A Distributed Security Model for Cloud Computing

Full Paper

Monjur Ahmed
Service and Cloud Computing Research Lab
Auckland University of Technology
monjur.ahmed@aut.ac.nz

Alan T Litchfield
Service and Cloud Computing Research lab
Auckland University of Technology
alan.litchfield@aut.ac.nz

Chandan Sharma
School of Engineering, Computer and Mathematical Sciences
Auckland University of Technology
em8149@aut.ac.nz

Abstract

Security in Cloud Computing is an ongoing concern and so security mechanisms play an important role. While mechanisms for security in the Cloud exist, new vulnerabilities for Cloud is an ongoing issue. For this, we contend that a need for a new type of security mechanism and model is required. Therefore, we propose a conceptual, high-level view of a distributed security model for Cloud Computing. This paper is part of an ongoing longitudinal study in which the model is outlined and discusses how the proposed security model is distributed among Cloud servers. The distributed security model presents multiple sacrificial targets in an attack and consequently eliminates any single point of failure.

Keywords

Cloud Computing Security, Distributed Security, Distributed Security Model, Security Mechanism.

Introduction

In Cloud Computing (CC), threats and vulnerabilities are major challenges and thus security is a key requirement (Liu 2012; Modi et al. 2013; Vaquero, Rodero-Merino & Morán 2011; Fernandes, Soares, Gomes, Freire & Inácio 2014; Sumter 2010). As with any computing environment, CC requires a robust security model and security mechanism but the range of vulnerabilities presents a confusing environment (Fernandes et al. 2014) and so it has been suggested that a new security solution is required for CC to retain its integrity and acceptability (Gritzalis, Mitchell, Thuraisingham & Zhou 2014).

CC makes use of the Internet to deliver services (Modi, Patel, Borisaniya, Patel & Rajarajan 2013; Grobauer, Walloschek & Stocker 2011) and has been referred to as Internet-based computing (Roberts & Al-Hamdani 2011). However, as a public network, the Internet-based computing approach presents a fair quantity of security concerns (Bottino & Hughes 2006) and in this regard, Cloud specific threats demand attention (Grobauer, Walloschek & Stocker 2011).

In order to address the demands of an Internet-based environment and to address those threats that are specific to CC, a security model is presented. The proposed model is distributed which makes it distinct from existing security systems that rely on a single application or centrally controlled server.

The paper is organized as follows: the existing security models or mechanisms are discussed in the next section with specific attention on existing security approaches that may or may not be distributed. The following section addresses the significance of a distributed security model for CC. The high level view of the distributed security model is illustrated and discussed in the next section. After that, validation of the security model is addressed and then a discussion and concluding remarks are made.

Existing Security Models in Cloud-based environments

The literature presents a number of cases of Cloud-based or Cloud-focused security solutions. Zissis and Lekkas (2012) discuss a generic Cloud security approach where they recommend the inclusion of trusted third parties. A security model involving multiple Clouds and called Multi-Cloud Databases (MCDB) is proposed by AlZain, Soh and Parded (2012). The model divides the Cloud into three layers: a management, application and presentation layer. The focus of this model is on data intrusion, service availability and data integrity.

A Cloud security approach by Zheng, Yang and Chen (2012) employs a strategy of noise generation to obtain obfuscation. The injection of random noise in this approach would make it hard for an attacker to distinguish between a real service request and noise. Consequently, it would be hard for an attacker to launch a successful attack. In a similar fashion, Mohamed and Abdelkader (2012) suggest using a Pseudo Random Number Generator (PRNG) with an encryption algorithm to generate critical data. The model evaluates a number of encryption algorithms and chooses the strongest.

Taking a different approach, Vaquero, Rodero-Merino and Moran (2011) discuss the Trusted Platform Module (TPM), which is a combination of hardware and software to bring data security into the Cloud. TPM does so by employing hardware based key binding and key wrapping. Also, the cybersecurity model for CC proposed by Rabai, Jouini, Aiss & Mili (2013) considers managerial perspectives of CC and takes the economic factors into account. The major aspects considered in this model are: mean failure cost, risk estimation matrices, system components and services, stakeholders' security requirements and providers' security concerns.

Chadwick and Fatema (2012) propose a privacy preserving authorisation system that enables users to set their own privacy policy. The proposed system puts user data and associated privacy policy together to enforce security. The intrusion severity analysis tool proposed by Arshad, Townsend and Xu (2013) concentrates on the security of virtual resource sharing over the physical platform. It uses a detection engine and an attack database where a system call handler works collaboratively with the attack database.

The Security Reference Architecture (SRA) proposed by Fernandez and Monge (2014) uses security patterns to ensure the integrity of a CC architecture. The security pattern maintains a misuse pattern with vulnerabilities, threats and security catalogue. It also includes best security practices where defences against the identified threats and vulnerabilities are defined.

For better access control and data security in the Cloud, the security model proposed by Ghebghoub, Boussaid and Oukid (2014) is based on encryption and Organization Role Based Access Control (ORBAC). With approaches conceptually similar to traditional access control mechanisms, it uses private key authorisation. Alternatively, AC_3 is an access control model for CC proposed by Younis, Kifayat & Merabti (2014) that is based on Role Based Access Control (RBAC) and Mandatory Access Control (MAC). AC_3 tries to find gaps in traditional access control mechanisms using security tags consisting of user role, permission, classification, time, random number, and location. As part of their proposed security model, Srinivasan et al. (2011) include a separate Cloud that is solely responsible for security and management and where access to the Cloud is determined by the presence of and controlled by security agents. Whereas the Category Based Access Control (CatBAC) proposed by Khamdja, Adi and Logrippo (2013), aims to provide integrated access controls. This model defines the security policy at an abstract level that can be used by a Cloud provider to build an access policy to enforce security. And, Xin, Song-qing and Nai-Wen (2012) propose a multi-dimensional security model that constitutes three defence layers that address the integrity of data encryption and authentication for the Cloud architecture.

What the above models do not address are the attributes or properties required for a distributed model for CC. That is, the literature we have found does not present a security model for CC that explicitly represents the distributed nature of the Cloud environment in its architecture. Some of the existing models can, to varying degrees, be treated as distributed. The Cloudgrid approach by Casola, Cuomo, Rak and Villano (2013) is such an example. It assimilates the idea of grid-on-Cloud and Cloud-on-grid. It implements a Cloud on top of a grid. It presents the Cloud architecture as three distinct layers: physical, Cloud, and virtual layer. Third party authentication authority is included in the scenario as part of this security model. As it includes third party authentication, the security approach may be thought of as distributed to some extent. Another example is the intrusion detection system proposed by Lo, Huang and

Ku (2008). The proposed model illustrates the collaborative approach of Network based Intrusion Detection System (NIDS) that reside in different Clouds. The NIDS housed in Cloud servers work collaboratively to update themselves on detection of an intrusion.

Importance of a Distributed Security Model

In this section, the argument that a distributed security model is feasible for a Cloud network is addressed. The Internet is the primary means for accessing Cloud services (Grobauer, Walloschek & Stocker 2011) and packet switching means data travels through various paths from source to destination. This aspect of the Internet requires a strong trust relationship among the networking elements (Kizza 2009). However, the Internet is a public domain and it is very difficult to ensure that all elements are trustworthy.

In cases where network elements are required to function collaboratively, such as on the Internet of Things (IoT) that also relies on Internet Protocol (IP) and packet switching (Raza, Duquennoy, Hoglund, Roedig & Voigt 2012), we submit that with just a single point of failure, total system security may be compromised, for example through a single point for authentication or validation, and that trust is lessened as a consequence. Therefore, a security mechanism in which components are distributed across the network and that presents no single point of failure provides the opportunity for an increase in trust.

Additionally, scalability, performance and elasticity in Cloud architecture are achieved by resource distribution but the remoteness or loose federation of resources may lead to poor control (Behl & Behl, 2012; Alfath, Baina & Baina 2013). It is the indirect nature of resource sharing in the public network that much of the Cloud is comprised that privacy and security concerns arise (Gritzalis et al. 2014). We suggest that rather than this aspect of the Cloud being a source for increased risk, that we take advantage of it and use loosely federated or remotely clustered nodes as a means to provide an improved security mechanism, such that cross border data security issues can be handled by a distributed security approach (Deibert 2012).

CC has a layered architecture with applications, APIs and Virtual Machines (VMs). The layered architecture makes security management complex, where the integrity and security of one layer may depend on the integrity and security of another layer. Such dependency enforces heterogeneity in security controls in the Cloud architecture and this results in the complex management of distributed resources (Bouayad, Blilat, Mejhed & Ghazi 2012). This scenario can be better addressed if a holistic security solution can act as a wrapper for multilayer security (Behl & Behl 2012). Thus, the argument is made that distributed resources and a layered, cross border architecture that offers no single point of attack may achieve more consistency if a distributed security model is deployed.

The core of this argument lies in the opportunity to benefit from distributed resources (Cascella, Morin, Harsh & Jegou 2012) and it is our stance that it is the centralized management of distributed resources for CC to be attractive to attackers (Yan, Zhang, Chen, Zhao & Li 2011). Thus, it is important to decide what security framework is needed. The model proposed here includes a distribution model that seeks the optimal use of distributed resources to counter distributed attacks and that eliminates a single point of attack because nodes are distributed among Cloud servers.

Distributed Security Model

The high level view of the distributed security model is illustrated in this section. Also, the terms ‘Security Model’ and ‘Security Mechanism’ are also defined in this section.

Security Model and Security Mechanism

In CC, the terms ‘Security Model’ and ‘Security Mechanism’ are not well distinguished and they are often used interchangeably and so we have set out to define the terms as follows.

Two approaches are found in CC when it comes to the discussion on security solutions. In the first approach, the architecture-wide scope is taken into consideration, and the arrangement of different elements of a Cloud architecture is dealt with to portray a holistic secured framework for CC. Examples of such approaches are the Cloud Cube model by the Jericho forum, the multi-tenancy model by NIST, and

the Cloud risk accumulation model by CSA (Che, Duan, Zhang & Fan 2011). These models define an architecture blueprint for a Cloud architecture and serves the purpose of a Reference Architecture (RA) for CC. Another approach addresses specific mechanisms or tools for particular security concerns for example, authentication (for example, the privacy preserving authorisation system by Chadwick and Fatema, 2011) and identity management (for example, the cooperative intrusion detection system by Lo, Huang and Ku, 2008). While the former can be termed as RA, the latter is the security mechanism.

We argue that a security model needs to have an RA and associated security mechanism(s) that are exclusive. Thus the proposed security model consists of an RA and its underlying distributed security mechanism. The high level concept of the security model is presented in the discussion that follows.

The Security Model

The security model is an amalgamation of an RA and a security mechanism. The security mechanism is split into a number of components that are distributed among the Cloud servers.

Assuming the following:

CA = Cloud Architecture	M = Security Model
C = A Cloud	λ = Component of the security model in one cloud server
S = A Cloud Server	N = Security Mechanism

The Cloud architecture for any organization may be made up of one or more Clouds. Thus an organization may have *C1, C2,… Cn* Clouds. Each Cloud can have one or more Cloud servers. So the Cloud servers are *S1, S2,… Sn*. As λ is the part of the security mechanism that resides in any given Cloud server, all the Cloud servers collectively form the total security mechanism. So in any given Cloud, the distributed parts of the security mechanism could be $\lambda 1, \lambda 2, \ldots \lambda n$. So the total part of the security mechanism for any given Cloud is,

$$N = \{\lambda 1, \lambda 2, \ldots \lambda n\} \quad 2.1$$

As security mechanisms will reside in Cloud Servers,

$$S = \{N \in S\} \quad 2.2$$

A Cloud may consist of one or more servers. Thus,

$$C = \{S \in C \mid 1 \leq S \leq i\} \quad 2.3$$

The security model may consist of one or more Clouds. Thus the security model is,

$$M = \{C \in M \mid 1 \leq C \leq i\} \quad 2.4$$

Thus a Cloud Architecture with the security model deployed can be expressed as,

$$\forall C\,[\exists M : \{(\lambda \subseteq N) \wedge (N \subseteq S)\}] \quad 2.5$$

In order to achieve a successful Cloud breach, this would result in having multiple points for the attacker to be compromised. As the security model will distribute its mechanisms across the nodes of a Cloud, and a high level of redundancy is achieved through the replication of individual elements, so if one element fails, the previously redundant component can be activated in another node. This eliminates the existence of a single point of failure within a Cloud architecture.

With the establishment of a high level view of the security model, it may be important to specify the type of security the distributed security model addresses. Ahmed, Litchfield and Ahmed (2014) state that the conveyance of data for a Cloud is through web services, thus the distributed security model will monitor web services of a Cloud architecture along with the life cycle of the data acquired and stored through web services. For example, a worm with a lifecycle similar to Stuxnet or Regin (Kushner 2013; Franceschi-Bicchierai 2014) spreads through web services. It is this type of attack that this model is intended to repel.

In Figure 1 is a conceptual illustration of the security mechanism. It is assumed that the security mechanism is split into three elements namely $\lambda 1$, $\lambda 2$ and $\lambda 3$. If there are three elements and three Cloud servers, each server will hold an element and redundant copies of elements also on other servers. For example, S1 has $\lambda 1$ as its operational part and $\lambda 2$ and $\lambda 3$ are redundant copies of operational elements on other servers.

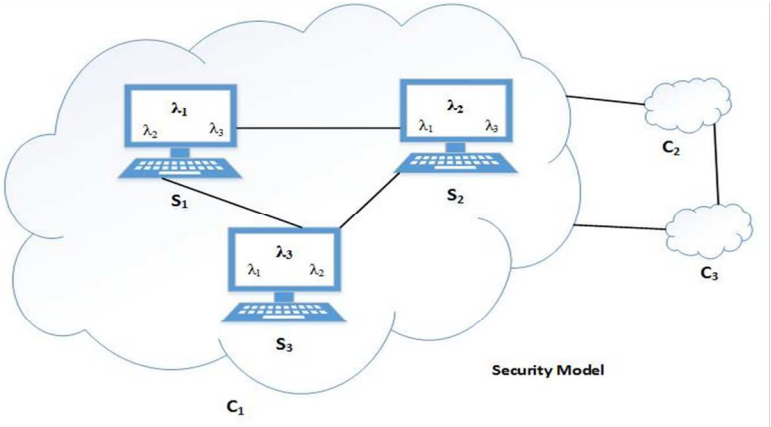

Figure 1: Conceptual View of the Distributed Security Model

As the components of the security mechanism will be distributed among the servers and all the parts of the security mechanism contribute equally, its distribution creates multiple points of attack and failure. However, for an attack to be successful all points are required to be taken offline and the attacker will not know which elements are operational or are merely redundant copies. This eliminates a single point of attack as well as single point of failure.

In reality, a Cloud architecture may have more nodes. Besides, a new node may become live at any time. As the number of nodes grows, a mechanism for distributing the security mechanism elements is required, with the capability to decide the dominant redundant part and the dormant redundant parts of the security mechanism that would be housed in the respective server. This is not addressed in this paper.

Model Validation

In this section, the concept of the security model is presented by means of Non-deterministic Finite State Automaton (NFA) and its equivalent Deterministic Finite State Automaton (DFA).

The birth and functioning of a Cloud may be initiated with one server and more servers can subsequently be added. The security mechanism is distributed if there are at least two components within any given server (Figure 2). Thus for a server, the security mechanism deals with three possible inputs: no change, addition of a component and deletion of an existing component.

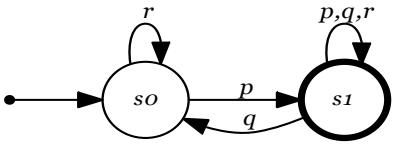

Figure 2: NFA representation of the distributed security model

The NFA has two states s_0, s_1 and three input functions *r, p* and *q***.** The states and inputs are:

r = No change: an input function which has no effect on the current state of the security mechanism.

p = Add server: an input function for the addition of a component.

q = Delete server: an input function for the deletion of a component.

*S*0 = a state where there is only one server in the security mechanism, (λ1, *λ2*).

*s*1 = the acceptance state, (*λ1* + *λ2*). Two servers in the security mechanism thus it is distributed. More servers can be added.

The NFA can be represented as $M = (Q, Z, R, q, F)$ where:

M = the NFA
Q = states of the NFA $\{s0 , s1 \}$
Z = input symbols $\{r, p, q\}$
$NULL$ = a state of $\{s0', s1', s0's1'\}$. This is a failure and from this state there is no return
q = initial state $\{s0\}$
F = acceptance state $\{s1\}$
R = transition $R: (Q * Z) \rightarrow Q$ (see Table 1)

$$\begin{pmatrix} R & p & q & r \\ s0 & s1 & NULL & s0 \\ s1 & s1 & s0, s1 & s1 \end{pmatrix}$$

Matrix 1: The transition function R for NFA

Matrix 1 is a 3*4 matrix and represents the transition function R of the NFA, where:

Elements in column 0 represent the initial state of NFA $\{s0, s1\}$

Elements in Row 0 represent the input symbols $\{p, q$ and $r\}$

The remainder of the elements represent the final state. For example, if $s0$ receives an input p, then it moves to $s1$, represented as $R: (s0 * p) \rightarrow (s1)$. Similarly, if $s1$ receives an input q, then it moves to $(s0, s1)$, represented as $R: (s1 * q) \rightarrow (s0,s1)$.

Figure 3 is the equivalent DFA of the NFA. The number of states in the DFA are the power set of the number of states in the NFA. If the total number of states in an NFA are n, then the number of states in a DFA would be $2^{**}n$. For the above DFA the total number of states are 2^{**} (number of states in the NFA $\{s0s1\}$). Therefore, the number of states in the DFA are $2^{**}2 = 4$.

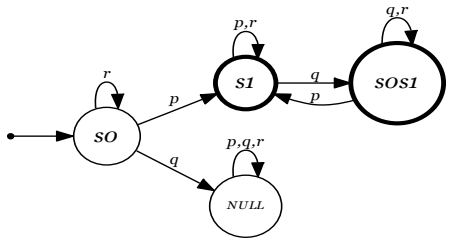

Figure 3: DFA representation of the distributed security model

The DFA can be represented as $M' = (Q', Z', R', q', F')$ where:

M' = the DFA
Q' = states of the DFA $\{s0, s1, s0s1, NULL\}$
Z' = input symbols $\{r, p, q\}$
$NULL$ = a state of $\{s0', s1', s0s1'\}$. This is a failure and from this state, there is no return.
F' = acceptance state $\{s1, s0s1\}$
q' = initial state $\{s0\}$
R' = transition R': $Q' * Z'$ (see Matrix 2)

$$\begin{pmatrix} R' & p & q & r \\ s0 & s1 & NULL & s0 \\ s1 & s1 & s0s1 & s1 \\ s0s1 & s1 & s0s1 & s0s1 \\ NULL & NULL & NULL & NULL \end{pmatrix}$$

Matrix 2: The transition function *R'* for DFA

Matrix 2 is a 5 * 4 matrix and represents the transition function R' of the DFA, where:

Elements in column 0 represent the initial state of DFA $\{s0, s1, s0s1, NULL\}$

Elements in row 0 represent the input symbols $\{p, q$ and $r\}$.

The remainder of the elements represent the final state. For example, if $s0$ receives an input p, then it moves to $s1$, which can be represented as $R\ (s0, p) \rightarrow (s1)$. Similarly, if $s1$ receives an input q, then moves to $(s0s1)$ which can be represented as $R\ (s1, q) \rightarrow (s0s1)$.

Turing Machine

In this section, the Turing Machine representation of the security mechanism is discussed. The Turing Machine can be represented as $M = (S, F, \delta, \text{ъ}, q0, \sum, \Gamma)$ where:

M = the Turing Machine, S = set of states of the Turing machine {$s0$, $s1$, $s2$, $s3$, $s4$}	$q0$ = initial state {$s0$}
F = acceptance state {$s4$}	ъ = a blank space on the tape {b} where ъ ∋ Γ and ъ ∌ ∑
δ = transition function and is the instruction set of the Turing machine	∑ = input alphabets {p, q} where ∑ ∈Γ
δ : (S* Γ) → (S * Γ * {*Left, Right*})	Γ = tape elements {p, q, b, X, Y}.
(See Table 1)	

A Turing machine consists of a head and tape that contains infinite spaces to the right. The head starts scanning the left most alphabet of the tape, then traverses the entire tape and, on seeing a blank symbol, halts. The head of a Turing Machine can read, write and move left or right over the tape alphabets, with the condition that it can only move right in the first step. The Turing machine accepts language of the form (p^*n q^*n) where:

p = addition of a component	q = deletion of an existing component
n should always be greater than or equal to two ($n \geq 2$).	

δ ($s0$, p) → ($s1$,X,*Right*)
δ ($s1$, p) → ($s1$,p,*Right*)
δ ($s1$, q) → ($s2$,Y,*Left*)
δ ($s2$, p) → ($s2$,p,*Left*)
δ ($s2$, X) → ($s0$,X,*Right*)
δ ($s1$, Y) → ($s1$,Y,*Right*)
δ ($s2$, Y) → ($s2$,Y,*Left*)
δ ($s0$, Y) → ($s3$,Y,*Right*)
δ ($s3$, Y) → ($s3$,Y,*Right*)
δ ($s3$, b) → ($s4$,*Halt*,-)

Table 1. Instruction set for the Turing Machine. δ : (S* Γ) →(S * Γ * {Left, Right})

The instruction set in Table 1 represents the working of the Turing Machine. Initially the head is at the left end of the tape and represents state $s0$. On reading an input symbol p on the tape, the Turing machine moves to state $s1$, writes X and moves right. On reading input p the Turing machine remains in state $s1$,

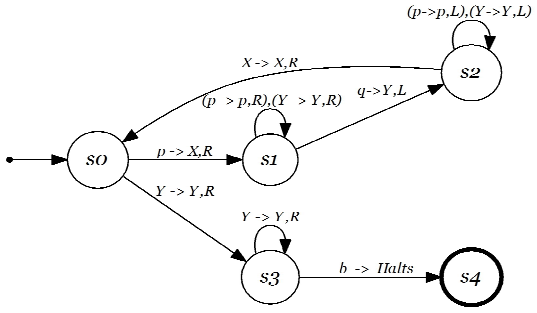

Figure 4: Turing Machine representation of the distributed security model

writes *p* and moves right. On reading input *q*, moves to state *q*2, writes *Y* and moves left. On reading *p* on the tape, the Turing Machine remains in state *s*2 and moves left. On reading *X*, it moves to the initial state *s*0 and starts moving right. On reading, the tape element, *Y,* remains in state *s*1, then writes *Y* and moves right, similarly, if it is in state *s*2 then it reads *Y*, remains in the same state and moves left. If the Turing Machine is in state *s*0 and reads the tape alphabet *Y,* this means that all the input symbols *p* have been read by Turing Machine and moves to state *s*3. The Turing Machine remains in state *s*3, reads and writes the same symbol, until it reads a blank symbol {*b*}, then moves to state *s*4 and halts.

Figure 4 represents the Turing Machine that is formed by the instruction set as represented in Table 1. It accepts strings of the form {*ppqq*, *pppqqq*, *pppppqqqqq*, *ppppppqqqqqq* ,…….,…} which belong to the language (*p*n q*n*).

Discussion

The security model discussed is a conceptual representation. As the study is ongoing, a number of aspects and functionalities of the security model have yet to be defined. The discussion identifies the security model to be made of different components and distributing the components among servers to eliminate a single point of failure. However, the model at this stage does not define how the security system would work, this requires further research. Also requiring further study is how componentization takes place, where the security model will be componentized and when servers are added, how to distribute the components. Additionally, how the components will interface among themselves and how change in a component in one server may affect redundant component(s) residing in other servers.

In the event of multiple copies of a component being compromised, it is important to embed some countermeasure to total system inconsistency. The goal of the proposed security model is to not only to provide a distributed defence or federated security model, but also to create a random scenario of the Cloud to ensure any malicious outsider is unable to map the pattern of a Cloud architecture of interest. If a component or server were to be compromised, then the security system within the proposed model would be able to detect the incident to take measure to avoid an architecture-wide effect. Thus, self-healing characteristics are part of the important future research developments for the proposed distributed security model.

Conclusion

In this paper we present a model for the distribution of components for a distributed security system. The distributed security model and its feasibility are discussed and existing security mechanisms are compared. The discussion also includes how the proposed security model introduces redundancy to eliminate a single point of attack (and subsequently, a single point of failure) within a Cloud architecture. The validation of the model is presented both in terms of automata and tested with a Turing Machine. The DFA and equivalent NFA as well as the Turing Machine representation of the distribution of the security model validate the security model. The model shown to be Turing complete.

References

Ahmed, M, Litchfield, A.T., & Ahmed, S. 2014. *A Generalized Threat Taxonomy for Cloud Computing*. 25th Australasian Conference on Information Systems (ACIS 2014), 8-10 December, Auckland, New Zealand.

Alfath, A., Baina, K. & Baina, S. 2013, April. Cloud Computing security: Fine-grained analysis and security approaches. In *IEEE 2013 national security days (jns3) conference* (p. 1-6). Rabat: IEEE. doi: 10.1109/JNS3.2013.6595465

Al Zain, M. A., Soh, B. & Pardede, E. 2012. A new model to ensure security in Cloud Computing services. *Journal of Service Science Research*, 4, 49-70. doi:10.1007/s12927-012-0002-5

Arshad, J., Townend, P. & Xu, J. 2013. A novel intrusion severity analysis approach for Clouds. *Future Generation Computer Systems*, 29, 416–428.

Behl, A. & Behl, K. 2012. An analysis of Cloud Computing security issues. In *World congress on information and communication technologies (wict)* (p. 109-114). Trivandrum, India.: IEEE. doi: 10.1109/WICT.2012.6409059

Bottino, L. J. & Hughes, W. J. 2006, October. Security measures in a secure computer communications architecture. In *25th digital avionics systems conference* (p. 6B3.1-6B3.18).

Bouayad, A., Blilat, A., el Houda Mejhed, N. & Ghazi, M. E. 2012. Cloud Computing: Security challenges. In *Colloquium in information science and technology (CIST)* (p. 26-31). Fez: IEEE. doi: 10.1109/CIST.2012.6388058

Cascella, R. G., Morin, C., Harsh, P. & Jegou, Y. 2012. Contrail: A reliable and trustworthy Cloud platform. In *Acm ewdcc '12*. ACM.

Casola, V., Cuomo, A., Rak, M. & Villano, U. 2013. The Cloudgrid approach: Security analysis and performance evaluation. *Future Generation Computer Systems*, 29, 387–401.

Chadwick, D. W. & Fatema, K. 2012. A privacy preserving authorisation system for the Cloud. *Journal of Computer and System Sciences*, 78, 1359–1373.

Che, J., Duan, Y., Zhang, T. & Fan, J. 2011. Study on the security models and strategies of Cloud Computing. In *International conference on power electronics and engineering application* (pp. 586–593).

Deibert, R. 2012. *Distributed security as cyber strategy: Outlining a comprehensive approach for Canada in cyberspace* (Research Paper). The Citizen Lab and Canada Centre for Global Security Studies, Munk School of Global Affairs, University of Toronto.

Fernandez, E. B. & Monge, R. 2014. A security reference architecture for Cloud systems. In *ACM ICSA '14*. Sidney, NSW, Australia: ACM.

Fernandes, D. A. B., Soares, L. F. B., Gomes, J. V., Freire, M. M. & Inácio, P. R. M. 2014. Security issues in Cloud environments: a survey. *International Journal of Information Security*, 13, 113-170. doi: 10.1007/s10207-013-0208-7

Franceschi-Bicchierai, L. 2014, November. *What we know about 'regin' spy malware*. Retrieved from http://www.stuff.co.nz/technology/digital-living/63550458/what-we-know-about-regin-spy-malware.html

Ghebghoub, Y., Boussaid, O. & Oukid, S. 2014. Security model based encryption to protect data. In *ACM ISDOC* (p. 50-55). ACM. doi: http://dx.doi.org/10.1145/2618168.2618176

Gritzalis, S., Mitchell, C., Thuraisingham, B. & Zhou, J. 2014. Security in Cloud Computing. *International Journal of Information Security*, 13, 95-96. doi: 10.1007/s10207-014-0232-2

Grobauer, B., Walloschek, T. & Stocker, E. 2011. Understanding Cloud Computing vulnerabilities. *Cloud Computing*, March/April 2011, 50–57.

Kizza, J. M. 2009. *Computer communications and networks*. Springer-Verlag London Limited. doi: 10.1007/978-1-84800-917-2_3

Kushner, D. 2013, February. *The real story of stuxnet*. Retrieved from http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet

Liu, W. 2012. Research on Cloud Computing security problem and strategy. In *2nd International conference on consumer electronics, communications and networks (cecnet)* (p. 1216-1219). Yichang: IEEE. doi: 10.1109/CECNet.2012.6202020

Lo, C., Huang, C. & Ku, J. 2008. Cooperative intrusion detection system framework for Cloud Computing networks. In *First IEEE international conference on ubi-media computing* (p. 280-284). IEEE.

Modi, C., Patel, D., Borisaniya, B., Patel, A. & Rajarajan, M. 2013. A survey on security issues and solutions at different layers of Cloud Computing. *Journal of Supercomputing*, 63, 561-592. doi: 10.1007/s11227-012-0831-5.

Mohamed, E. M. & Abdelkader, H. S. 2012, May. Enhanced data security model for Cloud Computing. In *The 8th international conference on informatics and systems* (infos2012) (p. cc12-cc17).

Rabai, L. B. A., Jouini, M., Aiss, A. B. & Mili, A. 2013. A cybersecurity model in Cloud Computing environments. *Journal of King Saud University — Computer and Information Sciences*, 25, 63–75.

Raza, S., Duquennoy, S., Hoglund, J., Roedig, U. & Voigt, T. 2012. Secure communication for the internet of things—a comparison of link-layer security and IPsec for 6lowpan. *Security and Communication Networks, Special Issue*, 1-15. doi:10.1002/sec.406

Roberts, J. C. & Al-Hamdani, W. 2011. Who can you trust in the Cloud? A review of security issues within Cloud Computing. In *Information security curriculum development conference*. ACM.

Srinivasan, M. K., Sarukesi, K., Rodrigues, P., Manoj, S. & Revathy. 2012. State-of-the-art Cloud Computing security taxonomies – a classification of security challenges in the present Cloud Computing environment. In *ICACCI '12 proceedings of the international conference on advances in computing, communications and informatics* (p. 470-476). ACM. doi: 10.1145/2345396.2345474

Sumter, L. 2010. Cloud Computing: Security risk. In *ACM SE 48th annual southeast regional conference*. NY, USA: ACM. doi: 10.1145/1900008.1900152

Vaquero, L. M., Rodero-Merino, L. & Morán, D. 2011. Locking the sky: a survey on IaaS Cloud security. *Computing*, 91(1), 93-118. doi: 10.1007/s00607-010-0140-x

Xin, Z., Song-qing, L. & Nai-wen, L. 2012. Research on Cloud Computing data security model based on multi-dimension. In *2012 international symposium on information technology in medicine and education* (p. 897-900). IEEE.

Yan, X., Zhang, X., Chen, T., Zhao, H. & Li, X. 2011. The research and design of Cloud Computing security framework. *Advances in Computer, Communication, Control & Automation*, LNEE 121, 757-763.

Younis, Y. A., Kifayat, K. & Merabti, M. 2014. An access control model for Cloud Computing. *Journal of Information Security and Applications*, 19, 45-60. doi: http://dx.doi.org/10.1016/j.jisa.2014.04.003

Zhang, G., Yang, Y. & Chen, J. 2012. A historical probability based noise generation strategy for privacy protection in Cloud Computing. *Journal of Computer and System Sciences*, 78, 1374–1381.

Zissis, D. & Lekkas, D. 2012. Addressing Cloud Computing security issues. *Future Generation Computer Systems*, 28, 583–592.