Master Thesis

INSURING LIABILITY RISKS OF SOFTWARE AS A SERVICE (SaaS)-PROVIDERS IN CLOUD COMPUTING

carried out for the purpose of obtaining the degree of

Diplom-Ingenieur (Dipl.-Ing. oder DI)

submitted at TU Wien

Faculty of Mechanical and Industrial Engineering

By

Murat OGURLU

Mat.No.: 1226156

under the supervision of

Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Walter Schwaiger

Institute of Management Science

Wien, August 2023

_____

Affidavit

I declare in lieu of oath, that I wrote this thesis and carried out the associated research myself, using only the literature cited in this volume. If text passages from sources are used literally, they are marked as such.

I confirm that this work is original and has not been submitted for examination elsewhere, nor is it currently under consideration for a thesis elsewhere.

I acknowledge that the submitted work will be checked electronically-technically using suitable and state-of-the-art means (plagiarism detection software). On the one hand, this ensures that the submitted work was prepared according to the high-quality standards within the applicable rules to ensure good scientific practice "Code of Conduct" at the TU Wien. On the other hand, a comparison with other student theses avoids violations of my personal copyright.

Wien, August 2023 Signature

Acknowledgements

I would like to express my sincere gratitude and appreciation to Prof. Walter Schwaiger for his exceptional support and encouragement throughout the journey of completing this thesis. His mentorship has been an instrumental force behind the successful completion of this thesis and the enrichment of my intellectual journey.

Under his mentoring, I acquired the invaluable skills of scientific thinking and acting, qualities that are becoming increasingly rare in today's academic landscape. His unique ability to foster independent thought was evident in allowing me to select my research topic, showcasing his remarkable mastery of diverse subjects.

Table of Contents

List of Tables

List of Figures

1. Introduction

The rapid evolution of cloud computing has brought about a significant transformation in the information technology landscape, reshaping how businesses and organizations handle data and services. This technology has gained widespread adoption and offers several advantages, including scalability, cost-efficiency, and improved accessibility. As a result, cloud computing is driving the next wave of digital business, with companies adopting emerging technologies such as artificial intelligence and virtual reality environment.

With the increasing popularity of cloud technology, more providers are entering the market to capitalize on the growing demand. As a consequence of this growing reliance on cloud-based services, it becomes crucial to address potential vulnerabilities associated with the technology.

The literature review indicates that there is a higher number of articles that primarily concentrate on topics related to technology and technical aspects, compared to business-focused ones. The existing articles on technological issues often delve into technical details from the standpoint of cloud computing specialists, providing informative content but lacking practical relevance for business professionals. On the other hand, current articles concerning business issues have mainly centered on 'cloud adoption' as a core theme in business-oriented literature. However, their approach tends to focus solely on inputs and outputs, overlooking the internal workings and structure of cloud computing, They also fail to consider the differentiations between various service layers and deployment models within cloud computing (Yang & Tate, 2012).

This is where this thesis comes into account. It will explore how to assess liability risks that arise in SaaS providers and calculate an insurance premium for them, bridging the gap in the literature and providing valuable insights for both technological and business-oriented professionals.

According to Gartner's forecasts, Software as a Service (SaaS) stands out as the largest segment of the cloud market in terms of end-user spending (Gartner Inc., 2023a). The SaaS business model is directly associated with numerous liabilities, unlike assets that contribute to a company's value, these liabilities lead to a reduction in both value and equity.

SaaS providers face significant liability risks, retaining them through customer contracts (Service Level Agreement), including interruption of information access, compliance failure, and intellectual property issues. These risks can arise from special legislation, such as telecommunication law, electronic commerce law, data protection law, copyright and trademark or patent issues. The risk for unwanted and unintentional issues in cloud computing is high, and attributing liability to cloud computing is not easy. In case of negligence, cloud provider could be held financially responsible for any harm caused to their customers.

To strike a balance between the risks and the costs each Cloud Service Provider (CSP) must carefully analyze its business operations. Accurately quantifying liability risks has proven to be complex, as it requires analyzing multiple factors that can be difficult to measure or predict. Additionally, obtaining relevant data to accurately assess liability exposure can be challenging, as SaaS providers may not have complete visibility into their customers' data or the risks associated with their service offerings (Eling & Schnell, 2016).

Liability insurance can be a valuable component of an overall risk mitigation strategy, to address potential liabilities and any resulting legal and financial consequences by transferring the risk to an insurance provider. However, the market demand for cyber risk insurance remains relatively small due to challenges associated with pricing such policies, the unique characteristics of SaaS liability risks compared to other operational risks, and information asymmetry between the insurer and the SaaS provider.

This thesis assesses the overall and partial liability risks that exist for SaaS-Providers, by performing a vulnerability assessment and expected loss analysis. Using collective risk model, the expected insurance premium for these liability risks is calculated.

This thesis aims to help SaaS-Providers to:

1) Assess the most vulnerable liability risks in their operations.
2) Quantify the potential financial impact associated with the identified risks.

This thesis also aims to assist insurance companies in:

3) Calculating the appropriate insurance premium for insuring SaaS-Assets.

The history and evolution of cloud computing are explored at the beginning of this thesis. Through tracing its origins and examining its evolutionary path, a better understanding of the development and progression of this innovative technology can be gained. This historical context sets the stage for understanding the definition of cloud computing and the various service and deployment models that exist within this domain.

The thesis then approaches the definition of cloud service providers, with a particular focus on SaaS providers. As key players in the cloud computing business, SaaS providers offer software applications to end-users via the internet, eliminating the need for local installations. The characteristics of SaaS providers explained and the specific security and vulnerability management challenges they face related to liabilities are pointed out.

To provide a systematic assessment of vulnerabilities, this thesis incorporates the principles of risk assessment outlined in the ISO 31000 standard. By applying the guidelines presented in this standard, the risks associated with SaaS assets can be systematically evaluated, considering factors such as data breaches, service interruptions, and compliance failures.

The SA-VA model, designed to assess vulnerabilities in cloud computing, especially for SaaS providers, exhibits an assessing nature that makes it applicable across different areas within cloud computing for vulnerability assessment. It takes into account the Information Security Management System (ISMS) standard and benchmarks to manage information security in businesses. Vulnerabilities are categorized into three types: interruption of information access, compliance failures, and intellectual property issues.

The mathematical formulation of the model involves representing vulnerabilities as nodes within a directed acyclic graph (DAG). Each vulnerability parameter is defined by seven discrete states, which represent various possible occurrences of attacks.

The statistical implementation of the SA-VA model is achieved through the use of a Copula-aided Bayesian belief network, which is employed to model and analyze the dependencies between various vulnerability factors. Additionally, Sklar's Theorem and Kendall correlation are utilized to estimate joint probabilities, enabling a more comprehensive understanding of the interconnectedness of vulnerabilities. The use of the Gaussian copula function in this thesis not only aggregates multiple normal distributions to determine vulnerability but also

simplifies the estimation of joint probabilities, further enhancing the effectiveness of the SA-VA model.

The SA-VA model is simulated using the MATLAB programming language, with detailed algorithms and code provided in the thesis. This ensures a transparent and reproducible approach to assess vulnerabilities in SaaS delivery model. Validation of the model confirms its reliability and ability to produce actionable results, thereby strengthening its application in real-world scenarios.

The results obtained from the SA-VA model implementation provide a vulnerability assessment report that includes calculated posterior probabilities and expected risk values for selected vulnerabilities. By using these findings as input and applying the concepts of collective risk modeling theory, the premium that an insurer can charge to provide insurance coverage for SaaS-Assets is determined.

The core contribution of this thesis lies in the design and implementation of a SaaS-Asset-Vulnerability Assessment (SA-VA) model. Following the principles of design science research methodology, the SA-VA model is developed to provide a structured approach for assessing and quantifying vulnerabilities specific to SaaS assets.

2. Cloud Computing History and Evolution

Cloud computing represents a novel approach to providing computing resources, rather than being a completely new technology (ENISA, 2009, p. 4). Cloud computing can help organizations expand their business capabilities and meet their computing needs without having to make large investments in infrastructure, training, personnel, and software (Horwath et al., 2012, p. 1).

International Data Corporation (IDC) forecasts whole cloud spending, which are total worldwide spending on cloud services, the hardware and software components underpinning the cloud supply chain, and the professional/managed services opportunities around cloud services, will surpass $1.3 trillion by 2025 while sustaining a compound annual growth rate (CAGR) of 16.9% (International Data Corporation, 2021). Worldwide end-user spending on public cloud services is forecast to grow 20.7% to total $591.8 billion in 2023, up from $490.3 billion in 2022, according to the latest forecast from Gartner, Inc. This is higher than the 18.8% growth forecast for 2022 (Gartner Inc., 2023b).

The idea of cloud computing can be traced back to 1961 when Professor John McCarthy mentioned it during MIT's centennial celebration; "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility… The computer utility could become the basis of a new and important industry." (Garfinkel, 1999, p. 1)

The evolution of cloud computing can be traced back to earlier systems such as utility computing and grid computing, which have been utilized in real-time applications long before the emergence of cloud computing.

Utility computing is a business model that revolves around the integration of servers, storage systems, and applications that are distributed globally and can be shared by multiple users, without necessarily relying on cloud computing. Essentially, it operates as a supercomputer that leases out its processing power to a diverse range of clients. Utility computing is primarily concerned with the business model of application infrastructure resources, which can encompass both hardware and software. Cloud computing, on the other hand, is focused

on the design, deployment, construction, and operation of applications within a virtual environment. It facilitates resource sharing and possesses the ability to dynamically scale up or down as well as self-heal when necessary. While cloud computing encompasses utility computing, not all utility computing is based on cloud technology. Utility computing represents a business model, whereas cloud computing refers to the underlying IT architecture.

Grid computing, another precursor to cloud computing, involves breaking down complex computing problems into smaller tasks and solving them on less powerful computers. Cloud computing is sometimes misunderstood and conflated with grid computing and utility computing. Grid computing refers to a distributed computing system where a collection of computers work together as a virtual machine to handle large-scale tasks. Utility computing, on the other hand, involves bundling multiple services, such as storage and facilities, into billable IT resources. The capabilities of cloud computing rely on the deployment of clusters (similar to grid computing) that encompass various functionalities found in utility computing. (Surbiryala & Rong, 2019). In Table 1 presented an overview of comparison of features between grid and cloud computing (Vachhani & Atkotiya, 2013, p. 55).

Feature	Grid Computing	Cloud Computing
Purpose	Collaborative sharing of resources	Use of service
Operational Approach	Grid needs processing from you	Cloud does the processing for you
Resource Distribution	Grid Computing divides everything	Cloud Computing Assimilates everything into one place
Network	Mostly internet with latency and low bandwidth	High-end with low latency and high bandwidth
Service Categories	CPU, network, memory, bandwidth, device, storage, etc	IaaS, PaaS, SaaS, Everything as a service

Table 1: Comparing Grid and Cloud Computing Features

2.1 Cloud Computing Definition

National Institute of Standards and Technology (NIST) defines cloud computing as: “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” (Mell & Grance, 2011, p. 2)

The ISO/IEC definition is also similar; “Paradigm for enabling network access to a scalable and elastic pool of shareable physical or virtual resources with self-service provisioning and administration on-demand.”(International Standards Organization, 2014)

To better understand the architecture and organization of cloud computing, it is essential to examine its underlying layers. Cloud computing can be explained in five layers (Youseff et al., 2008): Applications, Software environments, Software infrastructure, Software kernel and Hardware.

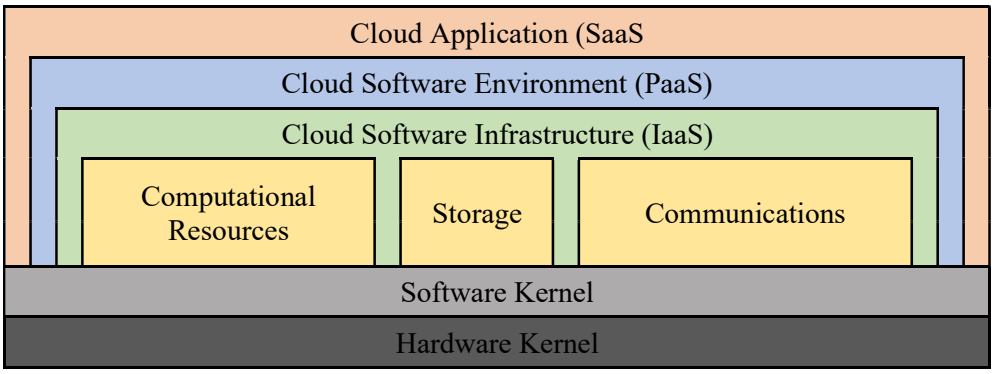

Table 2: The Layers of Cloud Computing

Hardware and Firmware Layer: At the foundation of the cloud stack lies the Hardware and Firmware layer. This layer comprises the physical infrastructure, including servers, storage devices, and network switches. These components provide the underlying computing power and resources required to support cloud services. Additionally, firmware, such as BIOS and device drivers, play a crucial role in ensuring hardware compatibility and efficient operation within the cloud environment.

Software Kernel Layer: This layer encompasses the software management systems that oversee and control the physical servers within the cloud. It provides essential functionalities like resource allocation, virtualization, and workload distribution. The Software Kernel layer acts as a bridge between the hardware infrastructure and the higher-level cloud software layers.

The cloud software infrastructure layer provides fundamental resources to higher-level layers. This layer offers essential services such as computational resources, data storage, and communication capabilities. Computational resources enable the execution of applications, while data storage facilitates secure and scalable data storage and retrieval. Communication services enable seamless connectivity and interaction between different components of the cloud ecosystem.

The cloud software environment layer is where developers interact with the cloud platform to build and deploy applications. Cloud providers offer programming-language-level environments that provide well-defined APIs, enabling developers to interact with the cloud infrastructure and services effectively. These environments streamline the deployment process and support the scalability required by cloud applications. They provide a range of tools, libraries, and frameworks to simplify application development, integration, and management.

The cloud application layer is the visible interface that allows users to access and interact with various services and applications provided by cloud providers. Typically, users access these services through web portals. It acts as a bridge between the users' terminals and the underlying cloud infrastructure, enabling the offloading of computational work to data centers and providing a seamless user experience.

2.2 Cloud Service and Deployment Models

Cloud computing has become a key enabler for businesses across various industries. Cloud Service Providers (CSPs) play a crucial role in delivering the necessary infrastructure, platforms, and software services to meet the evolving needs of organizations. In this section the different cloud services and deployment models will be explained.

The primary cloud cocomputing models featured in Table 3 and widely utilized today include Infrastructure as a Service, Platform as a Service and Software as a Service.

		IaaS	PaaS	SaaS
Managed by Customer		Applications	Applications	Applications
		Data	Data	Data
		Runtime	Runtime	Runtime
		OS	OS	OS
Managed by Provider		Virtualisation	Virtualisation	Virtualisation
		Servers	Servers	Servers
		Storage	Storage	Storage
		Networking	Networking	Networking

Table 3: Responsibilities in Different Cloud Service Models

Software as a Service (SaaS); The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. Examples: Google Apps, Adobe Creative Suite, Dropbox, Microsoft 365.

Platform as a Service (PaaS); customers will get the platform for the development of software applications. The capability provided to the consumer is to deploy onto the cloud infrastructure, consumer-created or acquired applications, created using programming languages, libraries, services, and tools supported by the provider. Google's App Engine is an example of PaaS, as anyone can build an app on Google's infrastructure.

Infrastructure as a Service (IaaS); Customers will get the services for a complete computing infrastructure over the Internet. The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. Examples: Amazon Web Services (AWS), Google Compute Engine (GCE).

Cloud computing offers different deployment models that determine how cloud services are provisioned and managed. These deployment models define the ownership, control, and accessibility of cloud resources. The main cloud deployment models are:

Public Cloud: In a public cloud deployment, cloud services are provided over the internet by a third-party cloud service provider. These services are accessible to multiple organizations or individuals. The infrastructure and resources are owned and managed by the service provider, who is responsible for maintenance, security, and updates. Users typically pay for the services they consume on a pay-as-you-go basis.

Private Cloud: A private cloud is dedicated to a single organization and is not shared with other users. It can be physically located on-premises within the organization's data center or hosted by a third-party provider. Private clouds offer greater control and customization options, allowing organizations to tailor the infrastructure to their specific needs. They provide enhanced security and privacy, as the resources are isolated from the public cloud.

Hybrid Cloud: A hybrid cloud combines both public and private cloud environments, allowing organizations to leverage the benefits of both. It enables seamless integration and portability of applications and data between the public and private cloud infrastructures. Organizations can utilize the public cloud for scalability, cost-effectiveness, and handling variable workloads, while keeping sensitive data or critical applications in a private cloud for enhanced security and control.

Community Cloud: A community cloud is a shared infrastructure that is designed and utilized by multiple organizations with common interests or requirements. It can be managed by the organizations themselves or by a third-party provider. Community clouds enable collaboration, resource sharing, and cost sharing among the participating organizations. They are often utilized by specific industries, such as healthcare or education, where there is a need for shared resources and compliance with industry-specific regulations.

These deployment models offer different advantages and considerations in terms of security, control, scalability, cost, and customization. The choice of deployment model depends on the specific requirements and goals of an organization.

By understanding their unique business needs, security requirements, and budget constraints, organizations can strategically select the appropriate cloud service and deployment models.

2.3 Cloud Service Providers: Definitions and Examples

The NIST identifies CSP as “A trusted entity that issues or registers subscriber authenticators and issues electronic credentials to subscribers. A CSP may be an independent third party or issue credentials for its own use” (Newhouse et al., 2019, p. 163).

Google defines CSP as; “A CSP is a third-party company that provides scalable computing resources that businesses can access on demand over a network, including cloud-based compute, storage, platform, and application services.” (Google Cloud, 2023) and for the Microsoft “A cloud service provider is a third-party company offering a cloud-based platform, infrastructure, application, or storage services” (Microsoft Azure, 2023).

Cloud providers of various sizes and types are part of the CSP market. The dominant players are the well-established leaders known as the big three: Google Cloud, Microsoft Azure, and Amazon Web Services. It is important to note that cloud environments often involve customized, mixed solutions tailored to the specific needs of customers, rather than strictly categorized PaaS, SaaS, or IaaS offerings. By examining following cases, we can better understand how customers derive value from cloud services and the distinct roles of cloud service providers and cloud providers.

An example of a cloud service provider (CSP) is Microsoft 365, previously known as Office 365. It is a Software as a Service (SaaS) solution built on Microsoft's Azure cloud platform. Microsoft 365 offers users access to a range of office productivity apps, including Microsoft Office, SharePoint for content and collaboration, Teams for chat-based communication, Exchange Online for email services, and Exchange Online Protection for protecting against spam, malware, and other threats. In the case of Microsoft 365, the CSP is the same company that owns the cloud computing platform (CP), Microsoft Azure.

To gain a clearer understanding of the differences among CSP, CP, and Customer, one can consider an additional example. Deloitte (CSP) offers cloud-based solutions that are built on Amazon Web Services (CP). These solutions provide companies (Customers) with applications like Smart Factory Fabric (SaaS-Solution). By leveraging these solutions, companies can enhance their operational performance, reduce costs, improve production

efficiency, enhance product quality, and minimize unplanned downtime in their smart factory operations.

“Smart Factory Fabric is a pre-configured suite of cloud-based, customized IoT applications built on AWS (e.g., machine monitoring, predictive maintenance) that can accelerate smart factory transformations.” (Deloitte, 2023)

Netflix relies on Amazon EC2 (Elastic Compute Cloud), an Infrastructure-as-a-Service (IaaS) solution provided by Amazon Web Services (AWS), to support its streaming platform.

When Netflix began providing online streaming services, its number of users grew significantly. To meet the needs of users with different interests, Netflix needed a solution beyond its existing data centers. AWS came to the rescue by enabling Netflix to dynamically scale its data warehousing capabilities in response to user demand. This allowed Netflix's engineers to focus on optimizing the application and enhancing the business, while AWS managed the underlying infrastructure, including computing, storage, and networking.

Netflix utilizes Amazon Web Services (AWS) to host their data and employs a Content Delivery Network (CDN) for serving user requests. The process begins when a video is submitted to Netflix and undergoes processing, encoding, and storage in the S3 system (cloud-based storage service) through the Lambda function (serverless computing service). The stored video is then transferred to a location within the CDN where it is more accessible to viewers. To handle playback requests, Netflix relies on the AWS Load Balancer, which directs these requests, sent by clients, to the API Gateway Service running on AWS EC2. When a playback request is received, the corresponding Play API is deployed, and similar APIs are deployed for subsequent requests.

Once the API is generated, it initiates a series of microservices to fulfill the request. However, there is a possibility of overlap between these microservices. To avoid this, Netflix has implemented Hystrix, a program that isolates each microservice from one another, ensuring they operate independently. Additionally, Netflix employs AWS Elastic Transcoder to format videos based on the specific needs and preferences of users (Infraveo Technologies, 2022).

2.4 Characteristics of SaaS Providers

SaaS providers offer software applications as a service, delivering them over the internet without requiring local installation. They handle the hosting, maintenance, and technical aspects of the software, allowing customers to access and use the applications remotely. By adopting a subscription-based model and leveraging cloud infrastructure, SaaS providers provide flexibility, scalability, and cost-effectiveness to their customers.

The main focus of this thesis will be on cloud service providers that primarily offer Software as a Service (SaaS) solutions. “SaaS is a difficult topic for a study since there is no one generally accepted definition of the concept. It is clear that SaaS is about selling a productized software service and doing this online. However, the exact borderlines between SaaS model and other e-commerce models are unclear.” (Mäkilä et al., 2010, p. 116).

If we review various definitions of Software-as-a-Service (SaaS) in the literature, typically, five distinct characteristics are linked to SaaS as discussed in (Mäkilä et al., 2010). These five characteristics are required for a firm to be considered a SaaS provider.

- The product is accessed via a web browser.
- The product is not customized for individual customers.
- The product does not entail the installation of software at the customer's premises.
- The product does not require complex integration and installation procedures.
- The pricing of the product is determined by the actual utilization of the software.

SaaS providers can enhance their competitive advantage through appropriate contracting strategies. They should develop applications that are highly compatible with various systems and programs. This can be achieved by writing applications using an open language, organizing them in a modular structure, and ensuring a flexible interface with other applications. Contracts should allow for smooth and quick exits, avoiding high cancellation fees and ensuring data transition in a timely manner. The future prosperity of the SaaS business model requires both technological advancements and strategic contracting (Ma, 2007).

2.5 The Role of SLAs in Cloud Liability and Insurance

NIST defines Service Level Agreement (SLA) as; "Represents a commitment between a service provider and one or more customers and addresses specific aspects of the service, such as responsibilities, details on the type of service, expected performance level (e.g., reliability, acceptable quality, and response times), and requirements for reporting, resolution, and termination." (NIST, 2021).

An SLA establishes the delivery capability of a provider, sets performance targets for user requirements, and outlines the guaranteed availability of services. The following are some reasons for using SLAs (Dash et al., 2014, p. 2900).

- It provides a clear understanding of cloud service providers.
- It provides a comprehensive list of the services offered by providers, including a detailed description of each service.
- It defines the purpose and objectives of business-level policies, including the roles of cloud service providers and users.
- It helps understand the essential security and privacy management policies in the cloud environment.
- It enables monitoring of service quality, performance, priorities, and responsibilities from a service-oriented perspective.
- It offers transparency by outlining the service management requirements in the event of a cloud service failure.

A typical Service Level Agreement (SLA) offered by a cloud provider comprises several key components that define the terms, conditions, and expectations of the service. These components include (Baset, 2012, pp. 57-58);

- Service guarantee: This outlines the specific metrics that the provider aims to fulfill within a designated service guarantee period. If these metrics are not met, the customer is entitled to a service credit
- The service guarantee time period refers to the timeframe within which the service guarantee must be fulfilled. This period can be measured in different units, such as a

billing month or the time elapsed since the last claim was made. It is also possible to have a relatively short time period, such as one hour.

- Service guarantee granularity refers to the level of detail at which a provider specifies a service guarantee in terms of resource scale. This means determining whether the guarantee applies to a specific service, a particular data center, an individual instance, or on a per-transaction basis. The service guarantee granularity can also be calculated by aggregating the resources considered, such as instances or transactions.
- Service guarantee exclusions refer to the specific instances or scenarios that are not considered in the calculation of service guarantee metrics.
- Service credit is a form of compensation provided by the service provider to make up for any shortcomings or failures in delivering the promised service.
- Service violation measurement and reporting refers to the way in which the violation of service guarantees is measured and reported, respectively.

Service Level Agreements (SLAs) often play a crucial role in addressing and mitigating risks. SLAs have gained significant importance as they establish the terms and conditions governing the provisioning and delivery of services, encompassing aspects related to security. These agreements incorporate financial penalties as consequences for unfavorable outcomes, allowing customers who have signed such agreements to seek compensation in the event of non-compliance. Non-compliance, particularly in the context of cloud computing, can manifest in various ways, such as the loss of quality of service, issues related to data privacy, vulnerabilities in the execution environment, conflicts arising from the execution locality concerning legal jurisdictions, and more (Morin et al., 2012).

Laws that impact the cloud cover both public and private aspects. Public law deals with the relationship between individuals and the government, as well as relationships that affect society as a whole. In relation to the cloud, this mainly involves the many rules created by governments worldwide to protect the privacy and security of their citizens' information. Examples include the Health Insurance Portability and Accountability Act (HIPAA) in the United States or the Data Protection Directive (EU 95/46/EC) in the European Union. Private law generally deals with relationships between individuals, like contracts or issues related to harm or injury. In the context of the cloud, this involves contracts between parties, such as cloud service providers and their clients. These contracts define their relationship and the

level of service expected from each other. It's common for these contracts to include obligations dictated by public law (Gordon, 2016).

It is important for both cloud service providers and their clients to be mindful of the numerous laws that impact their cloud operations. Among these laws, the most crucial ones are those established to safeguard the privacy and security of individuals' personal information. These laws dictate the recommended practices, guidelines, and limitations regarding how personal information should be handled and utilized. Unlike the United States, where data regulations are typically based on the specific industry it belongs to, the data protection laws in the European Union (EU) are considered nonsectoral. This means that these laws apply to all personal information, regardless of the industry or sector it is associated with.

In 2012, the European Commission introduced a preliminary version of the General Data Protection Regulation (GDPR) to establish consistent data protection laws across European Union member states and address concerns arising from new technologies, including cloud computing. Several changes in the GDPR are important for cloud service providers (CSPs) and their clients. These include higher penalties for noncompliance, requirements for notifications similar to those in the United States, adoption of privacy by design practices, the appointment of data protection officers within organizations, and the introduction of the "right to be forgotten." The "right to be forgotten" allows individuals to request the deletion of their data if there are no legitimate reasons for its retention. This right could have significant implications for data backups stored in the cloud. In the European Union, each member state has its own data protection authority. CSPs and their clients are likely to engage with these authorities to some extent, especially if they operate in a specific state or handle substantial amounts of personal information related to that state's citizens. It is important for them to be aware of the practices and guidelines set by the relevant data protection authority (Gordon, 2016).

Liability can arise due to various specialized legislations such as telecommunication law, electronic commerce law, data protection law (GDPR), and copyright, trademark, or patent issues. These laws create a legal framework that holds individuals or entities accountable for their actions in specific areas. For example, copyright and trademark laws safeguard intellectual property rights, preventing unauthorized use or infringement. Liability takes on different forms depending on the context. In the realm of cloud computing, civil liability

emerges from the contractual relationship between users and cloud service providers, governed by contract law in the user's country. Contract law establishes the foundation for regulating server availability, the consequences of direct and indirect data losses resulting from server downtime, as well as disaster recovery and backup measures (Weber & Staiger, 2014).

This legal framework outlines the obligations and responsibilities of both parties involved. Failure to meet these obligations can result in liability, potentially leading to legal consequences. Most cloud providers, in order to reduce their liability, enter into insurance contracts tailored to risks such as server downtime, power outages, and hacker attacks. To strike a balance between risks and costs, cloud providers analyze their business operations and categorize potential risks into security, data, privacy, regulatory risks, service failures, supplier risks, loss of operational control, and intellectual property liability. (Weber & Staiger, 2014)

While technological advancements continue, the damages covered by insurers tend to increase exponentially in the cloud, encompassing not only the provider's loss but also a portion of the customers' costs. As the market grows and legal frameworks, certification systems, and insurance schemes become more refined, insurance providers will expand their coverage accordingly.

2.6 Risk Assessment with ISO 31000

Organizations face a strategic challenge in managing risks, as they encounter a growing array of complex and varied threats. The ISO 31000 standard, introduced in 2009, aims to assist organizations in systematically and comprehensively managing different types of risks. By providing a universal framework, it enables organizations to integrate risk management into their overall management system. ISO 31000 offers a structured framework that caters to the requirements of any organization or situation. To accommodate the broad range of activities and risks, the standard's approach is fundamentally designed to be generic and logical (Lalonde & Boiral, 2012).

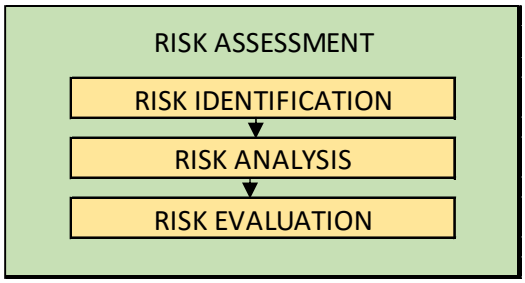

Table 4: ISO 31000 Risk Assessment Process

Risk assessment under ISO 31000 comprises as seen in Table 4 three steps, risk identification, risk analysis, and risk evaluation.

Risk identification serves the purpose of creating a comprehensive list of risks. It involves identifying events that have the potential to impact the achievement of objectives. These events can either create, enhance, prevent, degrade, accelerate, or delay the desired outcomes. The aim is to generate an inclusive inventory of risks that may affect the organization.

Risk analysis is a flexible process that can be conducted at various levels of detail, depending on the risk, analysis purpose, and available information, data, and resources. It can be qualitative, semiquantitative, quantitative, or a combination of these approaches, depending on the circumstances. The chosen method of analysis depends on factors such as the nature of the risk and the level of accuracy and precision required.

The next step is risk evaluation. This stage involves making decisions about the level of risk and determining its priority for attention. The evaluation process applies the criteria developed during the establishment of the risk context. It entails assessing the significance and potential consequences of the risks to determine the level of risk acceptability or tolerance. Through this evaluation, risks that require immediate attention or further treatment can be identified.

According to ISACA (Information Systems Audit and Control Association), the COSO ERM and ISO 31000 Frameworks lack emphasis on IT risk and fail to address specific IT domains (Isaca, 2009, p. 8). Large organizations face a challenge in effectively handling and managing IT risks due to the limitations of fully implementing the COSO ERM and ISO 31000 standards and recommendations. To address this issue, enterprises need to establish their own IT Risk Management framework. This involves the measurement, control, and management

of various IT risks, which presents a new and daunting challenge for companies. The registration and resolution of different risks prove to be difficult and unresolved (Kovácsné Mozsár & Michelberger, 2018).

IT risk is business risk. It refers to the risks associated with using, owning, operating, being involved with, influencing, and adopting IT in a company. It includes events and conditions related to IT that could potentially impact the business. These risks can happen at uncertain times and with varying levels of impact, creating difficulties in achieving strategic goals and objectives (Isaca, 2009, p. 11).

The assessment of risks in traditional IT environments differs from that of cloud computing. When it comes to migrating internal IT data and applications to the cloud or hosting customer data in the cloud, the level of risk varies depending on factors such as the sensitivity of the asset, the chosen cloud service/delivery model, the cloud architecture, and the security controls in place. To properly assess cloud risks, it is essential to proactively identify security risks throughout the supply chain and implement effective safeguards to mitigate them. Some cloud risks stem from the dynamic nature of the cloud's supply chain, while others arise from the immaturity of service providers' offerings, limited transparency, and the subjective nature of expert claims (Akinrolabu et al., 2019).

Although numerous studies have focused on investigating and addressing risks and concerns faced by cloud consumers, the perspective of Cloud Service Providers (CSPs) is seldom discussed in the literature. Because of the complexity involved in provisioning cloud services, traditional risk assessment frameworks are not effective in the context of cloud computing. CSPs need to adopt a comprehensive approach to managing cloud risks. This approach entails augmenting the abstract knowledge of risks derived from traditional frameworks such as ISO/IEC 27005 with specific technical factors that drive cloud risks, as provided by lower level conceptual models (Akinrolabu et al., 2019).

A vulnerability refers to the chance of a threat having a negative effect on an asset. When an asset is vulnerable to a threat, there is a risk involved (Rainer Jr et al., 1991). In cloud computing, an asset typically refers to any resource that is available in the cloud environment, such as computing power, storage, networking infrastructure, software and data that are hosted in the cloud. These assets are typically virtualized and can be rapidly provisioned on-

demand to meet the needs of cloud users. Assets can be owned and managed by the cloud provider or by the user, depending on the service model being used.

The decision to use ISO 31000 for risk assessment in this thesis was driven by the framework's international recognition, comprehensive approach, incorporation of best practices, and standardized communication. This comprehensive approach enables informed decision-making, effective risk mitigation, and improved overall risk management practices. Within this thesis, the process of risk identification is conducted through a detailed and comprehensive literature review, presented in Section 3.2.

3. Designing of SaaS-Asset-Vulnerability Assessment (SA-VA) Model

In this Chapter, first I introduce the design science research methodology (DSRM) used in this thesis. The DSRM is a problem-solving approach that seeks to create innovative artifacts to solve important problems in a specific domain. This thesis research utilizes the DSRM to develop a vulnerability assessment model for SaaS-Asset.

The identification and study of vulnerability factors in cloud computing are essential for businesses to set proper countermeasures to avoid and minimize them. Vulnerabilities within cloud liability are not only related to security breaches but also include lawsuits filed by or against the user. The vulnerability assessment approach developed in this thesis takes into account the ISMS (Information Security Management System) standard and benchmarks that provide a framework for managing and regulating information security in business organizations.

The vulnerability assessment model developed in my thesis aims to assess the essential cloud liability failure incurred by the SaaS providers in cloud computing. Based on a systematic literature survey, I classified cloud liability failure into three categories: interruption of information access, compliance failures, and intellectual property issues.

3.1 Design Science Research Methodology

For my thesis research, I have utilized the design science research methodology (DSRM) as presented by (Peffers et al., 2007), and aligned my work with the seven guidelines for design science research by (Hevner et al., 2008). The reason for selecting a design science approach for my thesis is that it seeks to solve important problems in a unique or innovative way. The arrived solutions is validated and demonstrated by showing its implementation in a use case.

In the Information Systems discipline, two paradigms are commonly used: behavioral science and design science. Behavioral science focuses on developing and verifying theories to explain or predict human or organizational behavior. Design science, on the other hand, aims to expand the boundaries of human and organizational capabilities by creating new and innovative artifacts. Design science is essentially a problem-solving process that aims to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently accomplished.

The core principle of design-science research is that knowledge and understanding of a design problem and its solution are gained through the building and application of an artifact. An artifact, in this context, refers to an innovative and purposeful creation that solves a problem in a specific domain. It could be a physical product, a software program, a new process, or any other tangible or intangible output. It must be evaluated thoroughly to ensure its usefulness in the specified problem domain. The artifact should also be novel, addressing a previously unsolved problem or finding a more efficient solution to an existing one. This sets design-science research apart from the practice of design, which may not have a specific purpose or novelty requirement.

The artifact itself must be rigorously defined, formally represented, coherent, and internally consistent. The process of creating the artifact often involves a search process to construct a problem space and propose a mechanism to find an effective solution. The results of the design-science research must also be communicated effectively to both researchers who will extend or study them and practitioners who will decide whether to implement them in their organizations (Hevner et al., 2008, p. 76).

3.2 Vulnerability Assessment Approach of This Thesis

Information technology (IT) in many businesses management plays a significant role. In the present world where IT plays a very considerable work in the development of any business organization, identification and study of vulnerability factors are very essential. Once the vulnerability factors are identified, then only proper countermeasures can be set to avoid and minimize them.

There is an important demand to identify, handle and fix the security issues of Cloud Computing. For fulfilling this demand, developer is trying to find out different types of cloud security vulnerabilities. There are different kinds of cloud vulnerabilities identified and listed by NIST on NVD (National Vulnerability Database). Vulnerability identification is a dynamic task that is identified frequently to handle this issue using different security tools/models that have been developed and designed to support the vulnerabilities identification process (Mishra et al., 2020).

Vulnerabilities within the cloud are partially related to security breaches. But it also includes lawsuits filed by or against the user. According to a survey, the current vulnerability in cloud computing via services is authorized liability and business continuity (Weber & Staiger, 2014).

Thesis´s vulnerability assessment model takes into account ISMS (Information Security Management System) standard and benchmarks, which govern information security in business enterprises. ISMS provides governance that enables the organization to establish, implement, maintain and improve the enterprise security system (Al-Dhahri et al., 2017).

Though there are several standards for IT governance, however I identified two important ISMS policies which govern my SA-VA Model as seen in Table 5 , first one is the COBIT (Control Objectives for Information and Related Technologies) developed by ISACA and the second framework is COSO developed by Committee of Sponsoring Organizations of the Treadway Commission. These policies governed the design, implementation, and maintenance of my SA-VA model and are aimed to reflect the alignment of the IT strategy with the business strategy (Mangalaraj et al., 2014).

The vulnerability assessment described in this thesis is aimed to measure the essential cloud liability failure incurred by the SaaS providers in cloud computing.

Based on the study from the literature survey (Lu & Xu, 2015), (Weber & Staiger, 2014), (Melzer, 2010), (Evans et al., 2016), I classified the cloud liability failure (CLF) into three (i) interruption of information access such as human error, power outage, maintenance, and threat in IT (ii) Compliance Failures such as the regulatory compliance and the security and privacy policy (iii) intellectual right issues such as patent, copyright, and trademarks.

My method is considered to be more comprehensive than that of many studies which only focus on threat and vulnerability that can cause mild to critical barriers to the network security components of the cloud infrastructure. This claim is supported by various research papers, including the example of a review on (Ahmed & Hossain, 2014) highlighting the importance of robust and consistent security in a cloud-based computing environment.

Vulnerabilities		ISMS frameworks			Literature
		ISACA	COSO	CSA	
Interruption of IA		**Y**	**Y**	**Y**	(Li et al., 2013), (Gunawi et al., 2016), (Weber & Staiger, 2014), (Soewito et al., 2022), (Hon et al., 2012)
	Maintenance	Y	-	-	
	Power Outage	Y	-	-	
	Human Error	Y	Y	Y	
	Dos Attacks		-	Y	
Compliance Failure		**Y**	**Y**	**Y**	(Muller & Supatgiat, 2007), (Yimam & Fernandez, 2016), (Martens & Teuteberg, 2011), (Weber & Staiger, 2014), (Soewito et al., 2022), (Hon et al., 2012)
	Regulatory Compliance	Y	Y	Y	
	Security & Privacy Policy	Y	Y	Y	
Intellectual Property		**Y**	**Y**	**Y**	(Weber & Staiger, 2014), (Lu & Xu, 2015), (Hon et al., 2012), (Melzer, 2010)
	Patents	Y	-	Y	
	Copyright	Y	-	Y	
	Trademarks	Y	-	Y	

Table 5: Vulnerabilities, Origins, and References

3.3 Interruption of Information Access as Vulnerability

This group of anomalies encompasses technical aspects of the information delivery, including risks caused, for example, by DOS attacks. These kinds of technical problems are most likely to frequently attract liability. An example of the interruption of information access (IA) is downtime due to routine maintenance, technical failures such as power outages, human error, and the denial-of-service (DOS) attack. This category of liability is most likely to frequently cause potential economic loss (Weber & Staiger, 2014).

SaaS providers are responsible for the maintenance of their IT infrastructure. Customers only pay for the services consumed, while CSPs continuously invest in improving their services. This includes both the scheduled and unscheduled maintenance and upgrades, which may involve service interruptions and even full system restarts. In such cases, the service may become temporarily unavailable to end-users, leading to a cloud outage.

Power outages refers to a disruption in the supply of electricity, whether caused by technical or natural factors, pose a critical challenge for the cloud-based industry. They are a major cause of service downtimes, leading to adverse reactions and severe consequences for service reliability and customer experience. To mitigate these issues, modern cloud service providers equip their power supply systems with multiple sources (Rahi et al., 2017).

Human errors pose risks to system reliability, leading to outages and disruptions. Literature sources highlight various categories of human errors, including misconfigurations of hardware, operating systems, applications, and devices, as well as failures to upgrade or right-size servers and neglecting patches and security updates (Gunawi et al., 2016). Additionally, classic mistakes like unplugging power cords or leaving critical ports open contribute to errors. Inadequate training, budget-driven decision-making, and lapses in judgment by executives and IT departments also play a role. Upgrade and configuration issues, such as incorrectly executing traffic shifts and neglecting necessary networking changes, are common sources of human errors. Post-failure situations requiring human intervention, like reintroducing faulty storage nodes without proper protection, also contribute to errors (Laura Didio, 2018).

DDOS stands for Distributed Denial of Service, and it involves multiple systems attacking a cloud service by flooding it with fake traffic. This attack is executed by the attacker who controls a botnet, which is a collection of compromised systems infected with malware. The attacker commands the botnet to send a large number of requests to the cloud service, making it unavailable to legitimate users (Suryateja, 2018).

On the other hand, a DoS attack, which stands for Denial of Service, is performed by a single source machine aiming to disrupt the availability of a system. In this attack, the attacker floods the target server with a high volume of requests, overwhelming the service queue and causing the service to become unavailable. DDoS attacks, being a variant of DoS attacks, are more prevalent and pose a significant threat (Somani et al., 2017).

It's important to note that while security measures can be implemented to prevent and mitigate these attacks, there can be unintended consequences. Disabling malicious applications to stop attacks may inadvertently affect well-behaving applications as well. Additionally, cloud services may need to temporarily shut down to apply security patches as a preventive measure against future security attacks (Gunawi et al., 2016).

3.4 Compliance as Vulnerability

Compliance, as discussed in (Mogull et al., 2017), is the adherence of an organization to established laws, regulations, standards, and specifications. In the context of cloud computing, ensuring compliance is very important due to the presence of various security and privacy laws and regulations at different levels—national, state, and local—within different countries (Jansen & Grance, 2011, p. 15). Failure to comply with these requirements can result in legal and financial consequences, reputational damage, and loss of customer trust.

Compliance requirements can vary across countries, with government and state regulations being mandatory and industry regulations serving as suggestions. Compliance issues in cloud computing arise from the cross-border nature of cloud services. Privacy laws can conflict between different jurisdictions, leading to challenges in meeting data protection provisions. For instance, a European cloud provider sending data to the United States must demonstrate

compliance with the EU-U.S. Data Privacy Framework to adhere to the General Data Protection Regulation (GDPR). This highlights the complexity of compliance in cloud computing, as cloud providers must navigate multiple legal frameworks and ensure adherence to each jurisdiction's specific requirements (Weber & Staiger, 2014).

The risks associated with non-compliance are significant. Non-compliance with privacy rules can infringe upon individuals' privacy and data protection rights. This can result in legal actions, regulatory penalties, and damage to an organization's reputation. Additionally, compliance failures can lead to data breaches, unauthorized access, and loss or theft of sensitive information, as stated in (Kumar et al., 2018). Such incidents can have severe financial and operational consequences, including financial losses, litigation, and erosion of customer trust.

To ensure compliance in the cloud, organizations should establish contracts with cloud service providers that define data protection standards and include service level agreements (SLAs) outlining security and privacy practices. These agreements should include provisions for technical controls like end-to-end encryption and data loss prevention tools to safeguard data and ensure compliance with regulations. Organizations must actively manage compliance by regularly monitoring and reporting on the provider's privacy and security practices (Kumar et al., 2018).

Service providers and customers in the cloud are expected to be fully compliant and may need to adhere to multiple regulations (Yimam & Fernandez, 2016). Cloud customers must also rely on third-party certifications and reports to assess the compliance alignment of cloud providers, as public cloud providers often engage third-party firms for audits and certifications. It's important to note that compliance in the cloud is a shared responsibility, with customers also responsible for their own compliance obligations (Mogull et al., 2017).

3.5 Intellectual Property as Vulnerability

NIST defines Intellectual property as (NIST): “Creations of the mind such as musical, literary, and artistic works; inventions; and symbols, names, images, and designs used in commerce, including copyrights, trademarks, patents, and related rights. Under intellectual property law, the holder of one of these abstract “properties” has certain exclusive rights to the creative work, commercial symbol, or invention by which it is covered.”

Intellectual property (IP) vulnerabilities in cloud computing can have significant implications for both cloud customers and cloud service providers. These vulnerabilities mainly involve patent, copyright, and trademark infringements, which can result in legal consequences and liability concerns. The interpretation and protection of these rights can vary across different jurisdictions. For instance, while the European Union (EU) clarified that the underlying ideas and principles of computer programs are not protected by copyright, the United States (US) patent laws may offer such protection. This disparity in legal frameworks creates complexities and conflicts for cloud customers and providers (Weber & Staiger, 2014).

Patents play a vital role in protecting inventions and innovations within the realm of cloud computing. However, issues related to patent infringement can arise, leading to potential liabilities for both cloud service providers and users (Phelps & Jennex, 2015). For example, patent issues in the cloud are exemplified by the rise of non-practicing entities (NPEs). These entities acquire patents solely for litigation purposes, targeting cloud service providers (CSPs) for infringement. NPEs' access to resources and immunity to counterclaims pose significant intellectual property risks to CSPs and their customers. The prevalence of NPE activities and the vulnerability of open-source software further exacerbate the challenges (Kemp, 2020).

Copyrights serve as a safeguard for original creative works, such as software, music, literature, and artistic creations. In the context of cloud computing, unauthorized use, distribution, or reproduction of copyrighted materials can result in substantial liability. For example, consider a situation where an employee of a company uploads a copyrighted document to a cloud storage service and shares it with unauthorized individuals. If the copyright holder discovers this unauthorized sharing, both the employee and the company may be held liable for copyright infringement (Phelps & Jennex, 2015).

The unauthorized use of trademarks can also lead to liability concerns in cloud computing. Trademarks are designed to protect brand identity, including logos, names, and symbols associated with goods and services. When cloud service providers employ trademarks without proper permission, it can result in trademark infringement and dilution (Weber & Staiger, 2014).

Embracing the cloud introduces potential security risks for sensitive intellectual property information. Cybercriminals, including intellectual property spies, target valuable IP assets. Malware and phishing are common techniques used in data breaches, but intellectual property spies, especially current or former employees, can be more sophisticated and malicious. System access abuse and privileges misuse are significant causes of IP breaches. Employee negligence and user errors also pose concerns, particularly as cloud-based file synchronization enables access to IP information on personal mobile devices (CIDON, 2015).

To effectively mitigate IP vulnerabilities, cloud customers should place a high priority on obtaining contractual indemnity to protect against potential IP violations arising from the software provided by the cloud service provider. It is essential to implement thorough employee training and enforce stringent controls to minimize the sharing of sensitive knowledge within file-sharing platforms. In terms of securing IP data stored in the cloud, encryption plays a central role by ensuring that data remains encrypted throughout its entire lifecycle. By employing access control mechanisms through encryption keys, authorized users can decrypt files while effectively preventing unauthorized access. (CIDON, 2015).

Intellectual property rights issues such as patents, trademarks, or copyrights are considered the most important aspect whose infringement is worth defending (Cooper & McCloskey). This is because IP is considered the most valuable asset of the organization. This thesis takes into account the assessment of security and privacy vulnerabilities in cloud computing as a potential contributor to intellectual property breaches.

3.6 SA-VA Model Mathematical Formulation

The mathematical formulation of the SA-VA Assessment model combines all the Vulnerabilities described in the DAG. The directed acyclic graph (DAG) described in Figure 1 shows that 13 different vulnerabilities are represented as nodes. The relationship between the variables is described by the arc that shows interdependent variables. Each of these variables $(X_1, X_2, \dots, X_{13})$ is an uncertain event that can be modeled with marginal or prior probability which can be obtained from several studies in the organization. The variables are discrete ordinal variables, and their correlation (ρ) is represented by the variance-covariance matrix.

Similar to the study from (Böhme, 2005), each risk is modeled as a random variable x_i with a non-negative distribution of claim amounts. A collection of the homogenous risk for the organization described in Figure 1 can be represented as a random vector with the length n.

Each vulnerability parameter has 7 states (very low, low, lowish medium, medium, highish medium, high, very high), which represent the discrete state for the number of occurrences of the attack. The aim is to measure the continuous probability of attack in the different states. An example could be the probability that there is low, moderate, or high compliance failure for a given month.

When nodes depend on each other, the vulnerable event is defined as a conditional probability that measures the likelihood of a security breach in one node being caused by other nodes. These causative nodes are represented by the observed set S_Q, while the node itself is represented by the evidence set, S_E. Each node event represents the cumulative probability.

The set of S_Q or S_E is assumed to be disjoint for all practical purposes. In the SA-VA modeling, the conditional probability defined between two nodes are described below:

$$\text{Based on} \quad S_E, S_Q \in U\ ; \quad U = \{X_1, X_2 \ldots\ldots\ldots X_{13}\}$$

$$0 \leq \rho(X_i, X_j) \geq 1, \text{when there is an arc}$$

$$\rho(X_i, X_j) = 0 \ \text{ otherwise}$$

$$\boldsymbol{E_i = P(S_Q | S_E)} \qquad \text{Equation 1}$$

The SA-VA formulation aimed to determine the conditional probability between the observed set and evidence set, as described in Table 6, based on the model in DAG.

Event	S_Q	S_E	Detail
E_1	Maintenance. MA	Interruption of IA, IA	P(MA,IA)
E_2	Power Outage, PO	Interruption of IA, IA	P(PO,IA)
E_3	Human Error, HE	Interruption of IA, IA	P(HE,IA)
E_4	DOS, DO	Interruption of IA, IA	P(DO,IA)
E_5	Regulatory Compliance, RC	Compliance Failure, CF	P(RC,CF)
E_6	Security and Privacy policy, SCP	Compliance Failure, CF	P(SCP,CF)
E_7	Security and privacy policy, SCP	Intellectual Property, IP	P(SCP,IP)
E_8	Patents, PA	Intellectual Property, IP	P(PA,IP)
E_9	Copyright, CO	Intellectual Property, IP	P(CO,IP)
E_{10}	Trademarks, TR	Intellectual Property, IP	P(TR,IP)
E_{11}	Interruption of IA	Cloud liability Failure, CLF	P(IA,CLF)
E_{12}	Compliance Failure, CF	Cloud liability Failure, CLF	P(CF,CLF)
E_{13}	Intellectual Property, IP	Cloud liability Failure, CLF	P(IP,CLF)

Table 6: Event-Related Vulnerabilities and Connections

4. Statistical Implementation of SA-VA Model

This Chapter begins with an explanation of Directed Acyclic Graph, as a type of graph used in Bayesian networks, allowing for efficient computation of joint probabilities. The Bayesian Belief Network (BBN) is employed to assess the vulnerabilities causing cloud liability failure associated with DAG. The BBN is a simple graphical model that models the relationship between causal variables. The success of the Bayesian method in assessing vulnerabilities is also explained by a literature review.

Copula Sklar Theory and Kendall Correlation are then presented. The copula function is used to express the joint distribution of random variables based on their marginal distributions. In Bayesian copula modeling, Kendall's correlation coefficient is often used as a measure of dependence between variables.

The use of the Gaussian Copula Function in the SA-VA model is also explained. The Gaussian copula is a commonly used copula function for modeling the joint distribution of variables with linear correlation, assuming normal marginal distributions and capturing the dependence structure.

After elucidating the statistical fundamentals of the SA-VA model, an overview of the algorithm is given. The algorithm comprises four functions employed in the SA-VA model for estimating vulnerabilities. Also, a brief explanation of the MATLAB implementation is provided. For readers interested in gaining a deeper understanding, the code samples and explanations are attached at the end of this thesis.

The SA-VA model will be validated at the end of this Chapter, where the data required for validation is also given and described as input. The results of the validation will be discussed in Chapter 5.

4.1 Directed Acyclic Graph

A Directed Acyclic Graph (DAG) is a type of graph used in Bayesian networks to model the relationships between random variables. In a DAG, each node represents a random variable, and each edge represents a direct probabilistic dependency between two random variables.

The acyclic property of the graph means that there are no cycles or loops in the graph. This is important because cycles would create feedback loops in the model, which can lead to inconsistencies and make it difficult to calculate the probabilities of the different events in the network (Heckerman, 2008).

One of the key advantages of using a directed acyclic graph (DAG) for Bayesian calculations is its ability to enable efficient computation of joint probabilities. This reason led to its inclusion in my thesis, as depicted in Figure 1.

By breaking down the relationships between the different variables into a set of conditional probabilities, the DAG can be used to calculate the probability of any event in the network by multiplying the appropriate conditional probabilities.

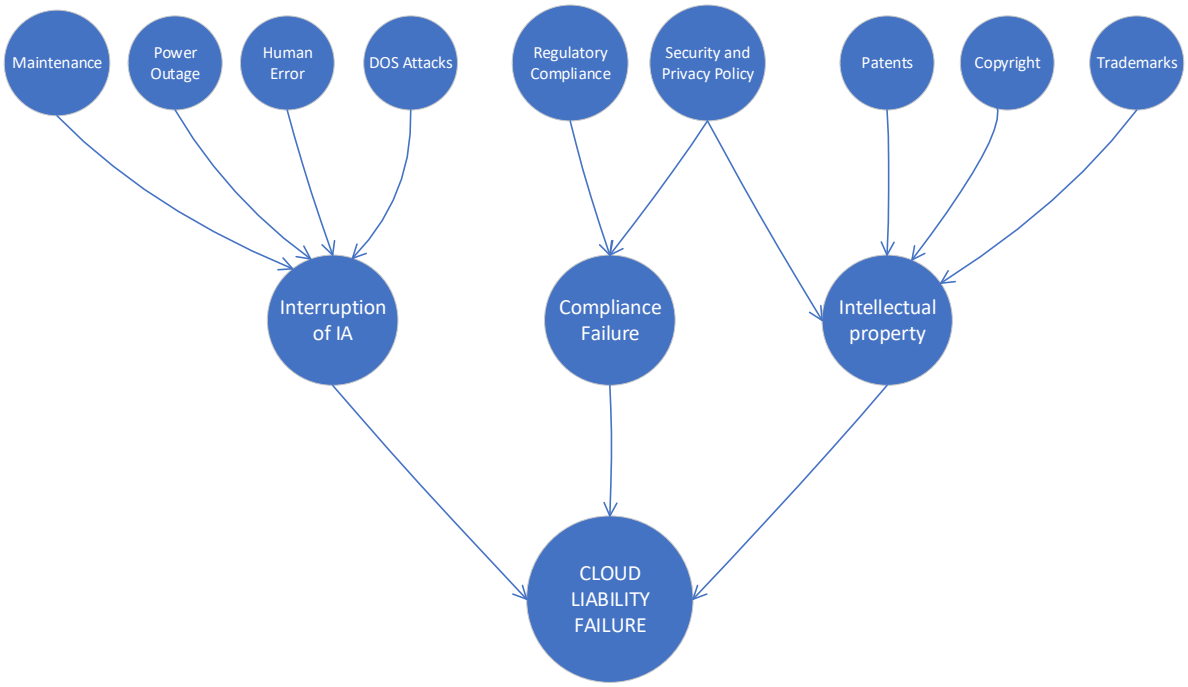

Figure 1: Directed Acyclic Graph for SA-VA Model

4.2 Bayesian Belief Network

Literature review showed, how the Bayesian network is effective for intuitive modeling and its capacity to assess the likelihood of attack when there are several IT breaches (Fenton & Neil, 2011, p. 1). Consequently, In order to examine the cloud liability vulnerabilities associated with the DAG depicted in Figure 1, I adopt the Bayesian Belief Network (BBN), as demonstrated by (Mukhopadhyay et al., 2013). BBN is an effective graphical model that captures the relationships between causal variables.

In order to apply BBN to SA-VA model, I begin by determining the prior belief of each node in the DAG. This can be obtained either through expert opinion or by utilizing the empirical method, as described by (Mukhopadhyay et al., 2006). BBN is a machine learning algorithm capable of learning from new data without the need for explicit programming. Therefore, it would also be suitable for interpreting data in the event of future modifications. It also considers the correlation between the nodes.

The marginal probability tables are utilized to define each node's prior belief. The root or parent node is regarded as the cause or observed in every pair of nodes connected by an arc and has a marginal probability table.

The child node, also known as the evidence node, has a conditional probability. The conditional probability between the query node and the evidence node is computed based on Bayes' theorem, which is defined as described below:

$$\boldsymbol{P(S_Q|S_E) = \frac{P(S_Q \cap S_E)}{P(S_E)}} \qquad \text{Equation 2}$$

$P(S_Q|S_E)$ is the conditional probability that defines the probability of occurrence of the query node, S_Q given that evidence node, S_E had already occurred.

$P(S_Q \cap S_E)$ represents the joint probability of both the query node and the evidence node occurring simultaneously.

$P(S_E)$ represents the probability of the evidence node occurring.

The conditional probability for a specific (S_Q, S_E) pair is calculated by incorporating the independence assumptions suggested by the directed acyclic graph. According to the study on the organization's IT security breaches, the prior probability of each evidence node is independent of the others. By incorporating the independence assumptions from the DAG, the computation of the conditional probability can be simplified (Norvig, 2016, p. 495).

Since there are several nodes connected to a query node, I apply the Equation 3 to aggregate the set of evidence nodes to the parent node. Therefore, conditional probability between a query node and a collection of evidence nodes is defined as the total of each conditional probability.

$$\boldsymbol{P(S_Q|S_E \dots)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{P(X_i|Parents(X_i))} \qquad \text{Equation 3}$$

As a result, the Bayesian intuitive model can be used to calculate the cumulative probability of each event in the SA-VA.

4.3 Copulas and Correlation Measurement

In Bayesian modeling, the prior knowledge is represented by a prior distribution, which is updated using the likelihood function to generate a posterior distribution. The copula function is used to model the dependence structure between random variables, and it is defined as a multivariate distribution function with uniform marginals. The copula function expresses the joint distribution of the random variables as a function of the marginal distributions. The SA-VA model would utilize the copular function to create a dependency structure between the vulnerabilities.

The Sklar theorem states that any multivariate distribution function can be uniquely decomposed into a marginal distribution function for each variable and a copula function that captures the dependence structure between the variables. Considering n random variables $x_1, x_2, x_3, \dots x_n$ that is defined on the same probability space such that each random variable has a distribution function F_i.

In the SA-VA model, the marginal distribution is derived from the joint probability distribution through a mathematical theory, as expressed in Equation 4 in terms of the marginal probabilities. The Sklar theorem illustrates this mathematical relationship as follows (Clemen & Reilly, 1999, p. 209):

$$\boldsymbol{F(x_1, x_2 \ldots, x_{13}) = C\,\{F_1(x_1), F_2(x_2) \ldots, F_{13}(x_{13})\}} \qquad \text{Equation 4}$$

Where,

$C = copula\ function$

$F(x_1, x_2 \ldots, x_{13}) = joint\ probability\ function$

$F_1(x_1), F_2(x_2) \ldots, F_n(x_{13}) = marginal\ prob.distr.for\ each\ of\ the\ \ var.\ x_1, x_2 \ldots, x_{13}$

The joint probability function $F(x_1, x_2 \ldots, x_{13})$ can be rewritten as described below:

$$\boldsymbol{F(x_1, x_2 \ldots, x_{13}) = f_1(x_1) \ldots \times f_{13}(x_{13}) \times c[F_1(x_1), F_1(x_2), \ldots, F_{13}(x_{13})]} \qquad \text{Equation 5}$$

From the Equation 5 it can be observed that copula density (c) encodes information about dependence among the variables $(x_1, x_2 \ldots, x_{13})$. This describes the joint probability function that can be expressed with the copula function on a 13-dimensional probability density function. However the vulnerability assessment model presented in this thesis simplified the estimation of joint probability between a query variable and the evidence variable.

Using the multivariate normal copula requires the correlation measurement for probability assessment between the nodes (Clemen & Reilly, 1999). The Correlation measurement is used to estimate the monotonic relationship, which is required to compute the joint probability of pair of variables. The common correlation measurements in Bayesian belief networks include Pearson, Spearman, and Kendall (Schober et al., 2018, p. 1763). The correlation among the vulnerabilities in the SA-VA model's random variables is obtained for every pair in the DAG using Kendall correlation measurement based. The obtained correlation matrix is referred to as R*.

4.4 Gaussian Copula to Model Correlated Risk

There are two types of copula functions commonly used to estimate the joint empirical probability: the Gaussian copula and the Archimedean copula. Among these, the Gaussian copula is considered the most prominent choice, particularly for risk management and financial analysis of assets (Elidan, 2010, p. 1). The Gaussian copula is derived from the multivariate Gaussian distribution and is completely determined by the correlation matrix. It uses a bivariate normal distribution with a mean of 0 and a variance of 1. Additionally, it requires the linear correlation among the nodes in the Directed Acyclic Graph (DAG). The parameters involved in describing the Gaussian copula are relatively simple to estimate.

In the SA-VA model, the Gaussian copula is used to estimate the joint probability between the dependent variables, assuming there exists a linear correlation, and the variables comply with the normal distribution function. The correlation matrix (R) is obtained from literature review conducted for the purpose of demonstrating the SA-VA model. Including the correlation matrix and assuming a multivariate normal density Equation 5 can be written as follows:

$$\boldsymbol{\phi}^{(13)}(\mathbf{y_1}, \dots\dots\dots\dots \mathbf{y_{13}} \mid \mathbf{R}^*) = \boldsymbol{\phi_1}(\mathbf{y_1}) \dots\dots\dots\dots \times \boldsymbol{\phi_{13}}(\mathbf{y_{13}}) \times \mathbf{c_{13}}[\boldsymbol{\Phi}(\mathbf{y_1}), \dots\dots\dots\dots \boldsymbol{\Phi}\mathbf{y_{13}}) \mid \mathbf{R}^*] \qquad \text{Equation 6}$$

Here ϕ and Φ denote univariate standard normal distribution respectively. The Gaussian Copula formula for the SA-VA model is derived as follows (Mukhopadhyay et al., 2013, p. 18):

$$\boldsymbol{c}[\boldsymbol{\Phi}(\mathbf{y_1}), \dots\dots\dots\dots \boldsymbol{\Phi}\mathbf{y_{13}}))] = \frac{\frac{\mathbf{e}^{\left\{-\frac{1}{2}\boldsymbol{y}\times\mathbf{R}^{-1}\times\boldsymbol{y}^{\mathbf{T}}\right\}}}{\sqrt{\mathbf{2}*\mathbf{13}\times|\mathbf{R}^*|}}}{\frac{\mathbf{e}^{\left\{-\frac{1}{2}\boldsymbol{y}\times\mathbf{I}\times\boldsymbol{y}^{\mathbf{T}}\right\}}}{(\sqrt{\mathbf{2}*\mathbf{13}})^{\mathbf{13}}}} = \frac{\mathbf{e}^{\left\{-\frac{1}{2}\boldsymbol{y}\times(\mathbf{R}^{-1}-\mathbf{I})\times\mathbf{y}^{\mathbf{T}}\right\}}}{|\mathbf{R}^*|^{1/2}} \qquad \text{Equation 7}$$

In SA-VA model, the choice of marginal distribution is obtained from a normal cumulative probability density function, which can be estimated from the mean and variance of the node.

Based on provided data, each node of the causal diagram modeled as normal distribution and aggregated using the Gaussian Copula.

For a more comprehensive understanding of the statistical aspects of the model, additional information can be found in following references: (Clemen & Reilly, 1999), (Nelsen, 2006, pp. 24-28), (Al-Adilee, 2014, pp. 201-202) and (Mukhopadhyay et al., 2013, pp. 16-18).

4.5 SA-VA Model and Risk Quantification Algorithm

The algorithm employed in the SA-VA model for vulnerability assessment utilizes a Copula-aided Bayesian belief network approach and consists of four functions, which are detailed as follows.

I. Node_df ()

This function returns the normal probability density function (pdf) and the normal cumulative density function (cdf) for each of the nodes using the mean, variance, lower bound and upper bound values of the node. The density functions represent the node distribution function that is useful to estimate the vulnerability report.

II. gen_jdf()

This function creates joint probability distribution between nodes that are connected with the arc in the DAG. The function takes the node probability distribution obtained in Node_df() and correlation Matrix, R as the input and produces a joint distribution using the copula function.

III. Gen_cp()

This function generates the marginal probability for the evidence set by summing all the joint probability distributions connecting the evidence set.

IV. Freq_of_failure

This function computes the posterior probability distribution of the event nodes, which is used to estimate the risk frequency considering the range of attack.

Procedure SA-VA ()

Input: DAG, Number_of _Nodes, Mean, Variance, lower_bound, upper_bound, Correlation Matrix (R)

Output: Risk frequency

```
Function Node_df (i, mean(i), variance(i), lowerbound(i),upperbound(i))
        return   pdf(i), cdf(i)
End Node_df
```

```
Function gen_jdf(R,node)
 For i = 1: Number_of_nodes
     For j = 1: Number of nodes
         if R(i,j) > 0
             Jdf(I,j) = Copula(Node_cdf(i), Node_cfd(j))
         End
    End
End gen_jdf
```

```
Function gen_cp(Jdf,MD)
        For i = 1:Number_of_event_node
          MD = ∑JDF(Node(1) … Node(n))
        End
        Return MD
End gen_cp
```

```
Function Freq_of_failure(JDF,MD)
        For i = 1:Number_of_event_node
            Risk_frequency = JDF(i)/MD(i)
       End
End Freq_of_failure
```

Figure 2: Algorithm of SA-VA Model.

In summary, the inputs of SA-VA algorithm are the DAG, mean, variance, lower bound and upper bound of each node. The first procedure in the SA-VA algorithm is to estimate the normal probability density function and the normal cumulative density function for every node in provided DAG using the function Node_df(). This is obtained using the mean, variance, lower bound, and upper bound for the nodes. The distribution is divided into seven states (very low, low, lowish medium, medium , highish medium, high, very high) based on the range between the lower bound and upper bound.

The correlation matrix is generated based on the relationship between the nodes in the DAG and the correlation coefficient obtained. The joint probability density function is created using the copula function by taking the correlation matrix and probability distribution (pdf and cdf) for each node. In order to simplify the Copula-based vulnerability assessment algorithm, I only estimate the joint probability density function between nodes that are connected with the arc in DAG.

The conditional or posterior probability is obtained between the set of the observed node and the evidenced node using the Bayesian inference model described in section 4.2 with the posterior probability, the risk frequency is estimated.

4.6 Matlab Implementation of SA-VA Modell

I will provide a methodical walk-through of the MATLAB implementation in this section. For those who are interested in applying the code themselves or gaining a deeper understanding, I have attached the codes and explanations at the end of this thesis, in Chapter 8.

Data Input: The code begins by reading data from an Excel file. It uses the `readtable` function to import columns containing mean failure (meanFailure), standard deviations (stds), lower range (lowerrange), and upper range (upperrange) of failure values. These values represent statistical information about different nodes/vulnerabilities.

Failure Distribution: The code proceeds by initializing variables and a loop to calculate the probability density functions (pdfs) for each failure variable. In this case, a normal distribution

is assumed for each failure scenario. The `normpdf` function is used to compute the probability densities based on the provided mean and standard deviation values.

Directed Acyclic Graph (DAG) Visualization: To visualize the relationships between different failure scenarios, the code constructs a directed acyclic graph (DAG) using the `digraph` function. A DAG is a graphical representation of a system where the nodes represent random variables (failure scenarios) and the edges represent dependencies between them. The `plot` function is then used to display the DAG.

Probability Distribution Function (PDF) Plot: In this part of the code, the normal probability distributions (pdfs) for a subset of nodes are plotted as seen in Figure 3. The code loops over the selected nodes, retrieves their respective pdfs, and plots them on separate graphs. Additionally, a red dashed line representing the median value is added to each plot.

Correlation Matrix: The code reads the correlation matrix from the Excel file. This matrix captures the relationships between different nodes in the DAG. The correlation values indicate the degree of linear association between the corresponding nodes.

Loss Function with Binomial Distribution: This section calculates the mean and variance of the loss for each node in the DAG. It uses a binomial distribution to generate random numbers based on the failure probabilities obtained from the pdfs. By simulating the failure events, the code estimates the mean and variance of the loss for each nodes.

Joint Probability Calculation: The code proceeds to compute the joint probability of all edges in the DAG. It iterates over the edges, extracting the child and parent nodes, and retrieves the corresponding correlation value from the correlation matrix. The joint probability is estimated using the copula density function, which takes into account the correlation between the nodes. The calculated joint probabilities are stored in a structure array called JDF along with the edge information.

Prediction Network: This part of the code computes the joint probability of specific sets of nodes in the DAG. It iterates over different sets of nodes (x1, x2, x3), matches them with the edges in JDF, and calculates the joint probability based on the conditional probabilities and

joint probabilities of the corresponding nodes. This step allows for predicting the joint probabilities of specific combinations of vulnerabilities.

There are four specific sets of nodes for which the joint probability is computed: x1, x2, x3 and x4. These sets are defined as follows:

x1: {'MA','IA'}, {'PO','IA'}, {'HE','IA'}, {'DO','IA'}
x2: {'RC','CF'}, {'SCP','CF'}
x3: {'SCP','IP'}, {'PA','IP'}, {'CO','IP'}, {'TR','IP'}
x4: {'IA','CLF'}, {'CF','CLF'}, {'IP','CLF'}

For each set, the code iterates through the JDF (Joint Distribution Function) array, which contains information about the edges (connections) between nodes and their corresponding joint probabilities. It matches the pairs of nodes in the sets x1, x2, x3 and x4 with the entries in JDF to retrieve the joint probabilities associated with those pairs. The joint probabilities are then used in Risk Frequency and Expected Loss Calculation.

Posterior Probability Calculation: The code calculates the posterior probability for selected nodes. It multiplies the joint probabilities obtained from the previous step with the corresponding prior probabilities. The resulting posterior probabilities indicate the updated probabilities of the failure scenarios given the observed data.

Posterior Distribution Plot: This section plots the prior and posterior distributions for selected nodes. It loops over the nodes, retrieves their respective pdfs, and plots both the prior and posterior distributions on the same graph as seen in Figure 4. This visualization allows for comparing the changes in the probability distributions before and after incorporating the observed data.

Risk Frequency Calculation: The code calculates the risk frequency mean and variance based on the posterior distribution. It determines the value with the maximum probability from the posterior distribution and uses it as the risk frequency mean. The risk frequency variance is predetermined as 4.

Premium Calculation: The function calculates the premium by multiplying the expected loss by `(1 + OV)` to incorporate the loading factor. It then adds the contingency loading, represented by `k` multiplied by the square root of the variance.

4.7 SA-VA Model Validation and Data

Data on cyber risk is not readily abundant, this might be because organizations that have been hacked don't want to share that information. In fact, even if historical data exists, the rapidly evolving cyber risk landscape could render it ineffective. The most valuable information for modeling would be the actual insurance claims that insurers encounter (Eling & Schnell, 2016, p. 478).

The primary objective of this thesis is to develop and demonstrate a vulnerability assessment model. This model is intended to assist SaaS providers in identifying the most vulnerable liability risks within their operations, understanding the associated financial impact, and facilitating the calculation of insurance premiums for insuring companies.

To achieve this, an extensive literature review was conducted to gather relevant information. Based on the findings, reasonable and well-informed estimations were made for the input data required for the SA-VA Model validation. Using MATLAB, the SA-VA Model was implemented and subsequently executed with the provided input data. The simulation results were thoroughly analyzed and presented in Chapter 5 of this thesis.

The probability density distribution presented in Figure 3 is a valuable tool for understanding the likelihood of attacks occurring at different frequencies and shows the prior knowledge of each node presented as probability density distribution which was estimated from the information provided in Table 7.

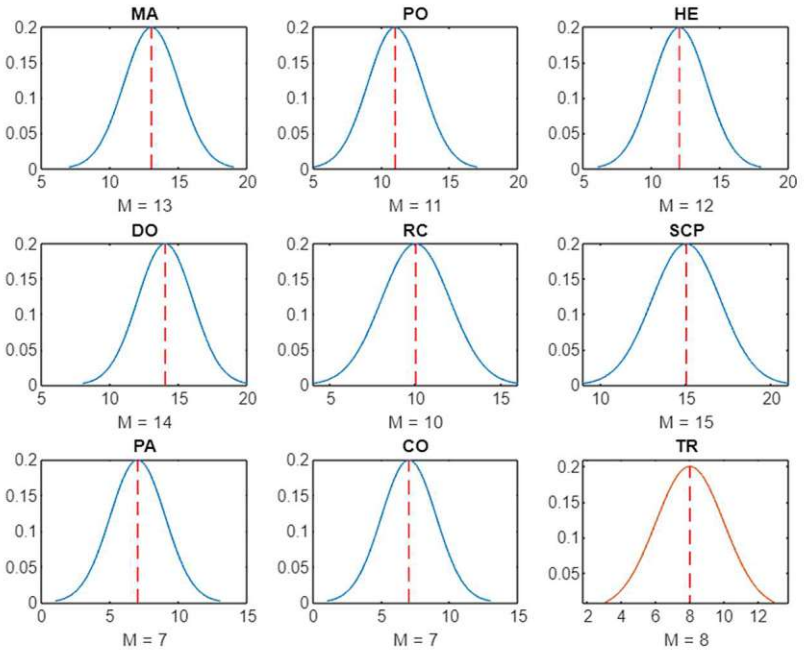

Figure 3: Normal Probability Distributions and Medians

The distribution allows us to identify the range of attack frequencies that are more likely or less likely to occur. For example, if the distribution shows a high likelihood of a low number of attacks, it indicates a relatively secure state. On the other hand, if the distribution demonstrates a higher probability of a large number of attacks, it suggests a vulnerable state that requires attention and appropriate countermeasures.

Prior marginal beliefs and loss distributions

Vulnerabilities	Failure distribution						Loss distribution
	Distribution	Failure parameters		Range		States	Parameters
		Mean Failure	Standart deviation	Lower	Upper		
MA	Normal	13	2	7	19	7	Binomial (1000,0.2)
PO	Normal	11	2	5	17	7	Binomial (1000,0.2)
HE	Normal	12	2	6	18	7	Binomial (1000,0.2)
DO	Normal	14	2	8	20	7	Binomial (1000,0.2)
IA	Normal	18	2	12	24	7	Binomial (1000,0.2)
RC	Normal	10	2	4	16	7	Binomial (1000,0.2)
SCP	Normal	15	2	9	21	7	Binomial (1000,0.2)
CF	Normal	17	2	11	23	7	Binomial (1000,0.2)
PA	Normal	7	2	1	13	7	Binomial (1000,0.2)
CO	Normal	7	2	1	13	7	Binomial (1000,0.2)
TR	Normal	8	2	3	13	7	Binomial (1000,0.2)
IP	Normal	16	2	10	22	7	Binomial (1000,0.2)
CLF	Normal	19	2	13	25	7	Binomial (1000,0.2)

Table 7: Prior Marginal Beliefs and Loss Distributions

As presented in Table 7, the failure distribution is divided into seven states: very low, low, lowish medium, medium, highish medium, high, very high. This division allows for the categorization of vulnerability levels.

The inclusion of the lower and upper ranges serves a critical purpose in ensuring that the Probability Density Functions (PDFs) accurately encapsulate the appropriate span of potential values for each variable.

The loss distribution of each node is modeled with the binomial random number (1000,0.2). The parameters used in the binomial model represent the loss distribution for each node in the vulnerability assessment. The probability of success indicates the likelihood of a vulnerability being successfully exploited or attacked. The number of trials represents the number of attempts or instances considered in the vulnerability assessment. For instance, if the probability of success is 0.2 and the number of trials is 1000, it means that, on average, there is an assumption of 200 successful attacks out of 1000 attempts.

	MA	PO	HE	DO	**IA**	RC	SCP	**CF**	PA	CO	TR	**IP**	**CLF**
MA	1,00												
PO		1,00											
HE			1,00										
DO				1,00									
IA	0.1	0.1	0.1	0.1	1,00								
RC						1,00							
SCP							1,00						
CF						0.1	0.1	1,00					
PA									1,00				
CO										1,00			
TR											1,00		
IP							0.1		0.1	0.1	0.1	1,00	
CLF					0.1			0.1				0.1	1,00

Table 8: Correlation Matrix.

The provided correlation matrix (R) among the vulnerabilities is depicted Table 8 shows the relationship between the nodes as the correlation coefficient. The correlations are assumed to lie between 0 and 1, presenting a more realistic representation of reality compared to assuming correlations of either 0 or 1. Although it remains a simplified representation, it proves to be a useful tool in understanding the relationships between nodes.

5. Results of SA-VA Model Implemention

The outputs from the SA-VA model are vulnerability assessment report, which includes the posterior probability at event nodes and the expected risk value, represented as the mean ($E(N_i)$) and variance ($Var(N_i)$), at each node.

The vulnerability assessment report will be discussed at the beginning of this chapter. It will include the posterior probabilities calculated, as well as the corresponding prior and posterior distribution graphs.

In Section 5.2, the quantification of the expected risk value derived from SA-VA results will be introduced. This will enable the calculation of the expected combined distribution of the loss ($E(S_i)$) using the collective risk model.

To demonstrate how the thesis's vulnerability model can assist insurance companies in estimating premiums for cloud liability failures incurred by SaaS providers, Section 5.3 will introduce the premium calculation model. An example calculation of the insurance premium for node CLF (Cloud Liability Failure) will also be provided, based on the expected loss derived from the collective risk model and simulation results obtained from MATLAB.

At the end of this chapter, the thesis's commitment to establishing a liability-risk policy for SaaS providers becomes evident.

5.1 Vulnerability Assessment Report

The SA-VA began with an initial assumption that there is a 20% probability of a moderate security breach occurring at each node. The copula Bayesian belief network-based model is used to compute the vulnerability of event nodes, this includes the Interruption of Information Access (IA), Compliance Failure (CF), Intellectual Property (IP), and Cloud Liability Failure (CLF).

The posterior probability was calculated using the frequency of failure algorithm as presented in Figure 2 and the assumption that each evidence nodes are independent of one another. For example, an interruption in IA could be caused by maintenance, power outage, human error or a DOS attack. The posterior probabilities of IA, CF, IP, and CLF were estimated to be 0.9627, 0.3867, 0.7207, and 0.4860, respectively, as shown in Table 9. The higher posterior probabilities observed were a result of intentionally using larger failure numbers in the input data. This intentional adjustment of the failure numbers was done to make the results of the assessment model more noticeable and to emphasize how well the evaluation process works.

Event	Details	Prior Belief	Posterior Belief
E1 IA (Medium)	Probability of moderate attack for IA	0.2	0.9627
E2 CF (Medium)	Probability of moderate attack for CF	0.2	0.3867
E3 IP (Medium)	Probability of moderate attack for IP	0.2	0.7207
E4 CLF (Medium)	Probability of moderate attack for CLF	0.2	0.4860

Table 9: SA-VA Model Assessment Report

The results show that the probability of compliance failure is relatively low when compared to the other event nodes, IA and IP. The results also show us the weakest link, which may cause cloud liability failure for SaaS providers. Similarly, the causal effect for any nodes presented in Figure 1 can be identified. Thus SA-VA provides an effective model for vulnerability analysis.

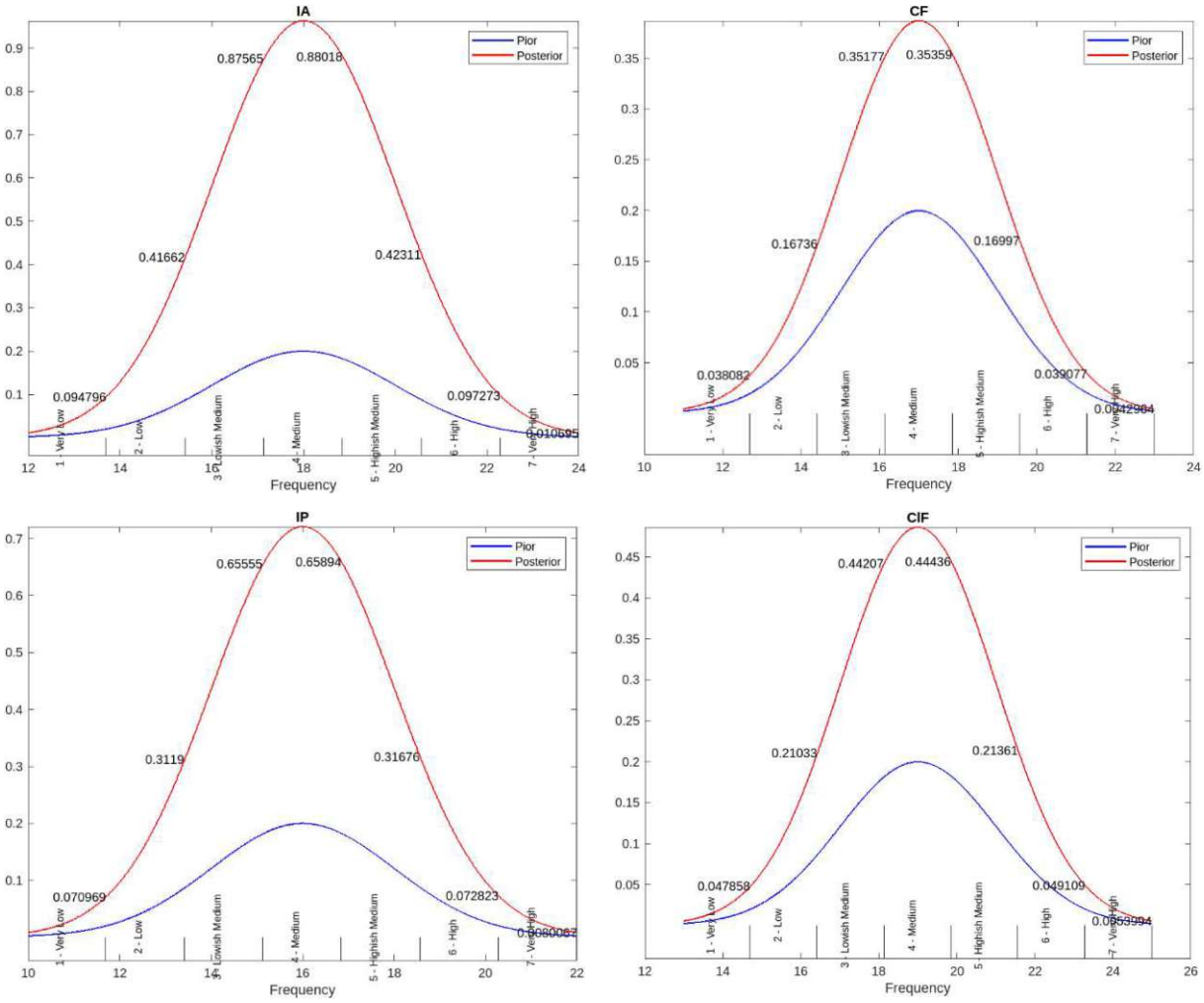

Figure 4: Prior vs. Posterior Probabilities for the Events

The presented results in Figure 4 and Table 9 highlight the potential of SA-VA model for improved decision-making and risk management. The IA node has a posterior probability of 0.9627, indicating high confidence in event occurrences. This confidence helps resources go to nodes/vulnerabilities with bigger risks. The IA node's average expected loss is $3617.69, which helps put resources to lower possible losses. The variance of expected loss, like IA's $165366.23, shows how uncertain things are. This helps decision-makers see possible financial impacts better and create stronger strategies to manage risks.

IA's calculated e-insurance premium of $4020.13, this approach matches insurance to the unique risk of each node. This makes sure insurance fits the risks well, giving the best protection.

Nodes like CF, with a posterior probability of 0.3867, demonstrate the potential for balanced resource allocation across nodes with varying confidence levels in event occurrences.

The IP node analysis yields a posterior probability of 0.7207, indicating confidence in event occurrences. The associated risk frequency of 15.9940 suggests an average occurrence of about 16 times within the specified period. The mean expected loss for IP is $3198.96, representing anticipated financial loss.

The CLF node analysis results in a posterior probability of 0.4860, suggesting moderate-to-high likelihood of related occurrences. The calculated risk frequency for CLF is 18.9940, indicating an average occurrence of about 19 times within the specified period.

5.2 Quantification of Vulnerabilities

The quantification of vulnerabilities causing liability failure is presented based on the collective risk model proposed by (Hossack et al., 1999). The model enables a successful analysis of potential losses arising from liability risks and incorporates the stochastic nature of both the frequency of events and the associated loss amounts.

To utilize the collective risk model for vulnerability quantification, the frequency of attacks ($E(N_i)$) and the associated loss ($E(L_i)$) are considered as stochastic variables. By doing so, the combined distribution of the loss, denoted as ($E(S_i)$), is constructed. The expected loss due to vulnerabilities is then calculated using Equation 8**,** which yields values for both the mean and variance as seen in Equation 9.

$$\mathbf{E(S_i) = E(N_i) \times E(L_i)} \qquad \text{Equation 8}$$

$$\mathbf{Var(S_i) = E(N_i) \times Var(L_i) + E(Li)^2 \times Var(N_i)} \qquad \text{Equation 9}$$

Where;

$E(S_i)$ represents the expected loss or claim severity.

$E(N_i)$ the loss frequency distribution.

$E(L_i)$ the loss amount distribution.

$Var(S_i)$ represents the variance of the claim severity.

$Var(N_i)$ the variance of the loss frequency distribution.

$Var(L_i)$ the variance of the loss amount distribution.

The process starts by gathering essential inputs, such as SA-VA report and the expected loss distribution at each node. These inputs enable the calculation of the expected loss or claim severity ($E(S_i)$) and its variance ($Var(S_i)$). The collective risk model relies on specific assumptions regarding the loss amount distribution. In this thesis, a binomial loss amount distribution with parameters (1000, 0.2) is assumed. Based on this assumption, the values of ($E(L_i)$) and ($Var(L_i)$) are calculated using input data in MATLAB.

The standard deviation σ=2 given in Table 7 is squared to specify the distribution in terms of its variance ($Var(N_i)$), which makes it 4 (Oliphant, 2006, p. 5).

Event	Details	Posterior Probability	$E(N_i)$	$E(S_i)$	$Var(S_i)$
E1, IA	Probability of moderate attack for IA	0.9627	17.9940	3617.69	165366.23
E2, CF	Probability of moderate attack for CF	0.3867	16.9940	3409.51	163543.46
E3, IP	Probability of moderate attack for IP	0.7207	15.9940	3198.96	162680.53
E4, CLF	Probability of moderate attack for CLF	0.4860	18.9940	3820.45	165629.92

Table 10: Risk Quantification Results

In summary, the loss distribution and risk frequency are taken as inputs for vulnerability quantification, while the resulting claim severity is presented as the output in Table 10, showing both the mean and variance.

5.3 Premium Amount Calculation

In this section, the premium calculation process for cloud service providers' cloud liability failure will be presented, aimed at assisting insurance companies in providing coverage. According to (Böhme, 2005, p. 6), a naive insurance company could set the premium amount as the mean claim amount $E(S_i)$, however, in this thesis, two important factors considered, which align with actuarial science, overhead loading (q) and the contingency loading (k).

Based on the claim amount obtained with Equation 8 and its variance with Equation 9, the insurance premium at any given node defined as presented in Equation 10 (Mukhopadhyay et al., 2006, p. 5):

$$\mathbf{Pr} = (\mathbf{1} + \boldsymbol{q}) \times \boldsymbol{E}(\boldsymbol{S_i}) + \boldsymbol{k} \times \sqrt{\boldsymbol{V}(\boldsymbol{S_i})} \qquad \text{Equation 10}$$

Where;

$q = overhead\ loading\ factor$

$k = contigency\ loading$

In actuarial science, the loading factor (q) plays an important role in insurance pricing. It covers various elements such as administrative costs, commissions, taxes, and profits. When setting insurance premiums, companies apply the loading factor, typically assumed to be 10%, to the net premium. This ensures that the insurer can cover its operating expenses and generate a reasonable profit while still offering the necessary coverage to policyholders. Additionally, insurers use the contingency factor (k), also assumed to be 10%, to address the unpredictability of individual claims. This factor helps establish appropriate reserves to cover potential future claims accurately. By applying the contingency factor to the average claim severity, insurance companies safeguard themselves against unexpected large losses and ensure they have sufficient funds to fulfill their claim obligations (Hossack et al., 1999, p. 122).

For the purpose of premium calculation, two functions have been included from the SA-VA algorithm, as depicted in Figure 5 below.

```
Function cliam_severity (E_mean,E_var)
        E_mean = E(N_i) * E(L_i)
        E_var = E(N_i) * Var(L_i) + E(L_i)^2 * Var(N_i)
Return E_mean, E_var
```

```
Function Premium_Calculate(E_mean,E_var)
        q = .1      // loading factor
        k = .1      // contingency factor
        Premium = (1 + q) × E_mean + k × √E_var
        Return Premium
End Premium_Calculate
```

Figure 5 : Insurance Premium Calculation Algorithm

The Functions for premium computation are cliam_severity() and Premium_Calculate(). The claim_severity() function enables the calculation of mean and variance severity based on the risk frequency, along with calculated values for the mean and variance of the loss distribution. The Premium_Calculate () function allows the premium to be determined using the output values from the cliam_severity() function, in addition to the assumed loading factor and contingency factor of 10%.

In Table 11, the result obtained from the MATLAB implementation of the premium calculation function is presented as 'premium,' inclusive of the posterior probability, claim amount, and its variance for each event.

Event	Details	Posterior Probability	$E(S_i)$	$Var(S_i)$	Premium ($)
E1 IA	Probability of moderate attack for IA	0.9627	3617.69	165366.23	4020.13
E2 CF	Probability of moderate attack for CF	0.3867	3409.51	163543.46	3790.90
E3 IP	Probability of moderate attack for IP	0.7207	3198.96	162680.53	3559.19
E4 CLF	Probability of moderate attack for CLF	0.4860	3820.45	165629.92	4243.19

Table 11: Risk Assessment and Premium Calculation Results

I will now illustrate the premium calculation using the results of SA-VA model MATLAB implementation with following example: Calculating Insurance Premium for Cloud Liability Failure (CLF) :

Risk Frequency $= E(N_i)$ $= 18.9940$

Risk freq. variance $= Var(N_i)$ $= 4$

Loss mean $= E(L_i)$ $= 201.14$

Loss variance $= Var(L_i)$ $= 200.1$

Expected loss,

$$\mathbf{E(S_i) = E(N_i) \times E(L_i)}$$

$E(S_i)$ = *mean risk frequency* × *loss mean*

$$E(S_i) = 18.9940 \times 201.14 = 3820.5$$

Variance of expected loss for CLF,

$$\mathbf{Var(S_i) = E(N_i) \times Var(L_i) + E(Li)^2 \times Var(N_i)}$$

$Var(S_i)$ = *mean risk frequency* × *loss variance* + *loss mean*^{2} × *risk frequency variance*

$$Var(S_i) = 18.9940 \times 200.1 + 201.14^2 \times 4 = 165629.92$$

Insurance Premium for CLF;

$$Pr = (1 + q) \times E(S_i) + k \times \sqrt{V(S_i)}$$

$$Pr = (1 + 0.1) \times 3820.5 + 0.1 \times \sqrt{165629.92} = 4243.2\ \$$$

5.4 SaaS-Asset Insurance

This thesis proposes liability-risk insurance for SaaS providers based on the quantification of vulnerabilities, allowing the measurement of risk exposure levels and estimation of claim severity. The insurance company would perform the initial calculation of the event's likelihood. Based on this calculation, coverage for the liability risk would be provided if the probability is acceptably low.

In cloud computing, if a provider were to assume complete liability within a multi-tenant, pay-as-you-go framework, the potential liability could escalate at such a rapid rate that the provider might find it financially unfeasible to bear the insurance premiums for covering such liability. Moreover, the liability associated with the provider's infrastructure could even surpass the overall value of the company.

A SaaS-Asset insurance not only offers a viable solution for managing liability risks but also presents a strategic approach for SaaS providers to mitigate potential financial burdens. By sharing the liability through an insurance policy, SaaS providers can confidently expand their services and scale their operations without facing overwhelming liability costs.

6. Conclusion

In this thesis, I propose a model designed to assist SaaS providers in making informed decisions about whether to transfer the liability risk associated with cloud liability failure or manage it in-house. The SA-VA vulnerability assessment model presented here evaluates the Cloud Liability Failure (CLF) that may arise due to vulnerabilities such as interruption of information access, compliance failure, or intellectual property issues.

This thesis aims to provide a comprehensive understanding of the liability risks faced by SaaS providers within the cloud computing landscape. The development and implementation of the SA-VA model offers a structured approach for assessing and quantifying vulnerabilities specific to SaaS assets. By integrating statistical analysis techniques and validation processes, this research provides practitioners, researchers, and policymakers with actionable insights to enhance the security of cloud-based services and mitigate potential risks effectively.

This thesis had the goal of providing valuable assistance to SaaS-Providers in two critical aspects. Firstly, it set out to help these providers assess their most vulnerable liability risks within their operations. The successful development and implementation of the SA-VA model represented the fulfillment in this regard. Secondly by quantifying the potential financial impact linked to these identified risks, the thesis fulfilled its objective to enhance the providers' understanding of the potential consequences. This achievement was highlighted through a clear and detailed explanation of the statistical basis for the SA-VA model, employing a copula aided Bayesian belief network, applying it through a collective risk model, and then delivering a vulnerability assessment report.

The thesis also holds a primary goal, which aims to extend its assistance to insurance companies by aiding in the calculation of suitable insurance premiums for insuring SaaS-Assets. This objective was met through the presentation of a precise premium calculation model. Moreover, the inclusion of an illustrative example calculation, along with its thorough explanation of the obtained results, contributed to the fulfillment of this specific aim.

In both cases, the thesis successfully fulfilled its objectives by providing practical insights and tools that can benefit SaaS-Providers and insurance companies alike.

Last but not least, the literature review conducted in this thesis represents a substantial investment of time and effort, reflecting a comprehensive exploration into the subject matter. This extensive literature review serves as a robust foundation, making it a valuable resource for individuals seeking a deeper understanding of this field or those aiming to undertake further research. Through an extensive examination of a wide variety of information sources, the integration of diverse perspectives, and critically analyzing existing literature, this thesis's literature section emerges as a robust reference point that not only highlights the challenges and nuances of the subject but also paves the way for future scholarly investigations.

7. Literature

Ahmed, M., & Hossain, M. A. (2014). Cloud computing and security issues in the cloud. *International Journal of Network Security & Its Applications*, *6*(1), 25.

Akinrolabu, O., Nurse, J. R., Martin, A., & New, S. (2019). Cyber risk assessment in cloud provider environments: Current models and future needs. *Computers & Security*, *87*, 101600.

Al-Adilee, A. (2014). Copula Functions and correlations Personal Contribution. *Journal of Kerbala university*, *10*(2), 196-205.

Al-Dhahri, S., Al-Sarti, M., & Abdul, A. (2017). Information security management system. *International Journal of Computer Applications*, *158*(7), 29-33.

Baset, S. A. (2012). Cloud SLAs: present and future. *ACM SIGOPS Operating Systems Review*, *46*(2), 57-66.

Böhme, R. (2005). Cyber-Insurance Revisited. Weis,

CIDON, A. (2015). Protecting intellectual property in the cloud. *WIPO MAGAZINE*(3), 14-17.

Clemen, R. T., & Reilly, T. (1999). Correlations and copulas for decision and risk analysis. *Management Science*, *45*(2), 208-224.

Cooper & McCloskey. *Intellectual Property Liability*. Retrieved 01.02.2023 from https://cmiprorisk.com/products/intellectual-property-liability/

Dash, S., Saini, H., Panda, T., & Mishra, A. (2014). Service level agreement assurance in cloud computing: a trust issue. *International Journal of Computer Science and Information Technologies*, *5*(3), 2899-2906.

Deloitte. (2023). *Cloud Services*. Retrieved 02.2023 from https://www2.deloitte.com/us/en/pages/consulting/solutions/cloud-consulting-services.html

Elidan, G. (2010). Copula bayesian networks. *Advances in neural information processing systems*, *23*.

Eling, M., & Schnell, W. (2016). What do we know about cyber risk and cyber risk insurance? *The Journal of Risk Finance*, *17*(5), 474-491.

ENISA, C. C. (2009). Benefits, risks and recommendations for information security. *European Network and Information Security*, *23*.

Evans, M., Maglaras, L. A., He, Y., & Janicke, H. (2016). Human behaviour as an aspect of cybersecurity assurance. *Security and Communication Networks*, *9*(17), 4667-4679.

Fenton, N., & Neil, M. (2011). The use of Bayes and causal modelling in decision making, uncertainty and risk. *CEPIS Upgrade*, *12*(5), 10-21.

Garfinkel, S. (1999). *Architects of the information society: 35 years of the Laboratory for Computer Science at MIT*. MIT press.

Gartner Inc. (2023a). *Gartner Forecasts Worldwide Public Cloud End-User Spending to Reach Nearly $600 Billion in 2023*. Retrieved 08.2023 from https://www.gartner.com/en/newsroom/press-releases/2023-04-19-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-reach-nearly-600-billion-in-2023

Gartner Inc. (2023b). *Gartner Forecasts Worldwide Public Cloud End-User Spending to Reach Nearly $600 Billion in 2023*. Retrieved 03.04.2023 from https://www.gartner.com/en/newsroom/press-releases/2022-10-31-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-reach-nearly-600-billion-in-2023#:~:text=Worldwide%20end%2Duser%20spending%20on,18.8%25%20growth%20forecast%20for%202022

Google Cloud. (2023). *What is a cloud service provider?* https://cloud.google.com/learn/what-is-a-cloud-service-provider#:~:text=started%20for%20free-,Cloud%20service%20provider%20definition,%2C%20platform%2C%20and%20application%20services.

Gordon, D. G. (2016). Legal Aspects of Cloud Computing. *Encyclopedia of Cloud Computing*, 462-475.

Gunawi, H. S., Hao, M., Suminto, R. O., Laksono, A., Satria, A. D., Adityatama, J., & Eliazar, K. J. (2016). Why does the cloud stop computing? lessons from hundreds of service outages. Proceedings of the Seventh ACM Symposium on Cloud Computing,

Heckerman, D. (2008). A tutorial on learning with Bayesian networks. *Innovations in Bayesian networks: Theory and applications*, 33-82.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2008). Design science in information systems research. *Management Information Systems Quarterly*, *28*(1), 6.

Hon, W. K., Millard, C., & Walden, I. (2012). Negotiating cloud contracts: Looking at clouds from both sides now. *Stan. Tech. L. Rev.*, *16*, 79.

Horwath, C., Chan, W., Leung, E., & Pili, H. (2012). Enterprise risk management for cloud computing. *COSO Juni*.

Hossack, I., Pollard, J. H., & Zehnwirth, B. (1999). *Introductory statistics with applications in general insurance*. Cambridge University Press.

Infraveo Technologies. (2022). *Netflix on Cloud with AWS*. https://aws.plainenglish.io/netflix-on-cloud-with-aws-amazon-web-service-fb035b992d52

International Data Corporation. (2021). *IDC Forecasts Worldwide "Whole Cloud" Spending to Reach $1.3 Trillion by 2025.* Retrieved 06.03.2023 from https://www.idc.com/getdoc.jsp?containerId=prUS48208321

International Standards Organization. (2014). *Information technology — Cloud computing — Overview and vocabulary*. Retrieved 13.02.2023 from https://www.iso.org/obp/ui/#iso:std:iso-iec:17788:ed-1:v1:en

Isaca. (2009). *The risk IT framework*. ISACA.

Jansen, W., & Grance, T. (2011). Guidelines on security and privacy in public cloud computing.

Kemp, R. (2020). *Intellectual Property in the Cloud: The Patent Troll Threat*. https://kempitlaw.com/insights/intellectual-property-in-the-cloud-the-patent-troll-threat/

Kovácsné Mozsár, A., & Michelberger, P. (2018). IT risk management and application portfolio management. *Polish Journal of Management Studies*, *17*(2), 112-122.

Kumar, N. G., Polala, N., & Kumari, D. A. (2018). New approach for securing cloud applications. 2018 2nd International Conference on Inventive Systems and Control (ICISC),

Lalonde, C., & Boiral, O. (2012). Managing risks through ISO 31000: A critical analysis. *Risk management*, *14*, 272-300.

Laura Didio. (2018). *Human Error is Top Cause of Downtime*. Retrieved 2023 from https://techchannel.com/SMB/10/2018/Human-Error-Top-Cause-of-Downtime

Li, Z., Liang, M., O'brien, L., & Zhang, H. (2013). The cloud's cloudy moment: A systematic survey of public cloud service outage. *arXiv preprint arXiv:1312.6485*.

Lu, Y., & Xu, X. (2015). Protecting intellectual property in a cloud manufacturing environment: Requirements and strategies. Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth: IFIP WG 5.7 International Conference, APMS 2015, Tokyo, Japan, September 7-9, 2015, Proceedings, Part II 0,

Ma, D. (2007). The business model of" software-as-a-service". Ieee international conference on services computing (scc 2007),

Mäkilä, T., Järvi, A., Rönkkö, M., & Nissilä, J. (2010). How to define software-as-a-service–an empirical study of finnish saas providers. Software Business: First International Conference, ICSOB 2010, Jyväskylä, Finland, June 21-23, 2010. Proceedings 1,

Mangalaraj, G., Singh, A., & Taneja, A. (2014). IT Governance Frameworks and COBIT-A Literature Review. AMCIS,

Martens, B., & Teuteberg, F. (2011). Risk and compliance management for cloud computing services: Designing a reference model.

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.

Melzer, M. A. (2010). Copyright enforcement in the cloud. *Fordham Intell. Prop. Media & Ent. LJ*, *21*, 403.

Microsoft Azure. (2023). *What is a cloud service provider?* Retrieved 2023 from https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-a-cloud-provider/

Mishra, N., Singh, R., & Yadav, S. K. (2020). Analysis and Vulnerability Assessment of Various Models and Frameworks in Cloud Computing. Advances in Data Sciences, Security and Applications: Proceedings of ICDSSA 2019,

Mogull, R., Arlen, J., Gilbert, F., Lane, A., Mortman, D., Peterson, G., Rothman, M., Moltz, J., Moren, D., & Scoboria, E. (2017). Security guidance for critical areas of focus in cloud computing v4. 0. *Cloud Security Alliance*.

Morin, J.-H., Aubert, J., & Gateau, B. (2012). Towards cloud computing SLA risk management: issues and challenges. 2012 45th Hawaii International Conference on System Sciences,

Mukhopadhyay, A., Chatterjee, S., Saha, D., Mahanti, A., & Sadhukhan, S. K. (2006). e-Risk management with insurance: a framework using copula aided Bayesian belief networks. Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06),

Mukhopadhyay, A., Chatterjee, S., Saha, D., Mahanti, A., & Sadhukhan, S. K. (2013). Cyber-risk decision models: To insure IT or not? *Decision Support Systems*, *56*, 11-26.

Muller, S., & Supatgiat, C. (2007). A quantitative optimization model for dynamic risk-based compliance management. *IBM Journal of Research and Development*, *51*(3.4), 295-307.

Nelsen, R. B. (2006). *An introduction to copulas*. Springer.

Newhouse, W., Johnson, B., Kinling, S., Kuruvilla, J., Mulugeta, B., & Sandlin, K. (2019). Multifactor authentication for e-commerce: Risk-based, fido universal second factor implementations for purchasers.

NIST. *intellectual property*. Retrieved April 2022 from https://csrc.nist.gov/glossary/term/intellectual_property

NIST. (2021). *service level agreement (SLA)*. Retrieved April 2023 from https://csrc.nist.gov/glossary/term/service_level_agreement#:~:text=A%20service%20contract%20between%20an,under%20Service%20Level%20Agreement%20(SLA)

Norvig, S. R. P. (2016). Artificial Intelligence: A modern approach 3rd ed. *Harlow Pearson Education*.

Oliphant, T. E. (2006). A Bayesian perspective on estimating mean, variance, and standard-deviation from data.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, *24*(3), 45-77.

Phelps, M., & Jennex, M. E. (2015). Yours, mine, or ours: discussing ownership of collaborative works in the cloud. 2015 48th Hawaii International Conference on System Sciences,

Rahi, S. B., Bisui, S., & Misra, S. C. (2017). Identifying critical challenges in the adoption of cloud-based services. *International Journal of Communication Systems*, *30*(12), e3261.

Rainer Jr, R. K., Snyder, C. A., & Carr, H. H. (1991). Risk analysis for information technology. *Journal of management information systems*, *8*(1), 129-147.

Schober, P., Boer, C., & Schwarte, L. A. (2018). Correlation coefficients: appropriate use and interpretation. *Anesthesia & analgesia*, *126*(5), 1763-1768.

Soewito, B., Gaol, F. L., & Abdurachman, E. (2022). A systematic literature Review: Risk analysis in cloud migration. *Journal of King Saud University-Computer and Information Sciences*, *34*(6), 3111-3120.

Somani, G., Gaur, M. S., Sanghi, D., Conti, M., & Buyya, R. (2017). DDoS attacks in cloud computing: Issues, taxonomy, and future directions. *Computer Communications*, *107*, 30-48.

Surbiryala, J., & Rong, C. (2019). Cloud computing: History and overview. 2019 IEEE Cloud Summit,

Suryateja, P. (2018). Threats and vulnerabilities of cloud computing: a review. *International Journal of Computer Sciences and Engineering*, *6*(3), 297-302.

Vachhani, M. K., & Atkotiya, K. H. (2013). similarities and contrast between Grid Computing and Cloud Computing. *Indian Journal of applied research*, *3*.

Weber, R. H., & Staiger, D. N. (2014). Cloud Computing: A cluster of complex liability issues. *European Journal of current legal issues*, *20*(1).

Yang, H., & Tate, M. (2012). A descriptive literature review and classification of cloud computing research. *Communications of the Association for Information Systems*, *31*(1), 2.

Yimam, D., & Fernandez, E. B. (2016). A survey of compliance issues in cloud computing. *Journal of Internet Services and Applications*, *7*, 1-12.

Youseff, L., Butrico, M., & Da Silva, D. (2008). Toward a unified ontology of cloud computing. 2008 Grid Computing Environments Workshop,

8. Attachment SA-VA Matlab Implementation

This attachment contains a detailed explanation and implementation of MATLAB codes. It is intended for individuals who want to gain a thorough understanding of the subject matter and follow the step-by-step processes. The document provides comprehensive explanations and code snippets for each MATLAB implementation. It serves as a valuable resource to aid in understanding the concepts and applying the methodologies. Readers will have access to a significant amount of information to support their research or practical pursuits.

```
data_input=readtable('data_input.xlsx')
```

This line reads provided excel file named 'data_input.xlsx' and stores it in a table.

```
meanFailure=data_input.meanFailure
stds=data_input.standardDeviation
lowerrange=data_input.lowerrange
upperrange=data_input.upperrange
```

These four lines extract specific columns from the `data_input` table and store them in separate variables for easier access later on in the code. Specifically, `meanFailure` contains the mean failure rate for each node in the system, `stds` contains the standard deviation of the failure rates, `lowerrange` contains the lower range of the failure rates, and `upperrange` contains the upper range of the failure rates.

```
n=length(meanFailure)
state=[4,4,4,4,4,4,4,4,4,4,4,4,4]
```

First the number of nodes are determined by calculating the length of the `meanFailure` vector. Subsequently, a `state` vector to assign discrete states for each node established. In this implementation, all nodes are initially set to a '4-medium' state.

```
samples =1000
pdfs=zeros(samples,n)
for i=1:n
    x=linspace(lowerrange(i), upperrange(i),samples)
    pdfs(:,i)=normpdf(x,meanFailure(i), stds(i))
end
```

These lines calculate the probability density function (PDF) for each node using the `normpdf` function. `samples = 1000` determines the number of sample points that will be used to generate the PDFs. The PDFs are calculated based on the mean and standard deviation values for each node, and the lower and upper range of the node. The `linspace` function is used to create a linearly spaced vector of values between the lower and upper range of each node. These values are used as the input to the `normpdf` function, along with the mean and standard deviation values for the node, to generate the corresponding PDF. The resulting PDF values are then stored in the appropriate column of the `pdfs` matrix.

```
nodenames={'MA','PO','HE','DO','IA','RC','SCP','CF','PA','CO',
'TR','IP','CLF'}'
```

A cell array called `nodenames` is created, containing the names of the 13 nodes.. The apostrophe at the end of the line transposes the cell array so that it is a column vector.

```
DAG=digraph([1,2,3,4,6,7,10,11,7,9,5,8,12],
[5,5,5,5,8,8,12,12,12,12,13,13,13])
```

I created a directed graph using the `digraph` function. The graph object is named `DAG` and it takes two arguments, a vector of source nodes and a vector of target nodes. The indices of the nodes in the `nodenames` cell array correspond to the values in the source and target vectors respectively. For example, the first edge of the graph connects node 1 ('MA') to node 5 ('IA').

```
DAG.Nodes.Name=nodenames
```

In this line, the collection of nodes in the DAG is referred to as `DAG.Nodes` By specifying `.Name`, the `Name` property of each node is accessed. By assigning `nodenames` to this property, the names in the cell array are assigned to the corresponding nodes in the DAG. This is done so that the graph can be plotted with the correct node labels.

```
figure(1)
plot(DAG, 'Linewidth',2)
grid on
title('DAG')
```

These lines generate a plot of the Directed Acyclic Graph using the `plot` function. The `Linewidth` parameter sets the width of the edges in the graph, and the `grid on` command adds a grid to the plot.

```
corr_matrix=readtable('corr._Matrix3.xlsx')
loss_mean=zeros(1,size(pdfs,2))
loss_variance=zeros(1,size(pdfs,2))
for i=1:length(pdfs)
b=[];
for j=1:100
b(j)=binornd(1000,0.2)
```

These lines of code read a correlation matrix from provided excel file and compute the mean and variance of a binomial distribution for each probability density function in `pdfs`. The binomial distribution is simulated by generating 100 random numbers between 0 and 1, and setting the probability of success to 0.2. The mean and variance of these random numbers are then computed and stored in `loss_mean` and `loss_variance`.

```
Edges=DAG.Edges
ed=table2cell(Edges
```

The edges of the `DAG` are stored in a table called `Edges`, which is converted into a cell array called `ed`. The `ed` cell array contains the child and parent nodes for each edge.

```
JDF = {}
for i = 1:size(Edges,1)
child = ed{i,1}{1}
parent = ed{i,1}{2}
```

An empty cell array called JDF is created, and a loop is started that iterates through each row of the Edges table. Inside the loop, the child and parent nodes for the current edge are extracted from the `ed` cell array.

```
childidx = find(strcmp(corr_matrix.Var1,child))
parentidx = find(strcmp(corr_matrix.Var1,parent))
corrvalue = table2array(corr_matrix(parentidx,childidx+1))
```

The correlation value between the parent and child nodes is then extracted from the `corr_matrix`, and is stored in the `corrvalue` variable. This process is repeated for each edge in the DAG.

```
rho = sin(corrvalue*pi./2)
RHO = [1 rho; rho 1]
c = copularnd('Gaussian', RHO, samples)
y = copulacdf('Gaussian',pdfs(:,[childidx,parentidx]),RHO)
```

Here computed the Kendall tau correlation coefficient between child and parent nodes and constructed a correlation matrix based on this coefficient. Random samples are generated from a Gaussian copula using this matrix, and the copula density is computed for the child and parent nodes based on their probability density functions and the Gaussian copula.

```
x    =    linspace(lowerrange(childidx),    upperrange(childidx),
samples)
k1 = lowerrange(childidx) + state(childidx)*2 - 2
idx1 = find(x >= k1)
idx2 = find(x < k1+2)
a = mean(c(idx1(1):idx2(end),1))
```

This code block extracts the portion of moderate probability density function for the child node and compute its mean probability.

```
x = linspace(lowerrange(parentidx), upperrange(parentidx),
samples);
k2 = lowerrange(parentidx) + state(parentidx)*2 - 2;
idx1 = find(x >= k2);
idx2 = find(x < k2+2);
b = mean(c(idx1(1):idx2(end),1));
```

The portion of the moderate probability density function for the parent node is extracted, and its mean probability is computed using these lines of code.

```
J = a * b
cnode = {parent, child}
JDF{i}.edge = cnode
JDF{i}.J = J
```

In these codes, the joint probability of a parent-child pair is computed by multiplying their mean probabilities `a` and `b` to get the joint probability `J`. The `cnode` cell array is used to store the names of the two nodes that form the `edge`, and the `JDF` cell array is used to store the joint probabilities for all parent-child pairs in the network.

```
X1 = 1
for i = 1:length(x1)
    for j = 1:length(JDF)
        jdf = JDF{1,j}
        if strcmp(jdf.edge{1},x1{i}{2}) &&
strcmp(jdf.edge{2},x1{i}{1})
            d = find(strcmp(corr_matrix.Var1,jdf.edge{1}))
            x = linspace(lowerrange(d), upperrange(d),
samples)
            k = lowerrange(d) + state(d)*2 - 2;
            idx1 = find(x <= k); idx2 = find(x >= k+2)
            p = mean(pdfs(idx1(end):idx2(1),i))
            X1 = X1*jdf.J/p
        end
    end
end
```

The purpose of this code is to compute the joint probability of a set of nodes specified by the cell array `X1`. For each node in `X1`, the code checks if there exists an edge in `JDF` that connects the parent and child nodes of the current node. If there is, the code proceeds to compute the conditional probability of the child node given its parent, and then accumulates the joint probability of the nodes by multiplying it with the joint density factor of the edge connecting the nodes. Below a brief explanation of the main functions used.

The line `d = find(strcmp(corr_matrix.Var1,jdf.edge{1}))` finds the index of the edge node `jdf.edge{1}` in the `corr_matrix`. The `linspace` function creates a vector `x` of equally spaced values ranging from the lower bound of the edge node to the upper bound of the edge node with `samples` elements. The `idx1` and `idx2` variables find the indices of the elements of `x` that correspond to the range `[lowerrange(d), k)` and `[k+2, upperrange(d)]`. The `mean` function computes the average probability density function (PDF) value of the `i`-th node in `x1` over the range `[idx1(end):idx2(1),i]`, and assigns it to `p`.

```
c=1
for i =10:12
x=linspace(lowerrange(i), upperrange(i),samples)
k=lowerrange(i)+state(i)*2 - 2
idx1=find(x<=k); idx2=find(x>=k+2)
p(c)= mean(pdfs(idx1(end):idx2(1),i))
c=c+1
end
```

The prior distribution is calculated by looping over nodes `i` that have indices 10, 11, and 12, (IA, CF and IP) and then creating a vector `p` that contains the prior probabilities for each node. For each `i` value, the code sets up a range of samples equally spaced values between `lowerrange(i)` and `upperrange(i)` using the `linspace` function. The mean function is a subset of the `pdfs` matrix corresponding to the current node `i`. This mean value is then stored in the `p` vector at index `c`.

```
X=[X1,X2,X3].*p
Posterior(1)=X(1); Posterior(2)=X(2); Posterior(3)=X(3)
```

After the `p` vector is created, the joint posterior distribution calculated by multiplying the prior distribution vector `p` with the vector `X`. The `X` vector contains the likelihood values for nodes `X1, X2,` and `X3.` The resulting joint posterior distribution is then stored in the vector `Posterior`.

```
X4=[];
for i=1:length(x4)
for j=1:length(JDF)
jdf=JDF{1,j}
```

As it has already been computed for `x1, x2` and `x3,` now the conditional and joint probability of the `x4` (CLF) will be computed. An empty array `X4` is initialized by `X4=[],`with `for i=1:length(x4)` a loop that will iterate over each element in the cell array `x4` is started. Another loop nested inside the first one that will iterate over each element

in the cell array `JDF` is started with `j=1:length(JDF)`. The current element of `JDF` is assigned to the variable `jdf`.

```
if strcmp(jdf.edge{1},x4{i}{2}) &&
strcmp(jdf.edge{2},x4{i}{1})
```

Here checked if the first and second elements of the `edge` field of the current element of `JDF` are equal to the second and first elements of the current element of `x4`. The `strcmp` function is used to compare these elements as strings.

```
d=find(strcmp(corr_matrix.Var1,jdf.edge{1}))
x=linspace(lowerrange(d), upperrange(d),samples)
k=lowerrange(d)+state(d)*2 - 2
idx1=find(x<=k); idx2=find(x>=k+2)
pp=mean(pdfs(idx1(end):idx2(1),i))
X4=[X4 jdf.J/pp]
end
```

The `find` function is used to locate the matching index in `corr_matrix.Var1`. A linearly spaced vector `x` is created between `lowerrange(d)` and `upperrange(d)` with `samples` points. The value of `k` is calculated. Indices are found for elements in `x` that satisfy the conditions `x<=k'` and `'x>=k+2`. The mean value is calculated from specific rows and column `i` in the `pdf` matrix.

As a summary; the code initializes an empty vector to store joint probabilities and iterates through the elements to find the joint distribution function corresponding to the root node. It identifies the index of the edge in a correlation matrix, generates a vector of evenly spaced frequency values, and determines the midpoint of the current edge state. Next, it finds the indices of frequency values within a range around the midpoint and calculates the probability density function within that range. The PDF is divided by the average probability density value in the range to obtain the conditional probability of the root node given the edge's current state. Finally, the joint probability of the edge is appended to the vector storing joint probabilities.

```
x=linspace(lowerrange(end), upperrange(end),samples)
k=lowerrange(end) + state(end)*2 - 2
idx1=find(x<=k); idx2=find(x>=k+2)
p(end+1)= mean(pdfs(idx1(end):idx2(1),end))
Posterior(4)= sum(X.*X4)*p(4)
```

This code calculates the posterior probability of node CLF, given the conditional and joint probabilities calculated before. It first generates a vector `x` of `samples` evenly spaced frequency values in the range of the last node, calculates the midpoint of the current state of the last node, and finds the indices of the frequency values that correspond to a frequency range around the midpoint of the current state. The code then calculates the prior probability of the last node by taking the mean of the PDF values in the range.

```
postprobs=zeros(samples,4)
postprobs(:,1)= pdfs(:,10); %IA
postprobs(:,2)= pdfs(:,11); %CF
postprobs(:,3)= pdfs(:,12); %IP
postprobs(:,4)= pdfs(:,13); %CLF

for c=1:4
for i=1:samples
postprobs(i,c)=pdfs(i,9+c)*Posterior(c)/max(pdfs(:,9+c));
end
end
```

In these lines of codes, the `postprobs` matrix is being computed. This matrix contains the posterior probability distribution for each of the four nodes IA, CF, IP, and CLF. After that the `pdfs` matrix is used to fill in the first row of each column of `postprobs`. The first row of the `pdfs` matrix corresponds to the prior probability distribution for each node, so the first row of each column of `pdfs` is copied into the corresponding column of `postprobs`. Then, a loop over each node is performed. For each node, another loop over each sample in the `pdfs` matrix is performed. Within this loop, the posterior probability for the current sample is computed. After all of the loops are completed, the `postprobs` matrix contains the posterior probability distribution for each node, given the observed evidence.

The SA-VA Model detects the weakest link based on the posterior probability computation. The risk frequency represents the expected frequency of an undesirable event or outcome. In the following code, the risk frequency is calculated. The risk frequency variance is a predefined value.

```
x=linspace(lowerrange(9+i), upperrange(9+i),samples)
[~,idx]=max(postprobs(:,i))
Risk_frequency_mean=x(idx)
Risk_frequency_variance=4
```

The expected loss associated with the risk frequency is calculated by multiplying the risk frequency mean with the corresponding loss mean for the variable of interest. The variance of loss is calculated by considering both the variance of the risk frequency and the variance of the loss distribution. The loss variance and the loss mean values are calculated using a binomial distribution.

```
E= Risk_frequency_mean*loss_mean(9+i)

Va= Risk_frequency_mean*loss_variance(9+i)+loss_mean(9+i)^2*
Risk_frequency_variance
```

The `premium_fixation` function calculates the premium fixation by multiplying the expected loss by `(1 + OV)` to incorporate the loading factor. It then adds the contingency loading, represented by `k` multiplied by the square root of the variance `V`.

```
PF=premium_fixation(E,OV,k,Va);
function premium=premium_fixation(Expected_loss,OV,k,V)
         premium=Expected_loss*(1+OV)+k*sqrt(V);
         end
```

The resulting value is the premium fixation, which represents the total premium or cost to be charged for the insurance coverage based on the expected loss, loading factors, and contingency considerations. Following results obtained from matlab implmentation.

Results of Matlab implementation;

```
================ IA ============
Posterior Probability for IA = 0.9627
Risk Frequency for IA = 17.9940
Mean Expected loss for IA, E(S) = $3617.69
Variance Expected loss for IA, Var(S) = $165366.23
Premium for insuring against IA = $4020.13

================ CF ============
Posterior Probability for CF = 0.3867
Risk Frequency for CF = 16.9940
Mean Expected loss for CF, E(S) = $3409.51
Variance Expected loss for CF, Var(S) = $163543.46
Premium for insuring against CF = $3790.90

================ IP ============
Posterior Probability for IP = 0.7207
Risk Frequency for IP = 15.9940
Mean Expected loss for IP, E(S) = $3198.96
Variance Expected loss for IP, Var(S) = $162680.53
Premium for insuring against IP = $3559.19

================ ClF ============
Posterior Probability for ClF = 0.4860
Risk Frequency for ClF = 18.9940
Mean Expected loss for ClF, E(S) = $3820.45
Variance Expected loss for ClF, Var(S) = $165629.92
Premium for insuring against ClF = $4243.19
```