Developing Dependable and Secure Cloud Applications

Authors: Ingo Weber, Surya Nepal, Liming Zhu
Data61, CSIRO / NICTA / University of New South Wales
firstname.lastname@data61.csiro.au

Introduction

An ever-growing share of applications are provided as SaaS solutions on the basis of public cloud services, such as IaaS or PaaS offerings. Cloud services and the ability to manage and control them programmatically through APIs enabled the rise of continuous deployment and DevOps. DevOps, the abbreviated combination of “development” and “operations”, has been defined as “a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality” [1]. This methodology is rapidly progressing towards mainstream adoption [2], as it allows to deliver new functionality to end users fast and often increasing quality along the way.

However, achieving security (confidentiality, integrity and availability) and dependability (availability, reliability, safety, integrity and maintainability) when developing applications in the context of cloud services and DevOps offers a separate set of challenges. For instance, DevOps typically includes implementing a continuous deployment pipeline (CDP), which automatically tests and deploys new versions of the software. This CDP needs to be secured and checked for errors – else it could spread malicious or erroneous code to all servers, thus easily multiplying any problems by orders of magnitude. Trying to achieve full automation in testing and deployment also puts additional challenges on security and dependability, including requiring better practices around quick recovery, rollback, and resilience.

In this article, we broadly discuss the challenges by analysing the security and dependability challenges for all phases of the software development and data security life cycles of SaaS solutions. We also provide an overview of our research and development that aims to alleviate some of the pain points.

SaaS Security Life Cycle (SSLC)

SaaS follows the standard software development life cycle (SDLC), which includes requirement gathering and analysis, design, implementation or coding, testing, deployment and maintenance or operation. Technologies like DevOps have significantly reduced the time needed from implementation over testing to deployment, through the use of a CDP and other means. The SDLC uses a number of artefacts (e.g., design documents, test data set, etc.) including the input/output data used during the operation phase. Hence, it is important to consider the dependability and security of cloud applications in a joint SaaS security life cycle (SSLC), which includes the SDLC and the data security life cycle (DSLC). The standard DSLC includes the phases of create, store, use, share, archive and destroy. Figure 1 shows the SSLC that combines the SDLC and DSLC. It has four “Development” and “Operation” phases and
one ongoing data storage phase that lies across to all the four other phases. We next describe the security and dependability issues in each of those five phases.

**Figure 1: SaaS Security Life Cycle**

**Development**

The development phase in SSLC includes the following four SDLC steps: requirement gathering, design, implementation or coding, and testing. The first two steps define the security and dependability requirements of the application, whereas the last step includes most activities of the CDP. The implementation and testing steps can be carried out in the spectrum between two extreme ways: fully traditional or fully in-cloud. In the fully traditional approach, the development phase occurs entirely in developers’ local machines or the organization’s network, and the cloud only comes into play in the deployment phase. In contrast, the fully in-cloud application uses cloud resources for implementation (development environment, revision control, coordination) and testing (using cloud-based testing environments and a cloud-based CDP). Most development teams choose a middle ground of sorts, e.g., with local development, revision control in the cloud, local integration testing and cloud-based acceptance testing. Though in-cloud solutions provides a number of benefits, such as agility and distributed collaborative coding, it also adds a number of threats with regard to security and dependability. For each component hosted in the cloud, the same threats apply as for the application itself, so we postpone the discussion until later in the execution and maintenance phase. Furthermore, inability to port developed applications from one cloud provider to another has become a critical issue. The standard techniques like static application security test (SAST) and run time application self-protection (RASP) should be a key part of cloud application development.
Delivery & Deployment

This phase includes delivery and deployment of applications to the cloud. The dependability and security of cloud applications can be enhanced by using continuous delivery or continuous deployment. Continuous delivery ensures that all changes in an application are seriously tested and ready to be deployed. Continuous deployment goes a step further and automatically deploys those changes, as long as the quality gates are passed. We have developed a solution for dependable deployment of cloud applications, called the Process Oriented Dependability (POD) framework – see sidebar for a brief description. We also worked on analysing security aspects of the CDP, and what it would take to secure the CDP [3]. Continuous deployment also requires better security and dependability, such as specific monitoring solutions and fully automated rollback plans so that a botched deployment can be recovered quickly.

Execution & Operation

Cloud applications are inherently vulnerable due to the underlying business model, which builds on multitenancy, and the virtualisation technology that enables it. Multitenancy introduces a new set of dependability and security challenges. The threats to integrity, reliability and secrecy of both processes and data stem not only from the remote adversaries, but also local adversaries such as providers’ administrators or other tenants.

Virtual machines (VMs) and VM images need to be secured throughout their life-cycles in all three states: creation, running and dormant. For example, dormant VMs are often forgotten when performing security upgrade/patches, resulting in increased risk when they are brought back online. The virtual machine monitor (VMM), also known as Hypervisor, can be used to enhance the security of VMs and applications running on those VMs. For instance, the VMM logs can be analysed, not only to detect attacks but also to replay the execution of VMs to detect vulnerabilities of the deployed applications.

Although the new security architectures built on VMMs enhance the security and dependability of cloud applications, the VMM also offers a single point of failure. If adversaries can get control of the VMM, the attackers can attack all tenant applications. VMMs themselves need to be secured. For example, Amazon EC2 uses the concept of security group where a particular rule can be applied to a VM to enhance security, e.g., by restricting communication with the target VM to a predefined set of IP addresses. Another approach is to use fully encrypted applications; for instance, Data61’s EncryptedDB [9] technology enables execution of encrypted SQL queries over encrypted relational databases on the cloud.

Termination

Secure termination of cloud applications remains a challenging problem. The secure termination not only needs to deal with the secure shutdown of VMs in due time, but also the secure archival and deletion of data. When the VMs allocated for a particular application are shutdown, we need to ensure that VM images are secured and the termination is dealt with properly so that there is no leakage of data through CPU and memory pages.

NIST has recognised the issue of secure deletion of data as a key computer security challenge and developed guidelines for media sanitization. This issue is further complicated in cloud storage services as the owner does not have full control over the data as well as the storage
media. Techniques such as policy-based file assured deletion and Proof of Erasability (PoE) have been proposed to address this challenge.

We have addressed the secure data deletion using a Key Management Service (KMS) [7] – see TrustStore sidebar for details. The basic idea behind the scheme is to decouple the keys and encrypted data, where the keys are managed by a key management service independently. Secure deletion is achieved by deleting the keys, which makes the encrypted data irrecoverable.

Data Storage

Data security is important not only during the execution of cloud applications, but also during other phases of the SSLC. The data needs to be protected from both remote and local adversaries while in motion as well as at rest. Data security is often defined as the CIA properties, i.e., confidentiality, integrity and availability.

Techniques proposed to achieve confidentiality of data fall into three general categories: architecture, privacy, and security. Hybrid-cloud techniques are in the first category, in which the sensitive processes and data are always kept in a private cloud and non-sensitive processes and data are sent to a public cloud. K-anonymization and differential privacy are examples of two key privacy-preserving techniques. The third category includes cryptographic techniques used to protect confidentiality of the data, such as homomorphic encryption.

The integrity of data in the cloud is a well-researched property within data security. The techniques proposed in the literature can be grouped in two categories: public auditing and verification. In public auditing, techniques such as homomorphic tokens and distributed erasure-coded data have been used. Verification is generally studied under two categories: Provable Data Possession (PDP) and Proof of Retrievability (PoR). However, there are two key shortcomings with these approaches: efficiency and reliance on the storage service.

High availability of data in the cloud can be achieved by using multiple cloud service providers and replicating data amongst them. Examples of such systems include HAIL and DepSky, which uses the Cloud-of-Clouds principle. Our solution, TrustStore, addresses some of these challenges – see the sidebar for details.

In summary, the lifecycle of the SaaS from development to termination, including the data consumed and generated, needs to be secured so that if things go wrong, one can detect and audit violations, and find the person who is accountable.

In terms of dependability, the cloud offers the opportunity to backup whole applications with their data. One solution, which originated within Data61 as the start-up company Yuruware, is now available as the product Unitrends Boomerang. It allows copying all VM images, virtual disks, and configurations from one cloud provider to another, e.g., from VMware-based clouds to Amazon Web Services. In the case of a cloud datacentre outage, the backed-up remote copy can be spun up within minutes, and minimizes data loss and unavailability.
The POD framework targets the dependability of cloud application deployment specifically. An overview is given in Figure 2, which splits the view into an offline and an online phase. At the core, it uses cloud metrics and logs from operations tools, such as the open source tool Asgard from Netflix. Such tools manage deployment of an application to a group of servers in the cloud, e.g., through rolling upgrade where old servers are iteratively terminated and replaced with servers running the new version of the application.

During good runs of such operations processes, we collect logs and metrics. These are used in the offline training phase, where we generate a process model for the operation from the logs, through Process Mining—a set of techniques originally developed for business processes. Besides the purely behavioural view, we also build precise timing profiles of activities. Further, we annotate the model with information how metrics change while the process is happening, such as the number of currently running servers decreasing by 1 whenever a server termination event has been logged.

In the online phase, we use current logs and metrics in combination with the annotated process model created offline. There are two POD-Detection services for this purpose. First, the conformance checker tracks if the behaviour and timing of the current execution is in line with the model. Second, assertion evaluation tracks the effects of the current execution on the metrics, and uses hand-coded additional assertions to check against the cloud API if process steps really had the desired effect. If any errors are detected, POD-Diagnosis and POD-Recovery are triggered directly. POD-Diagnosis uses conditional probabilities and diagnostic tests to find the root cause. POD-Recovery takes actions to bring the process back to a normal state, e.g., by terminating a server that is “hanging” during boot-up and replacing it with a new server.

A number of papers have been published on POD, including exhaustive experiments to test its effectiveness and efficiency. As a starting point, please refer to [4,5].
One of the systems that aims to provide a secure storage system on the cloud is Data61’s TrustStore [6,7]. The TrustStore architecture is shown in Figure 3. It provides a thin client middleware that ensures all data transferred from a client application to cloud storage services are always encrypted. Since it supports multiple cloud storage service providers, it enables high availability through RAIN (Redundant Array of Independent Nodes) where each cloud service provider acts as a logical node. Data integrity is managed through an independent service, called Integrity Management Service (IMS). The Key Management Service (KMS) not only solves the problem of managing keys, but also enables establishing root of data security to the user, sharing data to support collaboration, and secure deletion of data at termination by destroying the keys permanently. Once the keys in the key management service are destroyed, the data stored in the cloud storage service is irrecoverable. The access policies are attached with the keys to support policy based store, access and deletion of data. In addition, the IMS protocol can be used to implement the Proof of Erasability (PoE) by using hash values stored by it. A number of proof-of-concept applications were built using TrustStore, including TruXy (trusted Galaxy for scientific workflows) [6] and CloudDocs (secure sharing of documents over the cloud) [8].
Concluding remarks

In this brief article, we discussed the challenges and opportunities of developing secure and dependable applications in the cloud. Our discussion follows the SaaS Security Lifecycle phases, and how the cloud context impacts those phases. We highlight our own research and development efforts, where relevant. The sidebars present two such efforts in more detail: the Process-Oriented Dependability (POD) framework and TrustStore. Ultimately we believe the benefits outweigh the risks for most applications, and as the maturity increases further, more and more applications will be offered primarily as cloud services.

References