Aalto University
School of Science
Master's Programme in Security and Cloud Computing

Christian Yudhistira

# Online Platform for Interactive Tutorials: Authentication and Authorization

Master's Thesis
Espoo, August 11, 2020

Supervisor: Prof. Mario Di Francesco, Aalto University
Prof. Pietro Michiardi, EURECOM
Advisor: Prof. Mario Di Francesco, Aalto University

Aalto University
School of Science
Master's Programme in Security and Cloud Computing

ABSTRACT OF
MASTER'S THESIS

**Author:** Christian Yudhistira

**Title:**
Online Platform for Interactive Tutorials: Authentication and Authorization

**Date:** August 11, 2020 **Pages:** 46

**Major:** Security and Cloud Computing **Code:** SCI3084

**Supervisor:** Prof. Mario Di Francesco
Prof. Pietro Michiardi

**Advisor:** Prof. Mario Di Francesco, Aalto University

The development of human life relates mostly to the learning process that someone prepared. Each person takes different learning approaches to acquire new knowledge based on their needs. In a traditional class, teachers are typically responsible for providing knowledge to a group of students. This approach sometimes limits the learning process of students by relying on teachers to gain new knowledge.

A different learning approach has emerged for students by employing a virtual laboratory that is accessible from a web browser. Such an approach is beneficial for computer science students in running various experiments. They do not need to set up each experiment on their local machine as teachers implement each lab running in a data center. Furthermore, the recent development of cloud computing allows new possibilities to implement virtual laboratories in a public cloud infrastructure.

In this thesis, we developed an online learning platform that enables teachers and students to interact with virtual labs in a public cloud infrastructure. We implemented a web server using the Django web framework to handle users' interactions with the learning platform. The web server consists of two services that provide authentication and authorization to the online learning platform, including a user service and a course service.

In the current implementation, the user service supports the learning platform to authenticate users using two schemes, either through Django built-in authentication or OpenID Connect. On the other hand, the course service provides the learning platform with system authentication and authorization, primarily in accommodating teacher privileges. Users with teacher privileges can register multiple courses and update the contents of each course that is registered under their identity.

**Keywords:** Virtual Laboratory, Authentication, Authorization

**Language:** English

# Acknowledgements

First, I would say thanks to God for blessing my life and allowing me to finish my thesis at the right time.

Then, to my family members in Indonesia, who gave me support both morally and mentally from the beginning until the end of this thesis period.

Then, to my supervisor, Professor Mario Di Francesco, who allowed me to join as one of the members in developing OnPIT project. Thank you for helping me in completing my thesis through all feedbacks and suggestions.

And finally, to the committee of SECCLO master's program, particularly both from Aalto University and EURECOM, who gave me such an opportunity to complete my master's degree. To all friends from SECCLO 2018, thanks for providing such a precious moment in the last two years, I wish you all the best for the next journey.

Espoo, August 11, 2020

Christian Yudhistira

# Abbreviations and Acronyms

| | |
|---|---|
| API | Application Programming Interface |
| DAC | Discretinary Access Control |
| HTTP | Hypertext Transfer Protocol |
| IaaS | Infrastructure as a Service |
| IP | Internet Protocol |
| MAC | Mandatory Access Control |
| PaaS | Platform as a Service |
| RBAC | Role-based Access Control |
| REST | Rpresentational State Transfer |
| SaaS | Software as a Service |
| SSH | Secure Shel |
| TCP | Transmission Control Protocol |
| URL | Uniform Resource Locator |
| VM | Virtual Machine |
| VMM | Virtual Machine Monitor |

# Contents

# List of Figures

# Chapter 1

# Introduction

Learning is an essential part of human life development. In a traditional class, sharing of knowledge typically occurred in one way, primarily from teachers to a group of students. Students tend to feel that they learn more by only taking a passive role in this learning process [13]. A study from Deslauriers et al. [13] shows contrary evidence that students learned more when taking an active role in a so-called *active learning* classroom. A class with active learning encourages students to produce appropriate solutions for real-like problems [6]. Such a learning method is beneficial for students in science, as it positively supports their conceptual development through participation in different experimental activities [6]. In computer science, for instance, students in active learning courses gained better technical skills than those that were not [37].

During the implementation of active learning in a course, teachers usually need to provide students with guidance to help them run a personal experiment. For some courses that do not require frequent updates for the experimental content, the active learning activity demands less pre-class preparation time for teachers than the traditional class setup [37]. Unfortunately, this is not always the case for courses in computer science. In fact, the field of computer science has a variety of study topics with different rates of technology updates. As a result, it requires more time for teachers to prepare guidance for each study topic. Moreover, students need to set up different supporting tools in their machines for each experiment. In most cases, it is difficult for higher education institutions to provide experiments that are up-to-date if not supported by an adequate environment [21, 47].

A new approach has emerged to provide students in computer science with such a new learning environment. This approach employs a *virtual laboratory* that is accessible from a web browser. Teachers provide a virtual laboratory that run in a data center instead of on the local machine of stu-

dents. A web-based learning platform allows students to interact with each lab with more flexibility, even by using mobile devices [26]. Several studies define such a learning environment as *interactive labs* [39]. Moreover, the emergence of cloud computing brings new possibilities to realize interactive labs. Specifically, they allow teachers to provide different types of virtual laboratories with limited setup time. The deployment of interactive labs at such infrastructure is indicated as *cloud-based* interactive labs.

## 1.1 Research topic and goals

The deployment of interactive labs leverages the cloud infrastructure. Therefore both teachers and students require an interface to interact with virtual laboratories. For this purpose, some research suggests that online learning employs different underlying technologies in the cloud. For instance, research conducted by Kabiri et al. [27] proposes a cloud-based virtual laboratory based on IaaS (Infrastructure as a Service) service model at the private cloud. Despite the benefit of sharing multiple labs in a shared pool of computing resources, this solution lacks scalability when the number of student access exceeds the computing resource. Teachers need multiple high-end servers and network devices to handle a large number of students, which is not efficient in terms of maintenance cost. Consequently, other solutions need to take into account to provide a more feasible environment.

Another study proposes a different implementation of interactive labs through PaaS (Platform as a Service) and SaaS (Software as a Service) service models in a public cloud provider [14]. As opposed to the implementation in a private cloud, this solution relieves teachers from any maintenance of IT infrastructure when the number of requests from students demands more computing resources. This study shows the feasibility of implementing interactive labs in a public cloud service. However, less research has considered security aspects related to the implementation of interactive labs in a public cloud infrastructure. In this context, this thesis aims to describe security on the web application level of the online learning platform, particularly in implementing authentication and authorization to users and services located at a web server.

The goals of this thesis are the following:

1. Define a service that enables users authentication and access control in the learning platform.

2. Define a service that supports system authentication and authorization for different user roles.

3. Develop a dashboard for configuring interactive labs for users with a teacher role.

## 1.2 Structure

The thesis is organized as follows. Chapter 2 introduces the concept of cloud-based interactive labs and the relevant technology used to establish authentication and authorization for the developed online learning platform. Chapter 3 presents the design and the structure of the developed online learning platform. Chapter 4 provides security considerations of services on the web server. Finally, Chapter 5 concludes the thesis by providing directions feature for future work.

# Chapter 2

# Background

This chapter overviews the relevant technology employed to establish authentication and authorization to the cloud-based interactive labs, including role-based access control and public key management. Moreover, this chapter discusses webhooks and their use for relaying information between different web services.

## 2.1 Online learning

The development of Information Technology (IT) is very beneficial for many aspects of human life, including in education. The latest developments in Internet technologies allow everyone to collect heterogeneous learning materials using their mobile devices, such as smartphones and laptops. Furthermore, many educational institutions have started to leverage online learning systems in addition to the traditional classroom setup, primarily for delivering learning resources [8, 25]. Such learning environments enable teachers to share study materials to distance learners, either through native desktop applications or web applications. On the other hand, students have more options to learn new knowledge from different types of online learning resources. Such a new study possibility offers students more flexibility, in terms of both time and location, to review concepts and theories outside of class, not just limited to in-class learning at school.

Many studies use terms such as e-learning, distance learning, and online learning interchangeably. As Moore et al. [38] mentioned in their research, any form of those terms has a similar objective, which is improving access to educational resources for students via the use of technology. This thesis targets online learning as a learning environment accessible by students from the Internet.

Online learning has actually been evolving for more than a decade, accompanied by the advancement of computer technologies. This development has provided many benefits to both teachers and learners in modern education. Van Popta et al. [46] explored the learning benefits of peer feedback in online learning. The authors showed the improvement of critical insight and reflection of students from such an activity. In fact, online peer feedback helps students find strengths and weaknesses in their homework, more than in conventional learning situations. Another research from Rebholz et al. [41] presented an online analytic tool that allows teachers to perform formative assessment of students. This assessment gives an overview of the student's understanding and identifies their areas of misconception in a subject. For instance, the tool encourages teachers to restructure the course material for future teaching. Bruzual et al. [11] proposed a system for automating the assessment of Android exercises in a university-level course. Such an automated assessment provides insightful feedback for each exercise submission of students. As a result, the system enables students to perform independent learning of mobile application development. Many researchers from both fields of education and computer science believe in the potential of combining the two with online learning [30].

In an online learning environment, interactions between teachers and learners are not only limited to collecting assignments via online web applications and giving feedback to students after their homework submission. There is also the chance for students to interact through their browsers to perform virtual experiments. This environment requires substantially less effort to set up the laboratory compared to using a local machine [10, 39]. This type of learning environment is typically called *interactive labs*. Teachers establish a virtual laboratory running in a computing infrastructure, generally in public or private data centers. A web-based learning platform is publicly accessible for students to access labs with their browsers over the Internet. Web applications offer students more flexibility than native applications, as they are platform-independent [26]. In such a learning environment, teachers typically manage multiple laboratories that can be prepared as a course. Students can then follow any tutorial inside the laboratory after enrolling in a course.

Despite its benefits, this learning setup incurs in infrastructure costs. The growing number of users requires more resources to handle several services. Most educational institutions cannot afford more budget to set up and provide maintenance for the infrastructure [28]. To this end, the implementation of interactive labs demands different infrastructure solutions.

### 2.1.1 Cloud-based interactive labs

Another breakthrough in computer technology happened in the last decade since the emergence of cloud computing. The term cloud refers to a group of IT resources that allows users to share computing resources, either hardware or software, through an Internet connection [7]. Cloud providers in the public cloud infrastructure typically offer different delivery models such as SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service). Each is referring to different layers of resources that they provide to the customers.

Customers can pick one or a combination of several models to build an application. It means that every customer can build an application without the need to handle the IT infrastructure included in each chosen service model. The public cloud also allows users to scale resources on-demand and pay only for used services [20]. Furthermore, customers do not need to pay extra cost for hardware while their services demand more computing resources. This feature is enabled by virtualization technology that allows isolation between multiple services running on top of a single machine. With those benefits, the public cloud becomes a promising solution to deploy interactive labs. In the following, the deployment of interactive labs at the public cloud is indicated as *cloud-based interactive labs*.

Running interactive labs on a public cloud infrastructure has some advantages. One of the most important is to relieve course administrators of any IT infrastructure maintenance, as this is taken care of by the cloud providers [50]. As a result, teachers can concentrate on the design and management of course labs. Another benefit is the increased reusability by creating an instance of a virtual laboratory. This can be achieved by replicating each instance through virtualization technology for many students, without having to prepare it from scratch every time [27].

There are typically two types of virtualization technologies employed to provide virtual environments in the cloud infrastructure, including hypervisor-based virtualization and container-based virtualization [18]. A hypervisor or Virtual Machine Monitor (VMM) is a software running on a hardware layer of a host machine with a privilege to control underlying computing resources. Such a privilege allows the hypervisor to create multiple virtual machines (VMs) running on a single host machine by virtually sharing its resources. As a result, it provides near-complete isolation between VMs as each VM runs in an independent guest operating system (OS) even though it becomes an inefficient solution if several VMs are running with the same guest OS. On the other hand, a container provides virtualization at the operating system level. Such virtualization does not deploy multiple virtual environments

in separate OS. Thus, this type of virtualization is more lightweight than the previous virtualization technique. Each tutorial in the interactive labs usually requires different experimental requirements to accommodate a virtual laboratory [47]. A learning platform provides users with a service to deploy a virtual laboratory running in an isolated environment based on the experimental requirement.

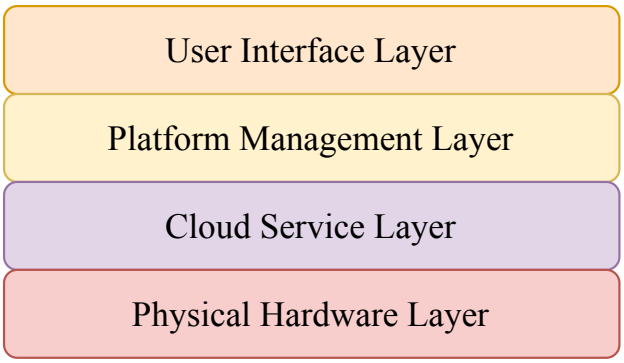



Figure 2.1: The structure of cloud-based interactive labs

The structure of cloud-based interactive labs in the literature can be represented as consisting of four layers [14, 27], as shown in Figure 2.1 and explained below.

- **The User Interface Layer** provides multiple interfaces for users. This layer enables learners to access web pages of a tutorial and interact with the virtual laboratory. On the other hand, this layer also provides interfaces for course administrators to access the dashboard page of a course.

- **The Platform Management Layer** includes several services that accommodate interaction between users and virtual laboratories. One of the services can trigger the provisioning of Virtual Machines or containers to accommodate the laboratory environment for each student. Another service supports course management, such as creating new courses, uploading course content, and organizing user accounts.

- **The Cloud Service Layer** accommodates every provisioning request from the upper layer by managing virtualization on a pool of computing resources in a datacenter. Furthermore, it routes data traffics for each pair of a user interface and an isolated environment of the virtual laboratory.

- **The Physical Hardware Layer** is the foundation infrastructure that provides all the computing resources to the upper layers. Such an infrastructure typically comprises of servers, network devices, and storage devices. In a public cloud environment, the physical hardware is not a user concern.

## 2.2 Access control

Access control is a security mechanism that regulates access of authenticated users to resources in a system. Such a mechanism grants or rejects a request to access resources, depending on the user's rights [44]. In the context of computer security, access control constrains the operations of both users and programs executing on behalf of the users. A suitable implementation of access control prevents any activity that could breach a security of a machine, such as a privilege escalation in a web application [52].

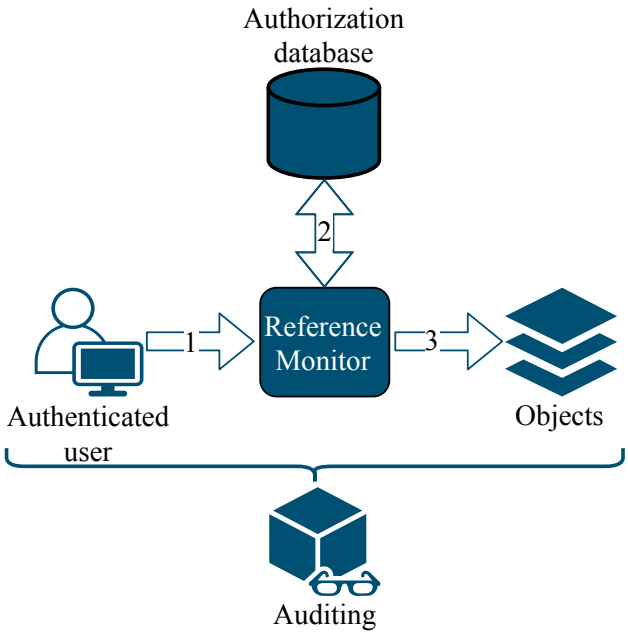



Figure 2.2: The structure of access control

To establish access control, different security components need to interact with each other, as shown in Figure 2.2. The *reference monitor* assesses every attempted access from authenticated users to objects in a system. Before allowing a user to access objects, the reference monitor typically checks the

user's access rights against an *authorization database*. The rules governed by the security administrator is stored in the database. In addition, there is an *auditing component* that monitors users actions to determine possible flaws in the system as a result of unauthenticated user activity.

Harrison et al. introduce a protection model for computing systems, and it is known as the access control matrix model [24]. This model abstracts an access policy, but it is not practical to implement. In fact, the access policy would need to be stored in a single table that contains the access rights (i.e., open, read, write, close) of all users for each file in the system. Moreover, this model has a security problem that might lead to the permission escalation of users: every user who obtains write permission could modify the matrix and grant write permission to other users without any approval from system administrators. Thus, other solutions are required to implement access control policies in computing systems.

One of the initial improvements over the previous model is Discretionary Access Control (DAC). DAC is an access control model that combines users' identity with corresponding authorization rules to determine the access right of users to the target information [44]. If the authorization rule asserts that users can access the object, then access is granted; otherwise, users do not have access permission for that object. DAC offers better solutions for granting access compared with the previous model. In fact, the owner of the object is the only entity that can determine the access rights of other users. Despite having that security property, DAC has the drawback of not imposing any restriction on the information flow in a system [36]. It allows unauthorized users to read information obtained from legitimate users without the awareness of the data owner.

Another solution from early access control models is Mandatory Access Control (MAC). MAC is an access control model that determines users' access rights based on the classification of subjects and objects in the system [44]. A system administrator usually assigns a security level tag to each user and object in the system. The assignment of security level tags forms a hierarchical security level (i.e., top-secret, secret, confidential, unclassified). Following the same rule as in the military setting, subjects obtain permission to access objects if they satisfy a particular security level associated with the two. This security property solves the issue in the DAC model by regulating the flow of information in a system using the security level hierarchy.

According to Zhu et al. [52], some web applications have a conditional link displayed if a specific access control examination is satisfied. In a multi-user web application, such as the interactive lab platform, having this security property is beneficial to separate web pages for teachers and students. Nevertheless, both MAC and DAC have limitations which make them impractical

for a multi-user web application: MAC is too rigid as it is designed for military settings, while DAC is not a practical solution as it takes control of each asset.

### 2.2.1 Role-based access control

In the 1970s, the emergence of multi-user applications in computing systems leads to the new model called role-based access control (RBAC) [16, 43]. This model introduces a new notion called role as a set of permissions to access objects in a system. Instead of regulating permissions for each user, users will be assigned to roles based on their responsibilities in a system. In the case of online learning applications, there are at least two distinct roles: teachers and students for each authenticated user. Each role possesses different capabilities to access features in the learning platform. Compared with the previous two access control models, the RBAC model simplifies the management of permissions in multi-user applications.

The RBAC model offers a better alternative to manage access to resources compared to the DAC and MAC models. In the DAC model, data owners apply access rights for each user associated with their resources. Therefore, the authorization rules are directly established between users and resources. Consequently, the DAC model is not practical, especially when a user's responsibilities change. Instead of revoking all user access rights and granting a set of new rules, the RBAC model only needs to revoke roles corresponding to the user and grant other roles. The latter solution requires less time than the DAC model for updating new permissions.

On the other hand, the MAC model is too rigid in the sense that there is only one-directional information flow in the implementation. There are two principles (read-up, write-down or read-down, write-up) that satisfy different security objectives (information integrity or information secrecy). In contrast, the RBAC model allows users to possess multiple roles that do not limit access to only one-directional flow. Moreover, new permissions can be added to roles as new resources are included. Thus, these properties give more flexibility to the access control in multi-user applications.

## 2.3 Public key management

Cryptography has been used in many parts of modern businesses, including maintaining the privacy of patients in hospitals [19], protecting electronic communication in the banking systems [51], and providing integrity and authenticity of customer data in financial transactions [22]. In such cases, a

cryptographic key is as valuable as the protected assets. Therefore, it is essential to manage cryptographic keys across their life cycle [9]. An adequate implementation of public key management allows preserving the resistance of computing systems from attacks [35].

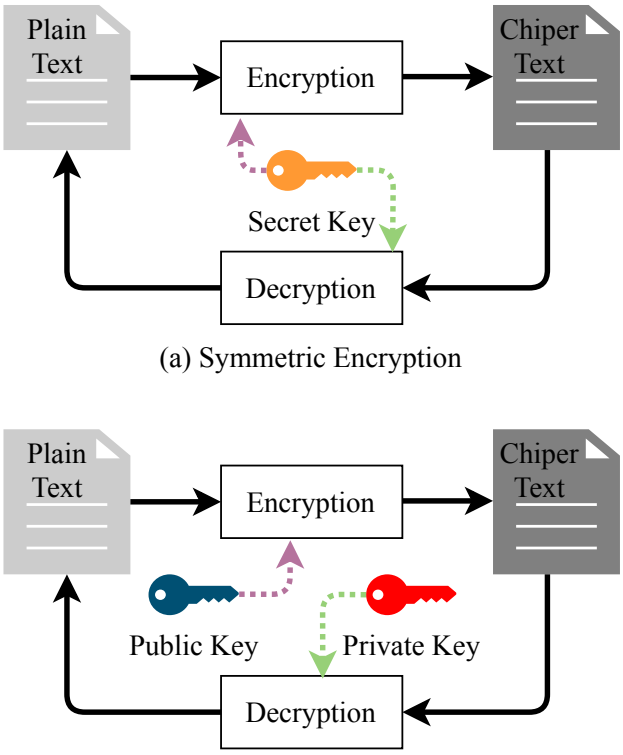



Figure 2.3: Symmetric and asymmetric encryption

Cryptographic keys can be categorized into *symmetric* and *asymmetric*. Each of these is extensively used in many services over the Internet, for instance, to improve security in cloud computing [48]. In symmetric encryption, there is only one key for encrypting and decrypting data between two systems. Typically, such a key is identified as a *secret key*. In contrast, asymmetric encryption uses a pair of keys: a public key generally used for encryption; and a private key for decrypting data. Figure 2.3 displays an illustration of symmetric and asymmetric encryptions. The generation of a key pair in the asymmetric encryption leverages cryptographic algorithms (e.g., RSA, DSA, and ECDSA) that rely on mathematical problems (e.g., factoring, discrete logarithm, and elliptic curve). Such a mechanism allows users who possess

the private key to prove the relationship with a public key but not vice versa. This security property of asymmetric encryption usually used to prove the authenticity of the user's identity without actually showing classical credentials (i.e., combining username and password). Algorithms for asymmetric encryption require more processing power for generating keys than those for symmetric encryption [12]. However, the public part of the asymmetric key allows the key exchange to occur in an insecure channel. As a result, there are many security protocols rely on the security property of asymmetric key.

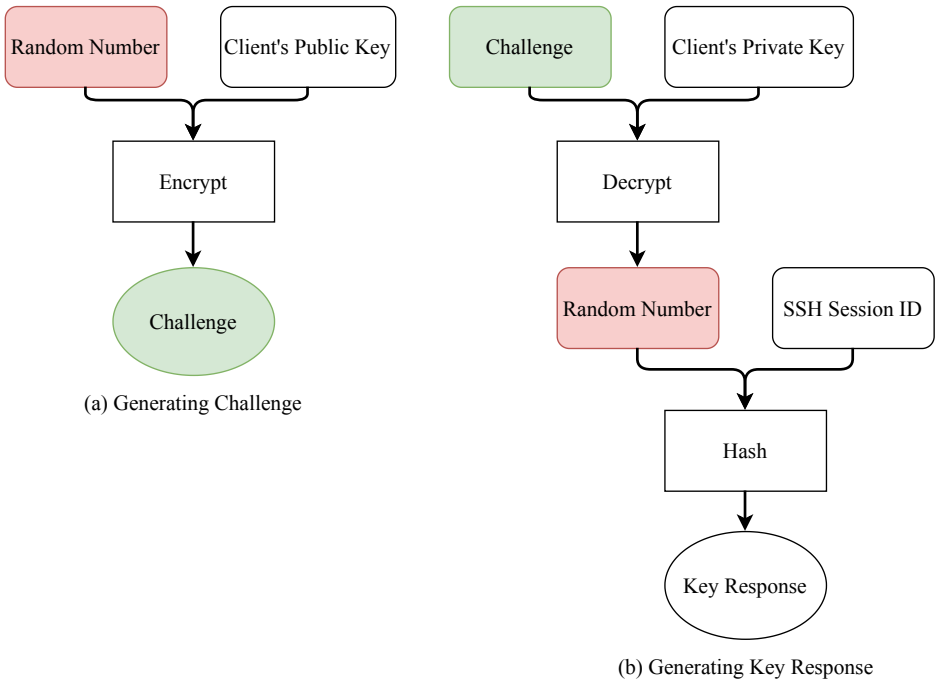



Figure 2.4: Generating (a) challenge and (b) key response in public-key authentication

Secure Shell (SSH) is a protocol that provides users secure access to remote systems over an insecure network. Public-key authentication is one of the authentication methods used in the SSH authentication protocol to authenticate a user who initiates remote access to a server using SSH [49]. The client generates a key pair based on a specific cryptographic algorithm using an SSH command-line tool. Then, the client copies the generated public key to many servers and usually stores it in the `authorized_keys` file of each server to consider the key trustworthy. In contrast, the private key remains in the client site as proof of the client's identity.

The public-key authentication process of SSH starts when a server receives

a request for access from a client. The client prepares a username and set up a key session inside the request. The server utilizes the username to locate the client's public key to create a challenge. The creation of the challenge includes a random number generation and encryption of the random number using the client's public key, as shown in Figure 2.4(a). In this case, only an authentic client who holds the corresponding private key can decrypt the server's challenge and obtain the random number. Then, the client computes a key response as a proof of holding private key by performing a hash function to the random number and SSH session ID, as shown in Figure 2.4(b). After the client sends the key response to the server, then the server computes the random number and SSH session ID with the same hash function and compares the hashed result with the key response received from the client. If both values are the same, then the request for access from the client is granted.

Additional security should take place to store the private key in the public-key authentication process of SSH. After the key generation process using a particular cryptographic algorithm, the private key is usually stored in `.ssh` directory of a user's home directory. The filename of each private key depends on the type of the cryptographic algorithm to generate the key pair (e.g., `.ssh/ssh_id_rsa`, `.ssh/ssh_id_ed25519`). To add an extra layer of security, a user uses a passphrase to decrypt the private key when logging in to the remote system through SSH. The implementation of public-key authentication in automated processes (e.g., automated deployment from Git) requires a system to handle multiple key pairs, and no available user is typing the passphrase. Consequently, the passphrase would have to be stored in another place or hard-coded the passphrase in a script [3], and it is not a practical solution. One feasible approach is to specify each private key directly in each authentication of automated processes though it reduces the security of the private key.

## 2.4 Webhooks

In the early stage of web development, web applications consist of static elements that fit into a single page. All interactions between clients and servers were limited: clients initiated all the requests, and servers sent a response with static files. The next stage begins with the implementation of Ajax in web applications that allows each component of the web application to interact with the server without reloading an entire web page. The recent development of the server handles not only static pages but also different types of web services (e.g., check balance, add pictures to social media).

Web applications of the client side obtain access for resources of each web service located on the server, usually through REST API.

A webhooks is a method designed to relay information between two web applications to enable a real-time interaction [33]. In the traditional approach based on REST APIs, an application needs to poll information from another web application frequently [29]. In contrast, webhooks offer a different method by sending information immediately, as an event occurs, in the form of a push-like notification. This solution is best suited for applications that rely on asynchronous events, such as pushing an update code to a repository or adding comments to a blog post [34]. Figure 2.5 displays the different implementation of pooling and webhooks in updating web contents.

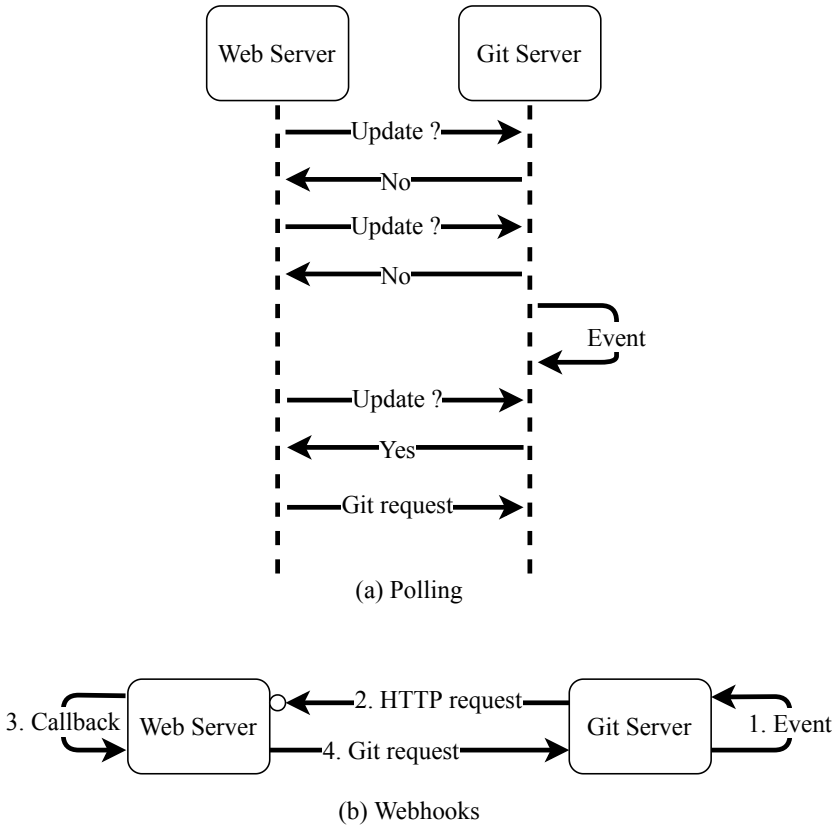



Figure 2.5: Comparison of polling and webhooks

Sending a notification for every occurrence of an event instead of polling makes webhook a less resource-intensive option in fetching information. This property also allows developers to build a web application with an event-driven architecture [40]. One of the main benefits is that two systems can

be decoupled while still being able to communicate, resulting in improved scalability and fault-tolerance. Moreover, this approach allows an event on one site to invoke actions on another site. The actions could be anything, for instance, starting the deployment process in continuous deployment or triggering an update of course contents as used in this thesis.

Typically, there are two elements involved in the webhook functionality: the receiver system and the sender system. The receiver system is responsible for providing a public endpoint that is used by the sender system as the destination for sending a notification. Moreover, the receiver system developer also provides a callback function associated with the public endpoint for responding to incoming requests from the sender system. On the other hand, the sender system prepares a service that will send a notification once a relevant event occurs on this site. Usually, the sender system makes an HTTP POST request to the URL endpoint configured for webhooks. It shows from the Figure 2.5, the web server operated as the receiver system and the Git server served as the sender system. Since the webhooks uses HTTP, other web services do not need different infrastructure to integrate a webhook [34].

From the fact that the implementation of webhooks requires the receiver system to expose a public endpoint, it allows many foreign entities to access a service behind this public link. This condition has opened vulnerabilities because any foreign services which have the URL of the webhook can send malicious information by impersonating a valid event. To protect the service from such a threat, webhooks has security mechanisms for the receiver system to validate the authenticity of the sender system. Some examples of the security mechanism employed by the webhook validation include [32]:

- Storing a whitelist of IP Address for known machines. This approach allows the receiver to reject each incoming HTTP request with IP address that is not registered in the authenticated IP list.

- Using HMAC [31] signature to validate each incoming HTTP Post request to the receiver. The receiver system generates a signature following the same steps as the sender system. Then, the receiver compares the generated signature with the received HMAC before handling the incoming HTTP request from legitimate machines to the callback function.

- Employing mutual TLS [5] authentication to authenticate each sender system. The receiver system verifies the sender machine's certificate when establishing an HTTP connection in the first place.

# Chapter 3

# Online platform for interactive tutorials

The goal of the thesis is to build a web-based interactive labs in the public cloud infrastructure. For this purpose, an online learning platform was designed to enable students to access the web page of a course, interact with the virtual laboratory, and provide an interface for course administrators to access the dashboard page.

This chapter outlines the system requirement and discusses the system architecture of the online learning platform.

## 3.1 Overview

The learning process of students usually includes two major approaches: theoretical lectures and practical work. The majority of educational fields deliver study materials in theoretical lectures that typically occur in face-to-face classes. Unfortunately, students from the field of science and technology do not obtain the full potential of learning experiences with such a learning system. In fact, an approach in practical works represents a crucial aspect of learning for students in science and technology, including engineering [15]. Such a study situation enables students to take an active role by working on hands-on exercises. Consequently, this type of practice helps students to improve their problem-solving skills [27]. Such a practice is beneficial for students in computer science too, as many new technologies are arising each year, and it would be challenging for students to keep their knowledge update if not supported by appealing labs.

The field of computer science has a wide area of study topics with different rates of technology updates. It is difficult for some educational institutions

to provide up-to-date experiments if not sustained with an adequate study environment [21, 47]. Interactive labs are best suited for accommodating such experiments for students in computer science. The integration of interactive labs and the underlying public cloud technologies enables students to access different types of computer-related experiments in isolated environments. To this end, an online learning platform is built to accommodate such an integration.

In the following, we discuss the primary components and features that follow to establish the online learning platform. The security considerations of the web application are detailed in Chapter 4.

A **web server** is the first component that provides an interface for both teachers and students to access the online learning platform. The user interface is built as a web application available to both users through their browsers as it provides flexible access in different types of mobile devices. The learning platform needs an authentication mechanism and access control to manage access rights for each user. Such a mechanism enables each authenticated user to have different sets of privileges when accessing the learning platform.

In our implementation, two user roles are supported: teacher and student. Users with a teacher privilege require to have a teacher dashboard for maintaining their courses. Using the dashboard, teachers can add a new course, update contents of a course, and manage student enrollment. On the other hand, users with a student privilege need a right to enroll in multiple available courses. Moreover, the interaction between students and each isolated virtual laboratory should be hosted by the learning platform. Thus, the learning platform provides students with different front-end types, primarily for accommodating such an interaction. This component also supports students with service to initiate a provisioning process of the virtual laboratory handled by the container orchestration component.

A **virtual laboratory** is the second component that provides an environment for students to do an experiment. Each virtual laboratory hosts an experiment that typically has particular experimental requirements. Therefore, we provide a specific set of experimental tools and supporting software for students based on the experiment's requirement every time they start a virtual laboratory. Moreover, each student needs to verify their progress when experimenting in a lab. For this purpose, we provide students with an automatic grading feature, primarily for verifying student completion of labs.

A **container orchestrator** is the third component that manages cloud resources leveraged by users in virtual laboratories. This component is responsible for providing an isolated environment using virtualization technol-

ogy (i.e., container) when students start a virtual laboratory from the web server. In addition, this component also orchestrates each virtual laboratory in the public cloud infrastructure to maintain the security of the cluster.

## 3.2 System architecture

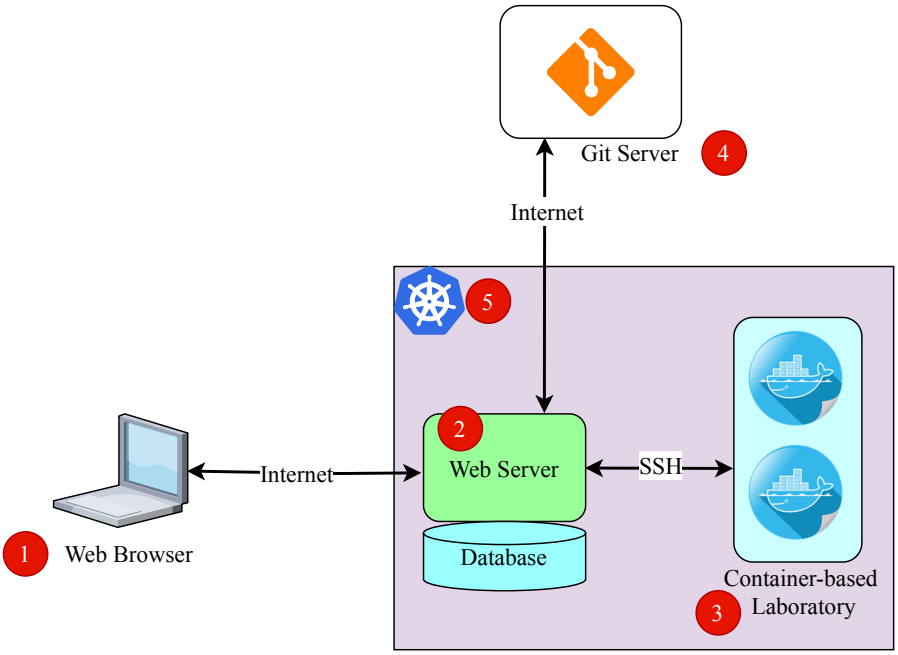



Figure 3.1: The architecture of OnPIT

The online learning platform developed in this thesis is referred to as *OnPIT* (Online Platform for Interactive Tutorials). Figure 3.1 shows the architecture of OnPIT that consists of 5 components:

1. **Web Browser.** This component displays an interface that users can use to interact with OnPIT.

2. **Web Server.** This is a place that handles each interaction of users and OnPIT. The web server is implemented with the Django framework. A PostgreSQL database integrated with the web server to store data related to users and courses. Both the web server and the database deployed as separate containers based on Docker.

3. **Virtual Laboratory.** This component provides an environment for students to perform each experiment of a laboratory. Each virtual laboratory run as separate containers based on Docker.

4. **Public Git Server.** This is a component that is used to store contents of a course.

5. **Kubernetes Cluster.** This component is responsible for provisioning each container in the cluster and orchestrating each container to maintain resources and the security of the cluster. We utilize Google Kubernetes Engine (GKE) that provides us with an environment for managing containerized applications using Google infrastructure.

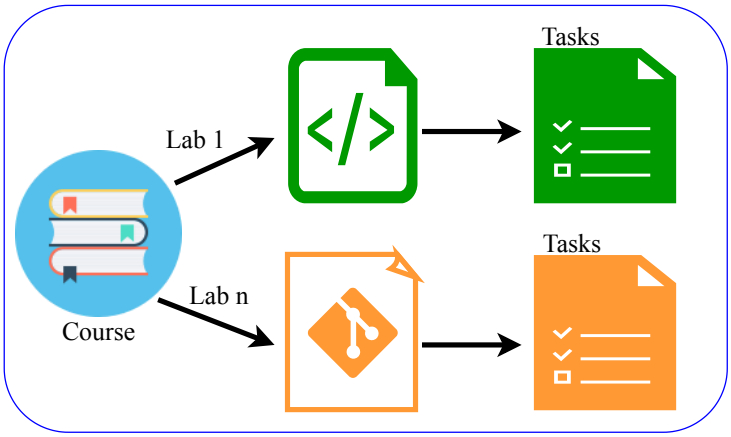



Figure 3.2: The structure of a course

We present the structure of a course in OnPIT, as shown in Figure 3.2. Each course consists of different type of labs which represent an individual virtual laboratory running in a container. Each lab includes a list of tasks that are provided to students when starting an experiment. The tools and supporting software of a lab are installed during the deployment step of the container.

Each role (i.e., teacher and student) has different privileges while accessing OnPIT. Generally, it can be represented in two different channels. Each channel consists of several services that reside in different components of OnPIT and establish a connection of different services to accommodate the privileges of each user.

Figure 3.3 shows a channel that is provided to accommodate the teacher role. Generally, the interaction between services in this channel leverages

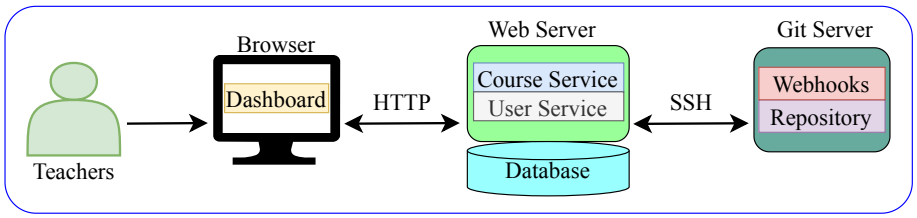



Figure 3.3: Channel for teachers

Hypertext Transfer Protocol (HTTP) and Secure Shell. Furthermore, there is only one type of front-end that is used by teachers to interact with the learning platform through the browser. As a teacher, three main privileges are available on this channel; the description of each privilege is provided below.

- **Adding a new course.** This privilege enables teachers to add a new course under their name. Teachers start the process by accessing the dashboard through their browser. In this step, teachers need to provide their credentials to the authentication system before getting permission to access other services. Once they pass that step, teachers can access a course service that is resided in the web server. In the implementation, the communication between the browser and the web server occurred over HTTP. The course service provides teachers with a course registration page that teachers can use to provide a Git repository Uniform Resource Locator (URL) of the new course. Finally, the web server clones the course contents from the remote repository designated by the repository URL using the SSH protocol, and the course service stores metadata of the course to the database.

- **Updating contents of a course.** This privilege allows teachers to renew contents of a course that belongs to them. The implementation of this privilege relates to the course registration step in the first privilege. In the registration step of a course, teachers need to configure the repository of the new course with the webhook details (e.g., a URL and a secret). Once the previous step has been performed, the course service receives a notification every time teachers push an update to the course repository at the Git server. The notification contains information related to an event in the remote repository that triggers the webhooks. The course service uses the event's information to request the web server copy the latest course contents from the remote repository using

the SSH protocol. Then, the course service updates metadata of the course in the database.

- **Managing student enrollment.** This privilege authorizes teachers to select a group of students who can access their course. As with the first privilege, teachers need to authenticate their identity before obtaining access to the user service located on the web server. The communication between the browser and the web server occurred in HTTP. The user service provides teachers with a web page to enroll students for each of their courses and then store each student's information to the database.

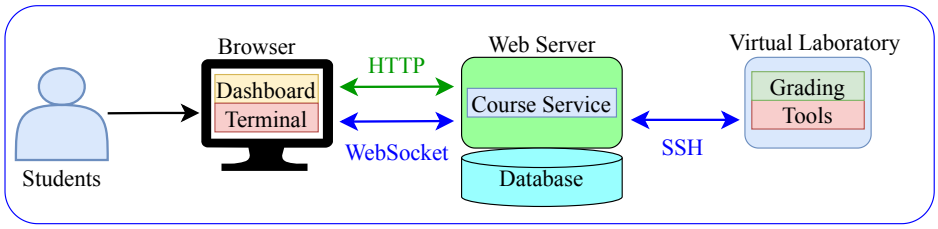



Figure 3.4: Channel for students

Figure 3.4 presents a channel that is implemented to support the student role. Typically, the interaction between services in this channel leverages three types of Transmission Control Protocol / Internet Protocol (TCP/IP) protocols, including HTTP, WebSocket, and SSH. In contrast with the previous channel, two types of front-ends are available for students to interact with the learning platform, including a course dashboard and a terminal. As a student, the front-end accommodates only one privilege; the description of the privilege is provided below.

- **Access to a virtual laboratory.** This privilege enables students to initiate a virtual laboratory and interact with the experiment in it. In the first place, students need to prove their identity by providing their credentials before gaining permission to access the course service in the web server. There are two types of front-ends which are accessible for students from their browser to the web server. The first one is the course dashboard that students can use to select all available labs from an enrolled course and read instructions from a given task. The communication between the browser and the web server for this type of front-end occurs over HTTP. The other one is the front-end terminal

that students can use to interact with the virtual laboratory. To establish such an experiment, the learning platform provides students with experimental tools, a grading system, and a Unix shell in the virtual laboratory. The Unix shell provides students with an interface to interact with the virtual laboratory using the terminal. To accommodate such an interaction, the SSH protocol is used between the web server and the virtual laboratory running in a container. Then, the bidirectional capability between the browser and the web server is achieved through the implementation of the WebSocket protocol.

# Chapter 4

# Authentication and authorization in OnPIT

This chapter discusses security considerations in the design and implementation of services located on the OnPIT web server. First, we present the user service design and the implementation of user authentication and access control in the learning platform. Then, we discuss the course service design and the implementation of system authentication and authorization to add new courses and update course contents.

## 4.1 User service

The user service allows users to authenticate their identity and get privileges to access resources in the online learning platform. We leverage two types of user authentication schemes: Django built-in authentication [4] and OpenID Connect [42]. Moreover, we also provide access control for each authenticated user. The following two sections discuss the design and the implementation of user authentication and access control in the online learning platform.

### 4.1.1 User authentication

The web application requires all users to prove their identity in advance to access the learning platform. As mentioned earlier, we provide two schemes to authenticate users in the online learning platform. Each user authentication scheme facilitates the learning platform to obtain attributes of logged-in users: name and email address. The learning platform provides those attributes to support teachers in the student enrollment process. The first option of user authentication uses a built-in authentication from Django.

Before getting access to the learning platform, each user should provide their personal information (e.g., full name, email address), username, and credentials in the web application's signup page – the web application stores those information to the local database. Once the sign up is complete, each user can use their username and credentials to validate their identity before accessing the learning platform. The second option utilizes OpenID Connect to perform user authentication on top of OAuth 2.0 [23]. We integrate the login page of the web application with the OpenID Connect service provided by the university. Such integration allows all users who have an active email account from the university to get direct validation into the learning platform. On the other hand, such integration enables the learning platform to obtain the attributes of each logged-in user directly from the university database.

### 4.1.2 User authorization

Once users prove their identity, each of them obtains different access rights to resources in the learning platform. In the current implementation, resources that are related to teacher privileges require limited access from authenticated users. Such privileges include adding new courses and managing the content of each course. To accommodate such privileges, we provide standard role-based access control (RBAC) to control each authenticated user's access rights. In standard RBAC, each user directly bounds to each one of the user roles.

In our implementation of RBAC, there are two roles available for each authenticated user: teacher and student. After login for the first time, all authenticated user obtains a student role before granted with a teacher role. Each user role represents different access rights to resources in the learning platform. The current implementation of access control using standard RBAC allows the system administrator of the learning platform to grant each user a role based on user status.

## 4.2 Course service

The course service provides teachers with an interface to register new courses and set up an environment to automate course updates. Two services are required to establish such features: a course service and a public Git service. Such a connection form a service-to-service communication, which takes no human intervention during the communication process. Nevertheless, such communication requires security considerations to maintain assurance in the communication of two services, as we can find in the communication of users

and web applications. Therefore, different approaches need to take place to provide authentication and authorization between two services. The following two sections discuss security aspects in the registration of courses and course updates.

### 4.2.1 Adding new courses

All users who are assigned a teacher role have a privilege to add multiple courses under their identity. Contents of each course are stored in a Git repository at a public Git service. The course service provides teachers with a dashboard where teachers can use to register a new course to the learning platform. The registration process takes place in the web server and requires teachers to provide a Git repository URL of the new course. Once teachers provide such information to the dashboard, the course service requests the web server to clone the repository of the new course designated by the Git repository URL to a local directory of the web server.

After teachers give the Git repository URL to the course service, the course service requests the web server to copy a repository located in the public Git service. In this phase, the web server operates on behalf of the user who initiated the registration process. There is no human intervention that checks the protection during the communication of both services. Therefore, we provide two security properties to maintain the assurance in the communication of the course service located in the web server and the public Git service, namely: an authentication and an authorization. The authentication guarantees that the public Git service communicates with an authentic web server. Such authorization allows the web server to access the repository in a public Git service on behalf of the repository owner.

In the implementation, teachers need the course contents in the public Git service not to be publicly accessible by everyone, particularly students. A private Git repository is best suited for this case as it provides a security mechanism that limits unauthorized users to get access to contents inside the repository. The repository owner should give access rights first to other users and systems before they can access the private repository of the owner. Public Git services, such as GitHub and GitLab, provide users with different forms of authentication, such as password, personal access tokens, and SSH keys [1]. The repository owner can utilize each type of authentication to give access rights to other users.

There are three security requirements in the design of adding new courses:

- First, the authentication method should allow the repository owner to grant limited access rights to avoid other users getting more access

rights more than needed.

- Second, the scope of access rights should be limited to a particular repository to avoid authenticated users getting access to unrelated content in another private repository.

- Third, the secret for authentication purposes should be kept secret to reduce the chance of being exposed by unauthorized users.

All users use password-based authentication when accessing their Git account from browsers. Such authentication requires users to provide their username and password for proving their identity before getting access to their Git account. This type of authentication usually gives users account-level access, which means each authenticated user obtains full access rights to all repositories in a single Git account. The same authentication method can be implemented to give access rights of a Git account to a web server. The web server sends an access request with the credential over the Internet to interact with the public Git service using a Git Application Programming Interface (API).

A personal access token is an alternative method for validating a user's identity when accessing the Git account. Instead of giving the password, an access token and an associated username are presented to the Git service over the Internet. The access token provides better security property in terms of giving access rights to other users than the password. In fact, it allows the repository owner to create multiple access tokens and manage different scopes of access for each token, such as read or write access. It also means that the repository owner can assign different access rights to each authenticated user. On the other hand, each authenticated user holds specific access rights that apply to all repositories in the Git account. Moreover, the repository owner can revoke access rights for a specific user at any time without affecting the permissions of other users.

SSH key authentication does not require an access token and a password to authenticate users when accessing their Git account. Such authentication leverages a pair of cryptographic keys (i.e., a public key and a private key) to replace security credentials as in password-based or token-based authentication. The pair of keys is generated explicitly for an individual user employing mathematical algorithms. As a result, the private key can relate a public key to a particular user, but it does not work the other way. This security property allows users to prove their identity without supplying their security credentials to the public Git service. Such an authentication method can be implemented as public-key encryption [17] is performed for authenticating users using cryptographic keys. Moreover, the repository owner can

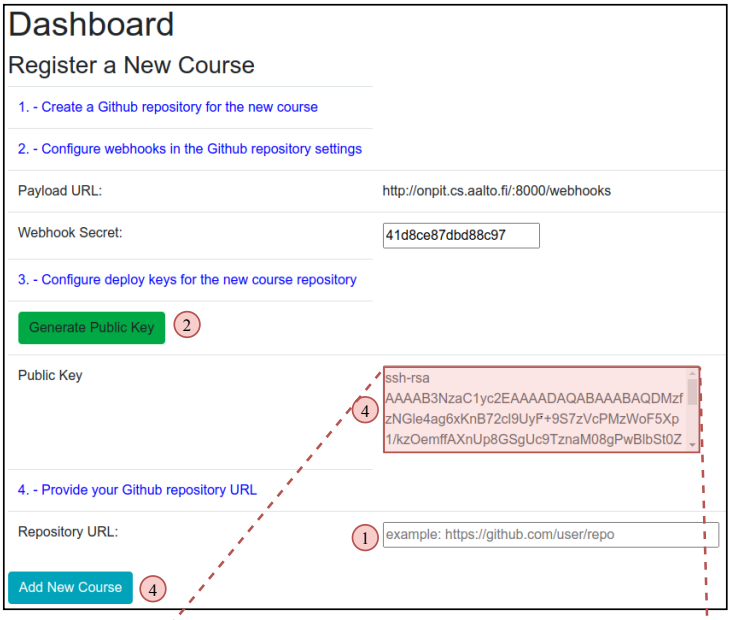
give access rights to a web server without giving the security credential. SSH key authentication is considerably more secure in authenticating a web server when accessing a public Git service. The web server does not need to reveal the private key over the network every time it interacts with the Git service.

Each authentication method has different security risks that are not directly suitable for the implementation of adding new courses. Password-based authentication does not require users to generate cryptographic keys, so it makes the configuration process is easier than using SSH keys. Token-based authentication has a similar configuration process as the password-based option. However, it has more security benefits than the password-based method as it allows the repository owner to provide different access rights for each authenticated user. This security property of the access token applies the least privilege concept [45], namely, limiting the access rights of authenticated users to the minimum privileges they need to perform any actions. Consequently, in the implementation of adding new courses, the web server obtains minimum access rights to merely copy course contents from a Git repository without getting access to modify course content in the repository.

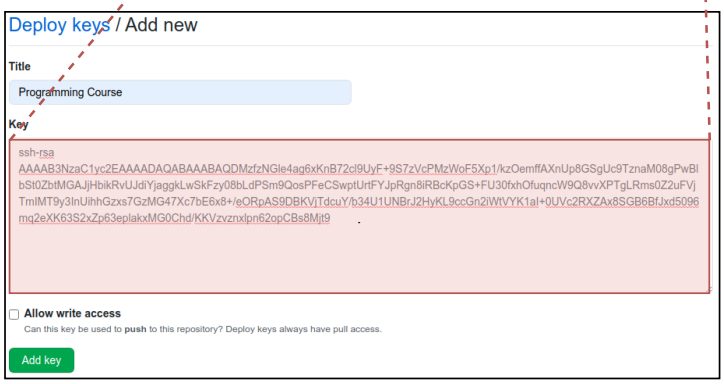
Nevertheless, token-based authentication grants access rights for all repositories in a Git account, including private repositories and public repositories. This property results in a vulnerability: if a malicious user manages to obtain the access token, then it obtains access to all repositories in a Git account. On the other hand, the implementation of adding new courses requires the web server to get access merely to private repositories that store course contents. Furthermore, additional security should be provided in the communication channel to secure the token, because the web server sends the token in every request to the public Git service. Even though token-based authentication allows the course owner to limit access rights granted to other users, but it is not a feasible solution to authenticate the web server.

In a public Git service such as GitHub, there is a feature to grant access to a server for deploying projects from only a single repository using an SSH key, which is called deploy keys [2]. By default, this feature limits the server to get read-only access to the repository. Moreover, the configuration process of this feature requires the repository owner to attach only the public key to the repository, which means retaining the private key still in the server. The implementation of deploy keys for each repository limits access of authenticated users to a particular repository. In addition, granting read-only access provides additional security for the course repository by limiting the access rights for each authenticated user to a minimum privilege. Furthermore, the implementation of public-key encryption in SSH authentication does not require to transfer the private key. In conclusion, the SSH key's implementation using deploy keys has adequate security properties to accommodate the

implementation of adding new courses.




(a) Dashboard




(b) Deploy key setting page

Figure 4.1: Steps for adding a new course

We developed a dashboard that allows teachers to register new courses based on the security considerations mentioned before. Figure 4.1 shows the process of adding new courses. First, teachers provide the Git repository URL of the new course to the web server through the dashboard. Second, teachers can request the web server to generate a pair of keys using the dashboard. Then, teachers collect the value of the public key displayed on the dashboard, while the private key is still stored in a local directory of the

web server. Finally, once teachers store the value of the public key to the deploy keys setting of the course repository, teachers can request the web server to send a clone request to the public Git service via the dashboard.

The authentication process of both servers can be seen in Figure 4.2. The web server sends a clone request which contains the target repository URL and the authentication method for this request. The public Git service receives the target repository URL and creates a challenge based on the requested authentication method. In the case of SSH key authentication, the Git service constructs a challenge based on the public key configured earlier in the deploy keys setting of the target repository. Then, the Git service sends the challenge to the web server. The web server receives the challenge and constructs a key response based on the private key corresponding to the public key that constructed the challenge. Then, the web server sends the key response to the public Git service. Once the Git service receives the key response, it validates the received key response before sending a Git response back to the web server for the initial Git clone request. The creation of the challenge and the key response in SSH authentication was detailed in Section 2.4

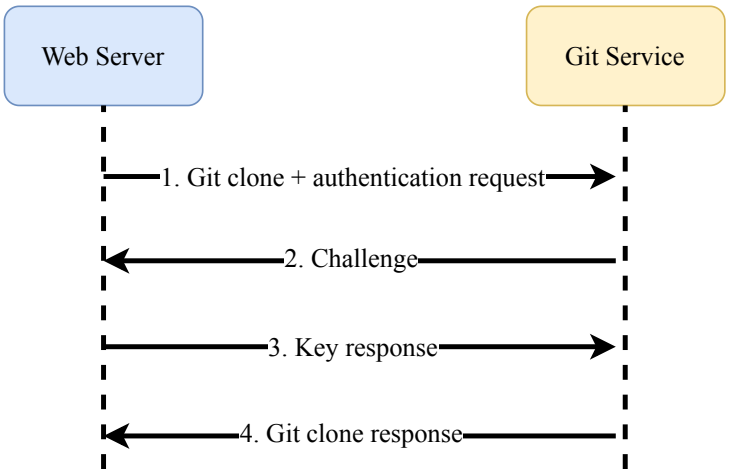



Figure 4.2: Authentication process in adding new courses

### 4.2.2 Updating course contents

In addition to registering multiple courses, each user who is assigned a teacher role has a privilege to update the contents of each course that is associated to their identity. To limit the number of access to the web server, teachers

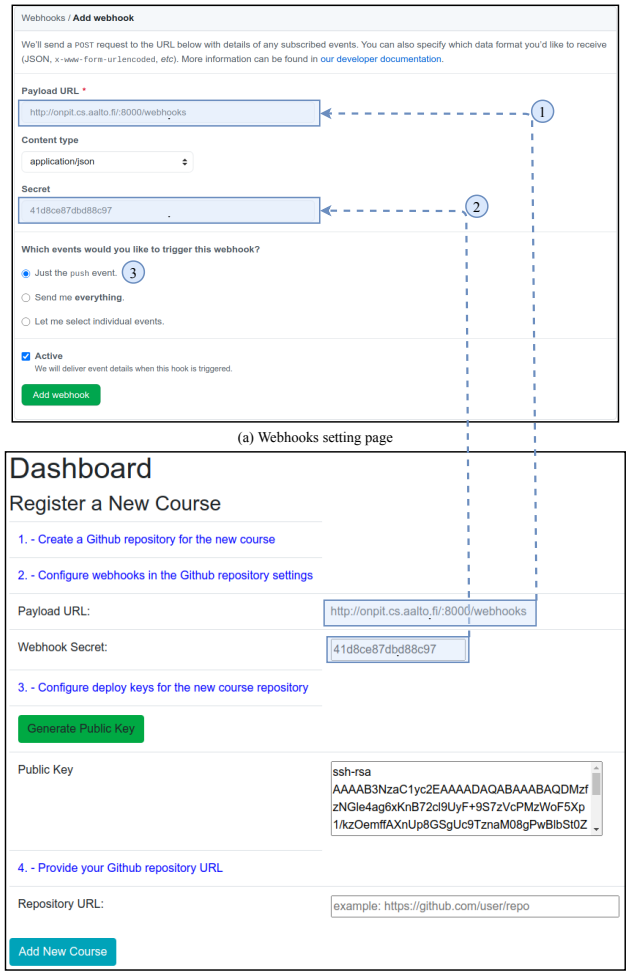
should not send the update directly to the web server. Instead, teachers provide new content to the repository of the course in the public Git service. By default, the web server does not get any information related to the update that occurred in the public Git service. The web server requires a notification sent from the Git service every time there is an updated content that occurred on this site. The notification is used to trigger the web server to copy the latest course's content from the public Git service. To bridge the gap of information between the web server and the public Git service, we implemented webhooks for updating the course content.

The implementation of webhooks requires us to provide a public URL endpoint on the web server. The public Git service sends a notification using an HTTP POST request to the public URL. Then, a webhook handler is prepared on the web server to send a response for each incoming request that is received on the public URL. The public URL is supposed to receive only the HTTP POST request from a legitimate public Git service. Nevertheless, since the URL is public, every internet service could send an HTTP request to this URL. Therefore, an authentication mechanism is required to limit the number of access that the webhooks handler should manage.

In a public Git service such as GitHub, each repository allows the repository owner to add a secret when configuring webhooks. The public Git service employs the secret to create an HMAC signature that is sent concurrently with the notification to the web server. The web server uses the HMAC signature to validate each incoming HTTP request before handling the request to the webhooks handler. As a result, the combination of the HMAC signature and the notification allows the web server to authenticate each HTTP request originating from the legitimate Git service.

The configuration process of webhooks in a repository requires the repository owner to provide a public URL endpoint, a secret, and specify events that trigger webhooks. Figure 4.3 displays the process of updating course contents. The dashboard that is used by teachers to register a new course has been designed to provide such information for the public URL endpoint and the secret. In the case of update course contents, the repository owner could only specify a push event. Once the configuration is complete, every time teachers push an update to the course repository in the public Git service, a notification and an HMAC signature are sent to the public URL endpoint on the web server.

The process of updating course contents using webhooks is illustrated in Figure 4.4. The update of a course starts once a push event occurred in the public Git service. Then, the Git service creates an HTTP POST request which contains relevant information such as an HMAC signature, a hash method to generate the signature, a type of an event, and a notification.




(a) Webhooks setting page

(b) Dashboard

Figure 4.3: Steps for updating a course

The first three components are stored in the HTTP POST header while the notification is stored in the HTTP POST body. The notification consists of information about the repository (e.g., repository name, branch, repository URL) where the specified event has been occurred in the public Git service. Then, the public Git service sends the HTTP POST request to the public URL on the web server. The web server receives the request and authenticates the source of the HTTP request by performing a signature check. The signature check starts by performing the same steps to generate the HMAC signature and comparing the value of the generated signature with the received HMAC signature from the incoming HTTP request. If both signatures have the same value, then the webhooks handler sends a Git request to the

public Git service to update the course contents. The subsequent Git request follows the same authentication process as in Figure 4.2.

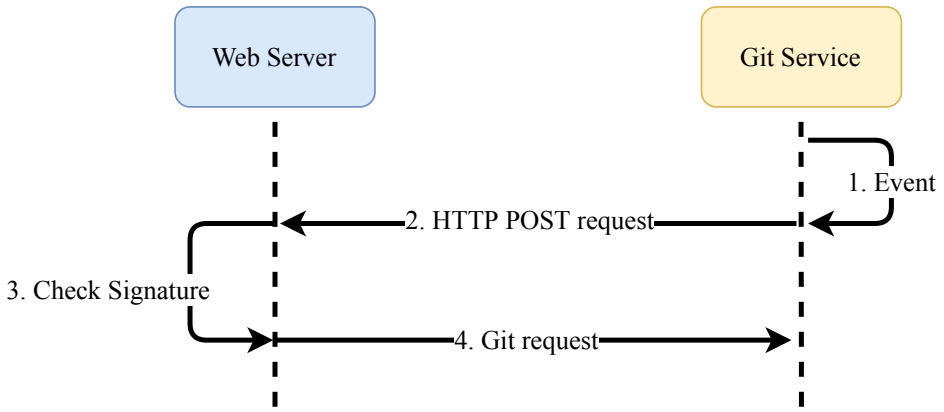



Figure 4.4: Updating course contents using webhooks

# Chapter 5

# Conclusions

In this thesis, we developed an online learning platform to provides teachers and students with an interface to interact with online labs in a public cloud infrastructure. The online learning platform consists of components that run as separate containers inside a Kubernetes cluster, including a web server, a database, and a group of virtual laboratories. Another component is a public Git service that mainly stores the contents of each course.

We developed the web server using the Django web framework. It consists of two services that implement authentication and authorization to the online learning platform, including a user service and a course service. The user service supports the learning platform to authenticate users using two schemes, either through Django built-in authentication or OpenID Connect. Both schemes provide each logged-in user with a role representing a set of access rights of each user in the online learning platform. Two roles are supported: teacher and student. Moreover, the user service provides each user with role-based access control to apply access rights in the learning platform.

The course service provides the learning platform with system authentication and authorization, primarily in accommodating teacher privileges. Users with teacher privileges can register multiple courses and update the contents of each course that is registered under their identity. In adding new courses, we use SSH keys in authenticating the web server to the public Git service. In addition, deploy keys based on SSH keys enable provision in the minimum access rights to the web server for cloning the Git repository of a new course. In updating course contents, webhooks enable content synchronization by sending a notification from the Git service to the web server every time teachers push an update. The web server generates an HMAC signature based on the webhooks secret to authenticate each incoming request from the Git service. A dashboard is implemented to provide teachers with an interface to configure courses.

## 5.1 Future work

some future works:

- Add learning analytics features to the teacher dashboard.
- Study other alternatives of key storage for SSH keys.

# Bibliography


[1] Authenticating to github. `https://docs.github.com/en/github/authenticating-to-github`. Accessed: 22 July, 2020.

[2] Managing deploy keys. `https://docs.github.com/en/developers/overview/managing-deploy-keys`. Accessed: 28 July, 2020.

[3] Passphrase. `https://www.ssh.com/ssh/passphrase`. Accessed: 8 August, 2020.

[4] User authentication in django. `https://docs.djangoproject.com/en/3.0/topics/auth/`. Accessed: 1 August, 2020.

[5] Aboba, B., and Simon, D. Ppp eap tls authentication protocol, 1999.

[6] Akinoğlu, O., and Tandoğan, R. Ö. The effects of problem-based active learning in science education on studentsâ academic achievement, attitude and concept learning. *Eurasia journal of mathematics, science and technology education 3*, 1 (2007), 71–81.

[7] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., et al. A view of cloud computing. *Communications of the ACM 53*, 4 (2010), 50–58.

[8] Auster, C. J. Blended learning as a potentially winning combination of face-to-face and online learning: An exploratory study. *Teaching Sociology 44*, 1 (2016), 39–48.

[9] Björkqvist, M., Cachin, C., Haas, R., Hu, X.-Y., Kurmus, A., Pawlitzek, R., and Vukolić, M. Design and implementation of a key-lifecycle management system. In *International Conference on Financial Cryptography and Data Security* (2010), Springer, pp. 160–174.

[10] Brusilovsky, P., Sosnovsky, S., Yudelson, M. V., Lee, D. H., Zadorozhny, V., and Zhou, X. Learning sql programming with interactive tools: From integration to personalization. *ACM Transactions on Computing Education (TOCE) 9*, 4 (2010), 1–15.

[11] Bruzual, D., Montoya Freire, M. L., and Di Francesco, M. Automated assessment of android exercises with cloud-native technologies. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (2020), pp. 40–46.

[12] Chandra, S., Paira, S., Alam, S. S., and Sanyal, G. A comparative survey of symmetric and asymmetric key cryptography. In *2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE)* (2014), IEEE, pp. 83–93.

[13] Deslauriers, L., McCarty, L. S., Miller, K., Callaghan, K., and Kestin, G. Measuring actual learning versus feeling of learning in response to being actively engaged in the classroom. *Proceedings of the National Academy of Sciences 116*, 39 (2019), 19251–19257.

[14] El Mhouti, A., Erradi, A. N. M., and Vasquèz, J. M. Cloud-based vcle: A virtual collaborative learning environment based on a cloud computing architecture. In *2016 Third International Conference on Systems of Collaboration (SysCo)* (2016), IEEE, pp. 1–6.

[15] Estriegana, R., Medina-Merodio, J.-A., and Barchino, R. Student acceptance of virtual laboratory and practical work: An extension of the technology acceptance model. *Computers & Education 135* (2019), 1–14.

[16] Ferraiolo, D., Kuhn, D. R., and Chandramouli, R. *Role-based access control*. Artech House, 2003.

[17] Friedl, S. An illustrated guide to ssh agent forwarding. `http://www.unixwiz.net/techtips/ssh-agent-forwarding.html`. Accessed: 29 July, 2020.

[18] García-Valls, M., Cucinotta, T., and Lu, C. Challenges in real-time virtualization and predictable cloud computing. *Journal of Systems Architecture 60*, 9 (2014), 726–740.

[19] Garson, K., and Adams, C. Security and privacy system architecture for an e-hospital environment. In *Proceedings of the 7th symposium on Identity and trust on the Internet* (2008), pp. 122–130.

[20] Goyal, S. Public vs private vs hybrid vs community-cloud computing: a critical review. *International Journal of Computer Network and Information Security 6*, 3 (2014), 20.

[21] Guerra, H., Cardoso, A., Sousa, V., and Gomes, L. M. Remote experiments as an asset for learning programming in python. *International Journal of Online and Biomedical Engineering (iJOE) 12*, 04 (2016), 71–73.

[22] Gupta, H., and Sharma, V. K. Role of multiple encryption in secure electronic transaction. *International Journal of Network Security & Its Applications 3*, 6 (2011), 89.

[23] Hardt, D., et al. The oauth 2.0 authorization framework. Tech. rep., RFC 6749, October, 2012.

[24] Harrison, M. A., Ruzzo, W. L., and Ullman, J. D. Protection in operating systems. *Communications of the ACM 19*, 8 (1976), 461–471.

[25] Ho, V.-T., Nakamori, Y., Ho, T.-B., and Lim, C. P. Blended learning model on hands-on approach for in-service secondary school teachers: Combination of e-learning and face-to-face discussion. *Education and Information Technologies 21*, 1 (2016), 185–208.

[26] Hu, W., Lei, Z., Zhou, H., Liu, G.-P., Deng, Q., Zhou, D., and Liu, Z.-W. Plug-in free web-based 3-d interactive laboratory for control engineering education. *IEEE Transactions on Industrial Electronics 64*, 5 (2016), 3808–3818.

[27] Kabiri, M. N., and Wannous, M. An experimental evaluation of a cloud-based virtual computer laboratory using openstack. In *2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI)* (2017), IEEE, pp. 667–672.

[28] Kanimozhi, S., Kannan, A., Suganya Devi, K., and Selvamani, K. Secure cloud-based e-learning system with access control and group key mechanism. *Concurrency and Computation: Practice and Experience 31*, 12 (2019), e4841.

[29] King, M. Polling vs. webhooks. `https://www.docusign.com/blog/dsdev-polling-vs-webhooks`. Accessed: 4 August, 2020.

[30] Ko, C. C., Chen, B. M., Hu, S., Ramakrishnan, V., Cheng, C. D., Zhuang, Y., and Chen, J. A web-based virtual laboratory

on a frequency modulation experiment. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 31*, 3 (2001), 295–303.

[31] Krawczyk, H., Bellare, M., and Canetti, R. Hmac: Keyed-hashing for message authentication, 1997.

[32] Krishnan, S., Varun, P., and Venkatasubramanian, B. Generic and configurable technique for webhook validation with arbitrary applications, Apr. 26 2018. US Patent App. 15/335,274.

[33] Leggetter, P. What are webhooks and how do they enable a real-time web? `https://www.programmableweb.com/news/what-are-webhooks-and-how-do-they-enable-real-time-web/2012/01/30`, 2012. Accessed: 1 July, 2020.

[34] Linton, T. F., Resinski, M. L., Felix, M. R., and Christopher, C. A. Systems and methods for utilizing webhooks integrated in paas supported application development and deployment, Sept. 27 2018. US Patent App. 15/465,475.

[35] Mazieres, D., Kaminsky, M., Kaashoek, M. F., and Witchel, E. Separating key management from file system security. In *Proceedings of the seventeenth ACM symposium on Operating systems principles* (1999), pp. 124–139.

[36] McCollum, C. J., Messing, J. R., and Notargiacomo, L. Beyond the pale of mac and dac-defining new forms of access control. In *Proceedings. 1990 IEEE Computer Society Symposium on Research in Security and Privacy* (1990), IEEE, pp. 190–200.

[37] McConnell, J. J. Active learning and its use in computer science. In *Proceedings of the 1st conference on Integrating technology into computer science education* (1996), pp. 52–54.

[38] Moore, J. L., Dickson-Deane, C., and Galyen, K. e-learning, online learning, and distance learning environments: Are they the same? *The Internet and Higher Education 14*, 2 (2011), 129–135.

[39] Navrat, P., and Tvarozek, J. Online programming exercises for summative assessment in university courses. In *Proceedings of the 15th International Conference on Computer Systems and Technologies* (2014), pp. 341–348.

[40] Rada, J. F. S., Iglesias, C. A., and Coronado, M. Maia: an event-based modular architecture for intelligent agents. In *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)* (2014), vol. 3, IEEE, pp. 87–94.

[41] Rebholz, S., Libbrecht, P., and Müller, W. Learning analytics as an investigation tool for teaching practicioners. In *Proceedings of the Workshop on Towards Theory and Practice of Teaching Analytics* (2012).

[42] Sakimura, N., Bradley, J., Jones, M., De Medeiros, B., and Mortimore, C. Openid connect core 1.0. *The OpenID Foundation* (2014), S3.

[43] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E. Role-based access control models. *Computer 29*, 2 (1996), 38–47.

[44] Sandhu, R. S., and Samarati, P. Access control: principle and practice. *IEEE communications magazine 32*, 9 (1994), 40–48.

[45] Schneider, F. B. Least privilege and more [computer security]. *IEEE Security & Privacy 1*, 5 (2003), 55–59.

[46] Van Popta, E., Kral, M., Camp, G., Martens, R. L., and Simons, P. R.-J. Exploring the value of peer feedback in online learning for the provider. *Educational Research Review 20* (2017), 24–34.

[47] Xu, L., Huang, D., and Tsai, W.-T. Cloud-based virtual laboratory for network security education. *IEEE Transactions on Education 57*, 3 (2013), 145–150.

[48] Yassein, M. B., Aljawarneh, S., Qawasmeh, E., Mardini, W., and Khamayseh, Y. Comprehensive study of symmetric key and asymmetric key encryption algorithms. In *2017 international conference on engineering and technology (ICET)* (2017), IEEE, pp. 1–7.

[49] Ylonen, T., and Lonvick, C. The secure shell (ssh) authentication protocol. Tech. rep., RFC 4252, January, 2006.

[50] Zhao, J., and Forouraghi, B. An interactive and personalized cloud-based virtual learning system to teach computer science. In *International Conference on Web-Based Learning* (2013), Springer, pp. 101–110.

[51] Zhou, R.-g., Li, W., Huan, T.-t., Shen, C.-y., and Li, H.-s. An online banking system based on quantum cryptography communication. *International Journal of Theoretical Physics 53*, 7 (2014), 2177–2190.

[52] Zhu, J., Chu, B., and Lipford, H. Detecting privilege escalation attacks through instrumenting web application source code. In *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies* (2016), pp. 73–80.