World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:15, No:6, 2021

Platform-as-a-Service Sticky Policies for Privacy Classification in the Cloud

Maha Shamseddine, Amjad Nusayr, Wassim Itani

Abstract**—**In this paper, we present a Platform-as-a-Service (PaaS) model for controlling the privacy enforcement mechanisms applied on user data when stored and processed in Cloud data centers. The proposed architecture consists of establishing user configurable 'sticky' policies on the Graphical User Interface (GUI) data-bound components during the application development phase to specify the details of privacy enforcement on the contents of these components. Various privacy classification classes on the data components are formally defined to give the user full control on the degree and scope of privacy enforcement including the type of execution containers to process the data in the Cloud. This not only enhances the privacy-awareness of the developed Cloud services, but also results in major savings in performance and energy efficiency due to the fact that the privacy mechanisms are solely applied on sensitive data units and not on all the user content. The proposed design is implemented in a real PaaS cloud computing environment on the Microsoft Azure platform.

Keywords**—**Privacy enforcement, Platform-as-a-Service privacy awareness, cloud computing privacy.

I. Introduction

CLOUD computing services are categorized into three main categories: Infrastructure as a Service (IaaS) [1,2], Platform as a service (PaaS) [2] and Software as a Service (SaaS) [3]. IaaS is acquired when physical resources are needed by a customer without going through the complexities of setting up and configuring hardware. This virtual service allows the customer to select, configure, and use virtualized computing resources and is then billed based on her usage of these resources. Customers can also choose the notion of hybrid clouds; based on the idea of customers connecting cloud services to their own network via a Virtual Private Network Connection (VPN). Amazon and Microsoft are the key players in the IaaS arena through their EC2 [4] and Azure [5] cloud platforms respectively. SaaS in the Cloud encompasses the traditional software services that customers used to consume but with major enhancements related to service usability and billing. Again, the concept of virtualization plays a significant role towards providing an "elastic" software environment that can charge customers in a pay-as-you-go fashion with high degrees of flexibility in resource upgrading or even downgrading. Companies such as Google and SalesFoce are considered the main pioneers and providers for such SaaS services. PaaS, on the other hand, provides customers with an environment for development and deployment. PaaS delivers virtualized hardware and software tools to the customer. Customers are not worried about the infrastructure in this case but rather the type and the degree of services provided. The notion of a hybrid PaaS also exists if a customer or company wishes to use a combination of their own systems with the virtual services. There are various types of PaaS services to developers, including but not limited to: public PaaS, private PaaS, mobile PaaS, and open PaaS. The usage of PaaS has proven to be cost effective on many levels. One example is cross-platform application development. PaaS allows developers to design, develop and test software with lower costs. Another example is using PaaS as the platform for web real-time communication (WebRTC); giving a web browser or a mobile application the ability to communicate over the network using customized and highly efficient application programming interfaces (APIs). Companies acquiring such a service save sizable costs in physical hardware investments and get relieved of the daunting concerns related to downtime or system upgrades since all of these aspects are on the PaaS server side.

In a PaaS environment, the software development and deployment are hosted on the server side and clients usually access this software using a thin web client. Many PaaS platforms charge their customers on a subscription basis. The customer is billed either monthly or annually or on some agreed usage of the time. This provides cost effective real-time access to software.

Privacy is a major hindering factor in the widespread adoption of Cloud computing. Outsourcing the clients' data and software for storage and processing on remote servers that are not managed or controlled by the customers themselves poses major security and privacy risks. Cloud providers, even reputable ones, cannot be fully trusted to store sensitive customer data or execute software operating on such sensitive data. For this reason, privacy enforcement should be given exceptional attention when designing Cloud software services to protect sensitive data from malicious or the least curious Cloud service providers. Privacy has many definitions. An early general definition provided by the American Institute of Certified Public Accountants (AICPA) is "The rights and obligations of individuals and organizations with respect to the collection, use, retention, and disclosure of personal information" [18]. When thinking about privacy, many considerations arise including data access, retention, data destruction, monitoring, and how to mitigate breaches. Privacy

Maha Shamseddine is with the Department of Electrical & Computer Engineering, Beirut Arab University, Beirut, Lebanon (e-mail: m.shamseddine@bau.edu.lb).

Amjad Nusayr and Wassim Itani are with the Department of Computer Science, University of Houston-Victoria, Victoria, Texas, USA (e-mail: nusayra@uhv.edu itaniw@uhv.edu).

and security go hand in hand, although some would argue that one cannot achieve privacy without security.

Privacy policies in cloud computing should be applied on every level in the software development lifecycle. Cloud providers must implement the necessary measures to ensure the integrity and confidentiality of the customer data as well as provide Cloud customers with the needed control on the different aspects related to the storage and processing of their sensitive information. This study utilizes a PaaS model for controlling privacy enforcement mechanisms that are applied on user data from GUI components when stored and processed in Cloud data centers. This is accomplished by creating 'sticky' policies to GUI components to be used during the development of software. Our model separates the notion of privacy policies and applying them in a modular approach to GUI components.

The rest of the paper is as follows. Section II shows the related work. Section III shows the design of the presented model. Section IV introduces the prototype implementation. Conclusion and future extensions are in Section V.

II. Related Work

A lot of research work has targeted the topic of privacy in cloud computing. The main breakthrough in this domain is represented by the introduction of the first fully Homomorphic encryption scheme by Gentry in 2009 [6]. Homomorphic encryption allows the processing operations to be applied on data while it is in the encrypted state. In this way the privacy of the information is enforced even when processed in the Cloud data center since there is no need to decrypt the data by the Cloud service provider to carry out the processing operations. It should be noted here that homomorphic encryption is not a new concept and has been known for over 30 years. However, the key contribution in Gentry's fully homomorphic encryption scheme is that the processing can be applied on both the addition as well as the multiplication micro operations constituting the service functions. Since any algorithm can be mapped to a set of circuit-level AND (multiplication) and XOR (addition) gates, Gentry's fully homomorphic scheme allows operations on encrypted data using any computable service function. The main obstacle in deploying homomorphic encryption schemes to secure the privacy of Cloud services is the high resource consumption required by the public-key cryptographic operations employed by these schemes. This fact has left homomorphic encryption implementations applicable in limited application domains with a very narrow practicability margin. Many research works have implemented improvements on Gentry's fully homomorphic encryption scheme to make it more feasible for real-world Cloud deployments [7]-[10]. Despite a major improvement of about 70X in the speed of homomorphic encryption operations [11], still this encryption scheme is not ready for use in large scale Cloud services. The other approach to target privacy challenges in Cloud computing is to employ tamper-proof cryptographic hardware to provide a set of physically and logically isolated execution containers to process sensitive data. The work in [12] presented a set of privacy protocols leveraging cryptographic coprocessors to provide the necessary privacy enforcement in the Cloud. The main concept in [12] is to develop a software division process that specifies the protected parts of software that need to process sensitive data and the non-protected parts that do not need any privacy enforcement. Accordingly, the protected software parts are allocated to be processed in tamper-proof cryptographic coprocessors installed by a trusted third party in the Cloud data centers. The privacy protocols specified provide a privacy feedback mechanism that informs users of the different privacy operations carried out on their sensitive data in the Cloud.

Many surveys considered data privacy in Cloud computing. Some important ones on this topic are presented in [13]-[15]. In [16] the authors present a PaaS content-based policy-driven security system for specifying the encryption range and strength on Cloud network data. The work presented in this paper is inspired by [16], however the main focus in this paper is on classifying the privacy of data storage and processing rather than on the data transfer over the network links.

III. System Design

The system model we assume in this work follows a traditional Cloud computing architecture consisting of a Cloud customer or client requesting a software development and execution services from a PaaS Cloud Service Provider (CSP). Fig. 1 presents the main system components for enforcing the privacy of customer data as it is processed and stored in the Cloud data centers. The main privacy enforcement components are described as follows:

1. The Privacy Policy: is a detailed XML-formatted specification of the privacy rules to be applied on the GUI data-bound component values. The privacy policy rules are established and linked to the respective GUI components at the software development phase. The policy rule enforcement is done at the software execution phase. The main rule syntax of the privacy policy is presented in Fig. 2. The privacy rules are wrapped in a policy configuration component giving software developers a set of flexible policy specification options to apply on the GUI data-bound components at the software development phase. The privacy rules on each GUI data-bound component specify the type of privacy enforcement mechanism to apply on the data value of that respective component. Three main rules are supported currently by the privacy policy:

a. Tamper-proof-processing: this option specifies that the sensitive customer data must be processed in tamper-proof processing containers in the Cloud data center. Tamper-proof execution typically leverages cryptographic coprocessors [12] installed in the data center to keep the data processing secure against tampering even form the CSP.

b. Homomorphic-enc: this option specifies that the sensitive customer data are to be processed using homomorphic encryption functions running in the Cloud. Homomorphic encryption allows the processing on data in the encrypted

state (by a customer key) without the need for decrypting it first.

c. Encrypt-for-storage: this option considers the privacy of data that do not require any processing in the Cloud and that need to be stored encrypted in the Cloud storage. The encryption is done on the customer-side using a customer encryption key and sent encrypted to the Cloud storage.

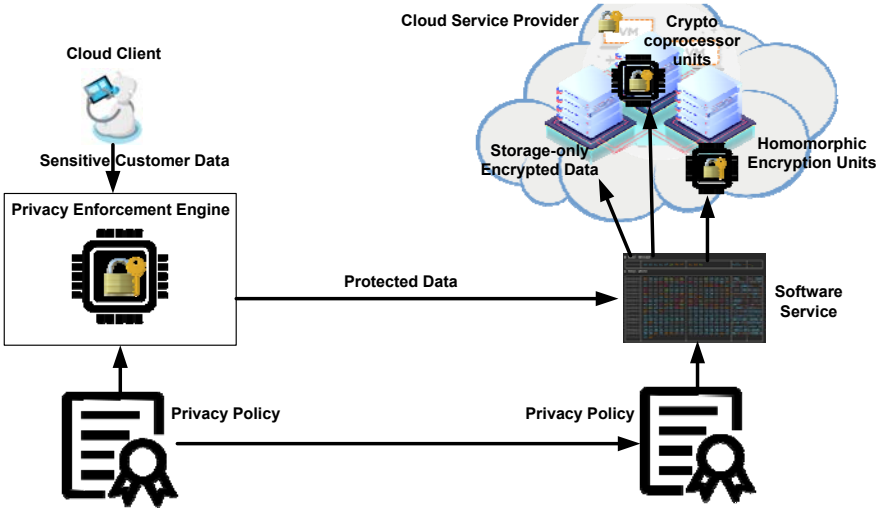

Fig. 1 Main components of the privacy-classification system architecture

```
<privacy-policy version=1.0>
<privacy-configuration number="1">
<Data-Bound-GUI-Comp name=TextBox1 value="5322-
1342-564-7843">
<tamper-proof-processing>true</tamper-proof-processing >
<encryption-key>ref:4265_client</encryption-key>
< /Data-Bound-GUI-Comp>
<Data-Bound-GUI-Comp name=TextBox2
value="$67990">
<homomorphic-enc>true</ homomorphic-enc>
<encryption-key>ref:3243_client</encryption-key>
< /Data-Bound-GUI-Comp>
<Data-Bound-GUI-Comp name=ComboBox1
value="Victoria, tx, 77901">
< encrypt-for-storage >true</encrypt-for-storage >
<encryption-key>ref:5693_client</encryption-key>
< /Data-Bound-GUI-Comp>
</privacy-configuration>
<privacy-configuration number="2">
                    .
                    .
                    .
</privacy-configuration>
</privacy-policy>
```

Fig. 2 Sample prototype policy configuration

2. The Privacy Enforcement Engine: applies the rules of the sticky privacy policy on the sensitive customer data received from the Cloud client. The enforcement is based on the privacy rules specified in the selected privacy configuration. The cryptographic operations are performed using a reference to the encryption key stored on the client side. The Cloud provider enforces the same privacy policy on the data received from the customer privacy enforcement engine and processes it based on the selected privacy configuration rules. So sensitive data are (1) processed in tamper-proof execution units controlled by a trusted third party on behalf of the Cloud customer, (2) processed using homomorphic encryption functions implemented in software in the Cloud, or (3) stored encrypted in the Cloud storage without the need for any processing operations.
3. The privacy execution containers: are software and hardware-based execution containers deployed in the Cloud data centers to process the sensitive customer data. As mentioned previously, the hardware approach relies on tamper-proof cryptographic coprocessor while the software approach employs homomorphic encryption processing units that can be operate on data in the encrypted form.

IV. Implementation

A sample prototype test-bed implementation is developed by employing the Microsoft.Net framework [17]. The privacy model implementation utilizes the libraries and classes that are a major component delivered as part of the framework. System.Windows.Forms is a Namespace that contains all the needed classes for creating Windows based applications operating under Windows operating system. All GUI components (Text Boxes, buttons, etc) inherit from the **Control** class that is basically an essential part of the System.Windows.Forms Namespace.

The privacy model modifies the GUI components by customizing a derived class to contain the privacy configurations. To specify the privacy policy, this derived class inherits all the members of the Control class and adds a new data member that encapsulates all the necessary fields and operations to implement privacy on a component level. The Control class is the base class for all visible GUI components.

Fig. 3 illustrates this by taking the base class, adding the extra members, and an example of using the ButtonBase class inheriting from modified Control class.

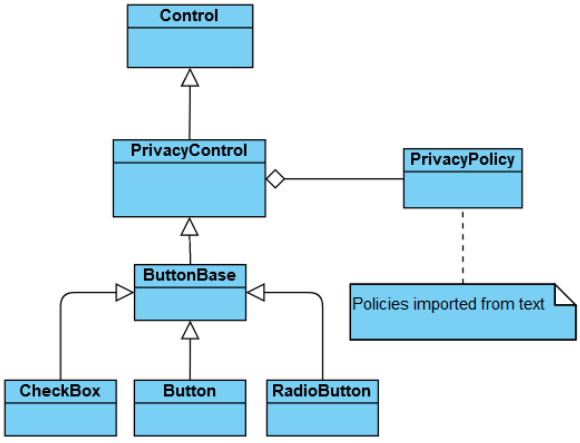

Fig. 3 UML diagram for the binding the security policy configurations to the .NET GUI data-bound components

Since GUI applications are event driven, the execution of the privacy configurations happens when a user generates an event. The Privacy model is not concerned with all events but rather with the events that initiate some kind of data transfer. This implementation is possible by creating a custom "EventHandler" delegate object that is responsible for calling events handlers for each GUI component.

A developer can add a component to a Windows Form Application and choose which privacy configurations to apply. Fig. 4 illustrates an example of using the button component. Once the user adds the button, an option will be provided to what privacy policy configuration the user wants to implement for that component.

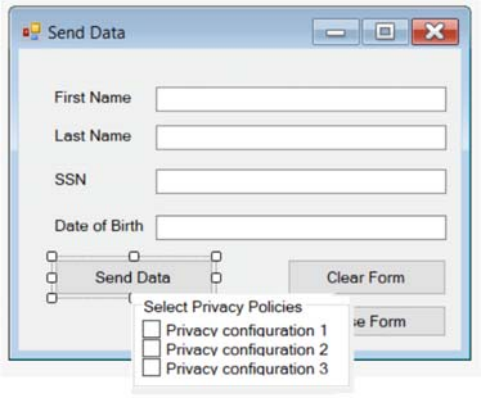

Fig. 4 Basic prototype for graphically specifying sticky privacy policy configurations to the GUI data-bound components

V. Conclusion and Future Extensions

In this paper we presented a PaaS policy-based system for privacy classification of user data as it is stored and processed in the Cloud data centers. The design is presented briefly together with a proof-of-concept implementation using the .Net framework. Due to the time limitation of this paper, many future extensions can be applied to enhance this work. This includes: (1) a more comprehensive design description of the system components, specifically the privacy enforcement engine and the operation of the homomorphic and tamper-proof privacy schemes in the Cloud, (2) a formal analysis of the privacy methods proposed and a proof of their effectiveness in enforcing the privacy of the sensitive user data, (3) a more extensive implementation scheme that includes all the cryptographic and policy-based mechanisms proposed on the client as well as on the Cloud sides.

References

[1] Bhardwaj, Sushil, Leena Jain, and Sandeep Jain. "Cloud computing: A study of infrastructure as a service (IAAS)." International Journal of engineering and information Technology 2, no. 1 (2010): 60-63.
[2] Keller, Eric, and Jennifer Rexford. "The" Platform as a Service" Model for Networking." INM/WREN 10 (2010): 95-108.
[3] Dubey, Abhijit, and Dilip Wagle. "Delivering software as a service." The McKinsey Quarterly 6, no. 2007 (2007): 2007.
[4] Amazon AWS homepage: http://aws.amazon.com
[5] Microsoft Azure homepage: http://azure.microsoft.com
[6] Gentry, Craig. "Fully homomorphic encryption using ideal lattices." In Proceedings of the forty-first annual ACM symposium on Theory of computing, pp. 169-178. 2009.
[7] Brakerski, Zvika, and Vinod Vaikuntanathan. "Efficient fully homomorphic encryption from (standard) LWE." SIAM Journal on Computing 43, no. 2 (2014): 831-871.
[8] Brakerski, Zvika, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping." ACM Transactions on Computation Theory (TOCT) 6, no. 3 (2014): 1-36.
[9] Brakerski, Zvika, and Vinod Vaikuntanathan. "Fully homomorphic encryption from ring-LWE and security for key dependent messages." In Annual cryptology conference, pp. 505-524. Springer, Berlin, Heidelberg, 2011.
[10] Fan, Junfeng, and Frederik Vercauteren. "Somewhat Practical Fully Homomorphic Encryption." IACR Cryptol. ePrint Arch. 2012 (2012): 144.
[11] Halevi, Shai, and Victor Shoup. "Faster homomorphic linear transformations in HElib." In Annual International Cryptology Conference, pp. 93-120. Springer, Cham, 2018.
[12] Itani, Wassim, Ayman Kayssi, and Ali Chehab. "Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures." In 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, pp. 711-716. IEEE, 2009.
[13] Zhou, Minqi, Rong Zhang, Wei Xie, Weining Qian, and Aoying Zhou. "Security and privacy in cloud computing: A survey." In 2010 Sixth International Conference on Semantics, Knowledge and Grids, pp. 105-112. IEEE, 2010.
[14] Sun, Yunchuan, Junsheng Zhang, Yongping Xiong, and Guangyu Zhu. "Data security and privacy in cloud computing." International Journal of Distributed Sensor Networks 10, no. 7 (2014): 190903.
[15] Xiao, Zhifeng, and Yang Xiao. "Security and privacy in cloud computing." IEEE communications surveys & tutorials 15, no. 2 (2012): 843-859.
[16] Itani, Wassim, Ayman Kayssi, and Ali Chehab. "SNUAGE: an efficient platform-as-a-service security framework for the cloud." Cluster computing 16, no. 4 (2013): 707-724.
[17] Microsoft .Net Framework Homepage: https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet-framework
[18] American institute of certified public accountants: https://www.aicpa.org/