2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing

Demo Paper: Automatic Provisioning, Deploy and Monitoring of Virtual Machines based on Security Service Level Agreement in the Cloud

Kazi Wali Ullah and Abu Shohel Ahmed
NomadicLab, Ericsson Research
Helsinki, Finland
e-mail: {kazi.wali.ullah, ahmed.shohel}@ericsson.com

***Abstract*—This demo presents a Security Service Level Agreement (SSLA) management solution for the cloud. In this work we aim to bind security in the Service Level Agreement (SLA) as a measurable and agreeable parameter between cloud service provider (CSP) and the customer. To achieve this, we allow cloud customer to choose between different security levels when negotiating the SLA and then our automated SLA engine finds the requirements from the SSLA and deploys the Virtual Machine (VM) based on that. Finally, we also provide monitoring of the security services from where the customer can review the current security status of the VMs. If there are any violations from the agreed SSLA, then the customer can immediately notice that and file a SLA breach complaint to the CSP.**

Keywords-Security Service Level Agreements; Cloud Computing; Security Monitoring;

I. Introduction

Service level agreements (SLAs) play an important role in todays cloud based service delivery. Nonetheless, it is intriguing to notice that, security, being one of the major concerns in the cloud arena, has not been included in the mainstream cloud SLAs yet. Though there have been quite a lot of early and recent works in this domain [1], [2], [3], [4]. To differ this, we aim to bind security in the SLA as a measurable and agreeable parameter. Binding security in the SLA has several effects. First of all, security can only be added to an SLA when it can be measurable. To measure the security, a client and the CSP has to agree on the security parameters that will be measured. So, it will help to gain a good understanding of the overall security of the system. Secondly, if we can bind security in the SLA, then it will bring more trust between the CSP and the user. Therefore, in this demo, we present an approach to establish SSLA between the CSP and the cloud customer.

II. Security Parameters

There are many different parameters that can be included in a SSLA. In our demo, we have chosen two different security parameters to work with. First one is the *VM container security profile*, which indicates the physical security of the machines where the VMs will be deployed. This parameter can take three different values, which are High, Medium and Low. These three levels depend on physical security parameters such as availability of trusted platform module (TPM), hypervisor hardening, other physical security issues of the data center, etc. The second security parameter is the *VM Application security profile*. This indicates the application level security requirements of the VM. Again, it can take three different values: High, Medium and Low based on what kind of application level security solutions are there in the VM image. For example, the level will depend on password policy, encryption algorithm in use, firewall status, antivirus status, etc.

Along with these two security related parameters, we have also considered a cost parameter associated with physical location. This indicates the running cost for the VM in a particular location. We have considered this parameter from the fact that, there may be similar locations to run a VM from the security standpoint, but they might incur different monetary cost.

III. Demo Overview

Figure 1 depicts the overall view of our demo. The web GUI serves as the automatic SSLA negotiating entity. Here, the customer can choose between different security levels for the security parameters mentioned above for their VMs. These requirements are then processed by our custom built

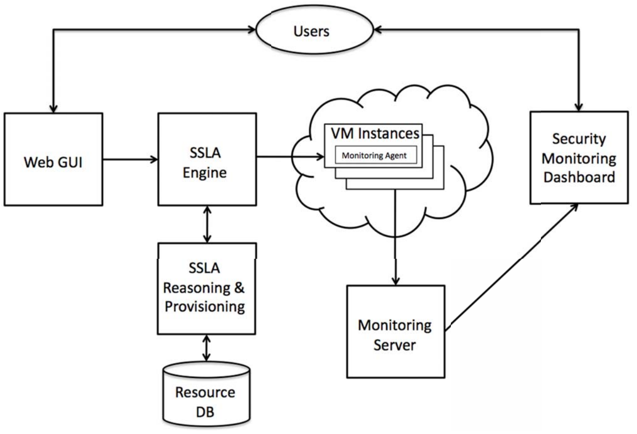

Figure 1. Overview of the SSLA solution

978-1-4799-2784-5/14 $31.00 © 2014 IEEE
DOI 10.1109/CCGrid.2014.39

SLA engine to deploy the VMs. To do this, it analyzes the technical requirements from the SSLA to provision the VM in an appropriate physical location and solves the constraints for optimal cost. Then it launches a VM in the OpenStack [5] cloud with the desired security configuration. While launching the VM, SLA engine pushes a custom written monitoring agent inside the VM, which will later monitor the security configuration of the VM periodically. The reports from this monitoring agent are stored in a centralized monitoring server. The customer of the cloud can review the current security status of the VM from this monitoring server through a security-monitoring dashboard.

The demo sequence diagram shown in Figure 2 is as follows:

1) Customer/tenant queries the SLA engine for SSLA parameters.
2) The SSLA engine replies with available security configuration options which will form the SSLA.
3) Customer chooses the desired level of security for the VM to be deployed in the cloud and sends it to the SLA engine.
4) SLA engine forwards this SSLA to the SLA reasoning engine. The reasoning engine splits the SLA into technical constraints and solves the constraints to find an optimal solution. During this phase it also considers the cost of the deployment and also the availability of physical resources that will satisfy the physical security requirements.

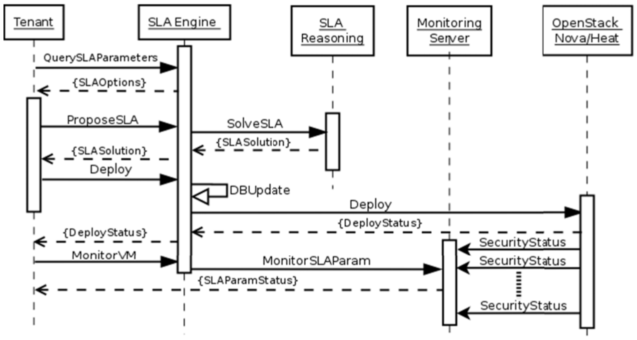

Figure 2. Sequence diagram for our SSLA solution in the OpenStack Cloud

5) SLA reasoning component replies to the SLA engine about the feasibility. If it is possible to deploy the VM with the customers desired SSLA, then it replies with the information where to deploy the VM, what is the running cost, etc. If it cannot satisfy the customers SSLA, then it sends a negative reply.
6) Solution received from the SLA reasoning component eventually forwarded to the customer by the SLA engine. Here, they customer can decide whether to deploy the VM with the given cost or go back and change the security requirements.
7) When the customer selects deploy option, the SLA engine updates its internal resource database to change the availability status of the physical resources.
8) SLA engine instructs the OpenStack to launch a VM. During this instruction the SLA engine also passes the script that will change the security configurations of the VM based on the SSLA. Also, a monitoring agent is launched within the VM which will monitor the security settings of the VM internally.
9) Success/failure notification of launching a VM is sent back to the SLA engine and eventually to the client.
10) The monitoring agent keeps sending the monitoring data to the centralized monitoring server periodically.
11) When the customer/tenant wants to review the current security status of the deployed VM, the request has to go to the SLA engine first.
12) The SLA engine forwards this request to the monitoring server which generates the current security status of the VM based on the monitoring data received from the monitoring agent. This result is displayed on a separate security-monitoring dashboard.

IV. Conclusion

Despite the fact that measuring security is hard, we have demonstrated a solution where cloud users can negotiate security at their required level. Based on the chosen security level, our SLA engine provisions and deploys the VMs in the cloud automatically. After the deployment, a monitoring engine takes over and reports about the actual security status of the deployed VM. Therefore the customer is able to check the current security status of the VM and if there are any violations from the agreed SSLA, then the customer can file a complaint with the vendor immediately with a proof of the violation.

References

[1] R. R. Henning, "Security service level agreements: quantifiable security for the enterprise?" in *Proceedings of the 1999 workshop on New security paradigms*. ACM, 1999, pp. 54–60.

[2] M. G. Jaatun, K. Bernsmed, and A. Undheim, "Security slas–an idea whose time has come?" in *Multidisciplinary Research and Practice for Information Systems*. Springer, 2012, pp. 123–130.

[3] M. B. Chhetri, Q. B. Vo, and R. Kowalczyk, "Policy-based automation of sla establishment for cloud computing services," in *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*. IEEE, 2012, pp. 164–171.

[4] K. Bernsmed, M. G. Jaatun, P. H. Meland, and A. Undheim, "Security slas for federated cloud services," in *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*. IEEE, 2011, pp. 202–209.

[5] "OpenStack >> Open Source Cloud Computing Software," http://openstack.org.