JOURNAL OF INFORMATION SECURITY AND APPLICATIONS 27-28 (2016) 103–116

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa

Formal verification of secure information flow in cloud computing

Wen Zeng [a,*], *Maciej Koutny* [a], *Paul Watson* [a], *Vasileios Germanos* [b]

[a] *School of Computing Science, Newcastle University, Newcastle upon Tyne, UK*
[b] *Department of Mathematics and Computer Science, Liverpool Hope University, Liverpool, UK*

ARTICLE INFO

Article history:
Available online 11 April 2016

Keywords:
Federated cloud system
Information flow security
Bell–LaPadula rules
Petri net
Diagnosability
Model checking

ABSTRACT

Federated cloud systems increase the reliability and reduce the cost of computational support to an organisation. However, the resulting combination of secure private clouds and less secure public clouds impacts on the overall security of the system as applications need to be located within different clouds. In this paper, the entities of a federated cloud system as well as the clouds are assigned security levels of a given security lattice. Then a dynamic flow sensitive security model for a federated cloud system is introduced within which the Bell–LaPadula rules and cloud security rule can be captured. The rest of the paper demonstrates how Petri nets and the associated verification techniques could be used to analyse the security of information flow in federated cloud systems.

1. Introduction

The extent and importance of cloud computing is rapidly increasing due to the ever increasing demand for internet services and communications. Instead of building individual information technology infrastructure to host databases or software, a third party can host them in its large server clouds. However, large organisations may wish to keep sensitive information on their more restricted servers rather than in the public cloud. This has led to the introduction of federated cloud computing (FCC) in which both public and private cloud computing resources are used (see Watson, 2012).

A federated cloud is the deployment and management of multiple cloud computing services with the aim of matching business needs. Data, services, and software are required to be allocated in different clouds for both security and business concerns. Although federated cloud systems (FCSs) can increase the reliability and reduce the cost of computational support to an organisation, the large number of services and data on a cloud system creates security risks due to the dynamic movement of the entities between the clouds. As a result, it is necessary to develop tractable formal models faithfully capturing information flow security within FCSs.

In this paper, we introduce a formal model of dynamic information flow in an FCS, where services and data can migrate and change their security status dynamically. We then explain how Petri nets (more precisely, coloured Petri nets (CPNs)) could be used to analyse the correctness of such system. We also show how one could use the notion of diagnosability investigated in Germanos et al. (2014, 2015) in order to detect malicious events violating the proposed security policy in FCSs. We also evaluate experimentally the efficiency of the proposed setup using model checking of Clarke et al. (1999).

The paper is organised as follows. Section 3 provides the basic notions about security policies. In Section 4, a model for secure information flow analysis in FCSs is presented. The basic definitions relating to Petri nets are given in Section 5. Section

* *Corresponding author.* School of Computing Science, Newcastle University, Newcastle upon Tyne, UK. Tel.: +44 191 208 7972; fax: +44 191 208 8232.
E-mail address: wen.zeng.wz@gmail.com (W. Zeng).
http://dx.doi.org/10.1016/j.jisa.2016.03.002
2214-2126/

6 outlines how Petri nets could be used to support property verification in FCSs. Section 7 describes the diagnosis of behavioural properties, and Section 8 presents experimental results obtained for the proposed approach. Section 9 concludes the paper.

2. Related work

There exist different methods for addressing workflow[1] security; for example, the flow-sensitive analysis of programs in Smith (2001) and Russo et al. (2009). Using Petri nets to model workflows, Knorr (2000, 2001) applied the Bell–LaPadula model to workflow security. In particular, Knorr (2000) considered the *read* and *write* security policies. In Knorr (2001), the deployment of blocks within a workflow across a set of computational resources was not considered. In addition, the paper considered the *clearance* level but not *location* level in its embodiment of Bell–LaPadula model.

Watson (2012) proposed to partition workflows over a set of available clouds in such a way that security requirements are met. The approach is based on a multi-level security model that extends Bell–LaPadula to encompass cloud computing. Watson (2012) also indicated that workflow transformations are needed when data are communicated between clouds. However, in this study, the concurrency of the events or the execution of tasks in the system, the dynamic movement of the services, and the changes of the clearance level were not considered. Zeng et al. (2014a,b) introduced a flow sensitive security model to capture information flow in FCSs systems, which can be captured by CPNs. However, the clouds and services were assumed to be fixed, and the dynamic movement of services was not considered. Zeng and Koutny (2014) proposed a formal model for data resources in a dynamic environment focused on the location of different classes of data resources and users. However, the Bell–LaPadula rules and server-side components were not considered.

As far as we are aware, there is limited work related to formal verification of security in cloud computing systems. As an example, Gouglidis and Mavridis (2013) proposed a methodology for the development and verification of access control systems in cloud computing. The authors verify the access control systems against organisational security requirements using techniques that are based on simple transition systems. As another example, Benzadri et al. (2014) employed Bigraphical Reaction Systems to formally specify cloud services and customers as well as their interaction schemes. However, they did not consider security policies.

3. Security policies in cloud computing systems

In this section, we recall some basic concepts concerning security policies in cloud computing systems.

[1] Information flow refers to paths followed by data from their original positions to the end users in computational processes. Workflows are used to specify the formation/implementation of such processes.

3.1. Information lattices

Throughout the paper, we will assume that the basis of a federated cloud is a set P of single deployment clouds. Moreover, S will denote subjects (e.g., services, programs and processes), and O will denote objects (e.g., data resources and messages). Subjects and objects will jointly be referred to as entities, and their set will be denoted by E.

We will assign a *security level* to any entity, which will in practice be related to the degree of security of its contents, as well as to any cloud which will be related to the maximal security level of the entities it can contain.

A lattice for security concerns, $\mathcal{L}_{sec} = (L_{sec}, \leq_{sec})$ consists of a set L_{sec} and a partial order relation $\leq_{sec}$ such that, for all $l, l' \in L_{sec}$, there exists a least upper bound $l \sqcup l' \in L_{sec}$, and a greatest lower bound $l \sqcap l' \in L_{sec}$. The lattice is complete if each subset L of L_{sec} has both a least upper bound $\bigsqcup L$ and a greatest lower bound $\bigsqcap L$ (see Denning and Dorothy 1976, 1982; Landauer and Redmond 1993). Following Landauer and Redmond (1993), we will assume that the security lattice $\mathcal{L}_{sec}$ is complete.

3.2. Security requirements: Bell–LaPadula

We adopt the Bell–LaPadula multi-level control model of Bell and Lapadula (1973), with services modelled as the subjects S, and data as the objects O (Knorr 2001). Such a security model consists of the following components:

- A set of possible access rights R. The commonly used access rights are *read* (=r) and *write* (=w). In addition to reading and writing, there can be other access rights, e.g., data items that can be executed and/or updated. In order to simplify the presentation, the access rights used in this paper are *read* and *write*, $R = \{r, w\}$.
- A complete lattice for security concerns, $\mathcal{L}_{sec} = (L_{sec}, \leq_{sec})$.
- An access control matrix: $B : S \times O \to 2R$. The access control matrix issues the subjects rights to access objects. For example, if a service s_1 reads a data item d_0, then there will be the following entry in the access control matrix: $(s_1, d_0) \mapsto \{r, \ldots\}$. Similarly, if a service s_3 writes a data item d_2, then there will be the following entry in the matrix: $(s_3, d_2) \mapsto \{w, \ldots\}$. Note that the empty set is a valid function value, e.g., $(s_9, d_7) \mapsto \varnothing$ means that the subject s_9 has no access rights to the data item d_7.
- A clearance map: $c : S \to L_{sec}$. This represents the maximum security level at which each subject (i.e., service) can operate.
- A security level map: $\ell : E \to L_{sec}$. This represents the security level of each subject and object.

The Bell–LaPadula model states that a system is secure with respect to the above model if the following conditions are satisfied for all subjects $s \in S$ and objects $o \in O$:

$$\text{clearance:} \quad \ell(s) \leq_{sec} c(s) \tag{1}$$

$$\text{no-read-up:} \quad r \in B(s,o) \Rightarrow c(s) \geq_{sec} \ell(o) \tag{2}$$

$$\text{no-write-down:} \quad w \in B(s,o) \Rightarrow \ell(o) \geq_{sec} \ell(s) \tag{3}$$