An Open-Source Proactive Security Infrastructure for Business Process Management

Angel Jesus Varela-Vaca, Department of Computer Languages and Systems, University of Seville, Spain, ajvarela@us.es
María Teresa Gómez-López, Department of Computer Languages and Systems, University of Seville, Spain, maytegomez@us.es
David Jiménez Vargas, Department of Computer Languages and Systems, University of Seville, Spain, davjimvar@outlook.es
Rafael M. Gasca, Department of Computer Languages and Systems, University of Seville, Seville, Spain, gasca@us.es
Antonio J. Suárez Fábrega, Department of Electronic Engineering, Computer Systems and Automation, University of Huelva, Huelva, Spain, asuarez@dti.uhu.es
Pedro J. Abad Herrera, Department of Electronic Engineering, Computer Systems and Automation, University of Huelva, Huelva, Spain, pedro.abad@dti.uhu.es

Abstract

Business Process Management Systems (BPMS) have emerged in the IT arena as cornerstone in the automation and orchestration of complex services for organizations. These systems manage critical information that is crucial for the organizations. The potential cost and consequences of security threats could produce information loss for the reputation of organizations. Therefore, the early response regarding to the non-compliance of security requirement is a real necessity overall during the business process execution. Currently, an active response requires a human intervention with high know-how and expertise in both business process management and security. In this paper, we propose an initial work which presents an open-source proactive infrastructure for the automatic continuous monitoring and checking compliance of security requirements at runtime of business processes.

Keywords: Business Process Management; Monitoring; Security; Log Management.

1. Introduction

The growing trend towards the automation and externalization of the activities that compose the Business Processes (BP) by means of Technology Infrastructure (TI), have been carried out frequently by means of Business Process Management Systems (BPMS). The problem is that this externalization has increased the security risks in the organizations. Nevertheless, security issues are mostly overlooked by default in BPMS, such as security controls (i.e. integrity checking) cannot be included in the BP model and there are no mechanism to check the compliance of security requirements at runtime.

Process-Aware Information Systems (PAIS), in the book by Weske (2007), is the most accepted framework to support the various stages of the business process life-cycle, including automated enactment and execution of business processes through Business Process Management (BPM). The enactment stage is the most critical phase in the BPM life-cycle. Since the cost and consequences of non-compliance of security requirements in that stage could range from mildly annoying to catastrophic, with serious injury occurring or systems destroyed, reputation losses, confidentiality losses, etc. Therefore, it is crucial to monitor business processes during the execution to check the compliance of security requirements in order to react as soon as possible. Currently more than 90 percentage of small and medium enterprises onto the Spanish market are not aware of the no compliance of certain security laws such as LOPD by AEPD (1999). The deployment of an infrastructure which helps to react automatically can raise a competitive advantage against competitors in a very changing market.

In order to be aware of the security issues, it is necessary to monitor the business processes. Weske (2007) distinguishes between *Monitoring* at enactment stage and *Business Activity Monitoring* (BPA) at evaluation stage of the life-cycle. On the one hand, *Monitoring* is used to visualise the state of process instances and log information during the execution. On the other hand, BPA is used to apply analysis techniques; such as Process Mining, in the logged data to check, for instance, the quality of models or the accuracy of the execution. The monitoring taxonomy proposed by González (2001) distinguishes two types of monitoring at runtime: *active monitoring* that provides information of the current state of business process instances; while *passive monitoring* provides status information about running business process instances upon request. Will van der Aalst (2012) highlights that monitoring is one of the case uses less considered in BPM arena in the last decade. Even less, BPA is apply completely disjoint of security issues.

Unfortunately, current monitoring and analysis techniques leave out security issues and data management of business processes such as mentioned by González et al. (2011). Traditionally, monitoring is only specified at operational level in order to check general properties of the business process execution (e.g. performance, number of instances, current state, etc.) Moreover, monitoring disregards information with respect to the domain-specific security problem. In general, this data involved in the monitoring process is captured on-line since most of variables are unknown at design either at implementation stage. On the other hand, the analysis of event trails is the final stage since reactions are not defined at all. It must be desirable to provide proactive solutions that enables to send a countermeasure to the business process once detected a non-compliance issue.

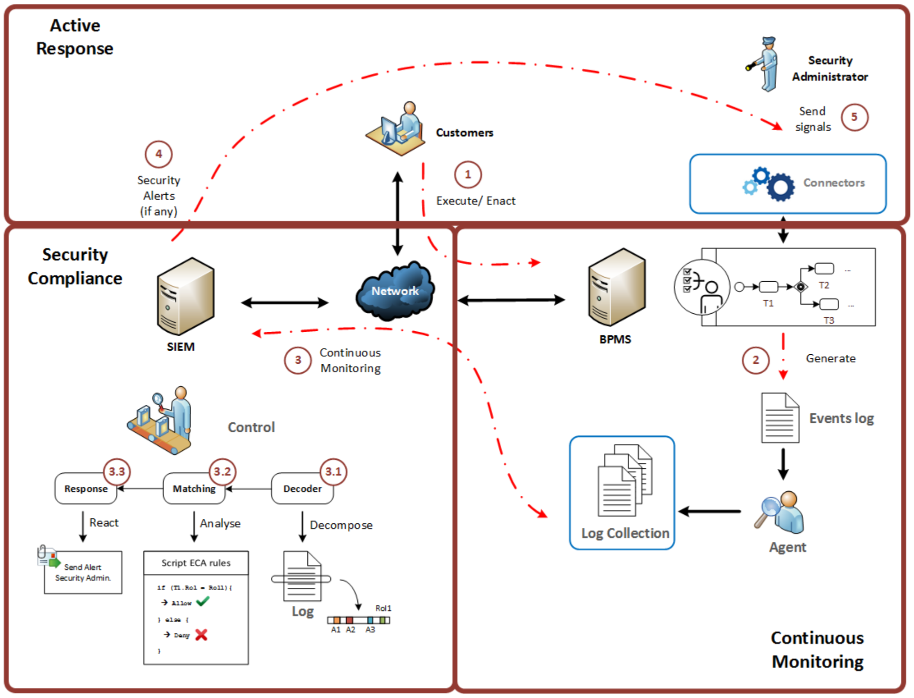
In order to reduce this limitation, we propose an infrastructure and architecture based on a Security Information and Event Management (SIEM) system to overcome the gap between monitoring, security compliance, and proactive response in a BPMS. The infrastructure aims to provide BPMS with features for the continuous monitoring, the automatic compliance checking of security requirements, and the active response at runtime of business processes.

The rest of the paper is structured as follows: Section 2 describes the different parts of the architecture; Section 3 shows an illustrative case study where all parts of the infrastructure are configured and a security non-compliance is detected and alarms are generated; in Section 4 conclusions are drawn and future work is proposed.

2. Architecture description

Nowadays, SIEM technologies aim to collect, store, analyse and report on log data for regulatory compliance and forensics. It implies the analysis of a set of complex events produced during the process execution. Montesinos et al. (2012) analyse how SIEM technologies can also help to automate several security controls from ISO/IEC 27002 and NIST SP 800-53 standards.

We propose to extend the infrastructure of a BPMS by means of SIEM such as shown in Figure 1. The proposed architecture is based on the runtime auditing layer included in Process Aware Information System (PAIS) such as proposed by Gómez-López et al. (2015). Gómez-López et al. provides an infrastructure to check the conformance of the persistent data managed and data-flow in a business process. Nevertheless this infrastructure only supports the data-flow and other information related to the business goals, but security aspects were not included.



```
if (T1.Rol = Rol1){
 → Allow ✓
} else {
 → Deny ✗
}
```

Fig 1. Framework for security compliance and monitoring of business processes.

The architecture proposed in this paper is encompassed of three separated parts that are responsible for: Continuous Monitoring, Security Compliance, and Active response. The architecture follows a control-flow which has been highlighted by means of slashed arrows also numbers has been included to show the step-by-step path for the information.

BPMSs facilitate business processes to be deployed and enacted, where the control information produced can be collected by a customer. Thereby the SIEM is responsible for collecting and recording the audit trails for the *agents*. The function of the *agents* is to monitor the events produced during the business process executions, sending these trails into log collections to the SIEM. Subsequently SIEM is able to analyse by means of checking the compliance of the security requirements. In case of non-compliance detection, the SIEM is already prepared to send signals or alarms (depends on the event) to the BPMS as a reaction. More detailed descriptions for each part of the infrastructure is given in the next case study where a case study shows how security problems are detected at runtime of business process execution.

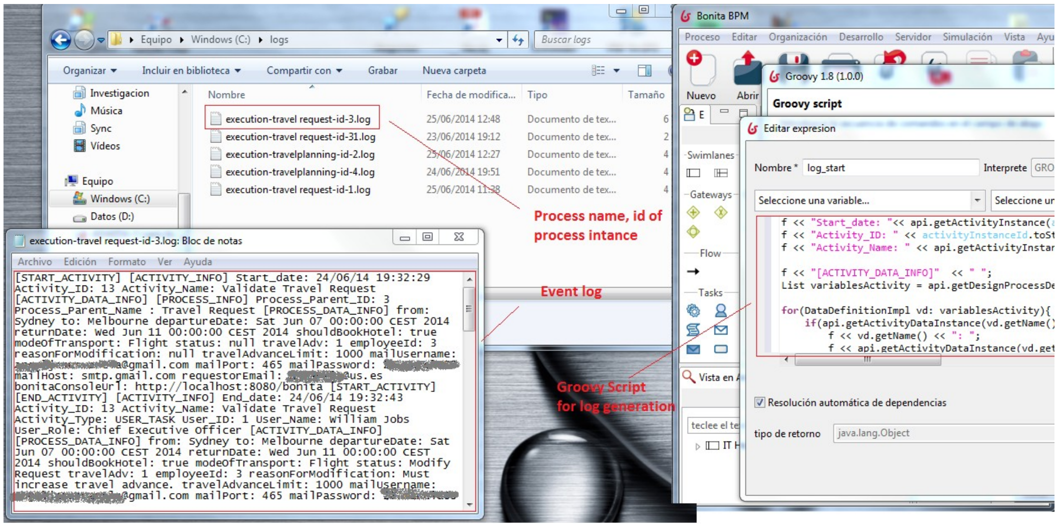
3. Case Study: Detection of Losing Data Integrity

The case study is based on the infrastructure shown in Figure 1, although equivalent software could be used, for our poof of verification in particular the BPMS is a *Bonita BPM framework* by BonitaSoft (2015) deployed over an Apache Tomcat server. Regarding the processes we have used the Travel Request blue-print included in the BPMS as proof of concept AlientVault (2015) OSSIM is used as SIEM system in order to collect and check the compliance of certain rules.

3.1. Continuous Monitoring

At the beginning (Step 1 of Figure 1, *Execute/Enact)*, Business process is enacted in a BPMS. During the execution of the process, several event traces are produced (Step 2 of Figure 1, Generate) event traces of execution. Monitoring those traces means to be aware of the business process execution state, collecting and analysing the events. The continuous monitoring can be carried out through the generation of event traces during all the life of each business process instance.

Event traces can be generated automatically in the format provided by the BPMS, or they can be customized in order to gather all the information needed by including, for example, Groovy (2015) connectors into the process definition. We could choose other formats such as XES traces as introduced by Günter (2015), as standard to store logs. Figure 2 shows an example of customized log trace obtained from the execution of Travel Request.



Nombre	Fecha de modifica...	Tipo	Tamaño
execution-travel request-id-3.log	25/06/2014 12:48	Documento de tex...	6
execution-travel request-id-31.log	23/06/2014 19:12	Documento de tex...	2
execution-travelplanning-id-2.log	25/06/2014 12:27	Documento de tex...	4
execution-travelplanning-id-4.log	24/06/2014 19:51	Documento de tex...	4
execution-travel request-id-1.log	25/06/2014 11:38	Documento de tex...	4

```
[START_ACTIVITY] [ACTIVITY_INFO] Start_date: 24/06/14 19:32:29
Activity_ID: 13 Activity_Name: Validate Travel Request
[ACTIVITY_DATA_INFO] [PROCESS_INFO] Process_Parent_ID: 3
Process_Parent_Name : Travel Request [PROCESS_DATA_INFO] from:
Sydney to: Melbourne departureDate: Sat Jun 07 00:00:00 CEST 2014
returnDate: Wed Jun 11 00:00:00 CEST 2014 shouldBookHotel: true
modeOfTransport: Flight status: null travelAdv: 1 employeeId: 3
reasonForModification: null travelAdvanceLimit: 1000 mailUsername:
[redacted]@gmail.com mailPort: 465 mailPassword: [redacted]
mailHost: smtp.gmail.com requestorEmail: [redacted]@us.es
bonitaConsoleUrl: http://localhost:8080/bonita [START_ACTIVITY]
[END_ACTIVITY] [ACTIVITY_INFO] End_date: 24/06/14 19:32:43
Activity_ID: 13 Activity_Name: Validate Travel Request
Activity_Type: USER_TASK User_ID: 1 User_Name: William Jobs
User_Role: Chief Executive Officer [ACTIVITY_DATA_INFO]
[PROCESS_DATA_INFO] from: Sydney to: Melbourne departureDate: Sat
Jun 07 00:00:00 CEST 2014 returnDate: Wed Jun 11 00:00:00 CEST
2014 shouldBookHotel: true modeOfTransport: Flight status: Modify
Request travelAdv: 1 employeeId: 3 reasonForModification: Must
increase travel advance. travelAdvanceLimit: 1000 mailUsername:
[redacted]@gmail.com mailPort: 465 mailPassword: [redacted]
```

Nombre * log_start
Interprete GRO

```
f << "Start_date: "<< api.getActivityInstance(a
f << "Activity_ID: " << activityInstanceId.toSt
f << "Activity_Name: " << api.getActivityInstar

f << "[ACTIVITY_DATA_INFO]"  << " ";
List variablesActivity = api.getDesignProcessDe

for(DataDefinitionImpl vd: variablesActivity){
    if(api.getActivityDataInstance(vd.getName()
        f << vd.getName() << ": ";
        f << api.getActivityDataInstance(vd.get
```

☑ Resolución automática de dependencias

tipo de retorno java.lang.Object

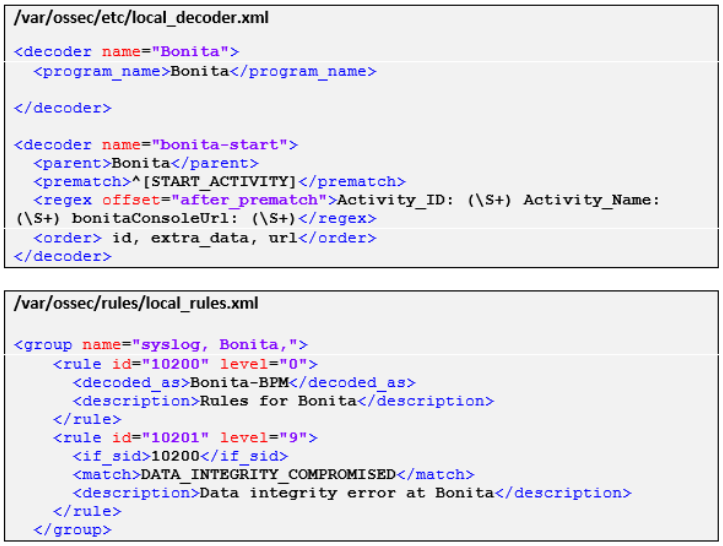
Fig 2. Event logs examples from various executions.

Simultaneously, an *OSSEC agent* deployed into BPMS server collects all traces generated to take them to SIEM (referred as Step 3, *Continuous Monitoring*). Logs can be analysed in order to check security requirements established, for instance Segregation of Duties (SoD) or Binding of Duties (BoD). Using this methodology even more complex requirements that require correlation of events could be customized at SIEM level to facilitate the analysis.

3.2. *Security Compliance*

When the OSSEC *agent* receives the logs collection, this is sent to be analysed in accordance to a set of rules that reflects security requirements to be checked. The analysis consists on applying a set of steps (Steps from 3.1 to 3.3 of Figure 1):

1. *Decoder*: To flatten and split the information in the event logs in order to extract certain data from the data-flow execution, such as tasks, variables, data, users, roles, etc.

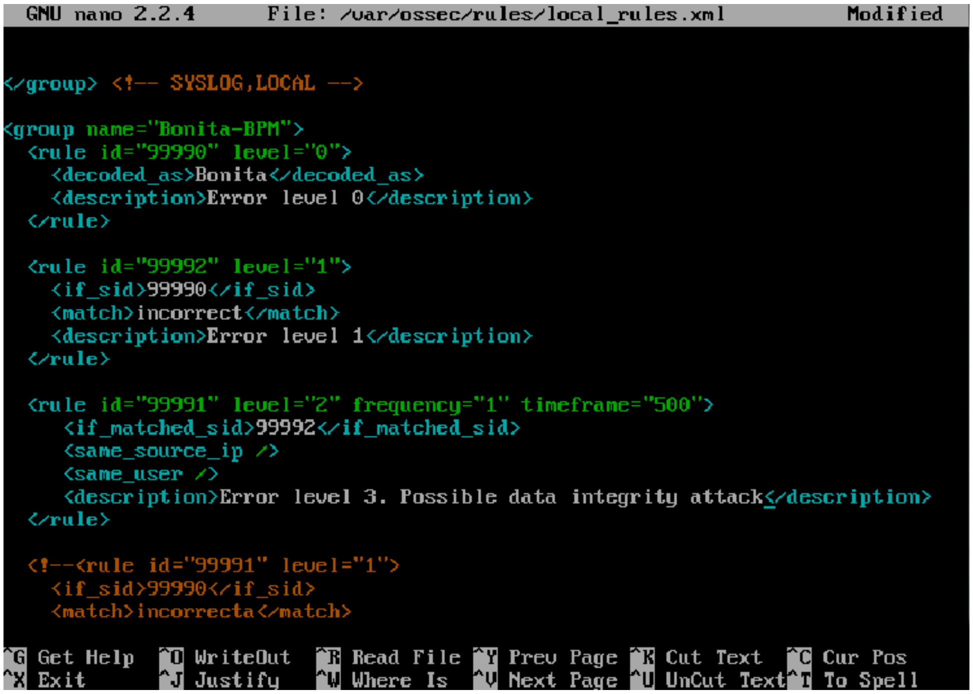


```
/var/ossec/etc/local_decoder.xml

<decoder name="Bonita">
  <program_name>Bonita</program_name>

</decoder>

<decoder name="bonita-start">
  <parent>Bonita</parent>
  <prematch>^[START_ACTIVITY]</prematch>
  <regex offset="after_prematch">Activity_ID: (\S+) Activity_Name:
(\S+) bonitaConsoleUrl: (\S+)</regex>
  <order> id, extra_data, url</order>
</decoder>
```

```
/var/ossec/rules/local_rules.xml

<group name="syslog, Bonita,">
    <rule id="10200" level="0">
      <decoded_as>Bonita-BPM</decoded_as>
      <description>Rules for Bonita</description>
    </rule>
    <rule id="10201" level="9">
      <if_sid>10200</if_sid>
      <match>DATA_INTEGRITY_COMPROMISED</match>
      <description>Data integrity error at Bonita</description>
    </rule>
  </group>
```

Fig 3. Configuration of rules at AlientVault OSSIM.

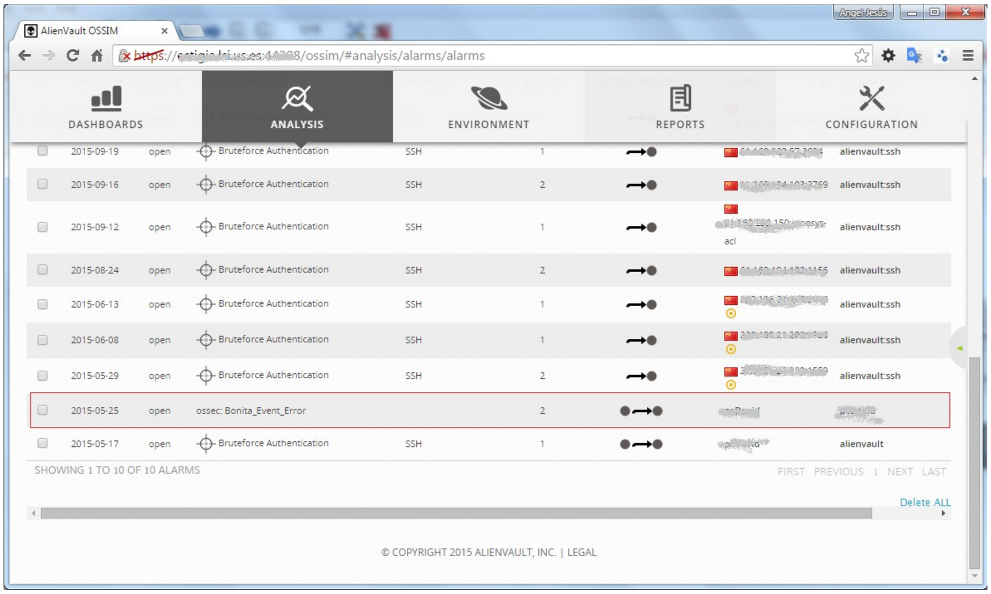
2. *Matching*: The information decomposed in the previous step provides the items to select the security requirements to be applied. If the data matched with the conditions established at the rule, and the order sequence of events (cf. Figure 4is also achieved, then the rule can be released. An example of rule configuration can be observed in Figure 3.



```
  GNU nano 2.2.4          File: /var/ossec/rules/local_rules.xml              Modified


</group> <!-- SYSLOG,LOCAL -->

<group name="Bonita-BPM">
  <rule id="99990" level="0">
    <decoded_as>Bonita</decoded_as>
    <description>Error level 0</description>
  </rule>

  <rule id="99992" level="1">
    <if_sid>99990</if_sid>
    <match>incorrect</match>
    <description>Error level 1</description>
  </rule>

  <rule id="99991" level="2" frequency="1" timeframe="500">
     <if_matched_sid>99992</if_matched_sid>
     <same_source_ip />
     <same_user />
     <description>Error level 3. Possible data integrity attack</description>
  </rule>

  <!--<rule id="99991" level="1">
    <if_sid>99990</if_sid>
    <match>incorrecta</match>

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Fig 4. Correlation rule set-up at AlientVault OSSIM.

3. *Response*: In case of matching a rule a response must be triggered in order to act in consequence of the non-compliance of security requirements. SIEM has been set-up to generate an alert and special security ticket in order to be checked by a security administrator. The alerts generated when a rule is triggered are registered in the Analysis → Alarms section of the SIEM such as shown in Figure 5



□	2015-09-19	open	Bruteforce Authentication	SSH	1	→●	[redacted]	alienvault:ssh
□	2015-09-16	open	Bruteforce Authentication	SSH	2	→●	[redacted]	alienvault:ssh
□	2015-09-12	open	Bruteforce Authentication	SSH	1	→●	[redacted] acl	alienvault:ssh
□	2015-08-24	open	Bruteforce Authentication	SSH	2	→●	[redacted]	alienvault:ssh
□	2015-06-13	open	Bruteforce Authentication	SSH	1	→●	[redacted]	alienvault:ssh
□	2015-06-08	open	Bruteforce Authentication	SSH	1	→●	[redacted]	alienvault:ssh
□	2015-05-29	open	Bruteforce Authentication	SSH	2	→●	[redacted]	alienvault:ssh
□	2015-05-25	open	ossec: Bonita_Event_Error		2	●→●	[redacted]	[redacted]
□	2015-05-17	open	Bruteforce Authentication	SSH	1	●→●	[redacted]	alienvault

SHOWING 1 TO 10 OF 10 ALARMS

FIRST PREVIOUS 1 NEXT LAST

Delete ALL

© COPYRIGHT 2015 ALIENVAULT, INC. | LEGAL

Fig 5. Security alarm triggered.

3.3. *Active response*

Following SANS (2014) definition, *Active Response* can be defined as the ability to automatically respond against a triggered event. Nevertheless, the response must be adapted with regard to the security requirement. Depending on the security requirement the response should require moderate actions or severe actions, such as send alerts messages or send signals to the business processes in order to lock the execution.

For instance, a rule is unsatisfied due to the compromising of a file integrity. The response could be sent an alert to security administrator to check possible corruption of the file system. In case of Segregation of Duties (SoD) could compromise the confidentiality of information. Hence, a signal has to be immediately sent to the running business processes instances in order to lock their execution while the error is being repaired.

The interaction with business processes at runtime can be carried out through services provided by the BPMS. For instance, Bonita Soft (2015) provides a *Web REST API* (2015) in order to interact with business processes, the BPM engine, and the BPMS.

4. Conclusions and forthcoming work

An integrated approach for the automatic continuous monitoring and checking compliance of security requirements at runtime of business processes has been proposed in this paper. The proposal describes how to set-up the infrastructure based on open-source solutions in order to reach automation of business processes, security requirement compliance checking at runtime, and the active response. In order to test the feasibility of the proposal the infrastructure has been tested in a case study where losing data integrity has been detected and alerts have been generated.

The main drawbacks are the flexibility regarding the mapping of security requirements and business process models and the limited syntax and semantic that *AlientVault SIEM* offers to represent rules and the engine to check certain rules. As future work we propose to extend the infrastructure more specifically the Security Compliance part with an Intelligence Engine based on Constraint Data Base (CDB); such as Labelled Object-Relational Constraint Database Architecture (LORCDB Architecture), proposed by Gómez-López et al. (2009), to store more complex security requirements. On the other hand, we propose to consider aspects related to flexibility such as proposed by Martinho (2010) in order to improve the mapping between security requirements and business process models.

References

AEPD - Agencia Española de Protección de Datos, (1999), 'Spanish Organic Act 15/1999, LOPD (Organic Act on Data Protection),' Boletin Oficial del Estado, Spain.

AlientoVault, (2015), 'OSSIM', [Online], [Retrieved October 19, 2015], https://www.alienvault.com/

BonitaSoft, (2015), 'Bonita Open Solution,' [Online], [Retrieved October 19, 2015], http://www.bonitasoft.com/

BonitaSoft, (2015), 'Bonita Web REST API,' [Online], [Retrieved October 19, 2015], http://documentation.bonitasoft.com/webrest-api

Gómez-López, M.T, Ceballos, R., Gasca, R.M., Valle, C.D., (2009), 'Developing a labelled object-relational constraint database architecture for the projection operator,' *Data Knowl. Eng.* 68(1), 146-172.

Gómez-López M. T., Gasca R. M., Pérez-Álvarez J. M. (2015), 'Compliance Validation and Diagnosis of Business Data Constraints in Business Processes at Runtime,' *Journal of Information Systems* 48, 26-43.

González, O., (2010), 'Monitoring and Analysis of Workflow Applications: A Domain-specific Language Approach,' Ph.D. thesis, Vrije Universiteit Brussel.

González, O., Casallas, R., Deridder, D. (2011), 'Monitoring and analysis concerns in work-flow applications: from conceptual specifications to concrete implementations,' *Int. J. Cooperative Inf. Syst*. 20(4), 371-404.

Groovy, (2015), 'Groovy Programming Language,' [Online], [Retrieved October 19, 2015], http://www.groovy-lang.org/

Günther, C.W., Verbeek, E., (2015), 'Extensible event stream,' [Online], [Retrieved October 19, 2015], http://www.xesstandard.org/

Martinho, R.; Domingos, D.; Varajão, J., (2010), 'Concept Maps for the Modelling of Controlled Flexibility in Software Processes,' *IEICE Transactions on Information and Systems.* E93-D, 8, 2190-2197.

Montesino, R., Fenz, S., Baluja, W., (2012), 'SIEM-based framework for security controls automation,'. *Inf. Manag. Comput. Security* 20(4), 248-263.

SANS Institute, (2014), 'What is Active Response?,' [Online], [Retrieved October 19, 2015], http://www.sans.org/securityresources/

van der Aalst, W. (2012), 'A decade of business process management conferences: Personal reflections on a developing discipline. In: Business Process Management,' Lecture Notes in Computer Science, vol. 7481, pp. 1-16. Springer Berlin / Heidelberg

Weske, M. (2007), Business Process Management: Concepts, Languages, Architectures. Springer, Germany.