A New Enterprise Security Pattern: Secure Software as a Service (SaaS)

Santiago Moral-García[1], Santiago Moral-Rubio[2], Eduardo B. Fernández[3] and Eduardo Fernández-Medina[4]

[1] Kybele Research Group, Dept. of Computer Languages & Systems II, Rey Juan Carlos University, Madrid, Spain

[2] Chief Information Security Officer, BBVA Group, Madrid, Spain

[3] Secure Systems Research Group, Dept. of Comp. and Elect. Eng. and Comp. Science, Florida Atlantic University, Boca Raton, FL, U.S.A.

[4] GSyA Research Group, Dept. of Information Technologies and Systems, University of Castilla-La Mancha, Ciudad Real, Spain

Abstract. In recent years, the hiring of Software as a Service (SaaS) from cloud providers has become very popular. The advantages of using these services seem to be many, but organizations need to know and handle a variety of threats. Before using SaaS, organizations should check the security measures offered by the service provider and the defense mechanisms included in their enterprise security architectures. Security patterns are a good way to build and test new security mechanisms, but they have some limitations related to their usability. In order to improve the usability of security patterns, we have defined a new type of security pattern called *Enterprise Security Pattern*. In this paper, we show a brief description of enterprise security patterns, and document a new pattern that the organizations could apply to protect their information assets when using SaaS.

1 Introduction

The practice of outsourcing business functions has been around for decades. In recent years, its realization as online software service has become very popular [1]. Online software delivery is now conceived and defined as Software as a Service (SaaS). SaaS focuses on separating the *possession* and *ownership* of software from its *use* [2]. The advantages to using SaaS seem to be many, because this online service model may prove cheaper than owning and maintaining an in-house IT system [3]. Companies expect to save money on support and upgrade costs, IT infrastructure, IT personnel, and implementation. However, this new environment has some threats that organizations should handle, in order to protect their information assets.

Before using SaaS, organizations should check the security measures offered by the service provider, and the defense mechanisms included in their enterprise security architectures. Security patterns provide the guidelines to support the construction and

Moral-Garcia S., Moral-Rubio S., B. Fernández E. and Fernández-Medina E..
A New Enterprise Security Pattern: Secure Software as a Service (SaaS).
DOI: 10.5220/0004088500140026
In *Proceedings of the 9th International Workshop on Security in Information Systems* (WOSIS-2012), pages 14-26
ISBN: 978-989-8565-15-0

evaluation of new security mechanisms [4]. The use of security patterns helps to incorporate security principles when building secure systems [5]. However, they have some limitations:

- They are small units of defense. They can only handle one (or a few) threats. Considering the number of threats that can affect current information systems, a security designer should tailor an extensive set of security patterns when building secure systems.
- There are different versions of the same pattern for each architectural level. As the building of secure systems need an extensive set of security patterns, this fact increases the complexity when a security designer is trying to select a pattern.
- Several instantiations of a pattern may have common aspects but the designer has to find them. This fact may cause unnecessary redundancies.

Because of these limitations, we have defined a new type of security pattern called *Enterprise Security Pattern*. This new type of pattern tries to improve the usability of security patterns by incorporating them in a more comprehensive pattern that can handle more threats. In this paper, we document a new enterprise security pattern that organizations could apply to protect their information assets when using SaaS. Companies which have already hired SaaS could also consult this pattern, in order to verify if they are correctly protecting their assets.

The remainder of this paper is organized as follows. Section 2 provides a brief description of enterprise security patterns, including their assets, context and solutions models. Section 3 documents a new enterprise security pattern called *Secure Software as a Service (SaaS)*. Finally, Section 4 presents some conclusions and future work.

2 Enterprise Security Patterns

An enterprise security pattern is described by four models describing generic enterprise security architectures that provide some security properties for a set of information assets in a specific context. These patterns combine in one cohesive pattern: (i) the information assets to be protected, including their sensitivity level, (ii) the context in which these assets are found, (iii) the threats associated with the assets, (iv) the security policies, patterns, mechanisms and technologies used to stop these threats, and (v) the stakeholders and systems involved in the solution. Here, we show the assets, context and solution models used by these patterns.

2.1 Assets and Context Model

When building secure systems, organizations should use an information assets classification, in order to facilitate the security designer's work. The information assets should be classified in groups, according to their sensitivity levels, which depend on the relative value of the asset for the organization. This value may depend on several aspects or factors. For this reason, when classifying assets, the organizations should seek support from a risk analysis methodology.

The organizations' information assets may be classified into three large groups:

data, *applications*, and *code and configuration*. A common characteristic for all information assets is that they have to be stored in and may be transported through one or more components in security realms.

A security realm can be defined as a logical and discrete entity that partitions the enterprise network. The main purpose of these realms is to standardize enterprise security in order to reduce cost, users' delay, and administrative overhead of redundant security procedures [6]. The main characteristic of security realms is that they have in common the same security policies.

Policies are management instructions indicating a predetermined course of action, or a way to handle a problem or situation [7]. Without policies it is impossible to build secure systems, we don't know what we should protect and how much effort we should put on this protection [8]. A specific system uses a combination of security policies according to its goals and environment. When building secure systems, designers have to consider many security policies of different types, such as confidentiality policies, integrity policies, availability policies, etc.

When classifying the security realms, we take into account the *Types of Realms* (TR) that can be found in an enterprise network, and who manages each of those realms, i.e., a *Characteristic of the Realm* (CR). The classification of Security Realms (SR) that we propose here can be defined as *SR: TR x CR*. The specific realms can be adjusted to fit different types of applications; what matters here is that we use a classification of this type. Table 1 shows with an "X" the security realms provided in our classification.

Table 1. Classification of security realm.

		Characteristic of the Realm		
		Managed	*Externally Managed*	*Public*
Types of Realms	*Customer*	X	X	X
	Employee	X	X	X
	Technical User	X	X	X
	Development	X	X	-
	Data	X	X	-
	Bastion	X	X	-
	Transport	X	X	X

The security level of the policies applied when protecting the confidentiality of an asset can vary, depending on the security realm in which the asset is found, so its integrity, availability, and auditability should be protected in all the realms. For these reasons, we have defined a group of security policies that the enterprise security patterns will use to define the sensitivity level of an information asset. To achieve this, we have combined the answer of four dependent questions related to the following security aspects: *access authorization*, *encryption*, and *storage authorization*. The four questions are:

1. Can the information asset A be transported through the security realm SR?
2. If it is so, should A be encrypted?

3. Can A be stored in SR?
4. If it is so, should A be stored in hidden form?

Table 2 shows the possible answers, the security policies associated to each combination, and a number that denotes the Security Level (SL) provided by each policy (1 is the lowest and 6 the highest). SL will help us when designing the solutions of the enterprise security patterns.

Table 2. Security policies of the sensitivity level.

		Answers Combinations			
SL	**Security Policies**	*1*	*2*	*3*	*4*
4	*Secure Channel (SC) & Hidden Storage (HS)*	Yes	Yes	Yes	Yes
3	*Secure Channel (SC) & Clear Storage (CS)*	Yes	Yes	Yes	No
5	*Secure Channel (SC) & Blocked Storage (BS)*	Yes	Yes	No	-
-	*Clear Channel (CC) & Hidden Storage (HS)*	Yes	No	Yes	Yes
1	*Clear Channel (CC) & Clear Storage (CS)*	Yes	No	Yes	No
2	*Clear Channel (CC) & Blocked Storage (BS)*	Yes	No	No	-
6	*Blocked Channel (BC)*	No	-	-	-

After removing the policy *Clear Channel & Hidden Storage* (*CC & HS*), shadowed row, we obtain six different security policies. The policy *CC & HS* has been removed, because it is not usual find a security policy in which the information assets require encrypted storage and may be transported in clear way. When protecting the information assets, enterprise security patterns may use additional security policies, such as integrity policies, availability policies, auditability policies, etc.

Different organizations could apply different sensitivity levels to the same asset. For example, when classifying the customers' account, a food industry organization could decide to apply security policies with low or medium security level in all its security realms. However, a banking organization could decide to apply throughout security policies with high or very high security level. Due to this, enterprise security patterns do not try to protect single information assets. They intend to protect information assets that have the same sensitivity level.

2.2 Solution Models

The four complementary models or viewpoints included in the solutions of enterprise security patterns are: the *Computationally Independent Model* (CIM), the *Platform Independent Model* (PIM), the *Platform Specific Model* (PSM), and the *Product Dependent Model* (PDM). We discuss below each of them.

Computationally Independent Model: this model provides a description of the security policies that the system should enforce, independently of its functional and technological characteristics. The security policies should be applied to the information assets and security realms.

Platform Independent Model: this model provides a conceptual description of the security mechanisms that should be incorporated into the system and the relationships that exist among them, independently of its technological characteristics and implementation details. The same CIM could be instantiated N times in this model, since a security policy may correspond to different security patterns. A good guideline which can be used as a basis to select the security patterns needed is the guideline developed by Schumacher et al. in [9] or Fernandez in [10].

Platform Specific Model: this model defines the architectural components included in the enterprise security architecture, independently of the technology used to solve the problem. The PSM should take into account how to place the security mechanisms within the architecture. The same PIM can be instantiated N times in this model, since a security mechanism may be placed in different architectural components. The security patterns described in the PIM are included within architectural security components. Two good guidelines which can be used as a basis to select the architectural component are the ISO/IEC-27000-series [11] and the IT Baseline Protection Manual [12].

Product Dependent Model: it is necessary to install the PSM in a specific technological architecture in this model. The same PSM could be instantiated N times, since the same architectural component may correspond to different technological products. The technological products must be reliable products made by known manufacturers in the security industry. The final solution may vary significantly depending on the technologies used.

3 An Enterprise Security Pattern: Secure Software as a Service

We document here an enterprise security pattern which could be used by organizations, in order to protect the information assets when using outsourced online applications, for example, *Google Apps for business*. We discuss below each of sections included in the pattern template. This template includes sections of the template provided by Buschmann et al. [13], and some new sections that we consider necessary when designing enterprise security architectures.

3.1 Intent

This pattern attempts to protect the confidentiality of the data included in the outsourced online applications of an organization.

3.2 Context

Employees of an organization access from home (*Public Employee realm*, P-E) outsourced online applications. The service provider has placed the applications in a data center (*Externally Managed Data realm*, EM-D). Employees access the service provider through Internet (*Public Transport realm*, P-T). The service provider has an applications gateway (*Externally Managed Bastion realm,* EM-B) between Internet and its data center.

In order to allow employees continue using the access credentials that they use in their organization, when an employee tries to access the online applications, the service provider redirects the employee's browser to the organization's gateway (*Managed Bastion realm*, M-B). In that moment, employees validate their credentials in the organization's systems (*Managed Data realm*, M-Da) to get a ticket to access the provider. Once the employee has the ticket s/he can access his/her online applications. Figure 1 shows the context diagram of this pattern.

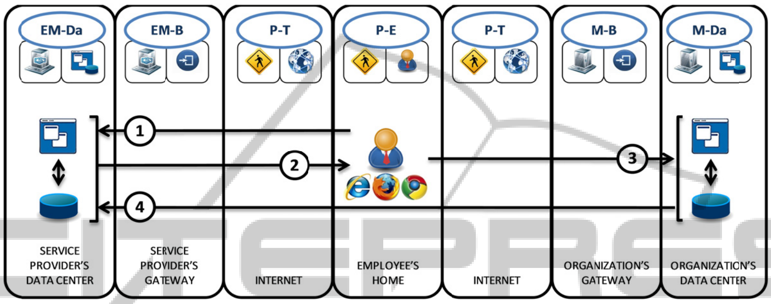
EM-Da
EM-B
P-T
P-E
P-T
M-B
M-Da
1
2
3
4
SERVICE PROVIDER'S DATA CENTER
SERVICE PROVIDER'S GATEWAY
INTERNET
EMPLOYEE'S HOME
INTERNET
ORGANIZATION'S GATEWAY
ORGANIZATION'S DATA CENTER

Fig. 1. Context diagram.

The sensitivity level (see Section 3) of the information assets (data) that this pattern attempts to protect are shown in Table 3.

Table 3. Sensitivity level of the information assets.

Security Realms	Security Policies	SL
Externally Managed Data	*Secure Channel (SC) & Hidden Storage (HS)*	*4*
Externally Managed Bastion	*Secure Channel (SC) & Hidden Storage (HS)*	*4*
Public Transport	*Secure Channel (SC) & Hidden Storage (HS)*	*4*
Public Employee	*Secure Channel (SC) & Clear Storage (HS)*	*3*
Managed Bastion	*Secure Channel (SC) & Hidden Storage (BS)*	*4*
Managed Data	*Clear Channel (CC) & Clear Storage (CS)*	*1*

As shown in Table 3, the data included in the outsourced applications should only be stored in clear form by the employee (P-E). The organization's data center (M-Da) could store the data in clear form, but these data are not related to the outsourced applications. The rest of realms should store the data in a hidden way. The data could leave the service provider, but the communication channels should be secure. The organization's data center (M-Da) could transport data in clear form, but these data are not related to the outsourced applications.

This pattern should only be used when the organization's employees use the outsourced online applications to store information assets that meet this sensitivity level. Organizations should ensure that no employee stores information assets with higher sensitivity levels.

3.3 Problem

In the past, mailbox and collaboration applications used by the organizations were placed within the organization. Employees accessed them from their home through the Internet. This context provoked a set of threats that the organizations had to handle. Some threats related to the confidentiality of data in this environment are (threats related to integrity and availability should also be handled):

- An attacker may read the accessed data by the employee via Internet. To prevent this, the organizations need to ensure that the communications between the employee and the data center are secure.
- An attacker may steal an employee's identity and access his/her applications. To prevent this, organizations need to ensure that the employee is who s/he claims to be.
- An attacker may take advantage of a vulnerability to access an employee's applications. To prevent this, organizations need to patch constantly the applications and the servers where the applications are hosted.
- A technical user who performs application maintenance may leak information. To prevent this, organizations need to ensure that the data are stored in a hidden way.

Once the cloud computing paradigm was born, the context for accessing the organizations' applications has changed significantly. In addition, some companies have as main objective providing and maintaining online applications of other organizations. The threats that the organizations have to handle in these contexts are similar to those of the context in which the employees accessed the organization's systems (listed previously), but how to handle them is different.

3.4 Known Incidents

Daily, there are many incidents of identity thefts. The main objective of these thefts is to obtain relevant information from the person (or company) attacked or steal his/her (or its) identity. One of the most notorious incidents was the case when hackers stole the access credentials of Fox News' Twitter account, and then they published that President Barack Obama was dead [14].

All companies are exposed to this type of theft. By using the pattern that we are describing here, companies could prevent that hackers can access the employees' online applications, even if they steal their identity. This is because the solution provided by the pattern ensures that the employee is who s/he claims to be (more detail in the next section).

3.5 Solution

We discuss below each of the models included in the solution:

Computationally Independent Model: we need to apply here the security policies included in the sensitivity level of the information assets. As shown in the diagram of the CIM (Figure 2), we could prevent that an attacker may intercept the applications' data, encrypting the channels and storing the data in a hidden way.

Platform Independent Model: we realize here the security policies of the CIM as security patterns. All security patterns included in the PIM are described in [9], except *Security Logger/Auditor* described in [15] and *Hidden Storage* which has not been described yet. The security pattern *Hidden Storage* should ensure that nobody unauthorized can read the information stored. Instantiations of the same pattern P are denoted as P_1, P_2, etc. The types of channels that we may find within of the PIM are: *clear channel* (single line), and *secure channel* (double line). In addition, these channels show a logical representation of the type of message that they transport. The type of messages that could be transported are: *request or response message* (solid line), and *record message* (dashed line).

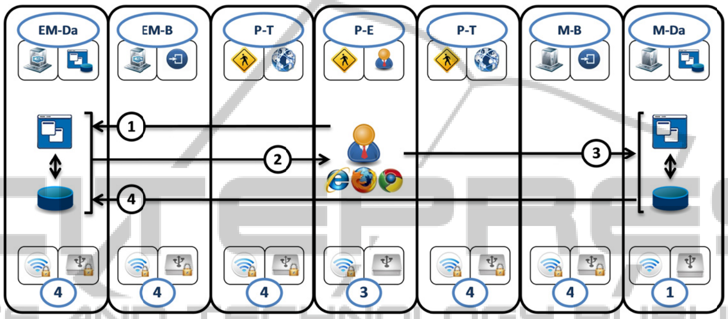

Fig. 2. Computationally independent model diagram.

We discuss below the sequence of actions shown in the PIM diagram (Figure 3):

1) The employee requests a secure channel through a browser to access his/her online applications (*Secure Channel_1*).

2) The service provider checks the employee's organization and redirects his/her browser to the organization's systems, including an access ticket associated with the employee.

3) The employee's browser requests a secure channel to access the organization's systems (*Secure Channel_2*).

4) The employee provides his/her credentials in the organization's systems (*Identification_2*).

5) The organization checks that the employee is who s/he claims to be (*Authentication*). If the validation is successful, the employee obtains the signed ticket to access his/her online applications.

6) The organization checks that the employee is who s/he claims to be (Authentication). If the validation is successful, the employee obtains the signed ticket to access his/her online applications.

7) The organization's systems redirect the employee's browser to the service provider's systems, including the signed access ticket. *Access Control* checks if the signed ticket is the same that the ticket generated previously.

8) If the access ticket is valid, the employee could access his/her applications.

9) The applications must make clear the data so that they can be shown the employee (*Hidden Storage*).

10) Before showing the online applications to the employee, the service provider rechecks that the employee has permissions to access those data and applications (*Access Control*).

11) The service provider shows the employee his/her online applications through a *Secure Channel*.

12) The service provider stores a session cookie in the employee's browser. From this point, the employee could access his/her online applications without re-authenticate.

In order to audit possible attacks, the mechanisms for identification, authentication, access control, and hidden storage should record all activity in the security patterns *Security Logger/Auditor*.

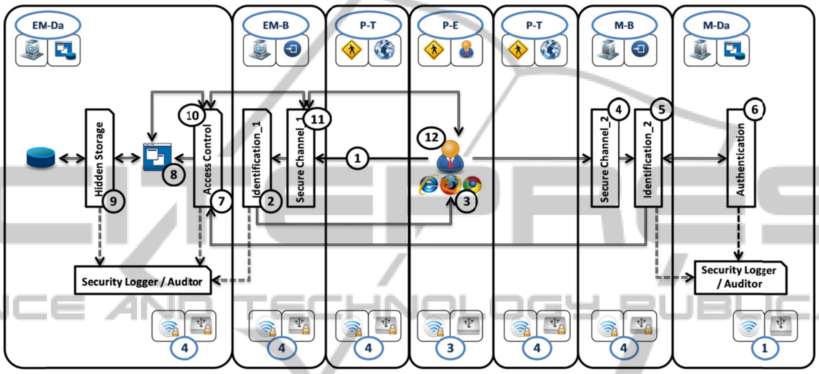

Fig. 3. Platform independent model diagram.

Platform Specific Model: we transform here the security patterns of the PIM into architectural components. As shown in the diagram of the PSM (Figure 4), the security pattern instantiations *Secure Channel_1* and *Identification_1* are transformed into a *Web Server*. The security pattern instantiations *Secure Channel_2* and *Identification_2* are transformed in a *Reverse Proxy*. The security pattern instantiation *Security Logger/Auditor* is transformed in a *Log System*. The security pattern instantiation *Access Control* and the applications are transformed into an *Application Server*. Finally, the security pattern instantiation *Hidden Storage* and the applications' data are transformed into a *Dissociation Data Server*.

In order to prevent that an attacker may access an employee's applications, after stealing his/her identity; we need to ensure that the employee is who s/he claims to be. To do this, an authentication system with high security level should be used (something stronger than passwords). In the solution of the pattern, we have decided to include *Token-Based Authentication Server*, because its use is currently more widespread, but we could also have used biometric authentication or some other kind of strong authentication.

Product Dependent Model: we transform here the architectural security components into technological products. The diagram of the PDM (Figure 5), shows the technological products that we have decided to include in the solution. We have selected these technologies, because we consider them reputable and currently used by many organizations; but we could have selected another set of technologies.

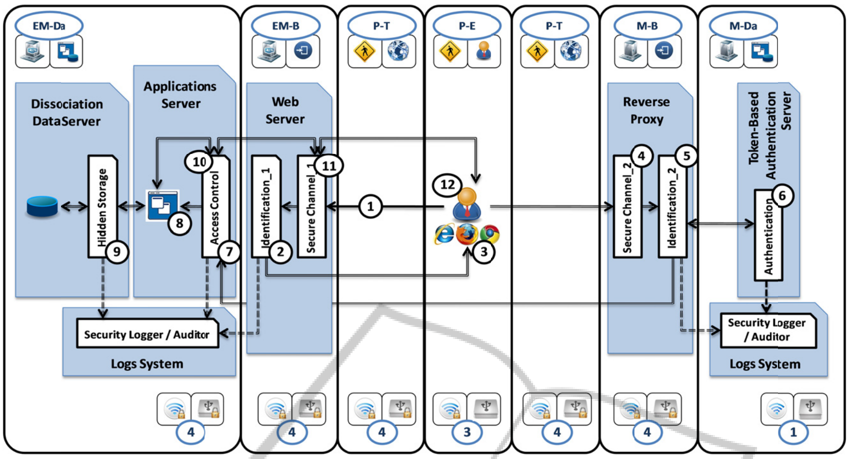

Fig. 4. Platform specific model diagram.

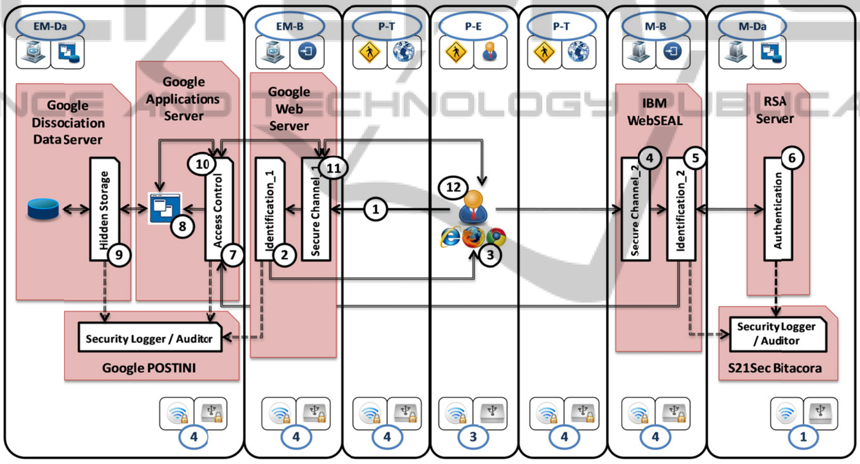

Fig. 5. Product dependent model diagram.

3.6 Considerations

The considerations of the pattern take into account the selected technologies in the PDM of the solution. Another set of technologies could change this analysis. Table 4 shows the result of the analysis for each of the relevant aspects.

We can see in Table 4 that, when deploying the solution, the performance overhead of the enterprise security architecture does not increase; even it could decrease because part of the IT infrastructure would be outsourced. The security and log administrator has to work in some cases outside the organization. This fact may mean a small increase of personnel in the security and log team. The installation cost does not increase; even it could decrease because IT infrastructure, IT personnel, implementation, and maintenance are outsourced. Once the solution is deployed the residual risk is minimal. This means that the solution does not need complementary measures to attain its initial objective.

Table 4. Considerations.

	Aspects to consider	Analysis
Performance overhead	*Storage*	*0*
	Primary Memory	*0*
	Processor	*0*
	Bandwidth	*0*
Complexity	*Security Administrator*	*1*
	Log Administrator	*1*
	End User	*0*
	Massive Expansion	*0*
	System Administrator	*0*
	Installation Cost	*0*
	Residual Risk	*0*

3.7 Consequences

As we said previously, threats found in the problem are related to the confidentiality of the assets. Integrity and availability threats could be handled in similar ways. We discuss below the security mechanisms that we have included in the pattern, in order to prevent or reduce the risk of the identified threats:

- An attacker may read the data accessed by the employee via Internet. The service provider's *Web Server* and the organization's *Reverse Proxy* using secure channels can prevent this.
- An attacker may access an employee's applications, after stealing his/her identity. To prevent this, we include a *Token-based Authentication Server*.
- An attacker may use a vulnerability to access an employee's applications. To prevent this, the service provider has to constantly patch the applications and the servers where the applications are hosted.
- A technical user who performs the applications maintenance may leak information. To prevent this, we include a *Dissociation Data Server* in the service provider.

This pattern would also be applicable in a context where the employees access their online applications from the organization (Managed Employee realm, M-E), rather than from their home.

3.8 Known Uses

As shown previously, Google is one of the online applications' providers that offer the security measures and architecture included in the pattern's solution. One of their most popular products is *Google Apps*. Forty million active users and four million of businesses are currently using it [16], including Florida Atlantic University and BBVA Group.

4 Conclusions and Future Work

Using Software as a Service (SaaS) has currently become very popular. This popularity is caused because companies could save money on support and upgrade costs, IT infrastructure, IT personnel, implementation, and maintenance. However, before using SaaS, organizations should check the security measures offered by the service provider.

Security patterns are a good way to construct and evaluate new security mechanisms, but they are not applied as much as they could be, because designers have problems in selecting them and applying them in the right places. Enterprise security patterns could improve the application of the patterns by incorporating them in a more comprehensive pattern that may handle more threats. There will be a smaller number enterprise security patterns, which makes their selection simpler for designers.

When adopting SaaS, organizations could consult the enterprise security pattern that we have presented here, in order to protect the data included in the outsourced applications from a common set of threats. Organizations which have already adopted SaaS could also consult this pattern in order to verify if they are correctly protecting their information assets. As future work, we will intend to document more enterprise security patterns which can be consulted by organizations when using Platform as a Service (PaaS) or Infrastructure as a Service (IaaS).

Acknowledgements

This research has been hosted at Florida Atlantic University (FAU) and carried out in the framework of the following projects: MASAI (TIN2011-22618) financed by the Spanish Ministry of Education and Science, BUSINESS (PET2008-0136) financed by the Ministry of Science and Innovation, and SISTEMAS (PII2I09-0150-3135) and SERENIDAD (PEII11-0327-7035), all financed by the Local Government of Castilla-La Mancha, in Spain.

References

1. Espadas, J., Concha, D., Molina, A.: Application Development over Software-as-a-Service platforms. In The Third International Conference on Software Engineering Advances (2008).
2. Turner, M., Budgen, D., Brereton, P.: Turning Software into a Service. Computer, 36 (10), pp. 38-44 (2003).
3. Ma, D.: The Business Model of Software-As-A-Service. In IEEE International Conference on Services Computing (SCC 2007) (2007).

4. Fernandez, E., Washizaki, H., Yoshioka, N., Kubo, A., Fukazawa, Y.: Classifying Security Patterns. In Progress in WWW Research and Development, pp. 342-347 (2008).
5. Hafiz, M., Adamczyk, P., Johnson, R. E.: Organizing Security Patterns. Software, IEEE, pp. 52-60 (2007).
6. Arconati, N.: One Approach to Enterprise Security Architecture. SANS Institute(2002).
7. Wood, C. C.: Information Security Policies Made Easy. Version 7 (2000).
8. Fernandez, E. B., Gudes, E., Olivier, M.: Policies and Models. In The design of secure systems (under contract with Addison-Wesley).
9. Schumacher, M., Fernandez-Buglioni, E., Hybertson, D., Buschmann, F., Sommerlad, P.: Security Patterns: Integrating Security and Systems Engineering. Wiley (2006).
10. Fernandez, E. B.: Security patterns in practice: Building secure architectures using software patterns. under contract with J. Wiley (To appear in the Wiley Series on Software Design Patterns).
11. ISO: International Organization for Standarization. http://www.iso.org (retrieved: March, 2012).
12. BSI: IT Baseline Protection Manual. Federal Agency for Security in Information Technology, Germany(2000).
13. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: Pattern-oriented software architecture: A system of patterns. Wiley, (1996).
14. GIZMODO: Fox News' Twitter Account Hacked. http://gizmodo.com/5817870/fox-news-twitter-account-hacked-claims-barack-obama-is-dead (retrieved: March, 2012).
15. Fernandez, E. B., Mujica, S., Valenzuela, F.: Two security patterns: Least Privilege and Security Logger/Auditor. In Asian PLoP (2011).
16. Google: Businesses share their stories - Google Apps. http://www.google.com/apps/intl/en/customers/index.html (retrieved: March, 2012).