (INTO_OUTFILE)
BinaryEdge Attack analysis
https://docs.google.com/spreadsheets/d/1QonE9oeMOQHVh8heFIyeqrjfKEViL0poLnY8mAakKhM/

ShieldFS
http://shieldfs.necst.it/continella-shieldfs-2016.pdf
Questions?