4-2021

QUANTIFYING SECURITY IN PLATFORM AS A SERVICE USING MEAN FAILURE COST: A STAKEHOLDER'S PERSPECTIVE

Anuradha Chauhan
achauhan1@go.stcloudstate.edu

Quantifying Security in Platform as a Service using Mean Failure Cost: A Stakeholder's Perspective

by

Anuradha Chauhan

Starred Paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science in

Information Assurance

May, 2021

Starred Paper Committee:
Abdullah Abu Hussein, Chairperson
Lynn Collen
Balsy Kasi

Abstract

Cloud computing is a scalable and cost-effective technology being adopted by organizations to increase profits and flexibility. Due to the nature of cloud service models, security is a big concern for organizations. In this paper, we discussed the security issues in the Platform-as-a-Service cloud service model and the various possible attack types on PaaS. PaaS has a number of stakeholders such as developers, testers, deployers, and administrators. PaaS stakeholders have different security goals, objectives, and tolerance to risks. Organizations need to understand the possible threats, source of those threats, and different security requirements of their stakeholders in a PaaS service model so that they can measure the risks accurately and implement effective security controls. The objective of the study is to provide stakeholders of PaaS with means to quantify the security threats in PaaS cloud services. We studied the different ways in the literature to measure the security risks in cloud computing. We proposed our methodology to quantify security in PaaS from a stakeholder's point of view, considering their interactions with various PaaS components and the tasks they perform. We identified the stakeholders, their interactions with different components, and the tasks they perform by creating a taxonomy. We then used a well-known metric, Mean Failure Cost, to quantify the failure cost of PaaS components as pertinent to stakeholders. This will help the organizations to assess their security controls and implement better security measures.

Keywords: Platform as a service, Mean failure cost, Quantification of risks, cloud security

Table of Contents

Chapter Page

List of Tables

Table Page

List of Figures

Figure Page

Chapter I: Introduction

Introduction

Cloud computing is increasingly being adopted by many organizations and users today. It offers convenient access to a shared pool of computing resources like infrastructure, platforms, storage, data, software, and applications as a service to its users. Many organizations are moving to cloud as it helps in collaboration, improves scalability, availability, flexibility, productivity along with reduced operational costs. Along with the multiple advantages of cloud computing, there are many security issues that are of great concern to organizations. Organizations are saving their critical applications and customer's personal data in the cloud, and securing those applications and data is critical for their business. Organizations are still struggling to understand how they can ensure security at all levels, network, host, and application levels in a cloud environment.

The cloud service models defined by NIST are of three types (Mell & Grance, 2011): Infrastructure-as-a-service (IaaS), Platform-as-a-service (PaaS), and Software-as-a-service (SaaS). IaaS provides hardware, storage, servers, and networking hardware to its users. PaaS provides a platform for developers to develop, test, and deploy their applications. SaaS provides users with access to cloud-based applications. In this paper, we are focusing on the PaaS service model and its security issues. IT organizations have gained significantly by adopting the PaaS service model and reducing their cost of purchasing and maintaining the underlying hardware, storage, and operating systems. Some examples of PaaS offerings are Microsoft Azure, AWS Elastic

Beanstalk, Oracle cloud platform, and Google App Engine. Security in a PaaS service model is a shared responsibility of the cloud provider and cloud customer (Liu et al., 2011). The cloud provider is responsible for providing security to the underlying infrastructure and the operating system. The cloud customer is responsible for securing its data and applications. Securing data is a big concern for every organization working in a cloud environment, and it is highly important in the case of a PaaS environment as it holds the most important information of an IT organization, its source code. Organizations need to have a better understanding of the security vulnerabilities and threats existing in a PaaS environment, especially the threats which are resulting from the interactions of different stakeholders with the components of the PaaS environment. They need to have a cybersecurity measure to quantify the security threats in their PaaS environment so that they can perform a more accurate risk assessment and implement better security controls.

Problem Statement

PaaS has several stakeholders, such as developers, testers, deployers, and administrators. PaaS stakeholders have different security goals, objectives, and tolerance to risks. This is highly dependent on the project domain, sensitivity of the data, and exposure to risks. The security threats largely depend on the stakeholders and how they interact with various components of the PaaS environment. Organizations need to understand the threats, source of the threats, and different security requirements of their stakeholders in a PaaS service model so that they can measure the risks accurately and implement effective security controls.

Nature and Significance of the Problem

The challenge that developers, testers and deployers face while developing, testing, and deploying applications is that they lack a foundational knowledge in application security. They focus on building the applications and making them work, but in the process, they often forget about the security of those applications. This leads to security vulnerabilities in the software and its implementation. These vulnerabilities are exploited by hackers for their benefits.

Code Spaces was a code hosting company that used to offer repositories for source code using code management tools like Git and Subversion. Code Spaces was using AWS for storage and server instances. They were attacked by a hacker who successfully gained access to their AWS control panel. The attacker asked for ransom, to which Code Spaces denied. When Code Spaces tried to take control of their AWS control panel, the attacker started deleting all their data along with backups. This attack destroyed Code Spaces, and the company was shut down. The company failed to secure its data and backups. There were many security issues that led the company to its failure, like not using multifactor authentication and lack of strict authorization policies for deleting primary data and backups (*Code Spaces: A Lesson In Cloud Backup* | *Network Computing*, 2014).

In 2019, hackers gained access to the Google Cloud infrastructure of Volusion and changed the JavaScript code to get card details of users from online forms (*Hackers Breach Volusion and Start Collecting Card Details from Thousands of Sites* | *ZDNet*, 2019). In 2017, personal details of 198 million American voters were exposed in

the Republican National Committee data breach. This attack happened because an engineer misconfigured the storage platform as public instead of private (*Top 5 Cloud Security Related Data Breaches! - Cybersecurity Insiders*, 2017). In another incident in 2017, millions of Verizon customers' data was exposed due to a configuration error on the AWS S3 bucket. This incident was due to a misconfiguration in security settings on the server, which allowed anyone to download files from the server.

These incidents indicate that organizations using cloud need to understand the security threats at each and every layer of cloud infrastructure and also at each interaction point of stakeholders with the cloud environment. This understanding can help them ensure comprehensive security of their critical applications and data in the cloud environment.

Objective of the Study

The objective of this study is to provide stakeholders of PaaS with means to quantify the security threats in PaaS cloud services. This work focuses on measuring the cost of PaaS components failure as pertinent to stakeholders of PaaS. In order to quantify the security threats in a PaaS model, we will consider the stakeholders, the tasks they perform, the different components of cloud infrastructure they interact with, and the security threats specific to them. This will help the organizations to assess their security measures and implement better security controls. The ultimate goal of this work is to improve security in cloud-based applications.

Study Questions/Hypotheses

Can failure be quantified in cloud computing?

How can failure be quantified from the stakeholder's point of view?

What are the existing methods to quantify failure out there?

How effective is Mean Failure Cost in quantifying security and privacy from the stakeholder's point of view?

Definition of Terms

Cloud Computing: Cloud computing is a model that provides convenient access to a shared pool of computing resources with minimal management effort (Mell & Grance, 2011).

NIST: National Institute of Standards and Technology is a non-regulatory federal agency of the United States Department of Commerce, and its mission is to promote innovation and industrial competitiveness by advancing measurement standards and technology.

SaaS: Software as a Service is a cloud service model in which cloud consumers can access applications running on the cloud infrastructure provided by a cloud provider (Mell & Grance, 2011).

PaaS: Platform as a Service is a cloud service model in which cloud consumers can develop and deploy applications created using programming languages, libraries, and tools provided by the cloud provider (Mell & Grance, 2011).

IaaS: Infrastructure as a Service is a cloud service model in which cloud consumer is provisioned with processing, storage, networks and other computing resources where the consumer can deploy software and applications (Mell & Grance, 2011).

Vulnerability: Vulnerability is a weakness in a system which can be exploited by an attacker to attack the system.

Threat: A threat is a potential danger that might exploit a vulnerability.

AWS: Amazon Web Services (AWS) is a subsidiary of Amazon which provides cloud services to cloud consumers (*Getting Started with Amazon Simple Storage Service - Amazon Simple Storage Service*, 2020).

AWS S3: Amazon Simple Storage Service is a service provided by AWS for storing and accessing data from anywhere on the web (*Getting Started with Amazon Simple Storage Service - Amazon Simple Storage Service*, 2020).

Git: Git is an open-source distributed versioning control system.

Subversion: Apache Subversion is an open-source software versioning control system.

Microsoft Azure: Microsoft Azure is a cloud computing service provided by Microsoft to build, test, deploy, and manage applications in cloud infrastructure (*Get to Know Azure* | *Microsoft Azure*, 2020).

AWS Elastic Beanstalk: AWS Elastic Beanstalk is a service provided by AWS to build and deploy applications in cloud infrastructure (*AWS Elastic Beanstalk – Deploy Web Applications*, 2020).

Oracle cloud platform: Oracle Cloud Platform is a platform as a service provided by Oracle (*What Is Oracle Cloud Platform* | *Oracle*, 2020).

Google App Engine: Google App Engine is a platform as a service provided by Google for developing and hosting web applications in cloud infrastructure (*App Engine | Google Cloud*, 2020).

Amazon EC2: Amazon Elastic Compute Cloud is a web service that provides scalable compute capacity in the cloud.

Data remanence: Data remanence is the residual data that remains even after the data has been deleted from the storage media.

Hypervisor: A hypervisor is a computer software or hardware that creates, runs, and manages virtual machines.

ARP: Address Resolution Protocol is a communication protocol used to convert an IP address into MAC address.

ARP poisoning: ARP poisoning is a network attack that manipulates host ARP tables allowing an attacker to intercept network traffic.

HTTP: HyperText Transfer Protocol (HTTP) is an application layer protocol used to transfer data over the web.

HTTP session hijacking: HTTP session hijacking is an attack in which an attacker takes control of a user's session (*Session Hijacking Attack | OWASP*, 2020).

DNS: Domain Name System is a distributed directory naming system of computers, services, and other resources connected to the internet, and it resolves domain names to IP addresses.

DNS cache poisoning: DNS cache poisoning is a cyber-attack in which an attacker changes the data in DNS servers so that the legitimate traffic is rerouted to the

attacker's websites (*What Is a DNS Cache Poisoning?* | *DDI (Secure DNS, DHCP, IPAM)* | *Infoblox*, 2020).

IP spoofing: IP spoofing is a technique used to create IP packets with modified source address to hide the identity of the sender.

SQL injection: SQL injection attack is an attack in which the SQL query is inserted from the client to the application through the input data.

LDAP: Lightweight Directory Access Protocol is an application layer protocol used for querying and manipulating directory servers.

LDAP injection: LDAP injection attack is an attack in which an attacker sends LDAP queries with code injections to gain unauthorized access to sensitive information stored on LDAP servers (*LDAP Injection*, 2020).

API: Application Programming Interface is a computing interface specific to an application or operating system that allows two applications to communicate with each other.

SLA: Service Level Agreement is a commitment between a service provider and a customer.

SOAP: Simple Object Access Protocol (SOAP) is a messaging protocol specification for exchanging structured information in web services.

XML: XML or Extensible Markup Language is a markup language designed to store and transport data.

WSDL: Web Services Description Language is an XML based interface description language used to describe the functionalities provided by a web service.

AHP: Analytic Hierarchy Process is a structured mathematical technique used in complex decision making (Saaty, 1987).

Summary

This chapter discussed cloud computing and cloud service models IaaS, PaaS, and SaaS, along with the benefits of cloud computing and the security concerns of organizations using cloud services. It also discussed the need for a cybersecurity measure to quantify the security threats in cloud computing from a stakeholder's point of view. The problem statement is how to quantify the failure cost in the PaaS service model considering stakeholders and their interactions with the various components of the PaaS environment. This will help organizations to measure risks accurately and implement effective security controls. The objective of the study is to provide stakeholders of PaaS with means to quantify security threats.

Chapter II: Background and Review of Literature

Introduction

This chapter discusses the research done in the field related to the security issues of cloud computing. The chapter first focusses on the different threats, security vulnerabilities, and attacks on cloud infrastructure, especially PaaS. Then the chapter discusses the research works done to quantify the security risks and failure cost in PaaS cloud service model considering stakeholders.

Literature Related to the Problem

Zissis et al. discusses the identification of threats in cloud computing (Zissis & Lekkas, 2012). Multitenancy in the cloud presents many confidentiality and privacy threats. The risk of data loss is more in the cloud due to the number of applications, devices, and users. An important characteristic of the cloud model is object reusability. These reusable objects must be controlled to decrease the chances of any security issues. Data confidentiality could also be compromised due to data remanence. Due to the multitenancy and virtualized nature of the cloud, data remanence may lead to the disclosure of confidential data. To secure a system, both data confidentiality and software confidentiality are important. The cloud provider should implement mechanisms to ensure data and software integrity. The threats at the physical level in the cloud include denial of service attacks, network attacks, hardware attacks, infrastructure misuse, and natural disasters. The threats at the virtual level affecting PaaS and IaaS services are programming flaws, software modifications, impersonation, session hijacking, traffic flow analysis, and connection flooding. The threats at the

application level affecting SaaS services are data modification, privacy issues, impersonation, session hijacking, and traffic flow analysis.

Ristenpart et al. showed how virtual machines over a shared physical infrastructure could be used to perform cross VM side-channel attacks to get information from a target VM (Ristenpart et al., 2009). If the VM of an attacker is located on the same physical server, the attacker can penetrate the isolation between VMs and attack target VM. The attack involves two steps – VM placement and information extraction. Placement of VM means getting a VM on the same physical machine as that of the target customer. The researchers demonstrated using Amazon's EC2 that by spending some extra money, an attacker can get a 40% chance of placing a VM on the same physical machine as that of the target customer. They mapped the AWS EC2 service to find the location of the target and the parameters needed during instance creation such that the new VM is located on the same physical machine as that of the target. An attacker can use the DNS service provided by EC2 to map the target's public IP address to a private IP address in order to find the instance availability zone. They used network probes to determine the services hosted on EC2 and to verify that VM placement on the same physical server as the target has been successful. They used hping, nmap, and wget for network probes. The attacker can brute force to launch multiple instances until the instance is located on the same physical server as the target instance. After the placement of VM, they used cross VM side-channel attack to successfully extract information from the target customer's VM.

Chen et al. discussed the data security and privacy issues during the entire data lifecycle in cloud computing (Chen & Zhao, 2012). The data lifecycle has seven phases: data generation, data transfer, data use, data share, data storage, data archival, and data destruction. Data needs to be secured from its generation until its destruction. During data generation phase, data ownership should be maintained in the cloud. During data transfer phase, data confidentiality and integrity should be maintained between the enterprise network and the cloud storage. Data encryption is possible for static data in the cloud, but it is not possible for data being used by PaaS and SaaS applications. In a multitenant cloud model, data being used is saved with other users' data, which could be a threat to data security. The data owners can also share the data with other parties in the cloud. The data owners should consider that the other parties maintain the original data protection measures and usage restrictions. During data storage phase, data should be encrypted to ensure confidentiality. Managing the keys could be a problem in the cloud. Verifying the integrity of data stored in the cloud is also an issue to be considered. When data is no longer needed, it should be destroyed. This needs to be ensured that data has been destroyed from all locations in a cloud environment.

Hashizume et al. has categorized the security issues as per the different cloud service models (Hashizume et al., 2013). They identified the main vulnerabilities and threats related to cloud computing, along with the possible countermeasures. Cloud computing has different features such as virtualization, distributed resources, large scale heterogeneous resources, as compared to traditional technologies. Traditional

security measures like access controls are not enough to secure cloud systems. Due to the different service models and technologies being used in the cloud, the risks are different from traditional infrastructures. SaaS provides applications access to its customers. In a SaaS service model, users have the least control over security. The SaaS applications are accessible through a web browser. The vulnerabilities in the web browser can be used to attack SaaS applications. The security issues in SaaS applications will be the same as other web applications. Securing data is the biggest security concern in SaaS. Securing stored data as well as data being processed is the responsibility of the cloud provider. PaaS provides the hardware and operating system for application development and deployment. Like SaaS, the security of PaaS also depends on a secure network and secure web browser. Data security is also a challenge of PaaS service model. PaaS offers a platform for developers to build, test, and deploy their SaaS applications, thereby increasing the security dependencies between them. Due to these dependencies, an attack on one cloud service layer will affect the other layers as well. A SaaS provider might be using platforms from a PaaS provider, which might be using infrastructure from an IaaS provider. These interdependencies in different cloud service models can give rise to new security threats. A provider is also a consumer. These can create confusion over security responsibilities. Securing PaaS includes securing PaaS platform and deployed applications. The authors have mentioned three main data security issues and challenges in PaaS: security issues in third-party web hosted development tools, the speed of change of applications in the cloud, the security of PaaS applications, and the

security of the underlying infrastructure. Developers are responsible for creating secure cloud applications, but the speed of change of applications in the cloud affects the security of those applications. Any change in PaaS components can affect the security of the applications. Developers should also consider the location at which they are storing their data as different locations have different laws related to data privacy and security. In the IaaS service model, cloud customer has better control over the security as compared to other cloud service models. The security issues in IaaS are related to virtualization, hypervisor, virtual machine lifecycle, virtual networks, and shared resources. As cloud infrastructure is based on virtualization, securing virtual machines is very important. Virtualization adds another attack surface to the cloud. The hypervisor is responsible for controlling and monitoring virtual machines and providing isolation between them. The vulnerabilities of the hypervisor should be handled properly. In a cloud environment, the virtual machines are migrated between different hosts for load balancing and maintenance, which could be a security threat. If an attacker can compromise the hypervisor, he can transfer the virtual machines to a malicious host. Virtual machines located on the same host share resources like CPU and memory. This resource sharing can cause security issues. An attacker can try to extract information from other VMs located on the same host as the attacker's VM. Offline virtual machines can also be vulnerable. A virtual machine can be instantiated with a malicious image. Virtual networks increase the connectivity between VMs and, as a result, increase the security risks. The authors listed the vulnerabilities in cloud computing. It includes vulnerabilities in virtual machines such as covert channels between VMs on the same

host, unrestricted allocation of VM resources, uncontrolled migration of VMs, and uncontrolled snapshots, which could lead to data leakage. The vulnerabilities related to data include the location of data, incomplete deletion of data, data backup, and use of plaintext data during storage, transfer, or processing. The authors also listed the threats in cloud service models. The threats included data leakage, service hijacking, account hijacking, denial of service, sniffing virtual networks, data manipulation, VM escape, and insecure VM migration.

Modi et al. discussed the vulnerabilities, threats, and security issues in the cloud (Modi et al., 2013). Vulnerabilities in virtualization can lead to cross VM side-channel attacks as well as denial of service attacks. Vulnerabilities in Internet protocols can be used to attack cloud systems. Some examples of such attacks are ARP poisoning, HTTP session hijacking, DNS cache poisoning, and IP spoofing. Unauthorized management interface access can also be used to attack cloud systems. Injection vulnerabilities such as SQL injection, LDAP injection, and OS injection can be used to attack cloud applications and get access to the private user data stored by those applications. As cloud services are provided to customers through web browsers and APIs, vulnerabilities in APIs and web browsers can also be exploited to attack cloud services. The authors further discussed the threats in cloud computing. The primary threat in cloud computing is the change in business model and loss of control over data. Abusive use of cloud computing is another big threat to the cloud environment. Attackers can use the bandwidth, processing power, and storage capacities provided by the cloud to launch denial of service attacks, password cracking attacks, etc. The

application programming interfaces which the cloud providers expose to the organizations to manage and interact with cloud services can also be a threat to cloud services. Improper use of these interfaces can affect SaaS, PaaS, and IaaS service models. A malicious insider such as a cloud system administrator can access sensitive organization's data. The multitenancy and shared resources add another threat to cloud computing. Due to the nature of cloud service models, data compromise can happen in multiple ways. Service hijacking, by using phishing and exploiting software vulnerabilities, can lead the clients to malicious websites. An attacker can launch a distributed denial-of-service attack on a target by flooding the target with requests or by compromising other virtual machines in the cloud. A service injection attack can affect the PaaS customers, in which a malicious service is injected into the cloud. The user is provided with the malicious service instead of a valid service which affects the integrity of the service. Attacks on hypervisor can be used to gain access to another customer's virtual machines. By attacking the data communication between two parties, an attacker can launch a man in the middle attack.

Kandukuri et al. (Kandukuri et al., 2009) focused on Service Level Agreements (SLA) between the cloud provider and cloud customer to include security issues. Security is an important feature in SLA for cloud services. The SLAs should include details about the methods implemented to secure the cloud infrastructure and user data. The cloud provider should provide details about securing physical, network, and application layers. As the organization's sensitive data is stored outside its premises, the cloud customer should ask information about the provisioning and monitoring of

privileged user access. The cloud customers are ultimately responsible for the security and privacy of their data, so the cloud provider should provide information about regulatory compliances to the cloud customer. In cloud computing, the cloud customer does not know the location where their data is being stored. The cloud customers can ask the providers to store and process their data in specific jurisdictions. In a multitenant cloud environment, the cloud provider should ensure that customer's data is segregated and secured from any possible attacks from other tenants. Investigations are very difficult in the cloud environment as multiple tenants might be having data on the same host, and user data often migrates over different hosts. SLAs should include support from cloud providers during investigations.

Gruschka et al. presented a taxonomy of attacks on cloud computing based on participants (Gruschka & Jensen, 2010). They divided the participants into three types: users, services, and cloud provider. Any interaction in a cloud environment involves two participants from the above-mentioned types, like a user requesting a service or a service requesting computational resources from the cloud provider. The types of attacks can be classified into three groups based on interactions and participants. Every participant provides a specific interface to the other participant in an interaction. A service provides web or secure shell interface to users. Similarly, a cloud provides APIs for services. For three types of participants, there will be six interfaces (service-to-user, user-to-service, service-to-cloud, cloud-to-service, cloud-to-user, and user-to-cloud). These six interfaces provide six attack surfaces. The most commonly exploited attack surface is service-to-user. This involves attacks like buffer overflow, SQL injection,

privilege escalation, etc. Attacks on a user-to-service interface include browser attacks, phishing attacks, etc. Attacks on service-to-cloud include attacks by any service on the cloud infrastructure, e.g., denial of service attacks, attacks on virtual machines. The attack surface on the cloud-to-service interface includes all attacks from a cloud provider on the services running on its hosts. A user-to-cloud interface includes attacks involving phishing emails to a cloud user with a spoofed email address of the cloud provider. It is not necessary that an attack involves only one interface; an attacker could exploit several attack surfaces combined together.

Jensen et al. discussed the technical security issues in cloud computing due to the usage of cloud services and underlying technologies used in building these services (Jensen et al., 2009). The two main technologies which are used to access cloud services are web services and web browsers. WS-Security is a standard that defines confidentiality, integrity, and authentication for securing SOAP messages. It defines XML digital signatures and XML encryption applied to SOAP messages. XML digital signatures are used to provide authenticity to messages. XML encryption is used to provide message confidentiality. An attack on XML signatures is known as an XML signature wrapping attack. Another type of attack on cloud systems is to inject a malicious service or a malicious virtual machine to the cloud. In this attack, the attacker creates its malicious service (SaaS or PaaS) or virtual machine (IaaS) and adds it to the cloud. The attacker tricks the cloud system to consider the new service as a valid service, and after that, the user requests are redirected to the malicious service. Another attack affecting PaaS and SaaS is the metadata spoofing attack in which the

attacker modifies the web services metadata files like WSDL to invoke another operation.

Sandikkaya et al. classified the security risks of PaaS and proposed some solutions for those risks (Sandikkaya & Harmanci, 2012). PaaS can be used to build and deploy applications. Resources are shared among users, and user objects are stored on multiple hosts. These user objects need to interact with each other to complete a task. The main source of security issues in PaaS is the spread of user objects on multiple hosts in the cloud. The authors classified the security issues in PaaS into four categories – issues related to resource pooling and rapid elasticity, issues related to broad network access, issues related to privacy, and issues related to continuity of service. In a PaaS service model, varied hardware and software resources are pooled together. These different resources can have different security settings. A setting might secure a specific resource but not another. Interoperability must be provided between different object interfaces, and these interfaces should support all possible access scenarios. The user objects are saved over multiple hosts, so in a multitenant cloud environment, the protection of hosts along with the objects is also important. In a PaaS cloud, user objects can be attacked by three types of adversaries. The first adversary is the cloud provider who can access user objects on its hosts. The second adversary could be another tenant on the same host who might attack user objects. The third adversary might be a third-party trying to access user objects. The interoperability problem and host protection issue can be solved using a trusted computing base. Trusted Computing Base is installed on all the hosts, and resources

are assigned through it. As objects use the standard interface to access the resources, interoperability is maintained. The assignment of resources is being checked by the trusted computing base, so attacks from objects to hosts can also be prevented. The user objects can be protected using encryption. As the PaaS services are accessed through broad network access, network communications must be secured, and access control must be applied to resources. The user objects need to communicate with each other to perform a task, and these communication channels are targets for attackers. A host has multiple objects residing on it, and it must apply policies on each object. The user objects in the PaaS cloud often migrate to different hosts, and this dynamic nature does not allow the hosts to apply updated policies on objects. To provide confidentiality over communication channels, the Transport Layer Security protocol must be used. Each object should have its own certificate. Access control policies must be combined with the user objects as the settings will be carried with the object. Policy enforcement points are used to manage access to objects using the object access control policies. If the policy enforcement point collects unnecessary private data, it could be a security risk. The policy enforcement point can also be attacked by a denial of service attack, which can stop it from responding to access requests. In traditional authentication methods, a lot of user information is collected than needed. This increases the risk of exposing user privacy. The objects should not collect more attributes than needed for authentication.

Literature Related to the Methodology

Rainer et al. discussed the different risk analysis methods for information technology (Rainer et al., 1991). The authors discussed quantitative risk analysis methods, which include Annual loss expectancy (ALE), Courtney, Livemore risk analysis methodology (LRAM), and Stochastic dominance. In the Annual loss expectancy, a list of all assets is created. The potential threats to the assets are analyzed along with the expected loss due to those threats. The annual loss expectancy from a threat is calculated as the product of the probability of occurrence of the threat event per year and the expected loss from a threat event. The Courtney method modified the ALE method and provided a dollar estimate of the annual expected loss. Livemore risk analysis method is similar to the ALE method except that it does not calculate total risk but instead calculates risk by individual elements involving the occurrence of single event loss. In Stochastic dominance method, it is assumed that disaster has already occurred, and analysis of the effects of the disaster is performed. The probability of the occurrence of disaster is not estimated, but the time to recover from disaster and the expected loss is calculated. In the qualitative risk analysis methodologies, the authors discussed scenario analysis, fuzzy metrics, and questionnaires. In scenario analysis, assets and possible threats are identified. Then scenarios are developed explaining the possible threats and respective loss to an asset. Fuzzy metrics use natural language values for assets, threats, and security mechanisms. Asset value might be small, medium, or large. The probability of

occurrence of a threat might be low, medium, or high. The authors proposed a risk analysis methodology by combining quantitative and qualitative methods.

Zhang et al. presented a risk management framework for the cloud environment considering all cloud deployment models and service models (Zhang et al., 2010). The framework consists of three phases: Architect and establish the risk management program, Implementation and operations, and Monitoring and review. In the first phase, which is architect and establish phase, planning for an effective risk management program is done by selecting critical areas of the cloud environment and then planning to establish risk management, program activities, define goals, requirements, and scope, creating steering committee and defining roles and responsibilities. The next phase is the Implementation and operations phase, which involves risk analysis, risk assessment, and risk mitigation. Risk analysis includes threat and vulnerability identification. The risk assessment determines the quantitative and qualitative outputs from the risk analysis phase. In this, the likelihood and impact of an attack are determined. The impact is determined in terms of high, medium, or low. The next step in risk assessment is to determine the level of risk. In risk mitigation, the cloud provider implements steps to mitigate the risks by either avoiding, accepting, transferring, or reducing the risk. The third phase is the monitoring and review phase, in which the risk management plan is monitored by the cloud provider.

A quantitative security evaluation metric using the Goal-Question-Metric (GQM) method is proposed by Halabi et al. (Halabi & Bellaiche, 2017). In GQM, a model is defined at three levels – conceptual level describing the purpose of measurement,

operational level describing a set of questions to characterize the measurements, and quantitative level with a set of metrics to answer the questions. These security metrics can be statistical or technical. The authors discussed three types of security evaluation metrics – implementation metrics, effectiveness metrics, and impact metrics. The implementation metrics measure the progress of implementation of security policies and procedures. The effectiveness metrics measure the security level and performance of deployed security controls. These metrics are used to validate that security controls are implemented and operating correctly. The impact metrics are used to evaluate the effect of the failure of security controls, and the effect on the performance of the cloud application. In order to evaluate the cloud security services, evaluation metrics are created for each security service, and weights are assigned to each metric.

Wu Chaoxia et al. proposed a risk assessment method using game theory, which consists of participants, action set, and utility function (Wu Chaoxia et al., 2015). The participants are the attackers and defenders in cloud computing. In the game between an attacker and a defender, an attacker can choose to attack or not to attack. Consequently, a defender can also choose to defend or not to defend. Four action states are possible between attacker and defender, (attack, defend), (attack, no defend), (no attack, defend) and (no attack, no defend). The utility function is defined as the difference between benefit from an attack and the cost of an attack. The benefit of an attacker is the damage to the network or data minus the punishment by the defender. The benefit of a defender is the damage of the network minus restore effort. The cost of an attack includes resource and time consumption. When the defender

detects an attack on the cloud system, he tries to recover the system. The recovery of the system includes the coefficients of confidentiality, integrity, and availability. A payoff matrix is calculated for all action states considering the penalty and recovery costs. This payoff matrix is used to calculate the probabilities of the success of the attacker and the defender.

Masky et al. propose the Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) Allegro framework to identify the information security risks in cloud computing (Masky et al., 2015). The methodology consists of eight steps across four phases. In the first phase, a risk management criterion is established, which will evaluate the effect of a risk on a cloud service provider. It could include financial, productivity, legal, and reputational areas. Areas are prioritized as per the impact from high impact to low impact areas. In the second phase, an information asset profile is created, which identifies the security requirements of the asset, and then locations are identified where information is stored, processed, or transferred. In the third phase, possible threats are identified. Cloud infrastructure elements are identified in connection to the information assets. The threats against an information asset are identified using questionnaires. An information asset risk worksheet is created with information about threats and impacts on different information assets. A relative risk score is computed, and risks can be compared as per the relative scores. The last step consists of identification, analysis, and mitigation of risks.

To assess security risks in cloud computing, a quantitative risk assessment framework called QUIRC is proposed by Saripalli et al. (Saripalli & Walters, 2010). The

authors used six security objectives for the cloud platform, confidentiality, integrity, availability, multiparty trust, mutual auditability, and usability. A list of common threats on a cloud platform is created, and another list comprising threats against web applications is created. They defined risk as a product of the probability of occurrence of a threat event and its impact. $R_e = P_e I_e$. Probability P_e is a fraction less than 1 and impact I_e is a value between 1 and 15. These values are relative. The impact is considered low with values between 1 to 5 if the loss of six security objectives mentioned above has a limited effect on cloud service operations. The impact is moderate, with values between 6 and 10 if the loss of six security objectives could have a serious effect on the operations and assets of the organization. The impact is considered high, with values between 11 and 15 if the loss of six security objectives could have a catastrophic effect on organizational assets and operations. Impact tables are prepared for possible threat events. Risk is calculated for each threat event from the probability and impact values. The security risk to an application will be the average of all security risks under each security objective category. Weights should be assigned to each security objective so that their total is 1. The weights describe the relative importance of a given security objective to the organization. The total security risk to the application is $R = \sum_{s=1}^{6} w_s R_s$ where w_s is the weight of the security objective, and R_s is the risk under a given category. To evaluate the impact of a security event, the wide-band Delphi method can be used. In the Delphi method, a questionnaire is prepared, and the participants provide a numerical estimate of the impact and map it to one of the

six security objectives. The moderator collects the answers from all the participants and shares the results with the participants. The participants are allowed to reconsider their decisions. The important characteristics of this method are the anonymity of the participants and iterative feedback to the participants. The final data is used in the QUIRC model to calculate the risks by security objectives and the net security risk.

Abuhussein et al. identified and categorized security and privacy attributes of cloud considering the two important stakeholders of cloud services, cloud service provider, and cloud customer (Abuhussein et al., 2012). The security and privacy categories used by authors are network, interface, data, virtualization, governance, compliance, and legal issues security attributes. The attributes include encryption, backup, authentication, and access controls, data isolation, monitoring, data storage location, data sanitization, disaster recovery, SLA conformity, performance and scalability, hypervisor security, client-side protection, standards and certifications, and nested services. These security attributes can be used to assess and compare cloud services. The authors compared three cloud service providers using their security and privacy attributes.

Chauhan et al. proposed a system for measuring the security of a cloud application (Chauhan et al., 2013). The proposed system, known as Security Measurement System, has two components, an off-cloud security measurement machine, and an on-cloud security server. The cloud security server connects to different cloud systems and gets the metrics details. The security measurement machine further processes that metrics information. The security measurement machine

has different modules that are used in measuring security. The application and cloud provider module manage the list of applications. Security control and metrics module manage the listing of the application-specific security controls and metrics. This module also applies weights to each security control. An application profile is created using security controls, metrics, and weights. The Customer internal security policy connect module provides an interface to connect to the organization's internal standards and guidelines documents to fetch information about application-specific controls. The Cloud connect and probe module connects to systems hosting applications like antivirus, log management to collect information about events and use them to calculate security metrics. The Measurement engine accepts inputs from all other modules and computes the security metrics and overall security level of the cloud application. The dashboard displays the results.

A framework to compare and rank security levels provided by cloud service providers based on the Analytic Hierarchy Process (AHP) has been proposed by Taha et al. (Taha et al., 2014). Security requirements in cloud services can be specified in Security Level Agreements or SecLA to provide transparency about user requirements and security services provided by cloud service providers. SecLA specifies security service level objectives that cloud service provider agrees to provide. In the proposed method, quantitative values are computed using the analytic hierarchy process for different cloud service providers based on service level objectives mentioned in SecLA. The assessment and ranking process works in three stages. In stage one, the cloud service provider SecLA and user SecLA requirements are defined with weights

assigned to each security level objective to represent its importance from the user's perspective. In stage two a model to measure the security attributes is defined. The cloud service providers answer the security service level objectives with a yes or no depending on if they provide the respective service. If the service level objective is a numerical value, the cloud providers need to provide a value based on the level of service provided. In stage three, all the data collected is used to rank the cloud service provider using the analytic hierarchy process. In stage three, first the SecLA are used to create a hierarchical structure to define the security attributes and then weights are assigned to each security level objective. Pairwise comparison of security attributes is performed to provide a relative ranking of cloud service providers for a specific service level objective. In the final phase, all the rankings are aggregated to give an overall ranking to the cloud service providers.

Karabacak et al. proposed a new method called Information Security Risk Analysis Method (ISRAM) for information security risk analysis (Karabacak & Sogukpinar, 2005). ISRAM is a quantitative paper-based risk analysis method allowing the participation of managers and staff of the organization. It is a survey-based method. It does not calculate Annual Loss Expectancy or Single Loss Expectancy but calculates risk as a numerical value between 1 and 25.

To manage the risks of any system failure, metrics, or key performance indicators are commonly used by organizations. Some of the commonly used metrics are the mean time to failure (MTTF), mean time between failures (MTBF), mean time to detect (MTTD) and mean time to resolve (MTTR) (*Comparing MTBF, MTTF, MTTR*,

2016; *MTTD vs. MTTF vs. MTBF vs. MTTR | Resolve Major IT Incidents Quickly*, 2018). These metrics help in understanding the failure rate of critical business systems. Mean time to failure or MTTF is the total time a system can operate when a serious defect occurs before it fails completely. Mean time between failures or MTBF is a measure of system reliability. It is defined as the average time between failures. Mean time to resolve or MTTR is the average time needed to restore a failed system. Mean time to detect or MTTD is the time needed to detect an issue in the system. It measures the time from the beginning of a system malfunction to the time to identify the issue.

Hale et al. created a framework called SecAgreement (Hale & Gamble, 2012), which extends the SLA negotiation standard to allow cloud service providers to include security metrics in SLAs. It uses a matchmaking algorithm that calculates and ranks SecAgreement enhanced SLAs by their risk, and the results allow organizations to identify gaps, quantify risks, and then select cloud services that best meet their security needs.

Mili et al. proposed a quantitative model called mean failure cost (MFC) to measure the reliability of a system (Mili & Sheldon, 2009). The authors discussed the problems of other metrics like mean time to failure (MTTF) and mean time to detection (MTTD) used to measure system reliability. These metrics assume that the failure cost of a system is independent of subspecifications. A system specification is made up of multiple subspecifications, and stakes could be different in meeting different requirements. In these metrics, any subspecification failure is considered equal irrespective of the stakes. Another problem with these metrics is that they don’t consider

different stakeholders while calculating the failure costs. The third problem of these metrics is that they don't consider the variance in the failure probability of subspecifications. Verification and Validation activities might result in more confidence in meeting some requirements than others, hence reducing the failure probability of those requirements. The authors proposed Mean failure cost (MFC) to measure the reliability and safety of a system. Mean failure cost varies by stakeholder, variance in stakes a stakeholder has in meeting different security requirements, failure probability of different system components, and different failure impact on different stakeholders. The authors considered a sorting program to illustrate the proposed metrics. The sorting program is made up of two subspecifications, Ord to specify if the final array is sorted and Prm to specify if the final array is a permutation of the initial array. They considered five stakeholders of the sorting program, binary searcher, linear searcher, brute force searcher, average finder, and median finder. These stakeholders have different use cases and different stakes in meeting the two requirements. The authors quantify the different stakes of the stakeholders in meeting Ord and Prm requirements by their corresponding failure costs. Then the mean failure cost is computed for each stakeholder as the sum of the failure costs of the two subspecifications, weighted by the failure probability of the subspecifications.

Summary

This chapter discussed the security issues in cloud computing, especially PaaS service model. The chapter highlighted the different ways to solve the security issues in PaaS by quantifying security threats, risks and failure costs. The next chapter include

details of our methodology to handle the security concerns in PaaS from stakeholder's point of view so that organizations can quantify the failure costs and implement better security controls.

Chapter III: Methodology

Introduction

This chapter discusses the methodology we used in our work. To quantify the security in the Platform-as-a-Service cloud service model, we identified the different stakeholders of PaaS, the various components of PaaS they interact with, and then quantified the failure cost from stakeholder's perspective.

Design of the Study

Our approach consisted of two main steps: identification, and quantification. In the first step, we identified the stakeholders and their interactions with various PaaS components. We created a taxonomy to achieve this goal in which we identified and classified PaaS stakeholders, components of PaaS, their interactions with those components and threats that can prevent them from performing their tasks. After identification, we quantified the failure cost from the stakeholder's perspective by using a well-known metric, Mean Failure Cost. After looking at different approaches to quantify failure costs and risks, we have selected Mean Failure Cost to quantify security risks in PaaS from the stakeholder's point of view. The advantage of using Mean failure cost is that it distinguishes between different stakeholders and provides failure cost for each stakeholder of the system. It also considers the different components of the system architecture, the variance in failure probability from one component to another, and the variance in failure cost from one requirement to another.

Data Collection

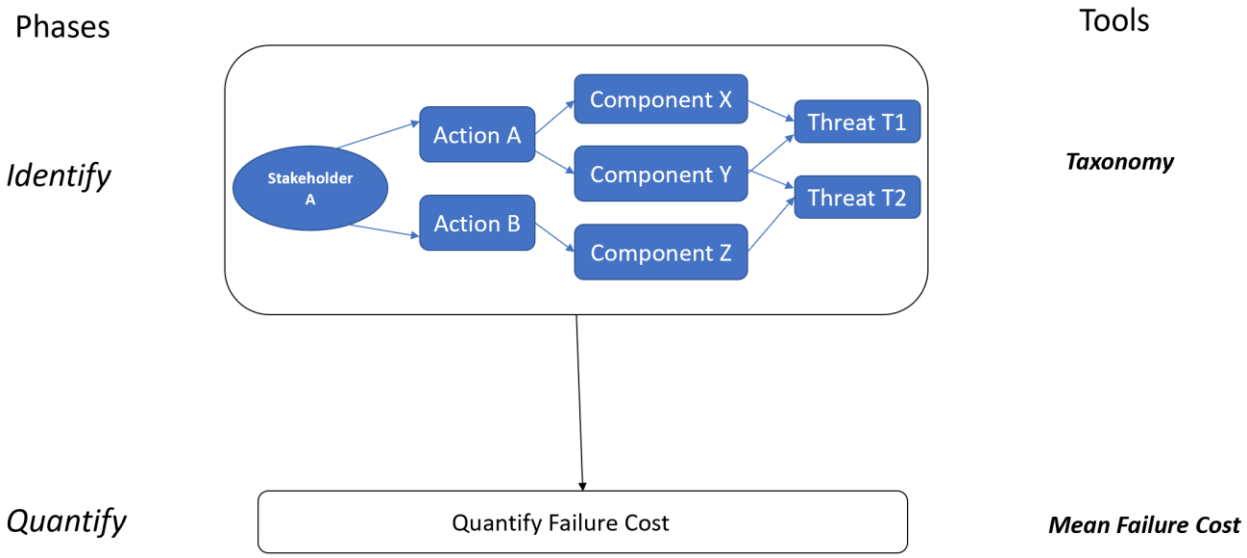
In the data collection, we collected data about the risks pertinent to various stakeholders of PaaS while interacting with the different components of PaaS. In order to achieve this goal, first, we identified different stakeholders of PaaS cloud environment, then collected data about the actions these stakeholders perform as part of their roles and responsibilities. We used job portal websites like indeed.com, glassdoor.com, LinkedIn and dice.com to get details about these actions. After identifying stakeholders and their actions, we collected data about different PaaS components these stakeholders interact with in order to perform the identified actions. We used software engineering Book of Knowledge and Internet search to identify the PaaS components. After identification of those components, we collected data about the threats that can prevent the PaaS stakeholders from performing the identified actions or interacting with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment.

Tools and Techniques

We used two tools in our work. We created taxonomies for the identification of the stakeholders, the actions they perform, the various PaaS components they interact with to perform those actions and the potential threats that can prevent the PaaS stakeholders from performing the identified actions or interacting with the identified PaaS components. We used a security metric, Mean Failure Cost, for the quantification of failure cost in a PaaS cloud service model from a stakeholder’s perspective. Fig. 3.1 shows the design of our study approach, along with the tools used in each phase.

Figure 3.1

Design approach and tools



Summary

This chapter discussed the design of the study and the various data collection methods used to identify PaaS stakeholders, the actions they perform, the different components they interact with in order to complete the actions, and the different threats that can prevent a PaaS stakeholder from performing those actions or interacting with those components. The chapter also discussed the tools and techniques that are used in the study. The next chapter discusses the details about the identification and quantification processes that are used to identify and quantify security risks in a PaaS cloud environment from a stakeholder's perspective.

Chapter IV: Data Presentation and Analysis

Introduction

This chapter discusses the identification of PaaS stakeholders, the actions they perform, the PaaS components they use to complete those actions and the potential threats that can prevent the PaaS stakeholders from doing those actions efficiently and securely. After identification, quantification of security risk is discussed using Mean Failure Cost as a security metric to calculate the failure cost of PaaS components from stakeholder's point of view.

Data Presentation

This section describes the two phases of the study in details, the identification phase, and the quantification phase. For the identification process, this paper proposes a taxonomical approach to identify PaaS stakeholders, their tasks, the PaaS components they need to interact with to complete their tasks and the potential threats that can prevent them from completing their tasks securely and efficiently. We are proposing a taxonomical approach as it is systematic and extensible. For the quantification of security risks from stakeholder's perspective, this paper proposes the use of Mean Failure Cost (MFC) security metric.

This section will discuss the taxonomies for different PaaS stakeholders, an overview of the Mean Failure cost (MFC) and then the proposed quantitative model is illustrated for one of the PaaS stakeholders.

Taxonomies for PaaS stakeholders

The taxonomies are classified into four levels that includes stakeholders, actions, components, and threats.

Stakeholder: A stakeholder is an individual, group or organization that uses PaaS as a software development environment for developing, testing, deploying and managing applications hosted in the cloud environment and can affect or be affected by the software development operations provided by the PaaS cloud provider. The focus of the study are PaaS consumers. PaaS cloud service providers are not included in the study.

Actions are the tasks performed by a PaaS stakeholder.

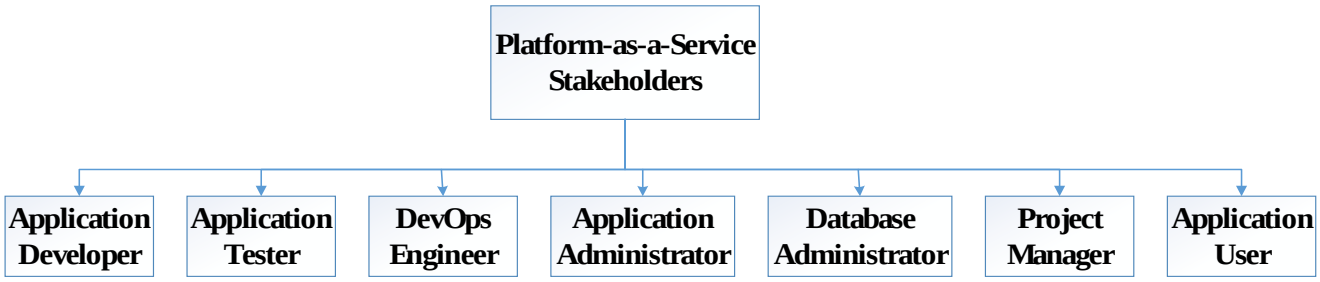
Components are software and hardware parts of the PaaS cloud environment that stakeholders interact with to perform the identified actions.

Threat: A threat is a potential danger that prevents a PaaS stakeholder from achieving his goals of developing, testing, deploying, and managing secure applications on-time, within budget and with quality in the PaaS cloud environment. In our study, threat also includes any potential danger that prevents an end user of the developed application from using the application securely in a cloud environment, or any potential danger to the overall security of the PaaS cloud environment.

The identified PaaS stakeholders are shown in the below taxonomy:

Figure 4

PaaS Stakeholders



Note. A taxonomy has been created for each PaaS stakeholder. These taxonomies are discussed further in this section.

Taxonomy for Application Developer

An application developer is a software development professional who designs and develops applications by creating or modifying source code for software applications.

The actions performed by the application developer in a PaaS cloud environment (Bourque et al., 2014), as part of his roles and responsibilities are:

a) *Analyze and Specify application requirements:* Requirements analysis is the elicitation, analysis, specification, and validation of the application software requirements. The application developer analyses and specifies the application software requirements and manage the application requirements during the entire lifecycle of the software development.

b) *Design application software:* In the software design process, the application software requirements are analyzed, and a software design is created that

describes the software architecture, its components, and interactions between those components. The application developer is responsible for designing application software.

c) *Create Applications:* The application developer is responsible for creating applications by writing code for the application software. Application software is created by using the application software design created earlier in software development lifecycle.

d) *Integrate Applications:* Application integration is the process of making your applications work and communicate with other applications. The application developer integrates the developed applications with other applications in the PaaS cloud environment.

e) *Test Software:* The application developer performs unit testing after writing the code for the application module. Unit testing is performed to ensure that the application module is behaving as expected and designed.

f) *Fix software bugs:* The application developer is responsible for debugging and fixing the software bugs or faults in the application software code.

g) *Maintain software code versions:* The application developer maintains different versions of software source code and configuration files, by using version control software. Many application developers work simultaneously on updating the application's source code and as a result, multiple versions of the source code are created in the application development phase.

The PaaS cloud components that the application developer interacts with to perform the identified actions are:

a) *Microsoft Word:* As part of the analysis and specifications of application requirements, the application developer creates application requirements documents. For creating these documents, Microsoft Word software is used by the application developer.

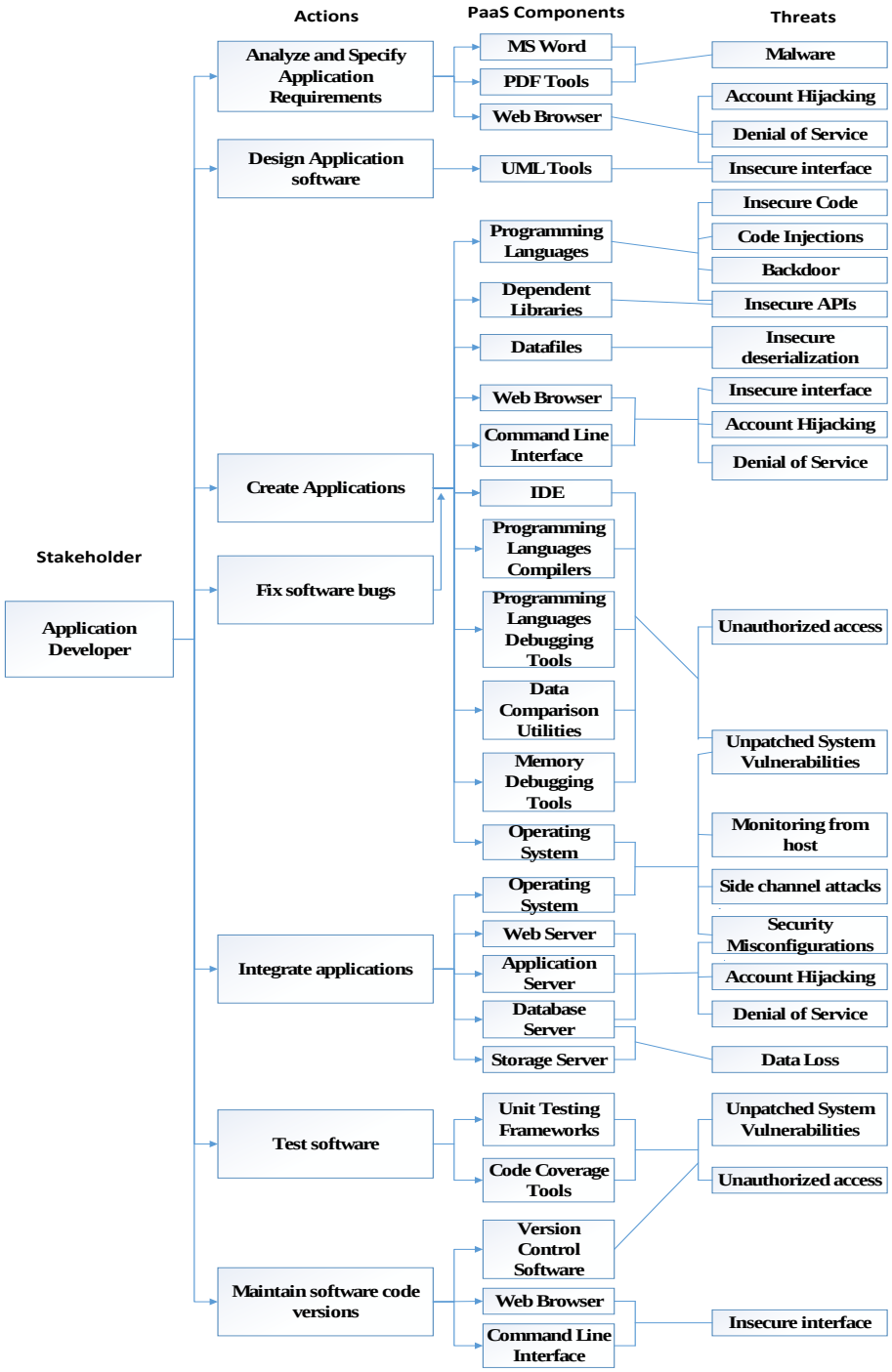
b) *PDF Tools:* For creating and saving application requirements and application design documents, the application developer uses PDF tools like PDF editors etc.

c) *Web Browser:* Web browser is the most important PaaS cloud component used by all the PaaS cloud stakeholders including the application developer, as it is the most used user interface to connect to all other PaaS components.

d) *UML (Unified Modeling Language) tools:* A Unified Modeling Language is a modeling language that is used to visualize the design of a system. A UML diagram is a diagram based on UML to show the system design and its main actors, roles, actions, and classes. The application developer uses UML tools like Visual Paradigm and Lucidchart to create application software design diagrams describing the details of system architecture, its components, and interactions among those components.

Figure 4.1

Taxonomy for Application Developer



e) *Programming languages:* A programming language is a language that sends instructions to the computer to implement algorithms. The application developer uses programming languages like JavaScript, C, C++, Java, Python, C# to develop applications in the cloud.

f) *Dependent libraries:* If a software program needs to use modules from a library or if a library needs to use resources from another library, a dependency is added on the required library. When developing and deploying applications in the cloud, the application developer might need to use dependent libraries and modules to incorporate the functionalities provided by those libraries in his application.

g) *Data files:* Data files such as JSON, yaml and csv files are used to store structured data. The application developer uses these files to send structured data between multiple processes and across multiple nodes.

h) *Command Line Interface (CLI):* Another user interface to connect to PaaS components is through command line interface. The application developer can use command line interface to connect to hosts and different tools used during the application development and maintenance process.

i) *IDE (Integrated Development Environment):* An IDE is a software application that provides features for code development, testing and debugging. The integrated toolset simplifies software development and increases the application developer’s productivity. There are several IDEs that can be used by the application developer such as Microsoft Visual Studio, Eclipse, PyCharm,

NetBeans, depending on the programming language used for software development.

j) *Programming language Compilers:* A compiler is a software that converts programs written in high level language into machine code that can be understood by a computer (Bourque et al., 2014). The main tasks of a compiler are preprocessing, lexical analysis, syntax analysis, semantic analysis, code generation and code optimization. The application developer uses compilers such as g++, Clang, LLVM, depending on the programming language used for software development.

k) *Programming language debugging tools:* A debugging tool is a software used to test and debug programs. The application developer uses a debugging tool to run its software program under controlled conditions, and track and monitor the operations performed by the program. Some of the commonly used debuggers are GDB, JDB, undoDB and DDD.

l) *Data comparison utilities:* Data comparison utilities such as Beyond Compare and WinMerge, are used to compare data and determines changes between code versions. The application developer uses these utilities to do side-by-side comparison of code versions, merge code changes and synchronize files during application development and maintenance.

m) *Memory debugging tools:* A memory debugging tool is a software used to find memory related errors such as memory leaks and buffer overflow in the application source code. These tools monitor memory allocation, memory access

and memory deallocation. The application developer uses these memory debugging tools to find and fix memory related issues in the application. Some of the memory debugging tools are Valgrind, IBM Purify and ElectricFence.

n) *Operating System:* Operating system is a collection of software and firmware, and it controls the execution of software programs. The operating system manages processing, memory, file system and input/output devices (Bourque et al., 2014). The application developer creates applications that runs on specific operating systems such as Windows, Linux and MacOS.

o) *Application Server:* An application server is a server that hosts web applications. The application developer uses application server and application server framework to create and run web applications.

p) *Web Server:* A web server is a server that stores, process and deliver web pages to clients. Some of the commonly used web servers are IIS, Apache and nginx. A webserver handles HTTP requests. The application developer uses webservers to deploy websites or web pages.

q) *Database Server:* A database server is used to store and manage database and provide access to the stored data to authorized users and applications. The application developer interacts with database server that is used to store application and user data.

r) *Storage Server:* A storage server is used to store and secure digital data and files over a shared network. The applications developed by the application developer might use storage server to store application data. The application

developer can also use storage server to save other application specific data and files.

s) *Unit test frameworks:* After creating an application, the application developer performs unit testing to test the application. Unit testing is a type of software testing in which individual components of a software are tested. Unit testing is performed by application developers during application development phase. Unit test frameworks are software tools that help in automating unit testing. The application developer uses unit test frameworks to perform unit testing of the software. Some of the commonly used unit test frameworks are NUnit, Junit and Embunit.

t) *Code coverage tools:* Code coverage tools are the tools that uses criteria such as function coverage, statement coverage, branches coverage, condition coverage and lines coverage to determine how much of the code is executed during testing. While testing the software, the application developer uses code coverage tools to understand the extent to which the source code of the software has been tested. Some of the commonly used code coverage tools are Bullseye, Cobertura and coverage.py.

u) *Version control software:* Version control systems help development teams to manage and track changes to the source code over time. The application developer uses version control systems such as GitHub, svn and ClearCase to manage different versions of the source code of the application.

The taxonomy includes two types of threats. First, the threats that can prevent the application developer from performing the identified actions or from interacting securely with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment. The taxonomy also includes the threats that can prevent an end user of the developed application from using the application securely in a cloud environment. The details of these threats are as follows:

a) *Malware:* Malware is a malicious software designed to cause damage or to gain unauthorized access to computer systems, networks, and data. The different types of malware are viruses, worms, trojans, ransomwares, spywares, adware etc. Malwares are used by cybercriminals to gain access to sensitive personal and business data. Microsoft Word and Adobe PDF have macro and scripting capabilities that allow them to install code on users' computer. A macro virus is a type of virus that embeds malicious code in the macros of documents and spreadsheets. In a PaaS cloud environment, the application developer uses Microsoft Word and Adobe PDF documents during requirements analysis and specifications. These documents could be infected with malware. These infected documents could be used to spread malware on users' computer and other applications in the PaaS cloud environment.

b) *Account Hijacking:* Account hijacking is a type of identity theft in which attackers gain access to a user's access credentials to a computer or application (Tirumala et al., 2015). It is one of the top security threats in cloud computing. Using account hijacking, cybercriminals can compromise the confidentiality, integrity,

and availability of cloud services. Cybercriminals use a variety of strategies such as phishing emails, popups, and fraud to steal user's access credentials. In PaaS cloud computing, the application developer uses web browser and command line interface to connect to hosts and cloud services. The application developer's credentials could be stolen and used to gain access to confidential data.

c) *Denial of Service (DoS):* Denial of Service (DoS) attack is an attack in which an attacker makes a computer system or network resources inaccessible to legitimate users. In a Distributed denial of service (DDoS) attack, an attacker compromises multiple hosts and use them to attack target host or network. A DoS or DDoS attack is performed by flooding the target host or network with traffic until the target crashes or cannot respond, preventing access to legitimate users (Darwish et al., 2013). DDoS attacks are one of the biggest security concerns in cloud infrastructure. DDoS attacks affect all the service models of cloud computing – IaaS, PaaS, and SaaS. DDoS attacks can be launched from outside the cloud environment or from inside the cloud environment. A DoS attack on any of the PaaS cloud components such as host, software tools, application server, web server, database server etc. will prevent the application developer from performing his tasks on time.

d) *Insecure interfaces and APIs:* In cloud computing, Application Programming Interfaces (APIs) provide communications between different cloud services (Kazim & Ying, 2015). The security of these cloud services depends on the security of the APIs used to communicate with these cloud services. Insecure

APIs and interfaces can be used by attackers to gain access to critical information in the cloud. Lack of robust identity and access management policies can further increase the security risk. In a PaaS cloud environment, the application developer uses APIs and software interfaces to communicate with different cloud services. These insecure interfaces and APIs prevents the application developer from creating secure applications.

e) *Insecure code:* The application software created by the application developer can consist of flaws in the code that can be exploited by attackers to compromise the security of the application and its data. These code vulnerabilities in the application can be exploited in attacks such as Cross-Site scripting, SQL injection, buffer overflow, Cross Site Request Forgery and many more. In a PaaS cloud environment, an insecure code written by the application developer will create vulnerable applications that can be easily attacked by cybercriminals.

f) *Code injections:* Code injection is a type of attack in which injected code is executed by the application. Code injections occur when an application sends untrusted data to the interpreter (*Code Injection Software Attack | OWASP Foundation*, 2020). These attacks happen due to lack of input/output data validation. Code injections are used by attackers to gain unauthorized access to a system or information, privilege escalation to root permissions, installing malware and modifying values in databases. In a PaaS cloud environment, if the application developed by the application developer does not handle input/output data validations properly, the application will be vulnerable to code injection

attacks. The users of the application will not be able to use the application securely. The confidentiality, integrity and availability of the application and critical application data can be easily compromised by attackers.

g) *Backdoor:* A backdoor is a method by which users gain access to a computer or application by bypassing normal authentication mechanisms. The application developer while creating the application can hardcode username and password in the code to quickly gain access to the system without normal authentication process. These backdoors can be exploited by cybercriminals to steal critical application and user data, install malware and hijack applications.

h) *Insecure deserialization:* Serialization is the process of converting object into a data format that can be restored later. Objects are serialized to save them on to storage, send them to multiple processes or to send them over the network. JSON, yaml and XML are the most used serialization formats in web applications. Deserialization is the process of converting the received serialized data back into an object. Insecure deserialization is a vulnerability in which untrusted data supplied by an attacker is deserialized by the application (*Insecure Deserialization* | *OWASP*, 2017). This vulnerability can be used to launch a denial of service attack or remote code execution attack on the application.

i) *Unauthorized access:* Unauthorized access means when someone gains access to a computer, server, or application without permissions. Unauthorized access of systems or applications is possible due to weak authentication mechanisms,

weak passwords, compromised accounts, malicious insiders, and social engineering. Unauthorized access of computers, applications or critical information by cybercriminals leads to the compromise of confidentiality, integrity and availability of systems, applications, and critical business information. In a PaaS cloud environment, the application developer uses several software tools such as debugging tools, memory debug tools, data comparison utilities, unit test frameworks, code coverage tools, version control tools during the development and maintenance of the applications. Unauthorized access by cybercriminals to any of these software tools prevents the application developer from performing his tasks and to create secure applications.

j) *Unpatched system vulnerabilities:* Servers and applications should be updated regularly by installing latest patches and versions. These patches contain fix for previous bugs and any new vulnerabilities. Unpatched systems and applications are a major risk to any organization. It is one of the top reasons of a successful intrusion by attackers in an organization. The application developer interacts with many software tools, hosts and services during the development and maintenance of applications in a PaaS cloud environment. An unpatched vulnerability in any of these hosts, services or software can compromise the security of the PaaS cloud environment.

k) *Security Misconfigurations:* Security misconfigurations happen when security settings are not implemented, and default settings are used. These misconfigurations are easy to detect and exploit. According to OWASP, security

misconfigurations is at number 6 in top 10 web application security threats (*Security Misconfiguration* | *OWASP*, 2017). An application is vulnerable if unnecessary features are enabled, disabled, or not configured properly on cloud services, or security settings of webservers, databases, operating systems etc. are not set to secure values. Attackers use these security misconfigurations to gain unauthorized access to applications, systems, and critical data. In a PaaS cloud environment, misconfigurations at any of the operating system, application server, database, webserver, or software tools can result in the compromise of the security of the PaaS cloud environment and the application being developed.

l) *Data Loss:* Data loss is the inability to access data or to lose control of the data. This can occur due to accidental or intentional deletion or overwriting of files, corrupted files due to abnormal device operation or damage, failed upgrades, software bugs, malwares, and storage device failures. In a PaaS cloud environment, data loss can happen on the database server and storage server, due to any of the reasons mentioned above. This data loss can prevent the application developer from performing his tasks.

m) *Monitoring from host:* In cloud computing, if the security of the host system is compromised then the security of the tenant's applications and data can easily be compromised. Attackers can passively monitor the tenant applications and its critical data from the compromised host.

n) *Side-channel attacks:* The multi-tenancy and shared resources properties of cloud computing introduces new threats from other customer's virtual machines

located on the same physical host. A side channel is a hidden channel created between two virtual machines on the same physical host. Any information obtained using this hidden channel is called side-channel information. An attack that benefits from side-channel information is known as a side-channel attack (Bazm et al., 2017). In a passive side-channel attack, an attacker continuously monitors and read information from target virtual machine collocated with attacker's virtual machine on the same physical host. Active side-channel attack forces the target virtual machine to perform abnormal operations. An attacker can use timing side-channel attack and cache-based side-channel attack to attack the target virtual machine. A side-channel attack can compromise the security of the PaaS cloud environment and the application being developed.

Taxonomy for Application Tester

An application tester is a person who evaluates and assesses applications to ensure that they function correctly. To accomplish this, the application tester creates and executes test cases to validate the application's functionality, usability, and consistency. The main role of the application tester is to find errors in the application and get them fixed. Application testing helps to create robust software applications.

The actions performed by the application tester in a PaaS cloud environment are:

a) *Analyze Application Requirements:* Application tester studies and analyzes functional and non-functional requirements of the application and understands the expected outputs to find testable requirements and defines them properly for test planning. It helps to identify the scope of the testing.

b) *Create test strategy:* A test strategy includes the tools needed, testing steps, roles and responsibilities, risk analysis, cost analysis and timelines for testing. The application tester helps in creating test strategy for testing the application in a PaaS cloud environment.

c) *Create test cases:* A test case defines the test procedures, test inputs, execution steps and expected results. The application tester creates test cases to test the application developed in a PaaS cloud environment.

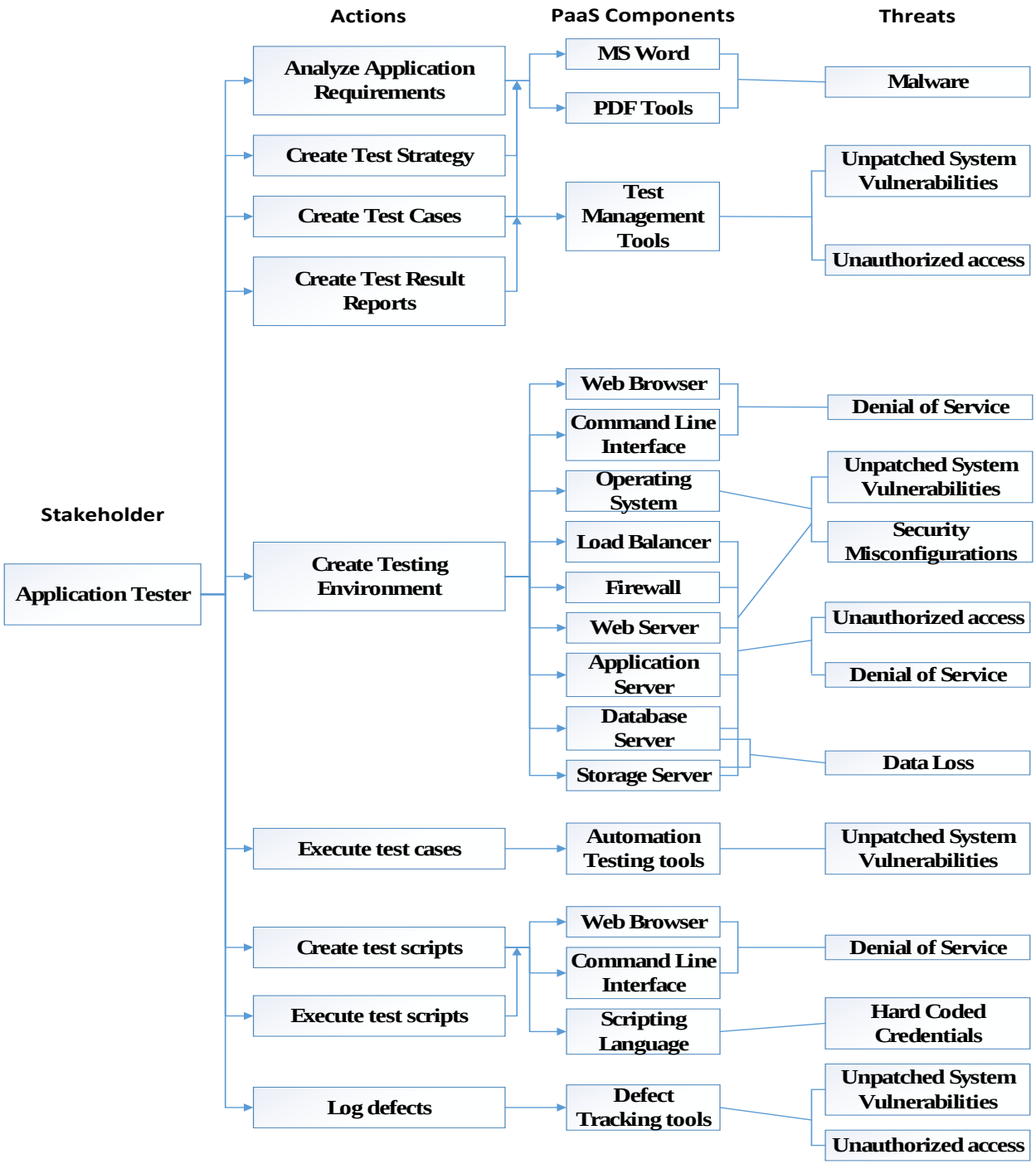
d) *Create testing environment:* To test the software applications, a testing environment needs to be created and configured. It includes installing and configuring the test servers and various testing tools. The application tester creates a testing environment to test the applications developed in a PaaS cloud environment.

e) *Execute test cases:* The application tester executes the test cases to test the features of the application in the testing environment created earlier. The actual test results are then compared to the expected results.

f) *Create test result reports:* After executing the test cases, a test results report is prepared. This report has details of the testing process, testing environment, and comparisons of actual and expected test results. The application tester creates the test results report after executing the test cases in a testing environment.

Figure 4.2

Taxonomy for Application Tester



g) *Create test scripts*: A test script is a set of instructions that can be run automatically to test the application in the testing environment. The application tester creates test scripts to test applications more efficiently.

h) *Execute test scripts:* The application tester executes the test scripts to test the applications and compare results of the test scripts with expected test results.

i) *Log defects:* After executing the test cases or test scripts, the application tester compares the actual test results with expected results. If the actual result is different from expected result in a test case, the application tester logs a defect for the corresponding application functionality mentioning the details of the error. These defects are sent to the application development team for fixing.

The PaaS cloud components that the application tester interacts with to perform the identified actions are:

a) *Microsoft Word:* The application tester creates documents during application requirement analysis, test strategy creation, test case writing and test results reporting. For creating these documents, the application tester uses Microsoft Word.

b) *PDF Tools:* PDF tools such as PDF editors are used by the application tester to create, edit, and save test requirements documents, test case documents and test results report documents.

c) *Test Management tools:* Test management tools help in planning, monitoring, and reporting test executions. These tools help in capturing requirements for testing, designing test cases, reporting test executions, and monitoring the

testing progress. The application tester uses these tools during testing the applications in a PaaS cloud environment. Some of the commonly used test management tools are qTest, Zephyr and Xray.

d) *Web Browser:* Web browser is the most used user interface to connect to cloud components. The application tester uses web browser to connect to different PaaS cloud components.

e) *Command Line Interface:* Command line interface (CLI) is another interface used to connect to cloud components such as hosts, applications, and software tools. The application tester also uses command line interface to connect to test servers and testing tools in a PaaS cloud environment.

f) *Operating System:* The applications that are tested by the application tester in a PaaS cloud environment runs on specific operating systems such as Windows, MacOS, Linux and Solaris. The application tester interacts with the operating system to run and test those applications.

g) *Load Balancer:* A load balancer is a device that distributes network and application traffic to a number of servers. It is used to increase the performance and reliability of applications. To test the performance and reliability of the applications developed in a PaaS cloud environment, the application tester uses a load balancer in the testing environment while executing the test cases.

h) *Firewalls:* A firewall is a network security device that monitors incoming and outgoing network traffic. It filters traffic based on a set of security rules. The

application tester uses a firewall as part of the testing environment to test applications in a PaaS cloud environment.

i) *Web Server:* A web server is used to host webpages. The application tester might be using webservers as part of the testing environment to test applications in a PaaS cloud environment.

j) *Application Server:* An application server is used to host web applications. The application tester uses application server to create a testing environment to test applications in a PaaS cloud environment. This testing environment is used to execute the test cases.

k) *Database Server:* A database server is used to store application and user data. The application tester uses database server in the testing environment and interacts with it while executing the test cases to test cloud applications in a PaaS cloud environment.

l) *Storage Server:* Storage server is used to store application specific data and files. The application tester uses storage server to save files and application specific data in a PaaS cloud environment.

m) *Automation Testing tools:* Automation testing tools are used to execute test cases automatically and produce test results. The application tester can use a variety of automation testing tools to test cloud applications. These tools can be used to perform load testing, performance testing, security testing, regression testing and many more.

n) *Scripting Language:* A scripting language is a programming language used for writing scripts. Scripts are created to automate the testing process. The application tester uses scripting languages to write scripts to automate the application testing in a PaaS cloud environment.

o) *Defect tracking tools:* The application tester logs defects for any of the errors found in applications while testing. To log these defects with detailed information, the application tester uses defect tracking tools. These tools help the testing team to report, manage, track, and resolve defects. Some of the commonly used defect tracking tools are JIRA, Bugzilla and zipBoard.

The threats that can prevent the application tester from performing the identified actions or from interacting with the identified PaaS cloud components on time, within budget and with quality are as follows:

a) *Malware:* The application tester uses Microsoft Word and Adobe PDF documents during requirements analysis, creating test cases and test reports. These documents could have embedded malwares that can harm the application tester's computer and other PaaS cloud components of the testing environment.

b) *Unpatched system vulnerabilities:* The application tester uses test management tools, automation testing tools, defect tracking tools, and the different types of servers such as webserver, application server, database server, firewalls etc. in a PaaS cloud testing environment to test applications. An unpatched vulnerability in any of these tools or servers can compromise the security of the testing environment and the application being tested.

c) *Unauthorized access:* The application tester interacts with multiple testing tools and test servers during the testing of cloud applications in a PaaS cloud environment. Unauthorized access by cybercriminals to any of these tools and servers can compromise the security of PaaS cloud environment and prevents the application tester from performing his tasks.

d) *Denial of Service:* Denial of service (DoS) attacks are a major concern in cloud computing. A DoS attack on any of the PaaS cloud component can prevent the application tester to connect to that cloud component through web browser or command line interface and perform his tasks on time.

e) *Data loss:* In a PaaS cloud environment, data loss from the database server or storage server of the testing environment can prevent the application tester from testing the cloud applications.

f) *Security Misconfigurations:* The application tester interacts with different types of servers such as webserver, application server, database server, firewalls, load balancers, and operating system while creating a testing environment and executing test cases to test the applications developed in a PaaS cloud environment. If the security configurations on any of these components are not implemented or are not set to secure values, attackers can use these security misconfigurations to gain unauthorized access to the testing environment. The attacker can compromise the integrity of the testing environment and the application being tested.

g) *Hard coded credentials:* Hard coded credentials, also known as embedded credentials, are plain text credentials in the source code. The application tester can include hard coded credentials in test scripts. These hard-coded credentials can be exploited by cybercriminals to get unauthorized access to the PaaS cloud components. This can compromise the security of the PaaS cloud environment and the application being developed.

Taxonomy for DevOps Engineer

DevOps is a software development strategy that integrates software development, deployment, and IT operations, using automated development and deployment, with the aim to develop software quickly and provide continuous delivery incorporating feedback. A DevOps engineer is a software professional who works with the software developers and IT operations staff to facilitate code releases and deployments.

The actions performed by the DevOps engineer in a PaaS cloud environment are as follows:

a) *Integrate new functionalities*: In a DevOps software development, DevOps engineers regularly merge their code changes to a central repository to add new functionalities or to provide fix for previous bugs. This software development practice is known as Continuous Integration (CI).

b) *Create packages:* After the code is integrated, the DevOps engineer reviews the code changes and package them into an artifact. The package is then moved to a production environment for deployment.

c) *Create deployment specification documents:* A deployment specification document consists of the details of the deployment along with the different components' details. The DevOps engineer is responsible for creating deployment specifications documents.

d) *Install and Configure applications:* The DevOps engineer is responsible for the installation and configuration of applications in the cloud. This is done through automated deployment tools. It is also known as Continuous Deployment (CD).

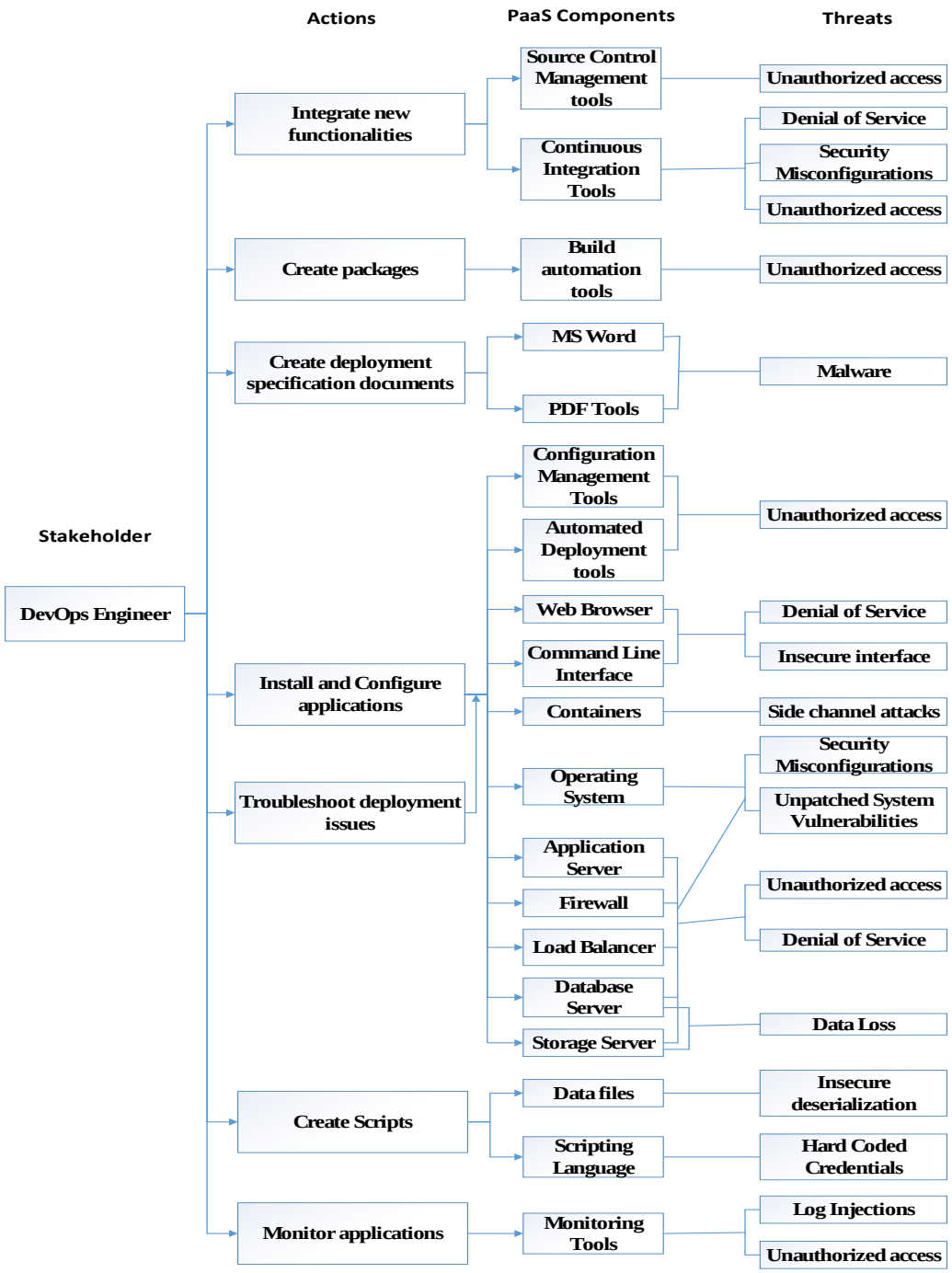
e) *Troubleshoot deployment issues:* The DevOps engineer is responsible for actively troubleshooting any issues during testing and deployment of applications and resolving the issues.

f) *Create Scripts:* The DevOps engineer is responsible for creating scripts to automate the different operational processes.

g) *Monitor applications:* Applications must be monitored continuously to detect any performance, compliance, and security issues. It helps to identify the cause of the error and maintain the security and availability of the applications. Continuous monitoring is done using automated tools. The DevOps engineer is responsible for monitoring applications in the cloud.

Figure 4.3

Taxonomy for DevOps Engineer



The PaaS cloud components that the DevOps engineer interacts with to perform the identified actions are:

a) *Source code management tools:* Source code control or version control is the process of tracking and managing changes to the source code. A source code management tool is a software tool that helps the DevOps team to manage changes to the source code. The DevOps engineer uses source code management tools such as Git, Subversion etc. to maintain the different versions of the source code. In the Continuous Development phase of DevOps lifecycle, a source code management tool helps in building stable versions of the application code.

b) *Continuous Integration tools:* CI tools are used to build projects, run tests, code analysis, code coverage, bug detection and many other features. The DevOps engineer uses these CI tools while integrating his code to the repository. Some of the commonly used CI tools are Jenkins, AWS CodePipeline, and Azure Pipelines.

c) *Build automation tools:* Build automation tools are used by the DevOps engineer to build packages from source code. Some of the commonly used build tools are Ant, Maven and Gradle.

d) *Microsoft Word:* To create deployment specifications documents and other documents, the DevOps engineer uses Microsoft Word.

e) *PDF Tools:* The DevOps engineer uses PDF tools to edit and save documents.

f) *Configuration Management tools:* Configuration management is the process of managing configurations of the systems and applications in an environment. Configuration management tools help in establishing and maintaining consistency and stability of systems and applications. The DevOps engineer uses configuration management tools while deploying applications in the cloud. Some of the commonly used configuration management tools are Ansible, Puppet and Chef.

g) *Automated deployment tools:* The DevOps engineer uses automated deployment tools such as Jenkins, AWS CodeDeploy, Octopus Deploy etc. to automate the deployment of applications in the cloud environment.

h) *Web browser:* The DevOps engineer connects to different PaaS cloud components using a web browser.

i) *Command line interface:* Command line interface is also used by the DevOps engineer to connect to hosts and services in a PaaS cloud environment.

j) *Containers:* Containers are operating system virtualization technology that packages all the necessary binaries, executables, and configuration files, and runs the application in an isolated environment. The DevOps engineer uses containers in the deployment of applications as they help in maintaining consistency of applications across the development, testing and deployment environments.

k) *Operating System:* The applications that the DevOps engineer deploy in the cloud runs on a specific operating system. The DevOps engineer interacts with the operating system during the deployment of applications in the cloud.

l) *Application server:* Application server is used to host applications in the cloud. The DevOps engineer interacts with the application server while deploying applications in the cloud.

m) *Firewalls:* A firewall is used to control traffic to the deployed applications in the cloud. The DevOps engineer configures the firewall to allow or deny traffic to the deployed applications from the outside network.

n) *Load Balancer:* The DevOps engineer uses a load balancer to distribute the incoming application traffic to several application servers in the cloud. It increases the performance and reliability of the deployed applications in the cloud.

o) *Database server:* A database server is used to store application and user data. The DevOps engineer interacts with the database server when deploying applications in the cloud.

p) *Storage server:* The DevOps engineer uses a storage server to save application specific data and files of the deployed applications.

q) *Data files:* The DevOps engineer uses data files such as JSON, yaml and csv files to store configuration variables to be used by other tasks in automated processes.

r) *Scripting language:* The DevOps engineer creates scripts to automate processes. Scripting languages such as Python, bash, and Ruby are used to create those scripts.

s) *Monitoring tools:* Monitoring tools are the software tools used to continuously track the status of the servers and applications in order to find any failures, defects, or problems so that those can be resolved as soon as possible. The DevOps engineer uses monitoring tools such as Splunk, ELK Stack and Nagios to continuously monitor applications in the cloud to ensure that they are working properly.

The threats that can prevent the DevOps engineer from performing the identified actions or interacting with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment are as follows:

a) *Unauthorized access:* The DevOps engineer uses multiple software tools during continuous integration, continuous deployment, and continuous monitoring of an application in the cloud. These tools include build automation tools, source code management tools, continuous integration tools, configuration management tools, automated deployment tools and monitoring tools. Unauthorized access by attackers to any of these tools or servers in the application deployment environment will prevent the DevOps engineer to create and deploy secure applications in the cloud.

b) *Denial of Service:* A DoS attack on any of the PaaS cloud components prevents the DevOps engineer to connect to that cloud component through web browser

or command line interface and perform his tasks on time. Jenkins, one of the popular CI/CD tool used by DevOps engineers can be abused to launch a distributed denial of service attack (Cimpanu, 2020).

c) *Security Misconfigurations:* Security misconfiguration is defined as failing to properly configure security controls for a server or a software tool. The DevOps engineer configures many types of servers, databases, firewalls and uses many software tools in the continuous integration and deployment of applications in the cloud. If any of these servers, firewalls or software tools are not configured properly, it can compromise the security of the deployed application and the data it handles.

d) *Malware:* The DevOps engineer creates deployment specification documents and uses Microsoft Word and PDF editors to create and save those documents. These documents could be infected with malware. These infected documents can spread malware to other PaaS cloud components.

e) *Insecure interface:* The DevOps engineer uses software interfaces and application programming interfaces (APIs) to communicate and integrate different cloud services. Insecure interfaces and APIs can be used by attackers to attack PaaS cloud services and applications and will prevent the DevOps engineer to create secure deployments of applications in the cloud.

f) *Side channel attacks:* The use of containers in the automated deployment of applications helps the DevOps engineer to increase the agility and speed of application development and deployment. But if the containers are not isolated

properly, a side-channel attack can be launched from another container residing on the same physical host in the cloud. A side-channel attack can compromise the security of the application running in the victim container.

g) *Unpatched system vulnerabilities:* Systems with unpatched vulnerabilities are a major risk to the security of the cloud environment. If any of the application server, firewall, databases, or operating system being used have an unpatched vulnerability, it can be exploited by attackers and further compromises the security of the deployed application.

h) Data loss: An application uses database server and storage server to store sensitive application and user data. Inability to access this data or to lose control of this data results in data loss. Data loss from any of these servers can prevent the application to function properly and securely.

i) *Insecure deserialization:* The DevOps engineer uses data files to save configuration variables in structured data format. These data files could be targets of insecure deserialization in which untrusted data is provided by an attacker, and this data is deserialized by the application. This untrusted data can further be used for remote code execution or denial of service attacks on the application.

j) *Hard coded credentials:* During the creation of scripts, the DevOps engineer can include hard coded credentials to connect to servers or applications. These scripts with hard coded credentials can be a target of cybercriminals to gain unauthorized access to PaaS cloud servers and applications.

k) *Log injections:* Applications use log files to maintain a history of events and transactions. These logs are used for debugging, data collection and optimizing application performance. The DevOps engineer uses automated monitoring tools to monitor these application logs. Log injection is a vulnerability which allows untrusted data to be written to a log file. An attacker can insert malicious data and false events into the logs. An attacker can also delete logs to cover the tracks of an intrusion. These forged logs can easily fail the purpose of monitoring the application.

Taxonomy for Application Administrator

An application administrator is an IT professional responsible for ensuring that applications used by an organization are functioning properly.

The actions performed by the application administrator in a PaaS cloud environment are as follows:

a) *Install and Configure applications*: The application administrator is responsible for preparing servers needed to deploy the applications in the cloud. He is also responsible for installing and configuring applications in the cloud.

b) *Update applications:* An application should be updated regularly by applying software patches and installing latest stable versions of the application. The application administrator creates maintenance plans for the administered applications and updates the applications regularly by installing latest patches and versions.

c) *Resolve incidents:* The application administrator responds to queries and issues in a timely manner. He is also responsible for resolving incidents and providing technical support to end users, troubleshooting their problems, and resolving them.

d) *Troubleshoot issues:* The application administrator continuously monitors the applications for any issues and proactively troubleshoots the issues to identify the root cause and implements corrective actions to resolve them.

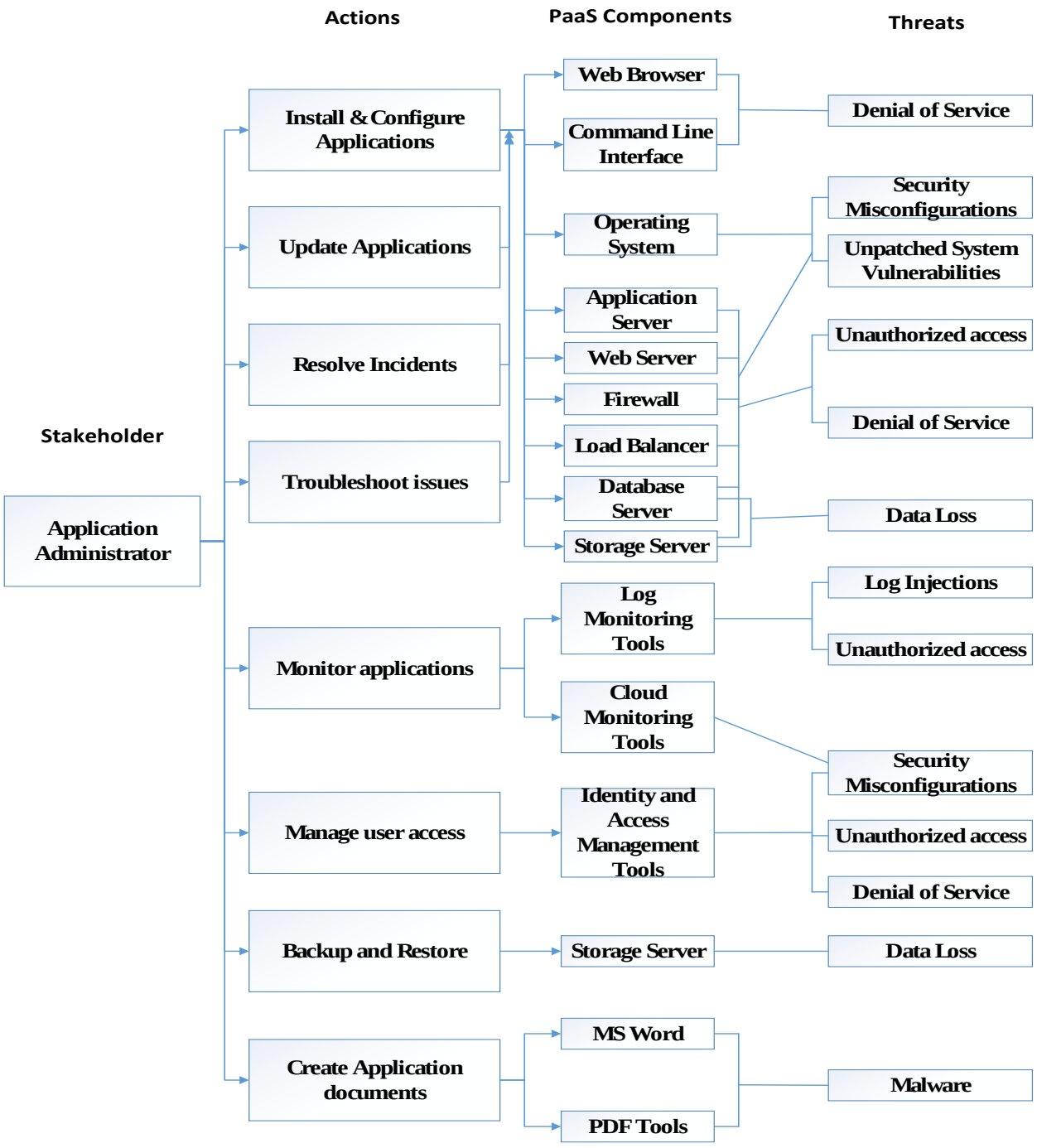
e) *Monitor applications:* The application administrator monitors applications daily to ensure the performance, reliability, and security of the applications.

f) *Manage user access:* The application administrator creates and manages the administrator and user access accounts of the application and provides appropriate authorization levels to these accounts. He regularly audits these accounts and deletes old user accounts and removes all access rights of that user account on the application.

g) *Backup and Restore:* The application administrator is responsible for creating application and user data backups and saving those backups on a storage server. These backups are used to restore the application in case the application crashes. The application administrator is also responsible for restoring the application after an application crashes by using the backup data.

Figure 4.4

Taxonomy for Application Administrator



h) *Create application documents:* The application administrator is responsible for creating and maintaining technical documentation for the application installation deployment procedures, standard administrative operating procedures, technical support procedures, and application issues and their resolutions.

The PaaS cloud components that the application administrator interacts with to perform the identified actions are:

a) *Web browser:* Web browser is the most used user interface to connect to all the PaaS cloud components. The application administrator uses web browser to connect to PaaS cloud services and cloud servers during the setup and maintenance of cloud applications.

b) *Command line interface:* Most of the PaaS cloud components also provide a command line interface to connect to them. The application administrator uses command line interface to connect to PaaS cloud servers and software tools.

c) *Operating System:* The applications that are installed and configured by the application administrator runs on a specific operating system. The application administrator interacts with the operating system while installing and maintaining these applications in the PaaS cloud environment.

d) *Application server:* The application administrator uses application server to install applications in the PaaS cloud environment.

e) *Web server:* The application administrator might use a web server while installing and configuring applications in the PaaS cloud environment.

f) *Firewalls:* Firewalls are used by the application administrator to allow or deny specific traffic to the administered applications as per the security rules.

g) *Load Balancer:* The application administrator uses a load balancer to distribute application traffic to multiple application servers in the cloud and increase the performance and availability of the application.

h) *Storage server:* The application administrator uses a storage server to store and secure application data and backups.

i) *Database server:* The application administrator uses a database server to store application and user data while installing and maintaining applications in the cloud.

j) *Monitoring tools:* The application administrator uses monitoring tools such as Splunk, ELK Stack and Nagios to continuously monitor servers and applications in the cloud.

k) *Cloud Monitoring tools:* Cloud monitoring tools are used to monitor and manage cloud computing resources and applications. It helps administrators to identify emerging defects and unexpected patterns to prevent minor issues from turning into major problems. The application administrator uses these tools to monitor the overall health and performance of applications. Some of the most used cloud application monitoring tools are Amazon Cloudwatch, Microsoft Cloud Monitoring and AppDynamics.

l) *Identity and Access Management tools:* To manage identities and access privileges to the applications, the application administrator uses identity and

access management (IAM) tools. These tools control access to protected data and applications, prevent data transmissions, implement multi-factor authentications (MFA), single sign-on (SSO) and password management. Some of the most used IAM solutions are Auth0, Ping Identity, Okta and AWS Identity & Access Management.

m) *MS Word:* The application administrator creates and maintains technical documents for the application installation and maintenance procedures. For creating these documents, the application administrator uses Microsoft Word.

n) *PDF tools:* The application administrator uses PDF editors to create, edit and save application installation and maintenance documents.

The threats that can prevent the application administrator from performing the identified actions or interacting with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment are as follows:

a) Denial of Service: A DoS attack on any of the PaaS cloud components can prevent the application administrator from connecting to the cloud component through web browser or command line interface and performing his tasks on time. A DoS attack on the identity and access management (IAM) solution will prevent users to access the application.

b) *Security Misconfigurations:* Security misconfiguration is one of the major concerns while configuring cloud resources and applications. The application administrator installs and configures servers and software tools such as log monitoring tools, cloud monitoring tools and identity and access management

tools to monitor and maintain applications. Any misconfigurations during configuration of these resources can compromise the security of the administered application and data.

c) *Unpatched system vulnerabilities: :* The application administrator interacts with different servers and software tools while installing and maintaining applications in the cloud. An unpatched vulnerability in any of these resources can compromise the security of the cloud application.

d) *Unauthorized access:* The application administrator interacts with different types of servers and software tools for application monitoring and user access control. Unauthorized access by attackers to any of these servers or software tools can compromise the security of the administered application and critical data.

e) *Data Loss:* The application administrator uses database server and storage server while installing and maintaining applications. Storage server is also used to store application data and user data backups. Data loss or inability to access data from these servers prevents the application from working properly and also prevents the application administrator from restoring the application in case the application crashes.

f) *Log injections:* The application administrator monitors applications daily using log monitoring tools. These logs are also analyzed to further optimize application performance. In log injections, an attacker inserts false events, add malicious data, and delete logs from the log files. These modified logs will mislead the application analysis.

g) *Malware:* The application administrator uses Microsoft Word and Adobe PDF editor to create application documentation. These documents could have embedded malwares that can harm other PaaS cloud components.

Taxonomy for Database Administrator

A database administrator (DBA) is an IT professional responsible for creating and maintaining a database environment. The DBA ensures that the organization's database and its related applications are functioning properly and efficiently. The DBA also ensures that the data is available to users and secure from unauthorized access.

The actions performed by the database administrator in a PaaS cloud environment are as follows:

a) *Design database systems:* The database administrator collects and analyzes the user requirements to make conceptual and logical data models. A conceptual data model is a high-level description that outlines data requirements. A logical data model represents the characteristics and relationships of data. The database administrator translates these data models into actual database implementations.

b) *Install and Configure database systems:* The database administrator is responsible for installing and configuring the database software such as Oracle, SQL Server etc. on servers.

c) *Update database systems:* The database systems should be updated regularly by installing the new patches and versions of the database software. The

database administrator updates database software and server when new versions or patches are available.

d) *Monitor database systems:* The database administrator monitors databases for performance issues and ensures data availability. Performance monitoring helps determine where the database should be modified to operate more efficiently. The DBA makes changes to data structures, application logic and Database Management Systems (DBMS) to optimize database performance.

e) *Migrate databases:* Database migration is the process of moving data from one database to another or from DBMS to an external data set. The database administrator is responsible for migrating databases efficiently and securely as per organizational needs.

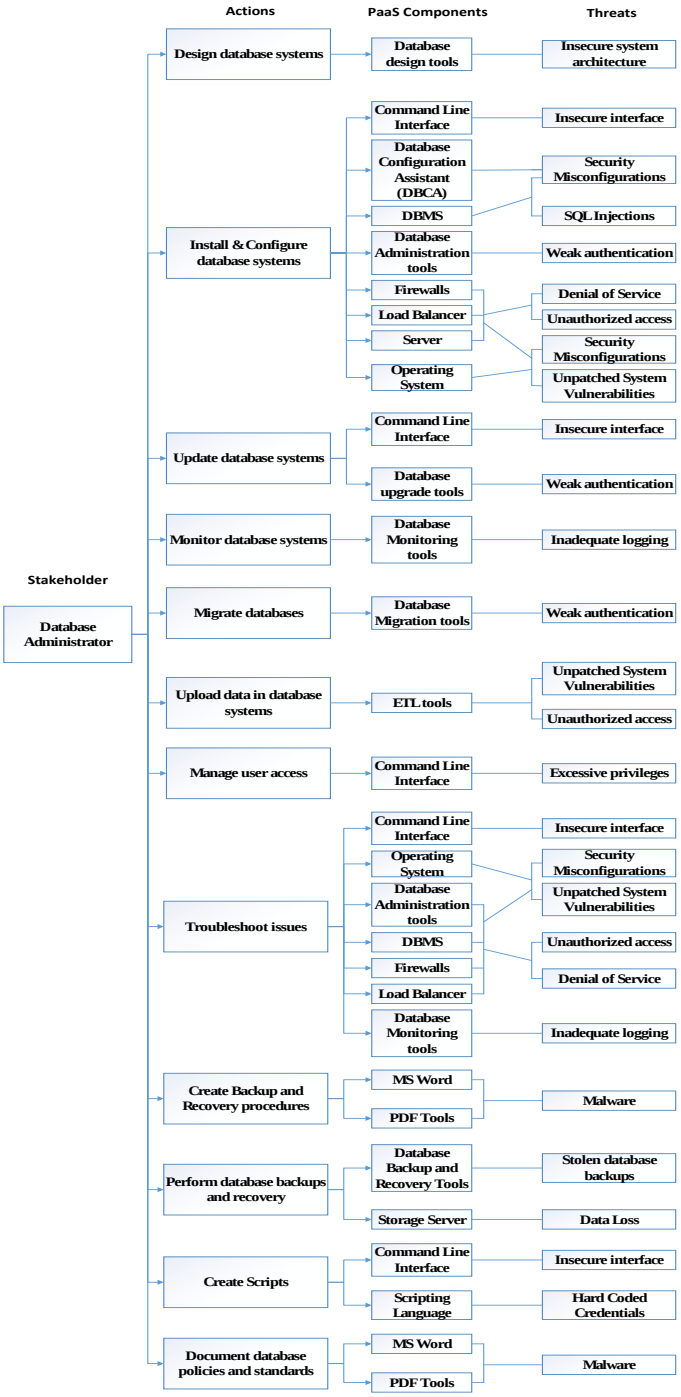
f) *Upload data in database systems:* The database administrator is responsible for efficiently importing large volumes of data from multiple systems into a data warehouse. This data is first extracted from multiple sources and then transformed into a desired format for import.

g) *Manage user access:* The database administrator creates and manages user profiles and user permissions to establish database access control levels.

h) *Troubleshoot issues:* The database administrator is responsible for troubleshooting database errors, resolving issues quickly to minimize damage and restoring lost data.

Figure 4.5

Taxonomy for Database Administrator



i) *Create backup and recovery procedures:* The database administrator creates backup and recovery plans and procedures based on the industry best practices and organizational needs.

j) *Perform database backups and recovery:* The database administrator performs regular backups of the database. In case of a database server failure or any other form of data loss, these backups are used by the database administrator to restore the database server. The DBA is also responsible for testing these backups at regular intervals to ensure that they can be used successfully to restore the database server and data when needed.

k) *Create scripts:* The database administrator is responsible for creating scripts to automate the operational and maintenance tasks related to the database and server.

l) *Document database policies and standards:* The database administrator is responsible for documenting database policies, procedures, and standards.

The PaaS cloud components that the database administrator interacts with to perform the identified actions are:

a) *Database design tools:* Database design tools help in designing database diagrams easily. They can be used to create a physical model or an Entity Relationship diagram (ERD) of the database. The database administrator uses these tools in designing database systems. Some of the popular database design tools are Lucidchart, Visual Paradigm ERD tools and Vertabelo.

b) *Command line interface:* The database administrator uses command line interface to install, upgrade and troubleshoot database systems. Command line interface is also used to create scripts and manage user access control of the database.

c) *Database configuration assistant (DBCA):* Database configuration assistant is a tool used to create, configure, and drop databases. The database administrator uses DBCA while creating and configuring databases.

d) *Database administration tools:* The database administrator uses database administration tools to automate database tasks such as looking up tables, find and replace or any other tasks to be run on the database. Some of the database administration tools are phpMyAdmin, MySQL WorkBench and SQL Server Management Studio.

e) *DBMS (Database Management Systems):* A DBMS is a software system designed to define, retrieve, modify, and manage data in a database. It interacts with users, applications, and the database itself to capture and analyze data. Some DBMS examples are Oracle, MySQL, and SQL Server. The database administrator installs and configures the DBMS on the database server.

f) *Firewalls:* Firewalls are used to control traffic to the database server. The database administrator interacts with the firewalls to allow or deny traffic to the database server on specific ports.

g) *Load Balancer:* A load balancer distributes incoming database traffic to several database servers. The database administrator uses a load balancer to increase the reliability and performance of the database.

h) *Server:* A server is used to install the database management system or DBMS such as Oracle or MySQL. This server will store the data and will be known as the database server.

i) *Operating system:* The database management system used to manage the database runs on an operating system and rely on the operating system to provide many functionalities. The database administrator interacts with the operating system while creating and managing database systems.

j) *Database upgrade tools:* Database upgrade tools are used to upgrade the database systems. The database administrator uses database upgrade tools to update the database system to a target version.

k) *Database monitoring tools:* The database administrator uses database monitoring tools to monitor performance of the database and check for anomalous behavior. Some of the database monitoring tools are SolarWinds Database Performance Analyzer, AppOptics APM and Datadog database monitoring.

l) *Database migration tools:* Database migration tools help in moving data from one type of database to another or to another destination or from on-premise to cloud. Some of the database migration tools are IBM InfoSphere, Apache NiFi,

Alooma and Snaplogic. The database administrator uses database migration tools to migrate the database as per organization needs.

m) *ETL tools:* ETL (Extract, Transform, Load) is the process of copying data from multiple sources, transforming data into proper format for querying and analysis, and then loading the data into the target database such as a data warehouse or a data store. The database administrator uses ETL tools to upload data to database systems.

n) *MS Word:* The database administrator uses Microsoft Word to create backup & recovery documents and database policies & standards documents.

o) *PDF tools:* The database administrator uses PDF editor to create, modify and save database documentation.

p) *Database backup & recovery tools:* Database backup solutions create and protect database copies. These database copies are used to restore the database in case of a database server failure or corrupted data. The database administrator uses backup and recovery tools to take database backups at regular intervals and to restore the database when needed. Some of these tools include NAKIVO backup & replication, dbForge Studio and SQL Backup Pro.

q) *Storage server:* The database administrator uses a storage server to save database backups.

r) *Scripting language:* The database administrator uses scripting languages such as bash and Python to create scripts.

The threats that can prevent the database administrator from performing the identified actions or interacting with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment are as follows:

a) *Insecure system architecture:* Database systems could be attacked due to the insecure database architecture and flaws in its features. Attackers find a particular weakness within the database architecture and use it to their advantage. If the database administrator creates a weak and poor database design, it could compromise the security of the applications using the database and the sensitive data stored in the database.

b) *Insecure interface:* The database administrator performs many of his tasks such as upgrading database system or managing user access by using the command line interface. Using a password on the command line interface to connect to the database can be insecure. It can compromise the security of the database.

c) *Security misconfigurations:* Cloud database configuration errors are a major reason for many data breaches. Misconfigurations at the database server or any of the other devices such as firewalls and load balancers can be used to compromise the security of the database server and the data. The database administrator configures database systems. Any misconfiguration in the operating system or any other server prevents him from creating a secure database system.

d) *SQL Injections:* In a SQL injection attack, an untrusted SQL query is inserted into the input data from the client to the application and these SQL commands are

executed on the database (*SQL Injection* | *OWASP*, 2020). It can be used to read sensitive data from the database, execute administrative actions on the database and to issue commands to the operating system. SQL injections are a critical problem for database administrators in the protection of enterprise databases.

e) *Weak authentication:* Weak authentication mechanisms in database tools such as administration tools, upgrade tools and migration tools, are a major security concern to the overall database systems. Weak authentication mechanisms prevent the database administrator from creating and maintaining a secure database system.

f) *Denial of service:* Denial of service attacks can also target database servers like any other server. The most common type of database denial of service attacks is launched by abusing database functions. The attacker sends requests to a database service until it reaches a threshold and collapses. There are multiple interdependent processes in a database, and crash of a single process can lead to the collapse of the entire database server. Attackers can also exploit complex database queries to launch a DoS attack. A few complex queries running simultaneously can consume a lot of database resources resulting in slowing or stopping the database. A DoS attack on the database can also be launched from the application using the database (A. Lane, 2013). A DoS attack against the database can prevent the database administrator from performing his tasks.

g) *Unauthorized access:* The database administrator interacts with different database tools and servers while creating and maintaining database systems. Unauthorized access by attackers to the database through any of these tools or servers will compromise the security of the database.

h) *Unpatched system vulnerabilities:* Unpatched systems is one of the significant attack vectors used by attackers. Leaving a vulnerable system unpatched can compromise the security of the application and the data. The database administrator interacts with database server and many tools to manage the database. An unpatched vulnerability in any of these tools or servers can compromise the security of the data and the database system.

i) *Excessive privileges:* If users have more privileges than needed for their job, these privileges can be misused by the user or by the attacker who compromises the user's account. If a user leaves the organization or move to a different role within the same organization, access rights of the user should be audited and deleted or updated as per the new job role. The database administrator manages user access for the database. If the database administrator provides excess privileges to users on the data, it could compromise the security of the database.

j) *Inadequate logging:* The database administrator monitors the database using database monitoring tools. Inadequate logging and poor auditing can defeat the purpose of database monitoring. Information collected at the application and database layers should be enough for investigating a suspected data

compromise. Inadequate logging prevents the database administrator from troubleshooting issues.

k) *Malware:* The database administrator creates database backup procedures, policies and standards using Microsoft Word and PDF editors. These documents could be infected with malwares and these malwares could be spread to other systems of the database infrastructure. These infected devices could be used to steal sensitive data from the database.

l) *Stolen database backups:* Anyone who has access to the sensitive data and the backup disks can steal it and misuse or sell the data.

m) *Data loss:* Database backups are stored at a storage server. Data loss from the storage server due to theft, server failure, deletion or overwriting of files, and malwares could compromise the security of sensitive data stored in the backups.

n) *Hard coded credentials:* The database administrator creates scripts to perform tasks on the database. If these scripts contain hard coded credentials to connect to the database, then these scripts and the credentials could be compromised by attackers to gain unauthorized access to the database.

Taxonomy for Project Manager

A project manager is responsible for initiating, planning, organizing, executing, monitoring, and closing projects on time, within budget, with quality and within scope for an organization.

The tasks performed by the project manager that can affect or be affected by the overall security of the PaaS cloud environment are:

a) *Manage project resources:* A project requires resources from various areas. Resources include people, hardware, software, and other assets. The project manager is responsible for the procurement, onboarding, and management of these resources.

b) *Manage project lifecycle:* A project lifecycle is a collection of project phases that defines the work to be performed in each phase, deliverables to be produced in each phase, resource requirements of each phase, and management control and approval process of work produced in each phase. The project manager is responsible for managing the project lifecycle along with the risks throughout the lifecycle of the project.

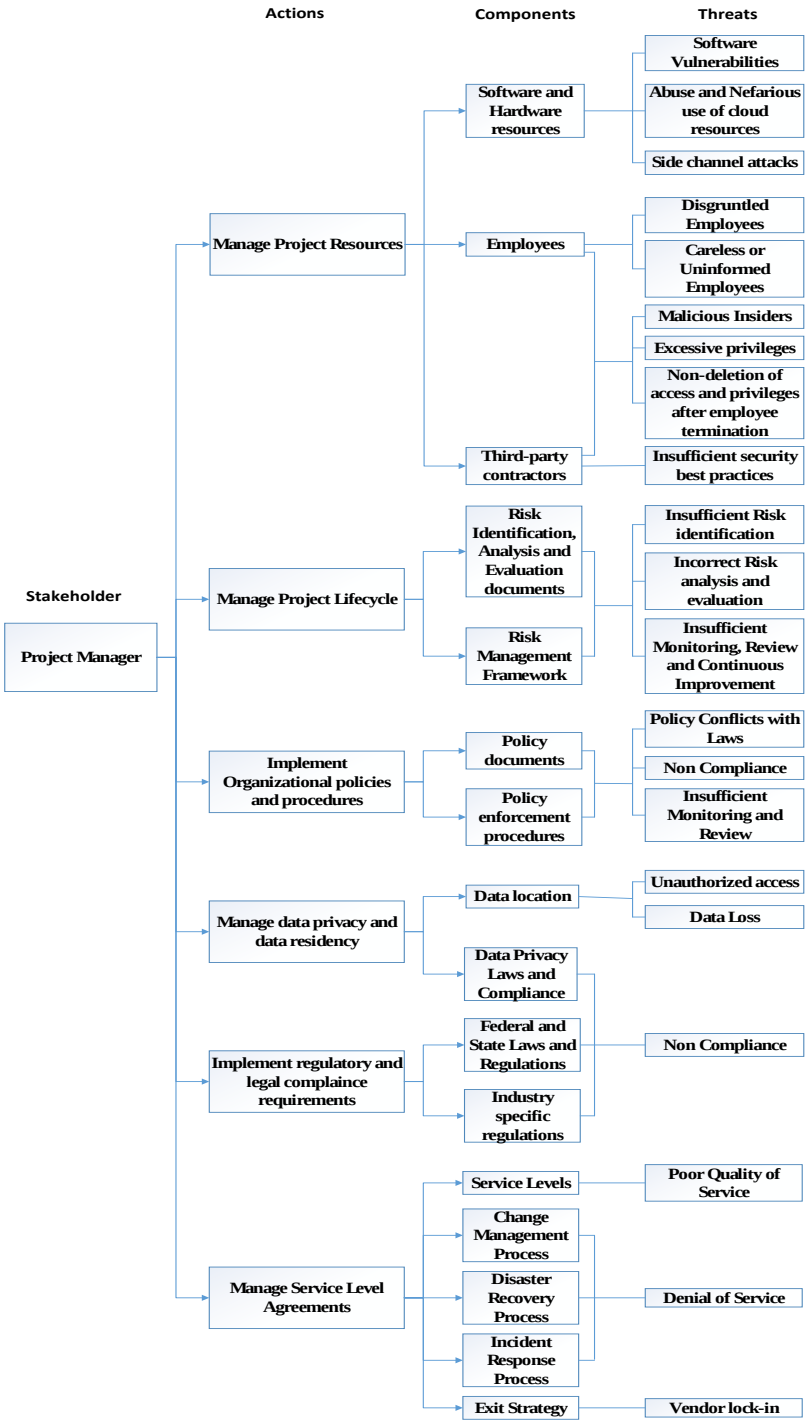
c) *Implement organizational policies and procedures:* Policy is a set of organizational guidelines that dictate certain behavior within an organization. The project manager is responsible for implementing organizational policies and procedures within the project.

d) *Manage data privacy and data residency:* In a PaaS cloud environment, the project manager needs to ensure the privacy of the user data and the data residency is as per the organization policies.

e) *Implement regulatory and legal compliance requirements:* Regulatory compliance is an organization’s adherence to laws and regulations relevant to its business processes. The project manager is responsible for implementing legal and regulatory compliance requirements within the project.

Figure 4.6

Taxonomy for Project Manager



f) *Manage Service-level agreements:* A service-level agreement (SLA) is a document signed by the service provider and the client that defines the level of service expected by the client, the metrics of service measurements, and the penalties if expected service levels are not achieved. An SLA plays a very important part in managing cloud services. The project manager is responsible for managing the SLA as per the organization and project requirements.

The components that the project manager interacts with, directly or indirectly, to perform the identified actions are:

a) *Software and hardware resources:* The project manager indirectly interacts with the software and hardware resources used in the project as he is a part of the procurement process. He analyses the proposals received from the vendors for the hardware and software products required for the project and selects the products and the vendors for the project.

b) *Employees:* Employees of an organization plays an important role in the overall security of the organization and its resources. The project manager directly interacts with the employees working for the project and manages their onboarding processes.

c) *Third-party contractors:* A third-party contractor is a person or company that provides services to another company. Many organizations need third-party contractors for some specialized tasks. The project manager interacts with the third-party contractors working for the project and manages their onboarding processes.

d) *Risk identification, analysis, and evaluation documents:* The project manager is responsible for managing risks throughout the project lifecycle. Risk management involves risk identification, risk analysis, risk evaluation and risk treatment. The project manager is involved in all these phases and reviews the risk identification, analysis, and evaluation documents.

e) *Risk management framework:* An organization's risk management framework defines the organization's planning and design for the risk management efforts. It also defines the organization's risk appetite and risk management program. The project manager uses the risk management framework to manage the risks within the project.

f) *Policy documents:* In an organization, policies are designed to create a productive and effective work environment. The project manager uses the policy documents to implement the policies within the project.

g) *Policy enforcement procedures:* The final component of the design and implementation of policies is the uniform and impartial enforcement. Policy enforcement procedures are created for this purpose. The project manager uses policy enforcement procedures to implement policies within the project.

h) *Data location:* An organization saves sensitive user data in the cloud. To manage the privacy of the stored data, the organization can specify a preferred data location to the cloud provider. The project manager should understand the data privacy requirements of the project and determine the preferred data location if needed.

i) *Data privacy laws and compliance:* The project manager needs to understand and implement data privacy laws and compliance requirements as applicable to the data created, processed, and saved within the project.

j) *Federal and State laws and regulations:* The project manager understands and interacts with the Federal and State laws and regulations that are relevant to the business processes of the organization.

k) *Industry specific regulations:* Regulations and compliance requirements varies by industry e.g. PCI-DSS and GLBA for financial industry, HIPAA for healthcare, FISMA for US federal agencies and HACCP for food and beverage industry. The project manager needs to understand and determine the compliance requirements of the project as per the industry.

l) *Service levels:* The project manager manages the service-level agreement (SLA) with the cloud provider. Service levels define the minimum level of service expected from the cloud provider. The project manager should understand the project requirements and determine the required service levels for different categories such as availability, performance, data encryption etc.

m) *Change management process:* An important process that needs to be established and defined in the SLA is the change management process. The project manager should understand and help in establishing a change management process for any changes or upgrades to the PaaS cloud infrastructure from the cloud provider such that it does not affect the project tasks and deliverables.

n) *Disaster recovery process:* Another important component of the SLA is the disaster recovery process. The project manager should determine the disaster recovery requirements and establish a disaster recovery process for any of the PaaS cloud infrastructure components provided by the cloud provider.

o) *Incident response process:* The project manager needs to work with the PaaS cloud provider to establish the expected incident response process, process to identify problems and resolution expectations.

p) *Exit strategy:* A very important component of the SLA in cloud computing is the exit strategy. The project manager works with the PaaS cloud provider to establish and define an exit strategy with expectations on the cloud provider to ensure smooth transition.

The threats that can prevent the project manager from performing the identified actions or interacting with the identified components on time, within budget and with quality in a PaaS cloud environment, and the threats due to the identified components that can cause a potential danger to the overall security of the PaaS cloud environment are:

a) *Software vulnerabilities:* The applications and the software being used in the project can have unknown or unpatched software vulnerabilities. These vulnerabilities can compromise the overall security of the PaaS cloud environment.

b) *Abuse and nefarious use of cloud resources:* Cloud resources can be abused by a malicious insider or by a cybercriminal. A PaaS cloud environment involves

storing and processing sensitive organization data such as source code and customer confidential data. PaaS is particularly susceptible to abuse of cloud resources because it allows cloud customers to create and deploy their applications. An attacker can inject a malicious service into the PaaS hosting environment and eventually affect the PaaS customers.

c) *Side-channel attacks:* Due to the multi-tenancy and shared resources in cloud, vulnerabilities in virtualization can lead to side-channel attacks from other virtual machines residing on the same physical server. These attacks can compromise the security of PaaS applications and sensitive user data being handled by those applications.

d) *Disgruntled employees:* An organization can have disgruntled employees due to many reasons such as an employee feeling overworked, underpaid, or not getting a promotion. Disgruntled employees can prove to be the biggest threat to the security of an organization as they have access to the confidential information and internal servers.

e) *Careless or uninformed employees:* Careless employees or uninformed employees who are not trained in security best practices and visit unauthorized websites, open suspicious email attachments, click on links in suspicious emails or have weak passwords, can be the biggest threat to the security of an organization.

f) *Malicious insiders:* Malicious insiders are an organization’s current or former employees, third-party contractors or other business partners who has or had

authorized access to organization's system, network or data and intentionally abuses that access to negatively affect the confidentiality, integrity and availability of the organization's information systems or information (CERN, 2020). A malicious insider can have access to all confidential data and can compromise the security of an organization very easily.

g) *Excessive privileges:* Excessive privileges given to an employee or third-party contractor can be misused by them or by an attacker who compromises these accounts. Regular audits of employees and third-party contractors' access rights and privileges should be conducted and updated as needed. These excessive privileges can be used to compromise the security of the PaaS cloud environment.

h) *Non-deletion of access and privileges after employee termination:* When an employee leaves an organization or a third-party contractor finishes his contract, their access rights and privileges to organization's resources should be deleted immediately. These access rights could be used to compromise the security of the organization.

i) *Insufficient security best practices:* Many of the third-party vendors do not follow security and privacy best practices and don't train their employees on security practices. An organization should review third-party security policies to reduce the risks of cyber-attacks from third-party contractors working for the organization.

j) *Insufficient Risk identification:* To manage risks including the information security risks in a project, the project manager is involved in the risk identification and analysis process. To identify risks, organization's information assets are identified and classified, and potential threats to them are also identified. Insufficient risk identification could be a big threat to the overall security of the organization. Insufficient risk identification can result from many reasons like not including all information assets, not classifying the assets properly, poor threat assessment of these assets etc. Insufficient risk identification can prevent the project manager from managing information security risks in the project.

k) *Incorrect Risk analysis and evaluation:* After risk identification, risk analysis and risk evaluation are performed. In risk analysis, likelihood and impact of an attack is calculated for each information asset. In risk evaluation, risk tolerance is calculated for each asset and organization's risk appetite is determined. Incorrect or poor risk analysis or risk evaluation can result in incomplete or weak security controls for the organization's information assets.

l) *Insufficient monitoring, review, and continuous improvement:* Organizations should monitor and review the risk management plan on a regular basis to compare outcomes from past and current performance against the desired outcomes and continuously improve the risk management efforts to improve the overall security of the organization.

m) *Policy conflicts with laws:* The project manager implements organization policies including information security policies within the project. If the information

security policy conflicts with government laws, then the project manager will not be able to enforce those policies within the project. Such a policy will not be able to stand up in court if challenged.

n) *Insufficient monitoring and review:* Policies should be monitored and reviewed on regular basis. They should be updated regularly as needed. Insufficient monitoring and review of policies can prevent the project manager from ensuring the uniform enforcement of policies within the project.

o) *Non-compliance:* Non-compliance of organization policies by employees can lead to security loopholes within the organization that can be exploited by cybercriminals. Organizations need to ensure compliance to government and industry regulations. The costs of non-compliance to any of the data privacy laws, Federal and State regulations or industry specific regulations can be extremely high for an organization. Non-compliance also leads to organization's reputational damage and loss of future business.

p) *Unauthorized access:* In a PaaS cloud environment, organization's sensitive user and application data is stored on storage servers located at a different geographical location. Unauthorized access to the data can compromise the security of the application and the user data.

q) *Data loss:* As data is saved on a storage server managed by the cloud provider, data loss can happen due to server failure, malware, or other reasons. Data loss can prevent the project manager from maintaining user data privacy.

r) *Poor Quality of service:* One of the issues in cloud computing could be poor quality of service as per the service levels mentioned in the SLA by the cloud provider.

s) *Denial of service:* Denial of service happens when the organization is not able to access PaaS cloud infrastructure components provided by the cloud provider. DoS can happen due to a disaster, incident, or an update on PaaS infrastructure components.

t) *Vendor lock-in:* One of the concerns for organizations moving to the cloud is vendor lock-in. Cloud vendor lock-in is a situation when the cloud consumer becomes dependent on a single cloud service provider and unable to migrate to another cloud service provider without significant costs and incompatibilities. The primary lock-in risks include data transfer risk, application transfer risk and infrastructure transfer risk. Cloud consumer organization should establish a clear exit strategy and agreed upon by both parties to make it easier for the organization to avoid vendor lock-in.

Taxonomy for Application User

In a PaaS cloud environment, an application user is an end user or client who is involved in user acceptance testing (UAT) of the developed cloud applications. User acceptance testing is the testing done to verify that the cloud application from the vendor meets the business needs of the organization.

The actions performed by the application user in a PaaS cloud environment are as follows:

a) *Test applications:* The application user tests the cloud applications developed in the PaaS cloud environment before the application is moved to the production environment.

The PaaS cloud components that the application user interacts with to perform the identified actions are:

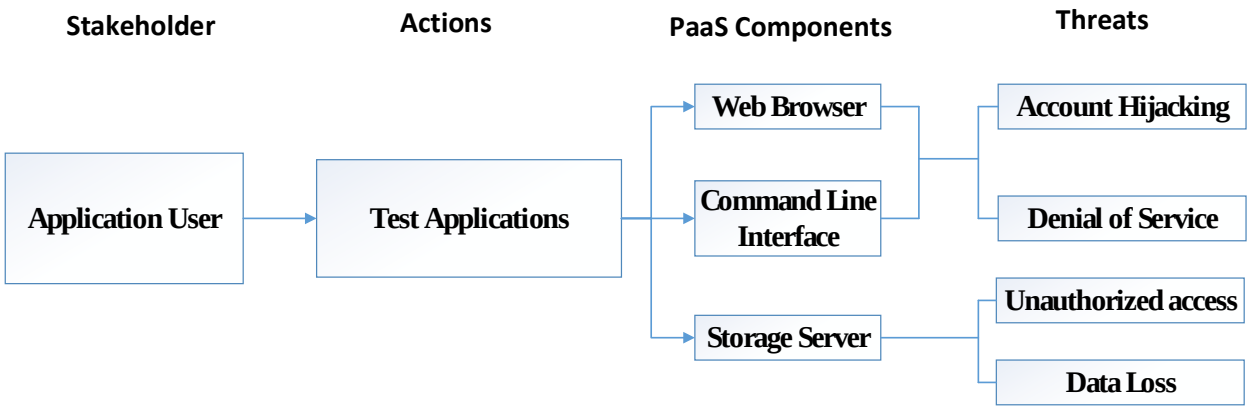
a) *Web browser:* The application user accesses cloud applications by using a web browser.

b) *Command line interface:* Cloud applications provide some functionalities through command line interface as well. The application user uses command line interface to access those functionalities and test them.

c) *Storage server:* The application user uses storage server to store data to be used during testing of the cloud applications.

Figure 4.7

Taxonomy for Application User



The threats that can prevent the application user from performing the identified actions or interacting with the identified PaaS components on time, within budget and with quality in a PaaS cloud environment are as follows:

a) *Account hijacking:* In account hijacking, an attacker gains access to a user's credentials to an application or system. In a PaaS cloud environment, the application user uses web browser and command line interface to connect to cloud applications for testing. The application user's credentials could be hijacked by an attacker to gain unauthorized access to cloud applications.
b) *Denial of Service:* A DoS attack on the PaaS cloud application can prevent the application user from connecting to the application and perform user acceptance testing.
c) *Unauthorized access:* The application user stores user data, used for testing the application, on the storage server. Unauthorized access of the storage server could compromise the security and integrity of the user data.
d) *Data loss:* Data loss of the data stored on the storage server can prevent the application user from testing the application.

Mean Failure Cost

To measure the reliability of a system, Mili et al. proposed a quantitative model called Mean Failure Cost (MFC) (Mili & Sheldon, 2009). The other metrics that are commonly accepted to measure reliability of a system, such as mean time to failure (MTTF) and mean time to detection of a vulnerability (MTTD), assumes that the failure cost and failure probability of a system are independent of its subspecifications and its

stakeholders. Mean failure cost or MFC varies by stakeholders, security requirements of stakeholders, failure probability of different system components and failure impact on different stakeholders. The quantitative model considers the system's stakeholders, the security requirements, functional components, and threats to calculate the average loss per unit of time ($/H) for a stakeholder because of security threats.

Ben Aissa et al. gave a formula to compute the values of mean failure cost for each stakeholder as below (Aissa et al., 2010):

$$MFC = ST * DP * IM * PT$$

a) ST is the Stakes matrix that represents the stakes that each stakeholder has in meeting security requirements. It has as many rows as the stakeholders (S_i) and as many columns as the security requirements (R_j). The values of the matrix represent the dollar value of the loss of the stakeholder if specific requirement is not met. We are proposing a change in the stakes matrix. In our case, the stakes matrix represents the stakes that each stakeholder has in performing his actions. The rows of the matrix represent stakeholders (S_i) and columns represent his actions (A_j). The values of the matrix $ST(S_i, A_j)$ represents the loss (in dollars) of stakeholder S_i if he is not able to perform action A_j per unit of time. The stakes matrix is to be filled by the stakeholder.

b) DP is the Dependency matrix that represents the probability of failure of an action if the specific components has been compromised or failed. It has as many rows as the actions of the stakeholder (A_i) and as many columns as the components (C_j) that stakeholder interacts with to perform his actions. The values of the

matrix DP(A_i, C_j) represents the probability of failure of action A_i when system component C_j has been compromised or failed. The dependency matrix is to be filled by the system architect who has a clear understanding of the different components a stakeholder needs to interact with to perform his actions.

c) IM is the Impact matrix that represents the probability of components failure given that specific threat has materialized. It has as many rows as the system components (C_i) and as many columns as the security threats (T_j) being considered. The values of the matrix IM(C_i, T_j) represents the probability that component C_i has failed or compromised if threat T_j has materialized. The impact matrix is to be filled by the security team after analyzing which threats affect which components and the likelihood of success of each threat.

d) PT is the Threat vector that represents the probability that a threat (T_i) materializes in a unit period of time e.g. an hour. The threat vector has as many rows as the threats being considered.

Quantification of security risks in PaaS cloud environment

To quantify security risks in a PaaS cloud environment from stakeholder's perspective, we propose the use of MFC because it gives the security risk in terms of financial loss per unit of time ($/H) and it calculates the risk by considering stakeholders, their requirements, components they interact with and the threats on those components. In this section, we will show how to calculate the mean failure cost for one of the PaaS stakeholders, the PaaS application developer. To compute MFC, we need to fill three matrices and the threat vector. Fig 4a shows the taxonomy of a

PaaS application developer that shows the tasks performed by the application developer, PaaS cloud components he interacts with to perform his tasks and the threats on those components. By using the taxonomy, the rows and columns of the matrices are developed.

a) Stakes matrix (ST)

The stakes matrix (ST) for the PaaS application developer is shown in Table 1. The stakes matrix represents the relation between the PaaS application developer and his tasks. The columns of the matrix are developed using actions from the application developer's taxonomy. The values in the matrix is to be filled by the stakeholder, in this case the application developer. Each value in the matrix represents the loss of the application developer or organization per hour if he is not able to perform that action. The values in the matrix can be changed depending on the organization and stakeholder needs.

Table 1

Stakes matrix ($/H)

Stakeholder	Actions						
	Analyze and Specify application requirements	***Design application software***	***Create Applications***	***Fix software bugs***	***Integrate Applications***	***Test Software***	***Maintain software code versions***
PaaS Application developer	200	300	500	300	100	100	50

b) Dependency matrix (DP)

The dependency matrix (DP) for the PaaS application developer is shown in Table 2. The dependency matrix represents the relation between the application

developer's tasks and the PaaS cloud components. The rows and columns of the matrix are developed using the actions and PaaS components from the application developer's taxonomy. The values in the matrix is to be filled by the system architect. Each value in the matrix represents the probability of failure of PaaS application developer's actions if component has failed or compromised. The values in the matrix can be changed as per the organization and stakeholder needs.

Table 2

Dependency matrix

Actions	Components																			
	MS Word	*PDF Tools*	*Web Browser*	*UML tools*	*Programming languages*	*Dependent libraries*	*Data files*	*CLI*	*IDE*	*PL Compilers*	*PL debug tools*	*Data comparison utilities*	*Memory debug tools*	*OS*	*App Server*	*Web Server*	*DB Server*	*Storage Server*	*Unit test frameworks*	*Code coverage tools*	*Version control software*
Analyze and Specify application requirements	0.5	0.3	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Design application software	0.2	0.2	1.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Create Applications	0.0	0.0	1.0	0.0	0.7	0.4	0.1	0.1	0.5	0.6	0.2	0.1	0.2	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Fix software bugs	0.0	0.0	1.0	0.0	0.7	0.4	0.1	0.1	0.5	0.6	0.2	0.1	0.2	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Integrate Applications	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.4	0.4	0.4	0.4	0.0	0.0	0.0
Test Software	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8	0.2	0.0
Maintain software code versions	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8

c) Impact matrix (IM)

The impact matrix (IM) for the PaaS application developer is shown in Table 3. The impact matrix relates components failure to security threats. The rows and columns of the matrix are developed using the PaaS components and threats from the application developer's taxonomy. The values in the matrix is to be filled by the cybersecurity expert after considering the threats that prevents a PaaS application developer from performing his tasks or interacting with the PaaS cloud components. Each value in the matrix represents the probability of failure of PaaS cloud component if

the corresponding threat has materialized. The values in the matrix can be changed as per the organization and stakeholder needs.

Table 3

Impact matrix

Components	Threats														
	Malware	*Account Hijacking*	*DoS*	*Insecure interface*	*Insecure Code*	*Code Injections*	*Backdoor*	*Insecure APIs*	*Unauthorized Access*	*Unpatched Vulnerabilities*	*Monitoring from Host*	*Security Misconfigurations*	*Side-channel attacks*	*Data Loss*	*Insecure Deserialization*
MS Word	**0.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PDF Tools	**0.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Web Browser	0.0	**0.8**	**0.5**	**0.5**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
UML tools	0.0	0.0	0.0	**0.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Programming languages	0.0	0.0	0.0	0.0	**0.8**	**0.5**	**0.8**	**0.5**	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Dependent libraries	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.8**	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Data files	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.6**
CLI	0.0	**0.8**	**0.5**	**0.5**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
IDE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.2**	**0.6**	0.0	0.0	0.0	0.0	0.0
PL Compilers	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.7**	**0.5**	0.0	0.0	0.0	0.0	0.0
PL debug tools	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.2**	**0.6**	0.0	0.0	0.0	0.0	0.0
Data comparison utilities	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.4**	**0.4**	0.0	0.0	0.0	0.0	0.0
Memory debug tools	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.2**	**0.5**	0.0	0.0	0.0	0.0	0.0
OS	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.6**	**0.6**	**0.8**	**0.6**	0.0	0.0
App Server	0.0	**0.8**	**0.5**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.8**	0.0	0.0	0.0
Web Server	0.0	**0.8**	**0.5**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.8**	0.0	0.0	0.0
DB Server	0.0	**0.8**	**0.7**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.8**	0.0	**1.0**	0.0
Storage Server	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**1.0**	0.0
Unit test frameworks	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.2**	**0.6**	0.0	0.0	0.0	0.0	0.0
Code coverage tools	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.2**	**0.4**	0.0	0.0	0.0	0.0	0.0
Version control software	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**0.6**	**0.4**	0.0	0.0	0.0	0.0	0.0

d) Threat vector (PT)

The threat vector (PT) for the threats under consideration for PaaS application developer is shown in Table 4. The threat vector reflects threat probability per unit of time. The values in the threat vector below shows the probability of attack for each threat during one hour (Jouini & Rabai, 2014).

Table 4

Threat vector

Threats	Probability
Malware	40.62×10^{-4}
Account Hijacking	17.27×10^{-4}
DoS	14.39×10^{-4}
Insecure interface	30.02×10^{-4}
Insecure Code	35.20×10^{-4}
Code Injections	20.05×10^{-4}
Backdoor	24.02×10^{-4}
Insecure APIs	29.02×10^{-4}
Unauthorized Access	54.5×10^{-4}
Unpatched Vulnerabilities	64.3×10^{-4}
Monitoring from Host	8.063×10^{-4}
Security Misconfigurations	48.5×10^{-4}
Side-channel attacks	5.8×10^{-4}
Data Loss	5.75×10^{-4}
Insecure Deserialization	6.2×10^{-4}

Using the values of the matrices and threat vector, we can compute the value of the mean failure cost for the PaaS application developer by using the below formula:

MFC = ST * DP * IM * PT

The mean failure cost (MFC) value for the PaaS application developer is shown in Table 5.

Table 5

Mean Failure Cost

Stakeholder	MFC ($/H)
PaaS Application developer	24.10

The same method can be used to calculate mean failure cost for any of the PaaS stakeholders.

Summary

This chapter discussed the details of the identification and quantification processes used to identify and quantify security risks in a PaaS cloud environment as pertinent to PaaS stakeholders. In the identification process, a taxonomical approach is used to identify PaaS stakeholders, their tasks, components they interact with to complete the tasks and the threats that can prevent them from completing their tasks efficiently and securely. For the quantification of security risks as pertinent to stakeholders, MFC security metric is used. The quantitative model is explained and the process to calculate mean failure cost was illustrated by calculating mean failure cost for PaaS application developer.

Chapter V: Results, Conclusion, and Recommendations

Introduction

This chapter discusses the results obtained and the conclusions drawn from the study. The chapter also provide recommendations for future work related to the study.

Results

The objective of the study was to provide PaaS cloud stakeholders with a means to quantify security threats as pertinent to them. The methodology of the study consisted of two main steps: identification, and quantification. In identification process, PaaS stakeholders were identified along with their actions, components they interact with and the threats that prevent them from accomplishing their tasks. For the identification process, a taxonomy was created for each PaaS stakeholder. The taxonomy for each stakeholder shows the actions performed by the stakeholder, components used by the stakeholder to complete those actions and the threats that can prevent the stakeholder from performing those actions. In the quantification process, MFC security metric was used to calculate failure cost from a stakeholder's perspective. Mean failure cost was calculated for PaaS application developer. The security threats depend on the stakeholders, their tasks and how they interact with various PaaS cloud components. Using the identification and quantification process of the study, organizations can quantify the security threats as pertinent to PaaS stakeholders and use that information to implement better security controls. Organizations can deploy countermeasures to enhance the security of PaaS cloud and reduce the mean failure cost for stakeholders. Organizations can reduce the mean failure cost for a stakeholder

by controlling the dependency matrix, impact matrix or the threat vector. The dependency matrix can be controlled by minimizing the impact of component failure on the stakeholder's actions. This can be achieved by using multiple servers for redundancy. The impact matrix can be controlled by minimizing the impact of potential threats on components by mitigating vulnerabilities of the components. The threat vector can be controlled by using better security controls and multilayered security.

The study questions are listed below:

Q1: Can failure cost be quantified in cloud computing?

Ans: The study shows that failure can be quantified in cloud computing by using MFC as the security metric.

Q2: How can failure be quantified from the stakeholder's point of view?

Ans: The study showed the identification of stakeholders, their actions, the components they interact with and the threats that can prevent them from performing their actions efficiently and securely, in a taxonomical approach. Using the information from the taxonomies, failure was quantified from a stakeholder's perspective by using MFC as the security metric. The paper also illustrated the calculation of mean failure cost for one of the identified PaaS cloud stakeholders, the PaaS application developer.

Q3: How effective is MFC in quantifying security threats from the stakeholder's point of view?

Ans: MFC can effectively be used by organizations to quantify security threats as pertinent to stakeholders. The study showed that MFC varies by stakeholders, their actions, probability of failure of system components due to different threats and

probability of potential threats. The paper also explained how MFC can be used by organizations to prioritize security countermeasures as per stakeholders and improve the overall security of the organization.

Conclusion

Cloud computing is a scalable technology being used by many organizations to gain access to a shared pool of computing resources. Organizations are using cloud to increase performance, flexibility, scalability, productivity, and availability. Security of information and information assets is a big concern for organizations using cloud computing. Security is highly important in a PaaS cloud environment because it holds the source code and processes sensitive user data. PaaS has several stakeholders with different security goals and tolerance to risks. The security threats depend on stakeholders, their actions, and interactions with different components. In the study, PaaS stakeholders have been identified along with their tasks and cloud components they use to accomplish those tasks. Security threats have been discussed as pertinent to different stakeholders, their actions, and interactions with various cloud components. To quantify security threats from a stakeholder's perspective, MFC metric have been used. MFC calculates the security risks for a stakeholder and it varies by stakeholders, their requirements, components, and threats. We illustrated the computation of MFC for PaaS application developer.

This study will help organizations to have a better understanding of PaaS cloud security issues from stakeholders' perspective. Organizations will be able to perform a more accurate risk assessment and implement better security controls.

Future Work

The paper discussed several threats specific to PaaS stakeholders and components but in a complex PaaS cloud environment the potential threats can be more than those discussed in the paper. As part of our ongoing work, we will continue to analyze potential threats specific to PaaS cloud stakeholders. As shown in the paper, calculating mean failure cost for a stakeholder requires complex calculations, so an automated tool can be created to help stakeholders calculate mean failure cost.

References

Abuhussein, A., Shiva, S., & Harkeerat Bedi. (2012). *Evaluating security and privacy in cloud computing services: A Stakeholder's perspective*. https://doi.org/10.13140/RG.2.1.2329.7761

Aissa, A. B., Abercrombie, R. K., Sheldon, F. T., & Mili, A. (2010). Modeling stakeholder/value dependency through mean failure cost. *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research - CSIIRW '10*, 1. https://doi.org/10.1145/1852666.1852727

App Engine | Google Cloud. (2020, March 14). https://cloud.google.com/appengine

AWS Elastic Beanstalk – Deploy Web Applications. (2020, March 14). https://aws.amazon.com/elasticbeanstalk/

Bazm, M.-M., Lacoste, M., Südholt, M., & Menaud, J.-M. (2017). *Side Channels in the Cloud: Isolation Challenges, Attacks, and Countermeasures*. 14.

Bourque, P., Fairley, R. E., & IEEE Computer Society. (2014). *Guide to the software engineering body of knowledge.*

CERN, C. (2020). *Insider Threat | Software Engineering Institute*. https://www.sei.cmu.edu/our-work/insider-threat/index.cfm

Chauhan, N. S., Saxena, A., & Murthy, J. (2013). An Approach to Measure Security of Cloud Hosted Application. *2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, 1–6. https://doi.org/10.1109/CCEM.2013.6684427

Chen, D., & Zhao, H. (2012, March). Data Security and Privacy Protection Issues in Cloud Computing. *2012 International Conference on Computer Science and Electronics*

Engineering. 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE 2012), Hangzhou.

Cimpanu, C. (2020, February). *Jenkins servers can be abused for DDoS attacks*. ZDNet. https://www.zdnet.com/article/jenkins-servers-can-be-abused-for-ddos-attacks/

Code Injection Software Attack | OWASP Foundation. (2020). https://owasp.org/www-community/attacks/Code_Injection

Code Spaces: A Lesson In Cloud Backup | Network Computing. (2014, July 3). https://www.networkcomputing.com/cloud-infrastructure/code-spaces-lesson-cloud-backup

Comparing MTBF, MTTF, MTTR. (2016, August 25). Get Certified Get Ahead. https://blogs.getcertifiedgetahead.com/mtbf-mttf-mttr/

Darwish, M., Ouda, A., & Capretz, L. F. (2013). *Cloud-based DDoS Attacks and Defenses*. 5.

Get to Know Azure | Microsoft Azure. (2020, March 14). https://azure.microsoft.com/en-us/overview/

Getting Started with Amazon Simple Storage Service—Amazon Simple Storage Service. (2020, March 14). https://docs.aws.amazon.com/AmazonS3/latest/gsg/GetStartedWithS3.html

Gruschka, N., & Jensen, M. (2010). Attack Surfaces: A Taxonomy for Attacks on Cloud Services. *2010 IEEE 3rd International Conference on Cloud Computing*, 276–279. https://doi.org/10.1109/CLOUD.2010.23

Hackers breach Volusion and start collecting card details from thousands of sites | ZDNet. (2019, October 8). https://www.zdnet.com/article/hackers-breach-volusion-and-start-collecting-card-details-from-thousands-of-sites/

Halabi, T., & Bellaiche, M. (2017). Towards quantification and evaluation of security of Cloud Service Providers. *Journal of Information Security and Applications*, *33*, 55–65. https://doi.org/10.1016/j.jisa.2017.01.007

Hale, M. L., & Gamble, R. (2012). SecAgreement: Advancing Security Risk Calculations in Cloud Services. *2012 IEEE Eighth World Congress on Services*, 133–140. https://doi.org/10.1109/SERVICES.2012.31

Hashizume, K., Rosado, D. G., Fernández-Medina, E., & Fernandez, E. B. (2013). An analysis of security issues for cloud computing. *Journal of Internet Services and Applications*, *4*(1), 5. https://doi.org/10.1186/1869-0238-4-5

Insecure Deserialization | OWASP. (2017). https://owasp.org/www-project-top-ten/2017/A8_2017-Insecure_Deserialization.html

Jensen, M., Schwenk, J., Gruschka, N., & Iacono, L. L. (2009). On Technical Security Issues in Cloud Computing. *2009 IEEE International Conference on Cloud Computing*, 109–116. https://doi.org/10.1109/CLOUD.2009.60

Jouini, M., & Rabai, L. B. A. (2014). A Security Risk Management Metric for Cloud Computing Systems: *International Journal of Organizational and Collective Intelligence*, *4*(3), 1–21. https://doi.org/10.4018/ijoci.2014070101

Kandukuri, B. R., V., R. P., & Rakshit, A. (2009). Cloud Security Issues. *2009 IEEE International Conference on Services Computing*, 517–520. https://doi.org/10.1109/SCC.2009.84

Karabacak, B., & Sogukpinar, I. (2005). ISRAM: Information security risk analysis method. *Computers & Security*, *24*(2), 147–159. https://doi.org/10.1016/j.cose.2004.07.004

Kazim, M., & Ying, S. (2015). A survey on top security threats in cloud computing. *International Journal of Advanced Computer Science and Applications*, *6*(3). https://doi.org/10.14569/IJACSA.2015.060316

Lane, A. (2013). *Securosis—Blog—Article*. https://securosis.com/blog/database-denial-of-service-the-attacks

LDAP Injection. (2020). Checkmarx. https://www.checkmarx.com/knowledge/knowledgebase/LDAP

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., & Leaf, D. (2011). *NIST Cloud Computing Reference Architecture*. 35.

Masky, M., Young, S. S., & Choe, T.-Y. (2015). A Novel Risk Identification Framework for Cloud Computing Security. *2015 2nd International Conference on Information Science and Security (ICISS)*, 1–4. https://doi.org/10.1109/ICISSEC.2015.7370967

Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing*. 7.

Mili, A., & Sheldon, F. (2009). Challenging the Mean Time to Failure: Measuring Dependability as a Mean Failure Cost. *2009 42nd Hawaii International Conference on System Sciences*, 1–10. https://doi.org/10.1109/HICSS.2009.107

Modi, C., Patel, D., Borisaniya, B., Patel, A., & Rajarajan, M. (2013). A survey on security issues and solutions at different layers of Cloud computing. *The Journal of Supercomputing*, *63*(2), 561–592. https://doi.org/10.1007/s11227-012-0831-5

MTTD vs. MTTF vs. MTBF vs. MTTR | Resolve Major IT Incidents Quickly. (2018, May 7). AlertOps | Resolve Major IT Incidents & Automate Real-Time Operations. https://alertops.com/mttd-vs-mttf-vs-mtbf-vs-mttr/

Rainer, R. K., Snyder, C. A., & Carr, H. H. (1991). Risk Analysis for Information Technology. *Journal of Management Information Systems*, *8*(1), 129–147. https://doi.org/10.1080/07421222.1991.11517914

Ristenpart, T., Tromer, E., Shacham, H., & Savage, S. (2009). Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. *Proceedings of the 16th ACM Conference on Computer and Communications Security - CCS '09*, 199. https://doi.org/10.1145/1653662.1653687

Saaty, R. W. (1987). The analytic hierarchy process—What it is and how it is used. *Mathematical Modelling*, *9*(3–5), 161–176. https://doi.org/10.1016/0270-0255(87)90473-8

Sandikkaya, M. T., & Harmanci, A. E. (2012). Security Problems of Platform-as-a-Service (PaaS) Clouds and Practical Solutions to the Problems. *2012 IEEE 31st Symposium on Reliable Distributed Systems*, 463–468. https://doi.org/10.1109/SRDS.2012.84

Saripalli, P., & Walters, B. (2010). QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security. *2010 IEEE 3rd International Conference on Cloud Computing*, 280–288. https://doi.org/10.1109/CLOUD.2010.22

Security Misconfiguration | OWASP. (2017). https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html

Session hijacking attack | OWASP. (2020). https://owasp.org/www-community/attacks/Session_hijacking_attack

SQL Injection | OWASP. (2020). https://owasp.org/www-community/attacks/SQL_Injection

Taha, A., Trapero, R., Luna, J., & Suri, N. (2014). AHP-Based Quantitative Approach for Assessing and Comparing Cloud Security. *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications*, 284–291. https://doi.org/10.1109/TrustCom.2014.39

Tirumala, S. S., Sathu, H., & Naidu, V. (2015). Analysis and Prevention of Account Hijacking Based INCIDENTS in Cloud Environment. *2015 International Conference on Information Technology (ICIT)*, 124–129. https://doi.org/10.1109/ICIT.2015.29

Top 5 Cloud Security related Data Breaches! - Cybersecurity Insiders. (2017, October 1). https://www.cybersecurity-insiders.com/top-5-cloud-security-related-data-breaches/

What is a DNS Cache Poisoning? | DDI (Secure DNS, DHCP, IPAM) | Infoblox. (2020). https://www.infoblox.com/glossary/dns-cache-poisoning/

What is Oracle Cloud Platform | Oracle. (2020, March 14). https://www.oracle.com/cloud/what-is-oracle-cloud-platform.html

Wu Chaoxia, Yuxia Sun, & Zhi Li. (2015). Cloud computing risk assessment method based on game theory. *Third International Conference on Cyberspace Technology (CCT 2015)*, 5 .-5 . https://doi.org/10.1049/cp.2015.0854

Zhang, X., Wuwong, N., Li, H., & Zhang, X. (2010). Information Security Risk Management Framework for the Cloud Computing Environments. *2010 10th IEEE International Conference on Computer and Information Technology*, 1328–1334. https://doi.org/10.1109/CIT.2010.501

Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. *Future Generation Computer Systems*, *28*(3), 583–592. https://doi.org/10.1016/j.future.2010.12.006