Mitigation of XSS Vulnerabilities in Add-ons for Cloud based Applications and a Forensic Framework for Investigating Incidents

Cloud Security

Muhammad Saad Malik

Master's thesis
May-2024
Master's Degree Programme in Information Technology, Cybersecurity

Malik, Muhammad Saad

Mitigation of XSS Vulnerabilities in Add-ons for Cloud based Applications and a Forensic Framework for Investigating Incidents

Jyväskylä: JAMK University of Applied Sciences, May 2024, 100 pages
Master's Degree Programme in Information Technology, Cybersecurity. Master’s thesis.
Supervisors: Jari Hautamaki, Jouni Huotari
Permission for open access publication: Yes
Language of publication: English

Abstract

The recent decade has witnessed a huge adoption of cloud among the IT Enterprises across the globe owing to its numerous advantages including but not limited Scalability and flexibility, Cost Efficiency, Disaster Recovery and Business Continuity and Reduced maintenance. This has also brought to light several security issues, many of which have not yet been studied comprehensively, especially in the context of specific Cloud Applications being used in popular cloud environments such as Microsoft 365, Google Marketplace, Shopify, and Amazon Web Services. This thesis aims to study one of those security vulnerabilities, namely the Cross-site Scripting attack, from both an offensive and defensive point of view.

The main research questions of this thesis are that what are the security architectures of popular cloud application suites, how does the Cross-site scripting attack can occur against the cloud application users via vulnerable add-ons, what are some good recommendations for cloud application developers and vendors to avoid such cases in future and secondly, in case of an XSS related security incident in a Security Operations Center, what are the frameworks in practice for the Security or DFIR analyst to follow and methodically investigate that attack, what are the pros and cons of each of these frameworks and whether an improved version can be presented for a more comprehensive and systematic analysis. The research method used for this thesis is Literature Review, i.e. Systematic Review, which entails a comprehensive seven-step plan for conducting the research.

The systematic review carried out in this thesis along with the empirical analysis of select add-ons from Microsoft 365, Google marketplace and Shopify has confirmed the presence of XSS vulnerabilities in a significant number of cloud application add-ons. And based on the results of this research, the recommendations for developers and vendors include rendering the user input as text rather than html, using a Content Security Policy and creating test items to check for XSS vulnerabilities throughout the development process. Furthermore, hardening the add-on iframe, implementing add-on logic in the add-on server, not in the clientside JavaScript, filtering the scripts in user input and not sharing access tokens to delegate all your permissions would prevent the exploitation of XSS vulnerabilities in cloud application add-ons. And secondly, the comparative analysis of forensic frameworks has revealed that the proposed framework is more comprehensive and systematic, and its adoption would reveal more meaningful insights into the investigation of the attack.

Keywords: Cross-site Scripting, Cloud Application add-ons, Digital Forensic frameworks

.

Malik, Muhammad Saad

XSS-haavoittuvuuksien lieventäminen pilvipohjaisten sovellusten lisäosissa ja rikostekninen kehys tapausten tutkimiseen

Jyväskylä: JAMK-ammattikorkeakoulu, toukokuu 2024, 100 sivua
Tietotekniikan, kyberturvallisuuden maisteriohjelma. Pro gradu tutkielma
Ohjaajat: Jari Hautamaki, Jouni Huotari
Lupa avoimeen julkaisuun: kyllä
Julkaisukieli: Englanti

Abstrakti

Viime vuosikymmenen aikana pilvet ovat ottaneet valtavasti käyttöön IT-yrityksissä eri puolilla maailmaa sen lukuisten etujen ansiosta, mukaan lukien, mutta ei rajoittuen, skaalautuvuus ja joustavuus, kustannustehokkuus, kattavuus katastrofien jälkeen ja liiketoiminnan jatkuvuus sekä vähäinen ylläpito. Tämä on myös tuonut esille useita tietoturvaongelmia, joista monia ei ole vielä tutkittu kattavasti, erityisesti tiettyjen pilvisovellusten yhteydessä, joita käytetään suosituissa pilviympäristöissä, kuten Microsoft 365, Google Marketplace, Shopify ja Amazon Web Services. Tämän opinnäytetyön tavoitteena on tutkia yhtä näistä tietoturva-aukoista, nimittäin Cross-site Scripting -hyökkäystä, sekä hyökkäävästä että puolustavasta näkökulmasta.

Tämän opinnäytetyön tärkeimmät tutkimuskysymykset ovat, mitkä ovat suosittujen pilvisovelluspakettien tietoturva-arkkitehtuurit, miten Cross-site scripting -hyökkäys voi tapahtua pilvisovellusten käyttäjiä vastaan haavoittuvien lisäosien kautta, mitkä ovat hyviä suosituksia pilvelle sovellusten kehittäjät ja toimittajat välttämään tällaisia tapauksia tulevaisuudessa ja toiseksi, jos tietoturvakeskuksessa tapahtuu XSS:ään liittyvä tietoturvahäiriö, mitkä ovat käytännössä ne puitteet, joiden avulla Security- tai DFIR-analyytikko voi seurata ja tutkia järjestelmällisesti kyseistä hyökkäystä, mitä ovat kunkin kehyksen hyvät ja huonot puolet ja voidaanko parannettu versio esittää kattavampaa ja systemaattisempaa analyysiä varten. Tässä opinnäytetyössä käytetty tutkimusmenetelmä on Literature Review eli Systematic Review, joka sisältää kattavan seitsemän vaiheen suunnitelman tutkimuksen suorittamiseksi.

Tässä opinnäytetyössä tehty systemaattinen katsaus sekä valittujen Mi-crosoft 365:n, Google Marketplacen ja Shopifyn lisäosien empiirinen analyysi ovat vahvistaneet XSS-haavoittuvuuksien olemassaolon huomattavassa määrässä pilvisovellusten lisäosia. Ja tämän tutkimuksen tulosten perusteella kehittäjille ja toimittajille annetut suositukset sisältävät käyttäjän syötteen näyttämisen tekstinä html:n sijaan, sisällön suojauskäytännön käyttämistä ja testikohteiden luomista XSS-haavoittuvuuksien tarkistamiseksi koko kehitysprosessin ajan. Lisäksi lisäosan iframe-kehyksen lujittaminen, lisälogiikan käyttöönotto lisäosien palvelimessa, ei asiakaspuolen JavaScriptissä, komentosarjojen suodattaminen käyttäjän syötteessä ja pääsytunnisteiden jakamatta jättäminen kaikkien käyttöoikeuksien delegoimiseksi estää XSS-haavoittuvuuksien hyödyntämisen pilvisovellusten lisäosissa. Ja toiseksi, rikosteknisten viitekehysten vertaileva analyysi on paljastanut, että ehdotettu kehys on kattavampi ja systemaattisempi, ja sen hyväksyminen paljastaisi merkityksellisempiä näkemyksiä hyökkäyksen tutkinnasta.

Avainsanat: Cross-site Scripting, pilvisovellusten lisäosat, Digital Forensic -kehykset

Contents

List of Tables

List of Figures

1 Introduction

In today’s world, nearly everything ranging from our Laptops, Personal Computers, TVs, Mobile Phones and Tablets devices is connected via the internet and that penetration is increasing every day to Embedded & IoT devices. Furthermore, the IT services and products providing companies have now started shifting from the traditional IT infrastructure to the cloud environments including AWS, Azure, and Google Online storage. And for the same reason, web-based applications are now also being migrated to the online storage owing to its numerous benefits, including enhanced scalability, cost efficiency, reliability, performance, global accessibility, and flexibility.

When sensitive data is moved to the cloud storage, it becomes more vulnerable, which enhances the need for robust security measures. Because these services are subject to external control, protection is essential. The majority of today's web applications employ HTML and JavaScript for their interface, with the user's data being processed and stored on the online storage. That provides both consumers and developers with a great deal of flexibility. Users can update the applications on a regular basis without having to install them locally. Take Google Docs as an example of a online storage application. In these situations, consumers entrust the online storage platform provider to protect their data, underscoring the necessity of online storage security (Singh and Chatterjee, 2017).

Nowadays, the majority of cloud storage applications use a microservice design, where the primary functionality is implemented as a separate service that is merely weakly connected to the main service using APIs. External developers can also access the APIs to add new features and functionalities. We refer to these features as extensions. For example, if the user has Google Docs, they could be able to translate the content to a different language by installing an extra extension. Nevertheless, the increasing number of extensions has led to new issues and a rise in the security landscape for online storage apps (Pandey, 2021). Due to the frequently pressing need for new features, untrained developers may create extensions, which could result in less robust security measures than for the main service. Thus, we examine the security risks resulting from potential security flaws in these extensions in this thesis.

In addition, a Security Analyst must be prepared to handle these incidents, such as “cross-site scripting” (also known as “XSS”) threats, in the Security Operations Center. Digital forensics and incident response (DFIR) focuses on finding forensic artifacts and human activity evidence on digital devices to identify behaviors of interest for security (Dimitriadis et al., 2020). Therefore, it's critical that a framework for handling these situations be in place. As a result, we also provide a digital forensic framework for looking into these instances in the second section of this thesis.

1.1 Problem Statement

In this thesis, I have done research in Online storage Security and explored the vulnerabilities associated with Online storage application extensions and more specifically the Cross-site Scripting (XSS) vulnerabilities. In addition, an analysis of current Digital Forensic frameworks is part of this study, which will also provide an enhanced framework for looking into XSS vulnerabilities in Cloud storage Application extensions. This thesis addresses the following primary research questions:

- How do insecure extensions enable XSS threats against users of online storage applications?
- An examination of the structure and layout of well-known software suites, including Microsoft Office 365, Shopify, and Google Suite.
- A discussion about secure product development best practices for extension developers.
- Which Digital Crime Investigation models are currently in use and is there a chance that a better model may be made available for a more thorough and organized examination, particularly when it comes to XSS attacks on online storage applications?

1.2 Thesis Type

The investigation of cross-site scripting attacks on online storage application extensions, the XSS vulnerabilities connected to them, and their fix are all included in this thesis. It also entails a comparative examination of the current Digital Forensic frameworks in use, research on how they work, and the creation of a new framework for looking into XSS-related security incidents in a Security Operations Center. This thesis, which is a literature review, examines extensions on three significant online storage application platforms through a case study.

1.3 Table of Databases used

During the research conducted in the thesis, the following research databases have been utilized, which include a few journals, conferences and some university databases.

Table 1: List of Used Databases

Sr. #	Database	Link
1.	JAMK Thesis Database	https://www.theseus.fi/
2.	Jyvaskyla University Thesis Dat	https://jyx.jyu.fi
3.	Aalto University Thesis Database	https://aaltodoc.aalto.fi
4.	University of Turku Thesis Database	https://www.utupub.fi/?locale=len
5.	IEEE Explore	https://ieeexplore.ieee.org/Xplore/home.jsp
6.	Journal of Cybersecurity	https://academic.oup.com/cybersecurity
7.	ACM transaction on Privacy and Security	https://dl.acm.org/journal/tops
8.	IEEE Transactions on Information Forensics and Security	https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=10206
9.	IEEE Security & Privacy	https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=8013
10.	Elsevier	https://www.sciencedirect.com/journal/computer-law-and-security-review
11.	BlackHat USA	https://www.blackhat.com/upcoming.html
12.	Infosec World	https://www.infosecworldusa.com/
13.	DEFCON	https://defcon.org/
14.	RSA Conference	https://www.rsaconference.com/usa

1.4 Planned Tools

During this thesis, there have been several tools utilized for the writing, editing, and proofreading of the work. Also, tools for the referencing system have also been used and lastly for doing the actual security testing of the cloud applications, some application specific tools have been utilized. These tools utilized for the purpose of this thesis include Zotero, MS Office 365, Quillbot, ChatGPT, Burpsuite, Metasploit, Nmap, XSSer, XSStrike.

1.5 Thesis Structure

The introduction chapter of the thesis is followed by the research methodology, the theoretical backdrop, a thorough description of cloud security, and a list of vulnerabilities related to extensions for cloud apps. Next, the research is put into practice, which is a two-step process. The first part covers a case study of sample extensions from three cloud application marketplaces, while the second part addresses cloud examination techniques. The results are then analyzed and synthesized, followed by the conclusion, future recommendations and finally the ethics.

2 Research Methodology

The collection of steps and methods used to find and examine data regarding a particular study topic is known as research methodology. This chapter introduces the Research Methodology that is being employed in our thesis titled “Mitigation of XSS Vulnerabilities in Extensions for Cloud based Applications and a Forensic Framework for Investigating Incidents”. The research methodology utilized in this thesis work is “Literary Review”.

2.1 Research Question

The security threats resulting from possible security flaws in online storage application extensions are examined in this thesis. In particular, how extension services handle the dubious user input. This is crucial because online storage services' emphasis on cooperation and data sharing makes it simpler to exploit flaws in the processing of dubious data. Our investigation focuses on script injection, also referred to as Cross-site Scripting. Secondly, which Digital Crime Investigation models are currently in use and if it is possible to present an upgraded version for a more thorough and organized analysis.

2.2 Research Methods

The common types of literature review include scoping reviews, narrative reviews, systematic reviews and annotated Bibliography. In this thesis, the research method being used is a systematic Review (Hoffman, n.a.). This review adheres to recognized review principles by methodically searching for, assessing, and synthesizing research information.

Conducting a comprehensive systematic review for the purpose of scientific research entails a series of steps which have been enlisted below: -

1. Defining the Research Question
2. Research Methodology
3. Conducting the **Research Q / Searching for Relevant Data Sources**
4. Assess the Eligibility of Data / Select Studies
5. Analyze the Data: Black Box Testing
6. Extract and Synthesize the Data
7. Disseminate your results

2.3 Research Objectives

The primary research objectives of this thesis are:

- Show how do insecure extensions enable XSS threats against users of online storage applications.
- Examining the architecture and design of popular software suites like Microsoft Office Online, Google Suite, and Shopify.
- A discussion on secure product development best practices for extension developers to adhere to.
- A framework for Digital Forensics that analyzes XSS attacks connected to extensions for cloud applications.

2.4 Related Research

With the assistance of pertinent research articles and thesis works, the background of the XSS attack against Online storage application extensions has been covered in this part. Their works and gaps will be analyzed, and these will be filled in this thesis.

Self-Evaluation of Cloud Storage Deployment Security

The first piece of writing is titled Self-assessment of Security in Online storage Deployment and was written by Simola Vesa. This thesis aims to pinpoint crucial elements businesses must consider throughout the lifecycle of a specific online storage service (Koskinen and Simola, 2019). It deals with the problem of identifying the risks directly associated with online storage computing. This work produced a checklist that can be used to recognize, accept, and reduce the risks associated with vulnerabilities related to online storage environments.

Amazon Cloud Adoption of Security Best Practices

Polar Electro, a firm that seeks to enhance the security of its online storage services and follow best practices for the security of its own services, hired Jukka Veijanen to work on this project (Veijanen, 2020). A vulnerability scan for the company's network and online storage-based instance was selected for this reason. One EC2 instance was picked, and Amazon Inspector was the tool of choice. Thus, the rules were made for scanning the tests and were executed after the tool was enabled. Both noteworthy and high severity findings were present in the inspector report. The thesis' findings demonstrated the importance of Amazon online storage security testing in the software development process. The assigner received a favorable recommendation for incorporating security best practices into their setting.

Industrial Internet of Things Security Analysis of Web Applications

The Security Analysis of Web Applications for IIoT is the focus of this paper by Naryana Srikar Bhava. Due to the growing number of cyberattacks on online applications worldwide, web applications must undergo security auditing by third-party businesses before being deployed on industrial networks. This research thesis develops a Secure Software Development Lifecycle (SSDLC) and security auditing standards for IIoT online applications using a constructive methodology. The implementation of GDPR laws, authentication, secure data transmission, and a security principles checklist can detect security risks early in the SSDLC, according to the study, which examines security standards and expert advice. As a result, when deployed in industrial networks, IIoT applications are resistant to significant cyberattacks (Narayana, 2022).

Problems and Difficulties with Cloud storage Security: An Overview

Another study on online storage security issues and challenges by Ashish Singh and Kakalu Chatarjee: The significance of online storage computing in the modern world is examined, along with the difficulties in transforming local computing to remote computing, or the security concerns and difficulties that arise for both service providers and users as a result of the distributed, virtualized, shared, and public nature of the online storage (Singh and Chatterjee, 2017). After that, it offers several defense strategies to deal with the security flaws at various online storage locations.

Cyber Incident Response in Public Cloud

The purpose of Loukasmaki Henri's thesis is to investigate and discuss the challenges associated with reacting to protection breaches in the context of the online storage domain. Additionally, practical advice and insights regarding the preparation, detection, analysis, containment, eradication, and recovery from online storage-based incidents are provided (Loukasmäki, 2023). This thesis also analyzed the services and products of two major online storage providers, namely AWS and Azure, in order to evaluate various incident response capabilities, tooling, techniques, and strategies.

The thesis works discussed above have researched various aspects of Online storage and Web Applications from a security perspective. The work by Simola Vesa merely discusses the threats there are to the online storage in the form of a checklist but does not discuss anything specific i.e. a certain kind of security attack and its resolution. The work by Jukka Veijanen is essentially on Online storage Security Testing, in which Amazon Inspector was chosen as the scanning tool for scanning the online storage environment and it was able to find some high severity findings. The Security Analysis of Web Applications for IIoT was the subject of the following study by Naryana Srikar Bhava, which emphasized the significance of authentication, secure data transmission, GDPR, security frameworks, and a checklist of security standards. The difficulties that develop for both consumers and providers as a result of the online storage's dispersed, simulated, exchanged and open nature are discussed in Ashish and Kakalu's thesis. Lastly, from the standpoint of a Security Operations Analyst, Loukasmaki Henri's work on Cyber Incident Response in Online storage has brought attention to security events pertaining to Online storage.

However, none of these thesis research works focus on any specific area of online storage security and online storage applications and performs a deep dive. For example, some specific vulnerabilities in Google Online storage that have not been addressed and therefore, have been taken advantage of frequently. Similarly, some specific attacks such as DDoS, Account Hijacking, Misconfigured Online storage Services, Phishing and MiTM Attacks. In other words, these works have covered the breadth of online storage security issues but lack depth. And there is a need for case studies to be conducted targeting some specific and in-depth area of online storage security. Therefore, by concentrating on a single area of online storage security—the XSS Attack in Online storage Application Extensions—this thesis has attempted to solve a few of these problems. This work discusses online storage applications first, followed by the several online storage markets and the extensions that are offered in those marketplaces. Lastly, it addresses the different risks that these online storage apps encounter, the most common threats, and goes into great length on the Cross-site Scripting Attack, a vulnerability that is common to online storage applications.

2.5 Importance of this Research

As was said in the preceding section, the focus of this study is on security threats resulting from possible security flaws in extensions, particularly script injection, also known as XSS. The dangers of XSS in connection with online storage application extensions have not received enough attention in the literature. The following is explained in the thesis in an effort to close this gap:

- A thorough explanation of how susceptible extensions might lead to an XSS attack against users of online storage applications;
- An examination of the Microsoft 365, G Suite, and Shopify application suites' architecture designs and security measures;
- Conducting an empirical investigation to determine how the issue has spread;
- For the security of their products, we provide extension developers certain recommended protective solutions and best practices. Additionally, I've shared the insights I gained from the analysis regarding design decisions and how they affect an extension system's security;
- Lastly, we suggest a digital forensic methodology for looking into the attack on online storage apps that is related to XSS.

3 Theoretical Background

The theoretical backdrop of the thesis is covered in Chapter 3 along with the literature review and the third stage of the systematic review, which is the research phase or searching for data sources.

The actual data collection from various sources is covered in detail in this chapter, which begins with an explanation of what the cloud storage is, the different types of online storage hosting, their service and deployment models, common online storage security issues, cloud storage application extensions, and cross-site scripting attacks in relation to different types of vulnerable extensions. Sample cases from well-known online storage application suites, such as Google Workspace, Microsoft 365, Amazon Web Services, Shopify, and Zoho Workplace, are also covered in this chapter.

3.1 Cloud and its Types, Service models and Deployment Models

Generally speaking, a cloud service is considered a product made out of services hosted on the Internet. This group may include servers, networks, storage units, software, and other services. These things could be sold and used anywhere in the world. Users don't need to make any changes to utilize online storage-based programs, and they can reach their locally saved data and services from any location on the internet. Moreover, users may periodically share files, data, and information with other users and several platforms via the online storage infrastructure (Koskinen and Simola, 2019).

There are several motivations for organizations to turn to the cloud (Koskinen and Simola, 2019), and among these are:

One on demand Self-service:

It is possible for a cloud user to allocate hardware resources, like network drives and server time, at will. Because the user may manage resources automatically and use them when needed without requiring contact between each service provider and people, there is no longer a need for an intermediary.

Wide Network Access:

Users gain from the cloud storage and have standard techniques to control them, regardless of the end-user platform.

Resource Pooling:

In a multi-tenant model, cloud storage resources including memory, computing power, storage, and network bandwidth are pooled to meet the needs of numerous clients based on user demand. Only offsite locations under the owner's control may be used for private online storages, while providers may let customers designate general server locations.

Quick Elasticity:

Resources can be assigned and released in the cloud storage in a dynamic and elastic manner. This offers automatic scalability for increased or decreased resources as needed. Denial-of-Service (DoS) assaults are declining in part because of this, as businesses who have sufficient online storage accounts are no longer at risk.

Measured service

Resources in the cloud storage are automatically managed and optimized based on storage, processing, bandwidth, and active user accounts thanks to metering features. Both clients and the online storage provider benefit from transparency as a result of the tracking, managing, and documenting service usage details.

Cloud Deployment Models:-

Four deployment models have been defined by Mell & Grance (2011) in the National Institute of Standards and Technology Definition of Online Storage: private, community, public, and mixed.

Public Cloud

According to the traditional mainstream definition of online storage computing, public online storages, also known as external online storages, are elsewhere external providers that share assets and charge on a utility computing basis. Through web apps or web services, resources are dynamically delivered on a fine-grained, self-service basis over the Internet. Public online storage are used by both individuals and businesses. Amazon Web Services (AWS), the Google Application Engine, Microsoft Windows Azure, and Salesforce.com are examples of public cloud storage service providers. Businesses like IBM's BlueCloud, Amazon EC2, and Rackspace's Cloud Offerings provide IaaS. Similar to the Windows Azure Services platform, Google's App Engine, Amazon's SimpleDB online storage hosting, S3 Simple Storage, and Online storage Front, PaaS is provided at the application layer (Vines and Krutz, 2010). Serving numerous customers at once, the public online storage is run by a service provider out of a single, centralized data center. It offers scalable pricing, shared infrastructure, opportunities for research and innovation, along with regular maintenance and upgrades. Saving money by allowing users to share resources is one of the public online storage's main benefits. Online storage technology adoption is strongly encouraged by the availability of shared infrastructure, remote hosting, dynamic licensing, and provisioning. Businesses who use the public online storage don't have to worry too much about infrastructure upkeep because it's the service provider's primary duty. Most tasks related to maintenance and security are managed by the supplier. However, it's crucial to set up Service-Level Agreements (SLAs) defining uptime and unique configuration needs prior to moving important apps to the public online storage. With the provider handling data and apps, clients have the least control while using the public online storage. Client control over sensitive data is diminished as a result of the provider's management of logging, monitoring, and control implementation. As a result, security features like encryption, access control, and identity verification are becoming essential parts of online storage architecture.

Private Cloud

Private networks that are hosted internally and frequently allocated to a single business are used in private (internal) online storages. To protect data security, extra precautions could be required, such as keeping data isolated between departments. Private online storage may also be accessible to distributors, internal Users/suppliers, business partners, and company offices. Private online storage configurations usually make use of virtualization technologies in the company's on-site data center (Winkler, 2012).

Community Cloud

A community cloud is a shared system utilized by multiple companies. It could be located on or off-site, under the direction of one or more organizations, or perhaps by a different entity entirely. Through the web or an open personal network, users can access the community online storage by using a Virtual Private Network.

Hybrid

Public, private, and community storages are examples of external online storage environments that are integrated into a mixed storage. According to NIST, mixed online storage combines two or more distinct online storage systems (private, community, or public) that retain their identities but are interconnected, allowing data and application transfers between them, such as storage bursting for load balancing. Krutz & Vines define a "online storageburst" as the dynamic deployment of an application primarily hosted on an organization's internal system, which can also be deployed to the online storage during high demand periods. An organization's private online storage becomes a mixed storage environment when it uses resources from the public or community online storages. While keeping sensitive or important apps and data on their own private online storage infrastructure, businesses can deploy secondary applications in a public online storage (Walter, 2013). Organizations use private online storages, usually housed in nearby data centers, to protect sensitive or important data internally. While private online storage systems are being upgraded, they might be phased out and used as a testing ground for public online storages. Winkler suggests that while media streaming services (such those for images or videos) could make use of the public online storage's capabilities, a private online storage platform should host an organization's website that contains important and confidential data. Enterprises that manage their own data centers might also reap benefits from online public storage. Compliance requirements prohibit certain industries, such as financial services, from hosting consumer data on external third-party online storage systems. In a similar vein, government organizations frequently believe that external online storage storage is too open to cyberattacks. It's important to remember, though, that information kept on laptops may be more prone to security breaches than information kept on online storage servers. A online storage developer can take on several responsibilities, such as Online storage Auditor, Online storage Service Provider, Online storage Service Carrier, Online storage Service Broker, and Online storage Service Consumer, in addition to the differences between different types of online storages. (Charif, 2014).

Cloud Service Models

Cloud deployment models are one of the three primary deployment types; they include a number of noteworthy benefits and drawbacks. From a business standpoint, by providing greater flexibility, these methods seek to improve client accessibility to IT. The following is a list of the three models' general characteristics.

Cloud infrastructure services (IaaS)

Online storage services fall under the most fundamental type known as system as a service, or IaaS. Typically, it refers to a service that includes infrastructure, real or virtual machines, other pertinent resources like image storage, networking, firewalls, load balancers, and other security features. One benefit of the IaaS online storage for the user is the ability to abstract and use specific data center-related functions from, say, a web interface or an API. Since it is no longer necessary to manage every aspect of the infrastructure, administrative tasks may now mainly focus on server-side operations, such as operating system administration and maintenance as well as third-party software maintenance. With this type of online storage service, the customer is in charge of all software development and administration because just the infrastructure is provided. It is crucial to emphasize that the customer is still responsible for maintaining and using the online storage resources appropriately, including setting up apps, even though they have minimal to no control over the foundational infrastructure that powers the online storage-based services (Kavis, 2014).

Cloud platform services (PaaS)

As was already noted, IaaS is unable to address the various automation challenges or scalability issues that businesses encounter, especially when it comes to software. The user is the source of all software infrastructure components. To some extent, software platforms are available from PaaS providers to assist in this process. Common software platforms that can be accessible via a range of APIs include databases, logging, and payment systems (Kavis 2014).
Many PaaS solutions also concentrate on optimizing the provisioning procedures for the virtual machines and containers that run the apps. To enable automatic scalability, for example, a Kubernetes platform offers an interface for containers. Applications and their dependencies are contained in manageable chunks for distribution and execution on online storage platforms using containers, a relatively new computing paradigm. Kubernetes or Docker Swarm are two orchestration solutions that can then be used to manage these containers. In summary, PaaS deployment can be considered a step above Software as a Service (SaaS) deployment because it removes the need for clients to own infrastructure in order to deploy software applications.

Cloud Software as a services (SaaS)

By distributing software programs online, SaaS reduces the amount of human effort required from the client. With SaaS, users and configuration are the only things that clients need to worry about; the service provider takes care of everything else. Customers can benefit from this strategy in a number of ways, such as the avoidance of platform maintenance and the removal of the requirement for staff to handle maintenance duties, especially for non-core functionalities. SaaS services, however, are not appropriate for software that requires significant modification in addition to preset configuration changes. An email service could serve as an example of how online storage services and SaaS are layered. Here, virtual machines housed in the provider's facilities and hypervisors are used to power customer-specific interfaces running within containers managed by the orchestration tool of the service provider. SaaS solutions are widely available in the market nowadays. Customers' lack of control over the underlying computing infrastructure and software distribution is a crucial component in comprehending the Software as a Service (SaaS) paradigm. SaaS and PaaS differ from one another in this essential way (Kavis 2014).

3.2 Security in Cloud

Respecting the three security principles—non-repudiation, authentication, and authorization—as well as confidentiality, integrity, and availability—is what it means to be secure. According to Ramgovind et al. (2010), there are several options for implementing security, including network and information security, personnel security, operations and communications security, and physical security.

Because online storage computing is globally distributed, its providers have global clientele. There are no industry standards for this kind of data processing and storage. The majority of vendors use extremely limited unique data formats. This is why companies often use non-standard data formats to keep customers from moving, which makes data translation and transfer difficult and expensive. In this respect, even though individual clients are dissatisfied with the vendor services and excessively reliant on their proprietary tools, they are unable to transfer. Chaudhary (2020) asserts that a lot of companies still rely on vendors with subpar security designs that can't survive cyberattacks as a result of bad choices made when selecting a trustworthy vendor to handle particularly confidential, sensitive, or financial data (Harkut 2020). It is essential for enterprises to assess online storage computing providers' security in order to prevent excessive vendor lock-in. A greater grasp of the architecture of online storage computing solutions provided by suppliers is necessary for the corporate decision-making processes.

Access control measures can be used along with control over people, logical, and physical security when sensitive data is stored using on-premises application deployment methods. Since firm data is stored offsite while using the online storage, the storage distributor must protect against hostile users and vulnerabilities in order to stop breaches and guarantee data security. Subashini & Kavitha suggested utilizing fine-grained authorization and strong encryption techniques to manage data access. Administrators were urged by Subashini & Kavitha to remove the OS Guest user and cancel access to client instances, in the same manner as Amazon does with its EC2. Subashini & Kavitha did draw attention to the fact that in order for EC2 administrators to access a host, each user need a unique Secure Shell (SSH) key that is robust enough to withstand cryptography. For this kind of access, audits and logging are done often. Before uploading, users should encrypt their data. Subashini & Kavitha advise assessing the security of data kept in the cloud by looking for vulnerabilities related to cross-site request forgery, OS and SQL injection, XSS, access control, cookie abuse, concealed field misuse, unsafe data hosting, and unsafe setup.

As an alternative, Wang et al. looked at how a online storage client authorizes an external auditor (TPA) to check the accuracy of dynamic data kept on the online storage. Furthermore, they conducted an impartial and unbiased evaluation of service quality.

3.3 Common Aspects of Cloud Security

Since online storage technology is a relatively new computing paradigm, some confusion exists over how security can be guaranteed at each level of the system. Because of this uncertainty, decision-makers are now giving security top priority as it pertains to online storage technology.

The following are some of the general issues with online storage computing that have been noted (Halpert, 2011):

- if users utilize more resources than expected, especially in public online storage environments, the availability of those resources could be jeopardized;
- denial of service assaults might be launched using the online storage's vast capacity;
- data residency is difficult since different nations and areas have varied laws governing the processing of personal data;
- when several clients share the same infrastructure, multi-tenancy allows for economies of scale but also raises compliance issues;
- keeping track of shared infrastructure logs could be difficult because multiple tenants' data could show up in the same log files;
- service level agreements regulate these parameters, which in turn affect the online storage's performance and service levels based on the services that are purchased;
- since data evacuation lays out the protocols for deleting information from shared infrastructure, it is important to take this process into account;
- supervisory access is problematic since it gives the service provider the most access to the infrastructure.

Following, we will delve into security concerns regarding various online storage deployment models, starting from broader issues and progressing to those more specific to deployment models.

Public Cloud

It is clear from the previous explanation that public online storages house several tenants on a single physical infrastructure. To maintain customer separation, the majority of public online storage providers use software-based isolation and permission constraints. Hardware-level isolation is feasible, although it could be more expensive. It is therefore essential to understand how the selected platform manages multi-tenancy. For example, does it allow various directory services to be used, such as LDAP or Microsoft Active Directory, where each tenant has their own directory?

Private Cloud

Multitenancy is not as much of a problem in private online storages as it is in public ones; in fact, private online storages alone may be considered a solution to the multitenancy problem (Bond 2018). According to Halpert (2011), online storage service providers and customers are typically housed within the same company, giving them greater control over factors like service quality. One illustration of this is that, depending on how important the task is to the company, workers of the client can more readily influence how it is managed. Because the infrastructure is devoted to this client alone, the customer must pay for the entire setup in order to have this control.

Hybrid Cloud

As was previously noted, a hybrid online storage is subject to the same considerations as previously outlined because it combines elements of storages that are both public and confidential. It's crucial to understand that all security precautions and guidelines should be followed as though the service were solely housed in a private online storage, even if some components of it function in a public online storage.

Security aspects in IAAS

It is important to understand the capacity of the service provider in an equipment as a service (IaaS) online storage to keep an eye on the performance of any virtual machine running inside the infrastructure (Halpert, 2011). In addition, even while the service provider handles patching at the hypervisor level, the customer is still responsible for implementing any necessary patches inside their virtual machines. As stated in Mather, Kumaraswamy, and Latif's book "Online storage Security and Privacy" (2009), there is another way to look at this division of labor. They break down IaaS security into two parts:

• The security of the virtualization software, which includes all virtualization-related software, including paravirtualization and hypervisors. The service provider is in charge of maintaining this layer.
• Security of virtual machines running particular operating systems and software stacks, or guest operating systems controlled by customers. The consumer is accountable for this.
This breakdown of responsibilities is fundamental and applicable across various service delivery models.

Security aspects in PAAS

The main difference between PaaS and IaaS is that the latter requires the service supplier to patch and configure both the guest operating system and the hypervisor. Scalability benefits and the usage of current system software are guaranteed when the service provider assumes this duty. Furthermore, less administrative overhead may result from assigning some maintenance tasks to the service provider rather than internal employees.

Scalability can be thought of as a security-enhancing feature that reduces the effect of particular attacks, including service disruption threat. In a similar vein, less administrative labor could free up staff members' time to improve software quality.

Platform as a service (PaaS) providers run the danger of future breaches, according to Jamsa (2012). It is underlined that applications operating on a PaaS platform may experience impaired performance, availability, and security if the supplier disregards service level agreements (Jamsa, 2012). Similarly, McGrath (2012) argues in his book "Understanding PaaS" that while PaaS may not fundamentally differ, customers may not be privy to the various activities occurring behind the scenes, such as monitoring, adjustments, and ongoing enhancements. Additionally, he says that while PaaS might not be appropriate in every situation, it is still a feasible choice that can improve the security of a sizable amount of the computing infrastructure needed for applications.

The aforementioned concerns are further reinforced by the findings reported in Mather et al.'s (2009) book, "Online storage Security and Privacy." They point out that in most cases, platform as a service (PaaS) companies don't reveal the configuration details of their security mechanisms. It also involves information about operating systems and the protocols used to protect the hosts that use the PaaS model. This practice is justified by the possibility that disclosing such information could enable attackers to take advantage of weaknesses and initiate assaults.

Based on all that has been said so far, it appears that host-level security management may not be necessary for consumers. It's crucial to keep in mind, though, that it is still the customer's duty to make sure the service provider satisfies any unique needs they may have.

Security aspects in SAAS

Typically, SaaS manifests as an application that is accessible through a web browser and is managed and supplied by a service provider. Customers can minimize administrative obligations by reducing their dependency on on-site data center software and applications. Furthermore, according to Jamsa (2012), because SaaS is frequently multi-tenant, modifying SaaS delivery may be difficult, expensive, and occasionally impractical.

Since the service provider hosts everything from applications to hypervisors and physical hardware, they might be able to access all user data for their software as a service (also known as SaaS) product.
Moreover, the remarks made earlier regarding PaaS are still applicable: it is the client's responsibility to ensure that the service provider meets any unique requirements they may have..

3.4 Cloud Application Add-ons

An extension provides custom commands and capabilities to an online storage program (also known as the host application). It is also commonly referred to as an integrated, plugin, extension, or app. Application-defined programming interfaces (Even though the extension is a stand-alone online service with its own server and client elements, APIs enable it to access user data and some fundamental features of the application that is being used.

Every extension has an online front-end that is built with HTML and JavaScript in addition to the web service. The host application imports the extension user interface (UI) into an iframe and presents it smoothly as part of the host program's user experience when the user activates the extension (Kolb 2016).

The extension user interface can connect with the host program in two different ways. It can communicate directly with the host-application user interface component or via the backend servers. In the latter case, the extension server situated in the online storage is typically contacted via the extension user interface. Through backend APIs that are hidden from the user, this server interacts with the host application server and retrieves user data.

Authorization management

Authorization management is a common tool used by online storage application providers to restrict the user data that their extensions can access within the host application. Each extension requires certain permissions in order to work. The host program frequently asks the user for explicit permission clearance when it is first launched or installed. Generally speaking, access control is quite coarse-grained, requiring the user to approve every request for permission to access either a particular document or all user data. Furthermore, the extension keeps its permissions until the user removes it.

Exchange, vulnerabilities, and solutions

Online storage application providers frequently showcase additional functions in an online exchange in which users can select and activate any functions they desire for their apps. For instance, a wide range of extensions compatible with Google products, such as Gmail and Google Docs, are available in the G Suite marketplace. Typically, the application vendor offers only a limited selection of extensions, with the majority being developed by third-party contributors. Some of the popular marketplaces for Online storage Application extensions include G Suite Marketplace, Salesforce AppExchange, Microsoft AppSource, Atlassian Marketplace, Amazon Web Services Marketplace, IBM Online storage Catalog, Zoho Marketplace, Shopify.

For the scope of our study, we have chosen the Google Online storage Marketplace, Shopify and Microsoft Office 365 and the Amazon Webservices marketplace. Some sample online storage application extensions will be chosen from each of these marketplaces, their possible exploits and vulnerabilities will be found, and then possible solutions and recommendations will be presented for the future developers and potential users of that application.

3.5 Types of Security threats in Online storage

The majority of businesses use largely online storage digital services, which makes online storage-deployed systems vulnerable to a variety of network assaults, including Brute Force, Man in the Middle, Distributed Denial of Service (DDoS), and Data Breaches. Cyberattacks on the internet or online storage networks seek to obtain private information while adhering to stringent authentication and authorization guidelines (Chou, 2020). These assaults usually target online storage system vulnerabilities and possible weak points with the intent of stealing user intellectual property or carrying out malevolent insider operations. SaaS, PaaS, and IaaS are the three categories into which online storage computing services fall; these terms relate to the provisioning models of software, platforms, and infrastructure, respectively. Each level serves different functionalities, such as real-time system operations and information processing, varying based on software, infrastructure, or platform emphasis. Online storage computing security attacks are categorized according to SaaS, PaaS, and IaaS levels, as depicted in Figure below: -

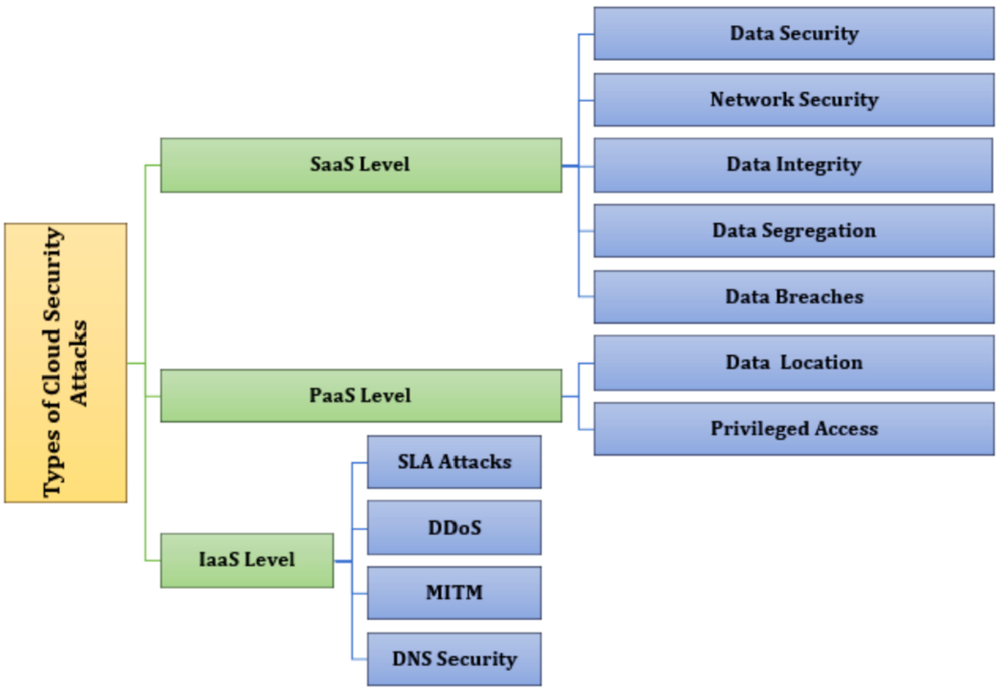


Figure 1: Types of Cloud Security Attacks

Figure above illustrates the primary types of online storage security attacks, each varying in terms of mediums, storage locations, communications, port numbers, and other factors. These threats and attacks are further delineated and succinctly described in the subsequent sections.

SaaS Level

SaaS is a software delivery mechanism that is licensed through subscription and is made available online by a third party. IaaS and HaaS are two aspects of online storage technology that let users outsource their IT systems to the online storage (Rani & Ranjan, 2014). PaaS is a online storage computing layer that developers mostly use to access a platform for creating and building online storage-based systems and applications. SaaS, often known as on-demand software, is a type of online storage computing in which software is hosted centrally and licensed by subscription. SaaS involves centrally hosted software that is licensed through subscriptions. The predominant method of data protection in the online storage is data encryption; however, significant risks of data breaches at the SaaS level include phishing and malware attacks aimed at stealing user credentials. Users of SaaS online storage services must familiarize themselves with various network attacks and threats, along with their potential impacts, when operating within the online system (Rani & Ranjan, 2014). The prevalent types of online storage security attacks at the SaaS level are briefly outlined below.

Data Security

Data security concerns are a significant consideration when data is stored online or within a real-time system hosted in the online storage. To mitigate the risk of Intellectual Property Theft, measures such as data encryption and protection against malware or unauthorized access protocols must be implemented (Jathanna & Jagli, 2017).

Data security over the Cloud

Creating rights and permissions to limit access to particular users is a basic procedure for guaranteeing online storage data security. Nonetheless, insufficient control and access management for sensitive data, as well as the incapacity to monitor data while it is in transit to or from online storage applications, are a few typical network security issues in SaaS (Aljumah & Ahanger, 2020).

Data Segregation

Inadequate monitoring of data during its transfer to and from online storage applications can complicate the process of data segregation, especially for sensitive information. The most effective preventive measure in such scenarios is to implement robust encryption for the data (Majeed, 2020).

Data Breaches

Data encryption has the potential to mitigate data breaches by rendering the data less accessible unless decrypted, even if it has fallen into the hands of an unauthorized individual.

PaaS Level

PaaS gives users access to an application system for creating, implementing, running, and overseeing online storage computing services. However, PaaS has a number of drawbacks, such as runtime problems, vendor lock-in, integration difficulties, data security issues, and customisation problems with older systems. Some of the dangers that are unique to the PaaS level are listed below.

Data Location

Data location has emerged as a prevalent online storage security concern across PaaS, IaaS, and SaaS platforms, underscoring the importance of robust data encryption and protection against malware.

Privilege Access

Ensuring privilege access and implementing effective access control protocols are vital for safeguarding data transmitted through online systems. Common practices for strengthening system security include establishing intricate passwords with robust encryption mechanisms and conducting continuous real-time monitoring and threat analysis under careful supervision (Blum, 2020).

IaaS Level

As-a-Service, or IaaS, refers to a web-based solution that provides sophisticated application programming interfaces (APIs) for abstracting different low-level details of a network infrastructure. IaaS includes geographical locations, physical computer resources, data segregation, backup mechanisms, security measures, and scalability. We briefly describe some of the most common dangers related to IaaS below.

Service Level Agreements SLA

The Service performance agreements plays a vital role in defending against DDoS attacks, necessitating thorough analysis to establish stringent configurations and protocols aimed at mitigating such attacks (Patrick & Satyanarayana, 2020).

Distributed Denial of Service (DDoS)

The DDoS attacks poses a significant threat to online storage computing by inundating a single IP address with an overwhelming number of ping connection requests, leading to system overload and collapse (Srinivasan et al., 2019).

Man in the Middle (MITM)

An attacker can eavesdrop by intercepting and stopping a communication or data transmission, which is known as an MITM attack (Shakshuki, 2016).

DNS Security

DNS security, also known as Domain Name System Security, is vital for safeguarding the DNS system, primarily utilized within a specific IP network. It serves to authenticate the origin of DNS data, thereby enhancing protection against attacks and ensuring data integrity (Aishwarya, Sannidhan, & Balaji, 2014).

Classification of Cloud Security Attacks

Below are additional online storage security threats outlined based on attacks originating both internally and externally to the organization.

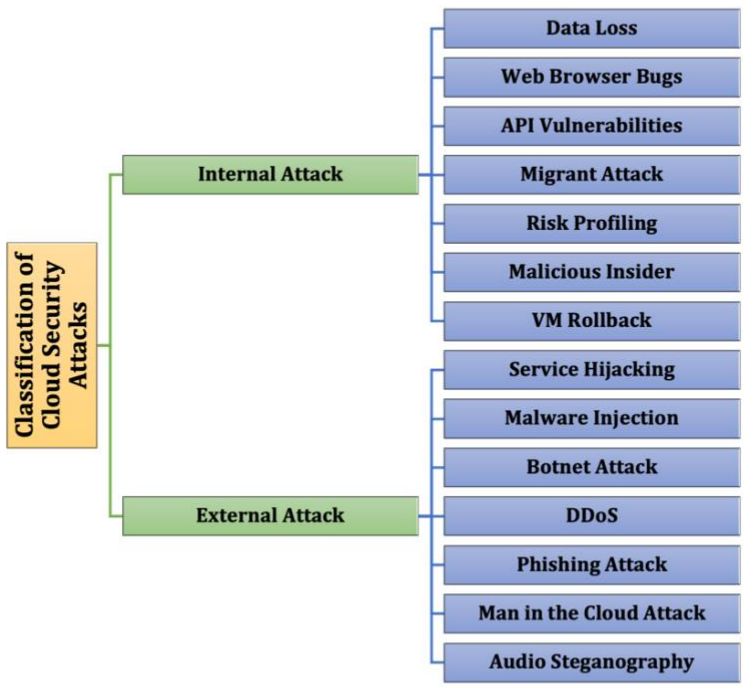


Figure 2: Classification of Cloud Security Attacks

Internal Attacks

Internal attacks typically stem from the negligence of employees and staff associated with a particular organization. Employee theft and sabotage, unauthorized worker access, inadequate cybersecurity actions, and risky behavior are the main causes of these threats (Javaid, 2013). Below are some examples of attacks initiated by insiders within the organization.

When a online storage system's hardware or software faults, usually as a result of human mistake, data loss frequently results. Processes involving data migration may result in this. When staff members utilize out-of-date software and services, web browser bugs present an additional risk to online storage computing (NSA, 2020). API Online storage computing vulnerabilities can occur when APIs don't have encryption, have unsecured endpoints, or have insufficient authentication because of business logic errors (Ariffin, Ibrahin, & Kasiram, 2020). A small number of infected virtual machines (VMs) cause excessive resource usage in order to fool online storage monitoring systems. This is known as the "migrant attack," which is a multi-resource denial-of-service attack on online storage virtual machine migration schemes (Chandrakala & Rao, 2018). Risk Profiling is crucial for insiders to assess the likelihood, impact, and mitigation of potential risks within the organization (Tadapaneni, 2020). Malicious insiders, such as former employees, contractors, or business associates, who possess privileges and authorization to the system, can undermine the organization's security by deploying malicious malware and computer viruses (Duncan, Creese, & Goldsmith, 2012). VM Rollback is an attack wherein a compromised hypervisor restores a Virtual Machine from a previous snapshot without the users' knowledge (Almutairy, 2019).

External Attacks

External attacks typically breach and exploit the online storage system from outside influences, often through internet connections (Usman, Awwalu, & Kamil, 2016). Launching an external attack on the online storage system can be devastating, potentially disrupting the organization's daily business operations and resulting in significant losses. Some external attacks pose severe risks for organizations utilizing online storage systems.

Service Hijacking is a significant security threat for online storage computing services, occurring when unauthorized individuals gain access to the system over the online storage (Tirumala, Sathu, & Naidu, 2015). Malware Injection is another external threat to online storage computing, where malware infiltrates the organization's online system through various methods, such as process injections (Watson et al., 2015). Botnet Attacks, akin to DDoS attacks, involve automated bots launching attacks on specified targets from numerous computers, potentially overwhelming and crippling the system (Anwar et al., 2014). Phishing Attacks are malicious attempts to trick authorized users with fake applications, often resembling legitimate ones, to obtain important information like user credentials and personal details (Basit et al., 2020). Man in the Online storage Attacks exploit vulnerabilities in online storage systems, enabling hackers and unknown parties to access and control stored data and information in popular file synchronization services such as Google Drive and Dropbox (Jabir et al., 2016). Audio Steganography is a rising and vulnerable attack on online storage-based systems, allowing data eavesdropping by attacking LAN hidden within audio file formats (Awadh, Hashim, & Hamoud, 2019).

3.6 XSS threat

It is now imperative for many commercial and governmental organizations to use web applications through the internet to cut expenses, expedite processes, enhance service quality, and expand the audience. Users profit much from internet services as well. These benefits do, however, include certain inherent hazards. Web apps that need users to register using input forms are vulnerable to a variety of hacking techniques that compromise their own and users' private information.

Since 1999, one of the main threats to the security of many web applications has been identified as XSS. Its attacks use vulnerabilities known as XSS to inject malicious code into trusted online apps, their plugins, or hosting servers. Because of this, malicious scripts can be injected into these programs to compromise them. Attackers can then take advantage of this to obtain elevated access privileges and sensitive user data, including usernames and passwords, when users view hacked pages on their browsers. XSS attacks usually happen as a result of insufficient user input control methods in input forms. This is due to a number of factors, such as the fact that developers are not well-versed in security standards, the necessity of quickly adapting online services to changing customer demands, and the attack's intrinsic simplicity, which views user inputs as possible attack vectors. XSS attacks are difficult to trace because they might come from the client or server side. As per the Open Web Application Security Project (OWASP) guidelines, most current report from 2021, cross-domain scripting (XSS) vulnerabilities have been regularly listed among the top 10 vulnerabilities since 2003.

We are doing a thorough mapping analysis because of the increasing attempts to mitigate XSS threats and their widespread incidence. The objective of this work is to classify all published studies that have addressed the XSS problem since it was first identified. By (1) detecting any biases towards specific types of XSS assaults or scientific remedies, and (2) developing an extensive classification of XSS attacks and associated mitigation measures based on a study of recent research, we want to discover any potential gaps in our knowledge. We characterize each class of solutions, highlighting their relative merits and drawbacks, and, when practical, we contrast related strategies. This project provides a comprehensive roadmap for future research and assists academics and industry practitioners in staying up to date with current efforts.

Apart from the three basic forms described by OWASP, more recent research has included different variations of XSS assaults. According to OWASP, a widely used classification involves differentiating between client-side and server-side XSS according to the input and processing locations of untrusted data. This classification is met by a number of the research that are part of this review. But in our analysis, we suggest a more comprehensive classification scheme depending on where the vulnerability that gave rise to these attacks originated. As such, we distinguish three different categories: Third party-based, Application-based, and Collaboration-based. This proposed classification of XSS assaults is shown in Figure 5, along with the number of research that cover each sort of attack. Then, we give a thorough explanation of each category along with a list of the particular assaults that fall under each one ("10 Useful scenarios for XSS attacks", 2024).

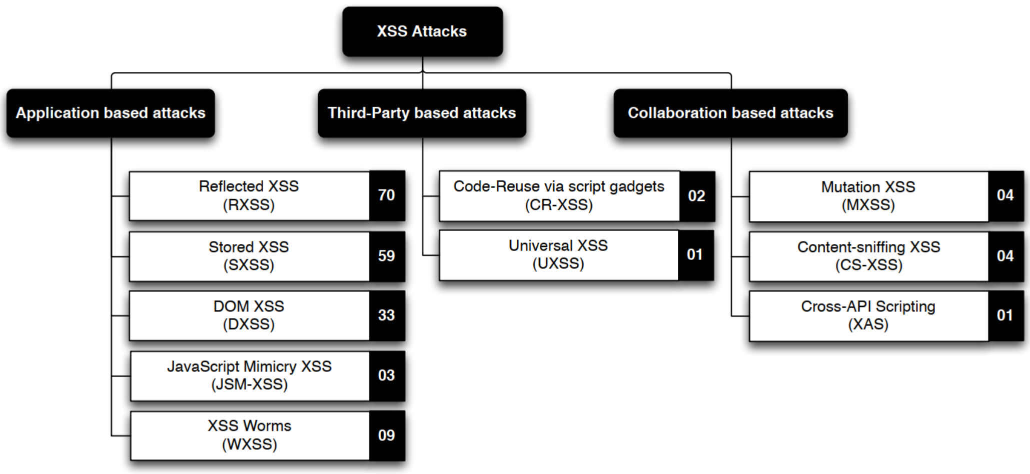

XSS Attacks
Application based attacks
Third-Party based attacks
Collaboration based attacks
Reflected XSS (RXSS) 70
Stored XSS (SXSS) 59
DOM XSS (DXSS) 33
JavaScript Mimicry XSS (JSM-XSS) 03
XSS Worms (WXSS) 09
Code-Reuse via script gadgets (CR-XSS) 02
Universal XSS (UXSS) 01
Mutation XSS (MXSS) 04
Content-sniffing XSS (CS-XSS) 04
Cross-API Scripting (XAS) 01

Figure 3: Typesof XSS Attacks

Application-specific XSS attacks

This kind of attacks is primarily caused by flaws found in web applications. Inexperienced developers frequently fail to recognize how crucial it is to provide sufficient sanitization procedures for all user-controlled inputs. Five contemporary threats fall under this category, three of which are the basic OWASP attacks that are explained in:

1. Reflected XSS (RXSS): their attacks happen when fraudulent information sent with an HTTP request gets incorporated into server replies without being properly cleaned up. As a result, once browsers have produced these malicious data components, they can run client-side. Web applications with search features are common targets for RXSS attacks because malicious material can be incorporated into search results or error messages. Non-persistent or Type I attacks are another name for these assaults. Because the malicious data is not kept on the server indefinitely, they are classified as non-persistent.

2. Stored Cross-Site Scripting (SXSS): SXSS attacks take place when malicious data is entered into forms and then stored in files or databases used by server applications without being cleaned up. As a result, when users access data from these applications, server answers containing malicious data are sent to them; once browsers render this data, it becomes hazardous. Internet applications and forums are popular targets for SXSS assaults because they permit the posting, storing in databases, and subsequent infection of all users who visit sites with malicious content. Another name for SXSS attacks is ongoing or Type II assaults. The harmful material might still do harm because it is kept on the server and is not removed or screened. This makes it persistent.

3. DOM XSS (DXSS): Attacks using DXSS provide a render-time risk. DXSS attacks, as opposed to RXSS and SXSS, It modifies the DOM trees that browsers build throughout the rendering process dynamically by using harmful information. Attackers can set specific values for DOM objects in internet apps that employ data from user-managed sources by using DXSS assaults. What distinguishes DXSS attacks is the potential for harmful material to be inserted into "Uniform Resource Locators" as values for DOM objects or HyperText Markup Language elements without ever making it to the server. The attacks referred to as Type 0 were initially recorded by Amit Klein in 2005.

4. JavaScript Mimicry XSS (JSM-XSS): To carry out JSM-XSS attacks, attackers use scripts that are already present in online applications rather than inserting malicious scripts. It has been shown and documented that when developers insert valid scripts into unexpected places, it might cause unwanted behaviors in internet-based program. JSM-XSS attacks are difficult to identify and can quickly get past filtering based on allowlisting.

5. XSS worms (WXSS): WXSS attacks are XSS assaults that have the capacity to replicate themselves. WXSS attacks are defined as RXSS and SXSS attacks with self-replication capabilities. This distinction is made because to the fact that certain weaknesses in other kinds of web applications are required for attacks to spread and turn into worms. Since WXSS threats spread among users of online applications and eventually infect more users before being discovered and stopped, they are more harmful. Attackers typically target online social network applications, which are networks of interconnected user profiles. To start a WXSS attack, online application flaws are employed to infect the initial user and upload an unwanted script. In order to elevate privileges and carry out operations on behalf of the user, the injected script takes advantage of other vulnerabilities in the program. This allows the script to replicate itself and infect more users that are linked to it.

Third-party based XSS Attacks

These types of threats take use of flaws in third-party components, such as browsers, browser extensions, and external libraries or programs that are used by web apps, rather than directly attacking flaws in web applications. This category includes two attacks that have been reported in the literature:

1. CR-XSS, or Code-Reuse via Script Gadget: A CR-XSS assault usually has to be executed by someone who is familiar with the scripts, or "script gadgets," that are incorporated into the Repositories and/or frameworks employed by the intended applications. In order for a CR-XSS assault to be successful, HTML codes that align with DOM components and trigger the execution of script gadgets must be injected with payloads that aren't executable, or covert payloads. When one or more devices are used, seemingly harmless payloads can become executable scripts. Lekies et al. introduced CR-XSS in 2013. Moreover, user-land code can produce script gadgets; in this scenario, CR-XSS assaults change to assaults based on online applications.

2. Universal XSS (UXSS): UXSS results from a browser's or one of its extensions' insufficient sanitization of URLs. An attacker uses a browser's or an extension's vulnerability to launch a UXSS attack. Therefore, by encouraging users to click on a link that launches installed extensions in their browser, the extension initiates the use of harmful scripts concealed in the connections. Malicious

scripts are referred to as universal since they are not explicitly associated with any particular online application and can be run in the context of any website.

Collaboration based XSS Attacks

This class of attacks depends on the existence of online applications as well as vulnerabilities in third-party components. Such attacks fail if none of these vulnerabilities exist. This classification applies to three attacks that have been reported in the literature:

1. Mutation XSS (MXSS): it attacks mostly take advantage of browsers' capacity to use the innerHTML property, which is used in web applications, to change prepared HTML strings into legitimate DOM elements. Attackers infiltrate target online applications by supplying their malicious material as structured HTML strings that are connected as values to the innerHTML attribute. When interpreted by browsers, this malicious data can pass through application filters and produce legitimate HTML text. They are run as part of rendered pages after being placed as new DOM elements. MXSS usually targets webmail programs that have the ability to transmit HTML content messages. As the message content is rendered by the recipient browsers, transmitted messages are changed, or mutated, into legitimate scripts. Heiderich et al. published the first report on MXSS assaults in 2017.

2. Cross-API Scripting (XAS): XAS is a type of XSS attack directed at websites that provide Restful APIs to outside developers, including social media platforms. Attackers make users of third-party programs susceptible to XAS by inserting malicious scripts into their own application profiles. Through APIs, malicious data is sent to victims' browsers in order to retrieve data from online applications. The inadequate data deletion by (1) the web application itself, which permits attackers to insert harmful data, and (2) external online applications, which accept malicious API answers without appropriate sanitization, is the primary cause of XAS assaults. As a result, harmful material is executed in the victims' browsers. Zhang et al. first used the term XAS in 2013 and then again in 2015.

3. Content-Sniffing XSS (CS-XSS): Content-sniffing XSS, often known as CS-XSS, is the term for XSS attacks that result from browsers misinterpreting certain file formats. Malicious material is inserted

into distinct media files (such as PDFs and photos) and posted to susceptible online applications in a cross-site scripting-XSS attack. Any person who loads these files in a vulnerable browser is at risk. Web applications need to permit the uploading of malicious files, and browsers need to use content-sniffing techniques to identify the sorts of files in order for a CS-XSS attack to be successful. If script files are present, browsers will incorrectly see them as HTML files, which will cause the files to be rendered and injected scripts to be executed in the victims' browsers.

The distribution of research analyzed across different XSS attack methods is shown in Figure 6. Application-based XSS assaults are the most studied of these, with 98 articles addressing them separately or in conjunction using other forms of threat. Remarkably, RXSS is the attack that is mentioned the most, followed by SXSS and DXSS. Given its frequency and relative ease of detection in comparison to other attack channels, the focus on RXSS is not surprising. Remarkably, compared to research from 2015, there is a discernible rise in interest in DXSS (Hydara et al. In contrast, less attention is paid to other attack variations. For example, because XAS and UXSS target particular apps or platforms, they are each addressed only once. UXSS attacks target particular browsers and browser plugins, whereas XAS attacks target online social network programs that use Restful APIs. It was discovered throughout the data extraction process that papers mentioning assaults that were not specifically defined were actually about simple attacks, like RXSS and/or SXSS. To be clear, these were not just categorized as basic attacks; they were also distinct.

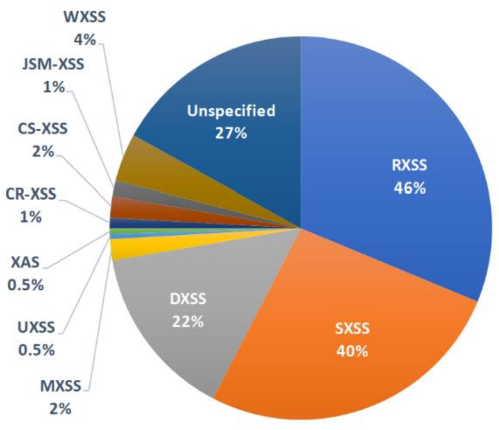


Figure 4: Share of Every XSS Attack

3.7 Cyber forensics and incident response

The theoretical foundation for the second portion of the previously described research is covered in this section. This part is mainly concerned with Cyber Defense or with the role of a Security Analyst sitting in a Security Operations Center. The Security Operations Center comprises of several different responsibilities mentioned below:

- cyber threat analysis;
- network protection and traffic monitoring;
- endpoint protection monitoring;
- security Information and Event Management (SIEM);
- cyber forensics and incident response;
- Phishing.

To carry out these works in the role of a Security Analyst, there are several different frameworks being used in practice. These frameworks help in the proper investigation for a certain kind of threat and make the process of Incident response more systematic. These frameworks include:

- Cyber Kill Chain
- Unified Kill Chain
- A framework for assessing microeconomic competitiveness (a model also named “diamond”);
- Famous framework also known as “MITRE”.

The primary aspect of the Security Operations Center that is the subject of this thesis work is online investigation and crisis management. While crisis managers use information from investigation to determine the behavior of interest from a security perspective, finding investigation objects or evidence of human interaction on digital devices is a specialty of DFIR staff (Satti & Rabail, n.d.).

Need for Digital Forensics

The following points serve as an overview of the main drivers behind and requirements for DFIR:

- Sifting false alarms from real situations and looking for network activity evidence of attackers;
- Securing the attacker's removal with such vigor that their network foothold is destroyed;
- Determining the duration and scope of a breach. This facilitates communication with pertinent parties;
- Identifying the gaps that allowed the breach to occur. What has to be altered going forward to prevent this breach?
- Recognizing the tactics of the attacker in order to prevent future attempts at penetration;
- Notifying the public of the attacker's details.

DFIR Terms

Some basic concepts related to DFIR include: system activity indicators, evidence safeguarding, digital evidence tracking, prioritizing preservation and cyber-attack chronicles. These terms are explained below:

Artifacts

Pieces of evidence that indicate a system activity are called artifacts. Artifacts are gathered during DFIR investigations to bolster theories or assertions on the actions of the attacker. For instance, we can utilize the aforementioned registry key to bolster our allegation that the attacker used Windows registry keys to keep persistence on a system. The registry key that was mentioned in this instance will be regarded as an artifact. Thus, gathering artifacts is a crucial step in the DFIR procedure. The file system, memory, and network activity of the Endpoint or Server might all provide artifacts.

The two most common operating systems seen in enterprise environments are Windows and Linux. You can visit the Windows Forensics 1, Windows Forensics 2, or Linux Forensics rooms to find out more about the forensic artifacts in these operating systems. Windows-based systems are mostly utilized for servers and endpoints, such as MS Exchange email servers and Active Directory Domain Controllers. Linux systems are mostly used by businesses as servers that host various services, like web servers and database servers.

Evidence safeguarding

The reliability of the data we are gathering ought to be maintained during the DFIR procedure (data preservation process). As a result, the industry has established a few best practices. It is important to remember that every forensic examination taints the evidence. As a result, the evidence is initially gathered and secured with a write-protect. After that, an analysis is conducted using a copy of the write-protected evidence. This procedure guarantees the safety and non-contamination of our original evidence during analysis. We could always go back and create a fresh copy using the evidence we had saved in case our copy under investigation gets tampered with.

Chain of Custody

Legal chronological documentation, or the process of following data from the moment of collection until now, is a crucial component of preserving evidence integrity. It is imperative to ensure that the evidence is stored in a secure location once it has been gathered. The evidence must not be in the possession of someone unrelated to the investigation, or it will taint the evidence's chain of custody. A tainted chain of custody casts doubt on the accuracy of the information and undermines the case by introducing unsolvable unknowns. As an illustration, let's say that during the transfer of a hard drive picture from the person taking it to the person doing the analysis, it ends up in the hands of someone who is not authorized to handle such evidence. In such instance, it is impossible for us to know if he handled the evidence appropriately and, as a result, avoided tainting it with his actions.

Order of Volatility

Electronic proof is frequently ephemeral, meaning that it may vanish irreversibly if not recorded promptly. For instance, as RAM retains data only while the computer is powered on, data in a computer system's memory (RAM) will be lost when the machine is turned off. Compared to other sources, some are more erratic. One type of persistent storage is a hard drive, which keeps its data intact even in the event of a power outage. A hard drive is therefore less volatile than RAM. Understanding the relative volatility of the various evidence sources is essential for executing DFIR so that the appropriate evidence can be preserved and collected. In the aforementioned example, we must prioritize protecting the RAM over protecting the hard disk because failing to do so could result in the loss of data in the RAM.

Timeline Creation

To make the most of the information contained in the artifacts, we must present them in an understandable manner after we have gathered them and preserved their integrity. For an analysis to be precise and efficient, a timeline of events must be made. Everything that has happened is arranged chronologically in this timeline of events. Timeline construction is the name of this task. Making a timeline gives the inquiry perspective and aids in gathering data from multiple sources to tell the tale of what transpired.

Let's now examine the static website that is attached to practice creating timelines and respond to the first query. Click the View Site button located in the task's upper-right corner to accomplish that.

The most obvious course of action following an XSS assault is to investigate and identify the incident's primary source before implementing the necessary controls for defensive measures. In the current digital era, investigators need to use standardized and well defined forensic processes in order to apprehend those responsible for cybercrimes.

The tools and techniques used to find online data are known as online investigation. As per Nikkel's (2020) definition, digital forensics involves the utilization of scientifically-derived and tested techniques to preserve, collect, validate, identify, analyze, interpret, document, and present digital evidence obtained from digital sources. The primary goal of this process is to aid in the reenactment of instances that have been found to be criminal or to anticipate illicit behaviors that have been proven to be detrimental to established operations.

Regretfully, forensic methodology is neither standardized nor uniform; rather, it is a collection of techniques and instruments developed from the insights of system managers, criminals, and law enforcement. Palmer (2002) proposed that the area of online investigation has developed through the use of spontaneous techniques and instruments, compared to its inception in the field of science, where many other conventional sciences have their roots. Over the past few years, a number of researchers and security professionals have sought to develop basic principles; nevertheless, their writing has been centered on technical issues rather than taking a broader process into account. As a result, a new framework that works well for both general investigations and targeted attacks like Cross-site Scripting has been developed in this thesis following a comparison of a few existing forensic frameworks.

3.8 Digital Forensic Frameworks in Practice

The most important defensive strategy against cybercriminals is digital forensics, which is vital in supplying critical evidence against them. To help identify suspects, data from the victim's computer is initially gathered in the digital forensics procedure. The gadgets used by the accused are then subjected to a forensic examination in order to collect further proof.

Our goal in writing this thesis is to provide an online investigation environment that can be employed to investigate XSS assaults on extensions for online storage applications. As a result, we will examine the current Digital Forensic frameworks in use in this section of the literature study. Analyzing, identifying, and stopping XSS assaults can be done in a number of ways. A few techniques also describe how to obtain digital proof after XSS assaults.

The section that follows presents the investigation and evaluation of relevant work.

Kruse and Heiser Approach

The Forensic paradigm put out by Kruse and Heiser (2001) serves as the foundation for the first Digital Forensic technique we will talk about. This method includes three stages of computer investigation, according to Kruse and Heiser (2001): gathering evidence, verifying the evidence, and analyzing the data. Here, the three A's of computer forensics are covered: The first is gathering evidence while preserving the original data and not changing or harming it. Verifying that the data you have captured matches the original data that was taken into custody is the second step. Analyzing the data without making any changes to the retrieved data is the final step.

US DOJ

The National Institute of Justice's David W. Hagy presented a lecture on the guidebook for incident participants, Digital crime scene examination (Nij, 2012). This guide offers an additional approach to conducting investigations into electronic crimes: Collection, Examination, Analysis, and Report Writing. The initial step of the investigation involves gathering evidence using computers, components, and other auxiliary equipment that is present at the crime scene. Hard drives, network storage devices, SATA, memory cards and thumb drives that are removable, portable electronics, mp3 players, wireless access points, and many more types of devices may be among them. The gathering of evidence would be examined in the second step using the instruments at hand, and the analysis would come in the third. The final stage would be to write a report detailing the entire online investigation crime investigation process and document the entire process.

Abstract Online investigation Model (ADFM)

The ADFM refers to an independent of specific technologies or instances of cybercrime. The fundamental components of this approach include the previously discussed DF protocols, concepts from classical forensics, and procedures for a real scene of the crime search by the Federal Bureau of Investigation (FBI). The following lists the key actions in this model:

- the detection phase is the initial stage: it entails identifying an incident based on its different clues and determining its nature. Although not strictly speaking a part of forensics, this is important since it influences later processes. Preparation is the process of assembling required equipment, warrants, authorizations, and management support;
- developing a dynamic approach: creating a dynamic strategy depending on the particular technology and possible implications on participants is the third phase. Gathering unaltered evidence while causing the victim as little harm as possible would be the primary objective here;
- securing and isolating physical and online data: isolating, protecting, and The objectives of the conservation phase are to preserve the state of both tangible and digital data. This also entails prohibiting the tampering of digital evidence and allowing the use of other EM equipment in the vicinity of the crime scene;

- collection is the process of using established processes to report the actual actual scene and replicate digital data;
- examining the situation in depth in order to find information is the sixth phase. Finding and recognizing potential evidence is the main goal, especially if it isn't in the usual places.
- the analysis phase of the digital criminal investigation process is the most important one that follows. This stage involves assessing the importance of the information gathered, reconstructing data pieces, and drawing conclusions from the evidence that has been discovered. To support the criminal idea, this phase may require multiple iterations;
- the presentation of evidence, which consists of a summary and an explanation of the conclusions, is the following stage. That is written in an easy-to-read style for the reader;
- returning the evidence, or giving the owner's digital and tangible property back, is the final step.

SDFIM

A systematic approach to online investigation was presented by Agarwal et al. (2011) in the Journal of information technology and Security. There are eleven phases in this concept. The first stage is preparation, which entails developing a fundamental understanding of what constitutes criminal action and crimes as well as gathering supplies for packaging evidence sources, among other things. Securing the Scene, the second phase, focuses on safeguarding the scene of the crime from unwanted access and preventing contamination of the evidence. The third stage, known as survey and recognition, entails an initial survey carried out by detectives to evaluate the situation, identify potential sources of data, and design an effective search plan. The fourth stage, called Documenting the Scene, entails mapping crime scenes, taking pictures, and making sketches. Maintaining an appropriate chain of custody throughout the whole inquiry process necessitates documentation. The fifth stage, known as communication shielding, calls for blocking every communication channel that the devices may have. Volatile and non-volatile evidence are gathered at the sixth step, which is called evidence collecting. Non-volatile data is found on memory sticks, SD cards, and other devices, whereas volatile data is found in ROM. Preserving involves packing, shipping, and storing items throughout the seventh stage. Phase eight, "Examination," is forensic experts going over the material they are still gathering information that is necessary to build their case. The technical assessment that the investigation team does in phase nine, Analysis, is based on the conclusions of the data examination. The results will be sent to a group of people that includes law enforcement officials, technology specialists, legal experts, and company management at the tenth phase, which is a presentation. The review procedure, which is the eleventh and last phase, identifies areas for improvement throughout all investigational phases. This can also refer to any improvements made in the collection, review, and analysis of data for upcoming outcomes, etc.

IDIP

An Integrated Digital Investigation Process with 17 phases altogether, organized into 5 groups, has been proposed by Brian Carrier. These are the five phases in order:

1. Preparation Phase ;
2. Implementation Phase ;
3. Actual Crime Scene Investigation Stages ;
4. Online Crime Scene examination Stages ;
5. Assessment Phases.

Infrastructure Preparedness and Operation Readiness make up the first step. Readiness for Operation. Infrastructure readiness makes sure that the necessary data is available for a thorough inquiry to take place, while operation readiness supplies the staff engaged with the incident and its investigation with equipment and training.

Offering a means of event detection and verification is the aim of the deployment phase. The first stage of this process is called " Identification and Alerting," during which an event is found and the relevant parties are informed. An SMS alert or a 911 call are a couple of examples. Authorization and Confirmation come in second. Getting permission to thoroughly examine the event and the scene of the crime is the main objective here.

During the Phase of the actual investigation of the crime scene, the scene is preserved, physical evidence is surveyed, document evidence is examined, physical evidence is searched for, Actual Crime Scene reconstruction is carried out, and finally the entire theory is presented.

The electronic crime scene investigation phase includes the following steps: the online scene is saved, online data is analyzed, document evidence and scene are searched for, online crime scene reconstruction is offered, and real crime scene reconstruction is the last one.

The last stage, known as the review phase, entails going over the study to find areas that could want improvement. This has to do with how effectively the digital and physical investigations were conducted and whether there was sufficient digital and physical evidence to solve the case. If all went according to plan, this step would result in new protocols, training, or nothing at all.

DSCFIPM

The Domain Specific Cyber Online Investigation Process Model (DSCFIPM) was proposed by Satti and Jafari (2015). In the event that a cybercrime scene report is made, The initial step in developing a policy and procedural flow for the online criminal investigation has been recommended to be DSCFIPM. There are ten phases in this concept. The creation of SOPs for cyber forensics investigations within the university realm is the initial stage. Phase two involves strategic planning and defining the scope, while Phase three involves sealing the crime scene. The fourth step involves the gathering of evidence by the investigating authorities; phase five involves preserving evidence; and phase six involves extracting relevant data; evidence is analyzed in phase seven; analysis and findings are presented in phase eight; case history records are archived in phase nine; and post-case review is the final phase.

AIDFF

Rughani (2017) put out an artificial intelligence system that is divided into three stages: Intelligent Gathering, Intelligent Evaluation, and Intelligent Display. Thus, in essence, the integration of AI in the Digital Forensics process transforms the original framework that Kruse and Heiser proposed into Smart. In this instance, instances that are comparable to the one at hand will be used to train the Smart Acquisition tool. In a similar vein, the Smart Analysis tools will also be trained on cases that are comparable to those that have already been resolved, and in the last stage of the AI-based online investigation environment, the same is true for the Smart Reporting tools (Rughani 2017).

Presented above were some of the Investigation models for Digital Crimes that are currently being used in practice. Now that we have the requisite background knowledge at our disposal, the next step would be to jump into the comparative analysis of each of these models, find out their pros and cons and suggest an improved version of the Investigation model for the Digital Crimes. The following chapters discuss the implementation part of these models.

4 Implementation of Research Part-1

This chapter covers the initial stage of research implementation, which focuses on identifying XSS susceptibility in extensions for online storage applications. The third stage of the systematic review, which is the research phase, is represented by this chapter.

4.1 XSS susceptibility in extensions for Cloud Applications

This study focuses on benign extensions, which are created by developers that may not have a strong background in security but have good intentions and do not include harmful intent. These extensions may still be vulnerable to outside threats like XSS even though they are innocuous.

Two types of XSS assaults that target insecure extensions have been classified:

1. Attacks having Shared Workspace
2. Attacks having Outside Input

Attacks having collaborative workspace

When it comes to using the same online storage-based application, the attacker and the target are acquaintances, coworkers, or even distant lovers. A shared workspace, such as a Google Docs page, is any area within the host application where modifications made by one user are visible to other users. This is where the victim and the attacker collaborate. The attacker has inserted malicious JavaScript code to this shared workspace. Text that is intended to be visible to the user can be hidden using common methods like utilizing a modest font size or matching the text color to the backdrop. If the attacker employs a susceptible extension that the victim activates within the shared workspace and the extension misinterprets the attacker's input, the injected script may integrate into the web page within the iframe of the extension. The victim's web browser then runs it after that. Thus, the target is effectively subjected to an XSS assault by the intruder (S and Manico, n.d.).

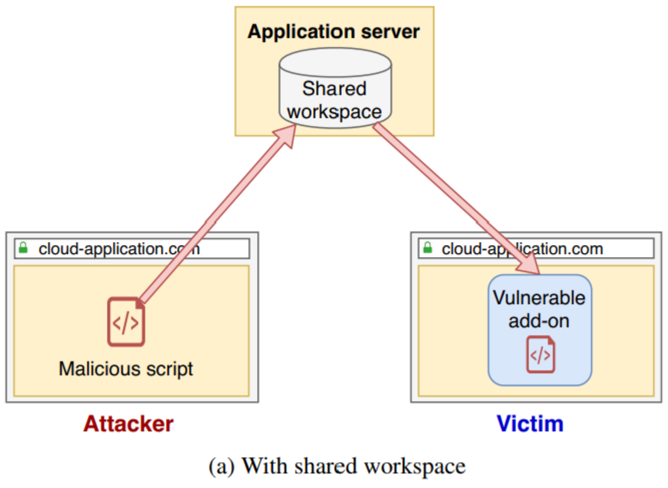
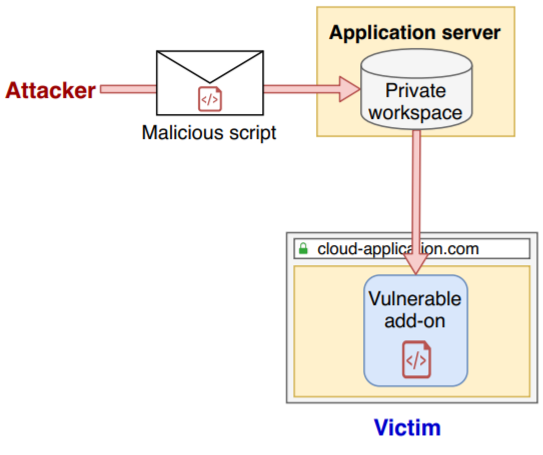


Figure 5: XSS Attacks having Shared Workspace

External input attacks

Certain host applications allow outside input, such as messages from non-platform users. For instance, the attacker can send the victim an email with the malicious script contained in it while they are using Gmail or Outlook. If the victim has allowed a susceptible extension to handle emails, the injected script may reach the extension's iframe and run there, just like any other JavaScript within that frame (S and Manico, n.d.).



(b) With outside input

Figure 6: XSS Attacks with Outside Input

The specific methods by which the attacker inserts the script into the shared workspace depend on the online storage application, the extension, and the vulnerability under attack. However, the underlying source of these attacks is the vulnerable extension's untrustworthy processing of user input, which directs it to the JavaScript data flow sinks in the user interface without the required sanitization. In essence, the attacker's code can be executed if the extension interprets malicious input from the attacker as HTML rather than plain text by using HTML element sinks like document.write or document.body.innerHTML. Similarly, an attack may be successful if the attacker's content is handled by JavaScript functions like eval and setTimeout that convert text input into executable code. However, the latter type of errors are less common since developers are better aware of the risks associated with these sorts of routines (S and Manico, n.d.).

Compared to the classic forms of XSS discussed in Section 2.1, the attacks discussed here are similar to stored XSS in that the malicious input of the attacker is stored in the host application's data repository. Furthermore, such assaults can be categorized as client-side or server-side XSS depending on whether the harmful information is handled by the extension server or the extension user interface.

Overall Impact

By exploiting the vulnerability in the extension, the attacker gains the capability to execute arbitrary scripts, enabling various types of attacks:

- ❖ Using the extension server's APIs to access data stored on the server; this is especially true in microservice architectures where the extension server likely maintains its own data storage. Possibly accessing data within the host application through cross-domain messaging, either directly or indirectly via the extension server. This is only possible if the host application and its extension APIs are designed properly.
- ❖ Conveying a fraudulent user interface within the iframe extension to deceive the user into disclosing sensitive information or credentials.
- ❖ Traditional XSS attacks are similar to using HTML5 APIs to request authorization to use external resources that belong to the victim or access to local resources like geolocation.

4.2 XSS Case Studies

This section will examine the architectures of several host applications and assess the potential benefits for the attacker in each case. It's crucial to keep in mind that the malicious script operates within an iframe with a different origin than the program hosting it. As a result, it is unable to access any related cookies or the host application's DOM structure within the browser. Any attempt to access host-application data must instead be made using exposed APIs on the extension server or cross-origin access techniques.

XSS attacks for Microsoft 365 online extensions

Microsoft Office Online is an online storage-based office suite that comes with Word, Excel, PowerPoint, and Outlook among other well-known office applications. The architecture of each application in the suite is the same, as Figure 2 illustrates. The extension interface seamlessly integrates into the user interface of the host application, making it easy for users to engage with.

To protect the extension interface from having direct access to user data within the host application, it is contained inside an iframe and has a different origin from the overarching program.

JavaScript APIs are used by the extension for client-side communication with the host application. MS Office Online apps in particular use windows.postMessage() for cross-origin communication between the parent application window and the extension's iframe. Table 1 describes the different levels of access that the extension can request to the host-application data. Only the document the user is currently working on can be viewed by the Word, Excel, PowerPoint, or OneNote extension. Conversely, Outlook extensions possess broader rights, allowing them to request access to the user's entire mailbox in addition to the item they are currently viewing (such an email or form) (o365devx, 2022). Additionally, Outlook extensions have the ability to contact a particular API called getCallbackTokenAsync(), which returns an access token together with the authorization level of the extension. By obtaining this token from the extension UI running in the browser, the extension server can connect to the email server on behalf of the extension for a predetermined period of time.

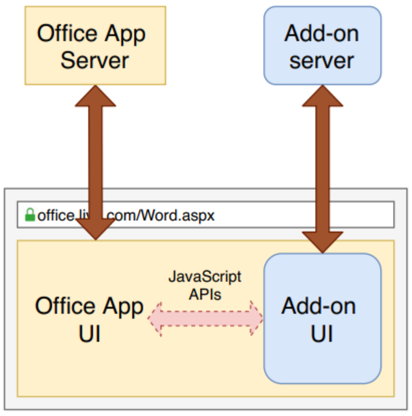


Figure 7: Microsoft 365 Add-on Architecture

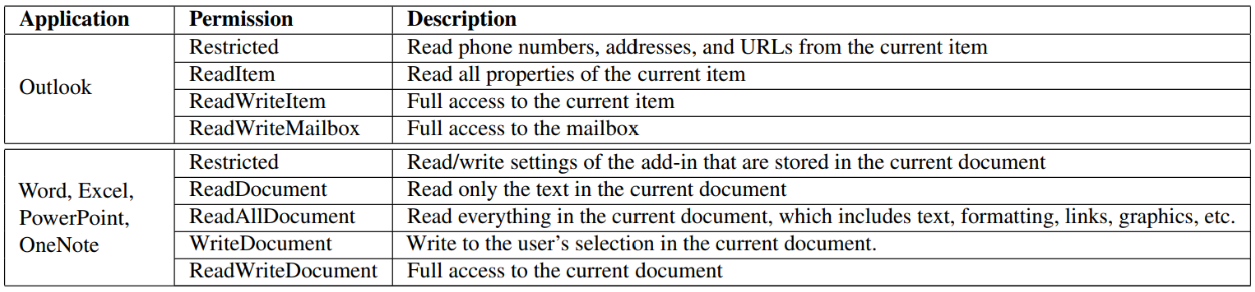
XSS Exploits

Let's examine how an intruder could leverage XSS to exploit a vulnerability in an MS Office Online extension. The extension must first be installed by the victim. Afterwards, the attacker can exploit the vulnerability by inserting malicious scripts into a document that is shared with the victim or, depending on the extension, sending the victim an email with the harmful scripts in it if the victim uses Outlook. In Section 3, these two attack techniques are explained. Below, we look into the extent of the attacker's access to the user's data within the host application.

Possess the extension's access rights. The attacker can use local communications with the host program window to access any resources authorized for the extension. The host application cannot discern between requests from the attacker that are malicious and those that are legal when the attacker's scripts are executing inside the iframe of the extension.

If Word, Excel, PowerPoint, or OneNote is the host program, the attacker's access is restricted to the active document. The attacker may or may not profit from this, given they were the ones who initially published the document, but they can utilize the injected script as a backdoor to maintain access even in the event that the victim revokes their permission to access. People may, for instance, print off a personal copy of a form or template before entering sensitive data. However, if the vulnerable extension in Outlook has the ReadWriteMailbox permission, the attacker can access the victim's mailbox in its entirety. As a result, they could be able to view all of the victim's emails and send emails on their behalf (o365devx, 2024).

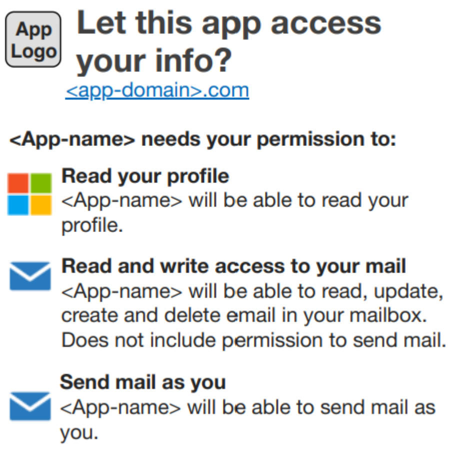
To obtain an OAuth 2.0 token, the attacker can mimic parts of the application's user interface using the iframe control. This kind of manipulation opens the door for other types of dishonest tactics, such phishing for sensitive information. However, we did find one specific strategy that can be utilized to target users of MS Office Online. Many extensions serve as bridges between MS Office Online applications and external web services that are hosted on the Azure network. In essence, these extensions give outside parties access to the backend server user interface. Unlike ordinary extensions, these servers interface with the host application through more potent APIs. To utilize these functionalities, the service provider needs to register an Azure application with the Microsoft identity platform and obtain user authorization for resource access. This authorization process is simplified using OAuth 2.0. The extension displays a pop-up window with the name, logo, domain, and permission requirements of the application (see Figure 3). If the user agrees to authorize it, the software receives an access token that enables it to access the required resources from anywhere.



Application	Permission	Description
Outlook	Restricted	Read phone numbers, addresses, and URLs from the current item
	ReadItem	Read all properties of the current item
	ReadWriteItem	Full access to the current item
	ReadWriteMailbox	Full access to the mailbox
Word, Excel, PowerPoint, OneNote	Restricted	Read/write settings of the add-in that are stored in the current document
	ReadDocument	Read only the text in the current document
	ReadAllDocument	Read everything in the current document, which includes text, formatting, links, graphics, etc.
	WriteDocument	Write to the user's selection in the current document.
	ReadWriteDocument	Full access to the current document

Figure 8: MS Office Application Permissions

As such users of the Microsoft Office Online extension are used to seeing the OAuth authorization screen, attackers may be able to deceive users into providing their login credentials by taking advantage of this familiarity. Initially, the attacker makes an Azure application with the same name as the weak extension. This is made feasible by Azure apps that don't need to have unique names. Then, using the injected script, the attacker asks for permission to use specific user resources. The attacker can access the victim's information from anywhere on the host application if the victim approves of their application. This token works similarly to the previously stated Outlook token and can be used with any MS Office Online application. As such this, it becomes challenging for the victim user to decide whether to grant an OAuth 2.0 token or grant a particular extension access to the extension APIs.


App Logo Let this app access your info?
<app-domain>.com

<App-name> needs your permission to:

Read your profile
<App-name> will be able to read your profile.

Read and write access to your mail
<App-name> will be able to read, update, create and delete email in your mailbox. Does not include permission to send mail.

Send mail as you
<App-name> will be able to send mail as you.

Figure 9: OAuth Authorization

Translator for Outlook Case Study

Microsoft's "Translator for Outlook" extension, which lets users translate emails into the language of their choice and is specifically made for the email service Outlook, will be used to demonstrate the vulnerabilities. This extension often operates in the following ways:

1. the extension is launched by the user and appears as a side panel in the interface of the host application ;
2. The language into which to translate the email is chosen by the user ;
3. the extension displays the translated content after translating the entire email into the selected language.

Unfortunately, there is a fault in the extension that causes translated content to be printed as HTML without the proper text escape. Therefore, if the victim receives an email with malicious scripts from an attacker—possibly anyone on the internet—and uses the extension to try to translate it, the scripts will execute. It's interesting that the extension's only permission is ReadItem. Consequently, if the attacker uses the host application only to leverage local communications, they are unable to read the victim's mailbox or send emails on their behalf. As was indicated in the previous section, the attacker can gain these kinds of access capabilities by requesting an OAuth 2.0 token through their malicious scripts.

Google Workspace extensions

The G Suite is a suite of apps developed by Google that includes Gmail, Sheets, and Docs. Prior to investigating the capabilities of G Suite extensions, it is imperative to comprehend Google APIs. These APIs allow for programmable access to several Google applications, including Maps and Drive. Before a client (website, for example) may use these APIs to access private user data, it must be linked to a Google Online storage Platform (GCP) project ("Google Online storage projects | Apps Script"). After that, the client needs to use OAuth to get an access token. The user is redirected to the Google Authorization website where they must first log in using their Google account. After that, the client-requested rights and the name of the GCP project are displayed in an authorization window on the website. If the user accepts the permissions, the Google Authorization server gives the access token to the client; if not, an error is raised.

The architecture of G Suite extensions is shown in Figure 4. The main way that G Suite extensions differ from those for MS Office Online is that they are fully hosted on Google online storage. The extension server allows for direct user data interaction by acting as a client for Google APIs. The extension server specifies the interfaces to which the extension UI makes requests, and the extension server responds by carrying out the desired operations on the user data. The only extension code that can access these server interfaces comes from the same server. To illustrate, Google's Translate extension for Google Docs consists of a server with two main interfaces: one for translating text that the user selects and returning the result, and another for substituting the translated text for the current selection ("Google Docs Editor Extension quickstart | Apps Script").

The extension server needs permission to access the user's personal information when it functions as a client for Google APIs. This authorization usually happens the first time the user launches the extension, as shown in Figure 5 where a typical prompt for authorization appears. If the user authorizes, the extension server receives an access token with the necessary permissions.

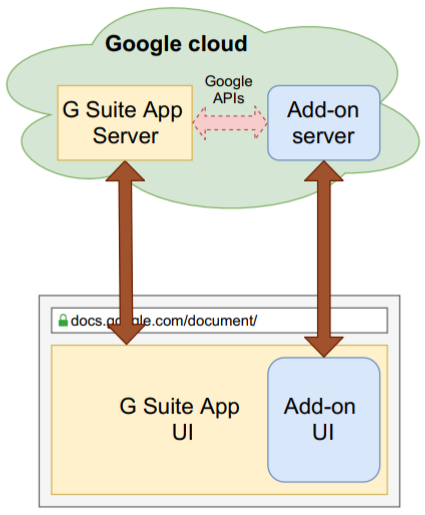
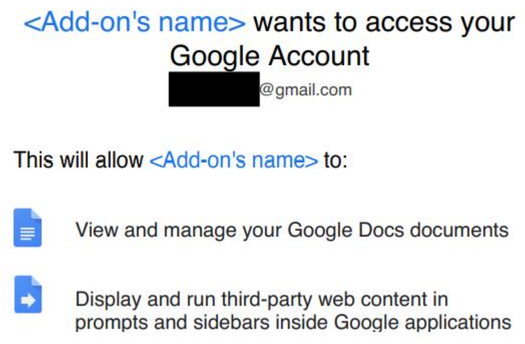


Figure 10: Google Cloud Add-on Architecture

G Suite extensions can request permissions to access user data across all G Suite applications. A Google Docs extension requesting to send emails from the user's Gmail account is an illustration of this. The cross-application permissions paradigm makes extensions more flexible and useful, but it also makes it more likely that malicious extensions will take advantage of it. Many privileges, including "Read, compose, send, and permanently delete all of your email from Gmail" and "See, edit, create, and delete all of your Google Drive files," may be requested by these dishonest companies, giving them access to a vast amount of customer data. To mitigate these risks and guarantee compliance with the Google API User Data Policy, Google personally reviews extensions that ask for sensitive permissions.



<Add-on's name> wants to access your Google Account

[redacted]@gmail.com

This will allow <Add-on's name> to:

View and manage your Google Docs documents

Display and run third-party web content in prompts and sidebars inside Google applications

Figure 11: Google Account Add-on

XSS Exploits

We then investigate the various ways in which a proficient XSS hacker could obtain user information.

Reach the same degree of accessibility as the add-on. It can first appear that the XSS attacker is unable to access the victim's data since the host application window does not accept local messages from the extension UI. However, the attacker can circumvent this restriction by using a frequently used Google API that extension UIs use: Google provides the Picker API, which allows users to select files or folders stored on Google servers. Like other Google APIs, the Picker API needs an access token in order to operate. In reality, it is advised that extension servers provide an interface via which the browser's extension code can retrieve a copy of the server's token. Most notably, the server token is not exclusive to the Picker API. Therefore, the injected XSS code may also request the access token in order to get the same permissions to the user's data as the extension server.

Get the OAuth 2.0 token request procedure started. If the vulnerable extension does not use the Picker API, the hacker can change the script that is injected into a JavaScript client for Google APIs and ask the user for an OAuth 2.0 token. When this happens, the victim is presented with an authorization prompt, just like when an extension requests rights when it is first executed (see Figure 5). While the attacker's malicious client requires them to use their own GCP project, they are free to use a project name that sounds similar to the extension. This method is similar to the phishing exploit described in Section 4.1.1 for Office Online extensions. Since G Suite customers are accustomed to the authorization question, we predict a high success rate for this attack. This could persuade the target to believe that the extension requires updated permissions.

Form Ranger Case Study

It is noteworthy that the most popular Google Forms extension is Form Ranger. Form Ranger automates the process of filling out Google Forms, an online tool that facilitates data collecting through quizzes and surveys, employing any spreadsheet that is saved on the user's Google Drive to extract data. The typical sequence of actions with this extension unfolds as follows:

1. the extension launches into the host application's user interface as a side panel ;

2. the extension lets users fill in the response possibilities for each question by displaying a list of questions in the form and importing data from a spreadsheet ;

3. after choosing an inquiry, the extension shows a list of every spreadsheet on the user's Google Drive. Notably, this makes use of the Picker API ;

4. Users are able to choose the sheet and column to load data into after choosing a document from the list. A sneak peek at the data in the column is also displayed ;

5. after selecting a spreadsheet and column, the extension requests that data be entered into the question from the selected column.

Readers may already be aware of Step 4's flaw: rather than filtering or sanitizing the data, the extension renders it as HTML from the selected spreadsheet. Thus, if an attacker manages to access the document (for instance, through a joint venture with the victim), they have the ability to include malicious JavaScript code into the area of the document meant for form inputs. This code executes when the victim uses the extension. Given that the extension makes use of the Picker API (Step 3), the attacker will be able to acquire the extension's access token along with all associated permissions, including "See, edit, create, and delete all of your Google Drive files," "View and manage your forms in Google Drive," and "See, edit, create, and delete your spreadsheets in Google Drive."

It's clear that the extension has more rights than it need. For instance, reading access to spreadsheets should be adequate; it doesn't need to be able to view all Google Drive files or write emails on the user's behalf. The attacker can access all of the victim's Google Drive data and send emails on their behalf thanks to these superfluous rights. As stated in the previous section, the attacker can employ phishing to obtain an OAuth 2.0 token if they wish to achieve more access permissions, such as reading the victim's emails.

Shopify extensions

Shopify is an e-commerce platform that allows small business owners to set up online storefronts. It provides services like payment processing, marketing, and customer engagement. The owner of each business can use the built-in features of the platform to process orders, interact with customers, and add new products through a web admin interface. Third-party services can be integrated with this admin interface through Shopify extensions.

The extensions' architecture is displayed in Figure 6. While operating in the online storage, the extension server interacts with shop data using the Shopify REST APIs over HTTPS. By following these steps, you can obtain an OAuth 2.0 access token, which is typically required for the store server to provide access to the extension server. When the user starts the extension for the first time, the shop's admin interface displays an authorization prompt detailing the data the extension intends to access (such as orders and goods). If the user grants permission, the Shopify server will deliver an access token to a pre-registered endpoint on the extension server. This token is used by the extension server to access the required data, and it is possible to renew access tokens without requiring further user input at a later date ("Staff permissions").

The admin interface of the shop has a menu that incorporates the extension user interface. The extension UI employs a set of local JavaScript APIs provided by Shopify to perform resource-picking operations. Window is used by the extension UI to communicate with the admin interface of the shop.postMessage(). After prompting the user to select the required sort of resource, the admin interface returns the generated data objects to the extension.

An owner and several employees can oversee a Shopify store. The store's owner has complete access to it and can give employees different levels of access, including rights for admin tools and extensions. The view that shows through the extension's user interface depends on how closely the member's permissions and the extension's permissions coincide.

XSS Exploits

An intruder on Shopify has to be a staff member with the capacity to insert malicious scripts into shop resources in order to use the "shared workspace" attack vector. Scripts can be added to product and order descriptions even if Shopify prohibits using HTML or scripts in customer data. These scripts will execute when other employees use a weak extension that puts them in danger.

Install a Malicious Extension

The Shopify extension architecture combines the greatest elements of MS Office Online with the G Suite. Specifically, the access token is sent directly to the extension server, from which the extension user interface cannot access it by default. Furthermore, since shop data is handled in the admin interface, resource-picking procedures can be carried out by the extension UI with the help of the local JavaScript APIs, protecting shop data. Attackers may still trick the proprietor of the store or a legitimate user into installing a malicious extension, though. Unlike Microsoft Office Online and G Suite, which only allow extensions to be installed through official marketplaces, Shopify allows extra extensions to be installed with a simple URL visit. This suggests that an attacker may design a malicious extension that looks like a legitimate one and install it using an injected script. If the victim believes the extension has been updated, they may approve it again. Afterwards, depending on the permissions set, the malicious extension can access any shop data.

Get Printer Pro in order

Order Printer Pro is one of the most popular extensions in the "Orders and Shipping" section of the Shopify marketplace. Invoice App - Order Printer Pro - Shopify Order Printer Pro Using the Invoice

App: send Custom Docs | Shopify App Store, users can quickly create order-related documents, such return forms and invoices, and then print or email them to customers.
The procedure for printing an order is as follows:
1. The extension is launched by the user from the admin interface, where it is visible as a component of the user interface.
2. A list of the shop's orders appears when the user chooses the "Orders" menu in the extension user interface.
3. After the user chooses the order they wish to print, the extension shows the order information along with buttons to print or export the order as a PDF. The customer's shipping address, the contents of their order, and any notes made by staff are all shown.
4. The order is printed when the user hits the "Print" button.

The extension's problem is that when showing order information, it does not output the order notes safely. Consequently, any JavaScript snippets that are present in the notes will be performed. As previously stated, Shopify does not prohibit the inclusion of HTML or JavaScript in this area. Consequently, XSS attacks against other employees, including the store owner, can be carried out by any staff personnel who has write access to the orders of the establishment.

5 Implementation of Research Part-2

Due to the sharp rise in cybercrimes and frauds, legal professionals must adopt appropriate procedures for gathering digital evidence. Using useful frameworks and techniques, forensic examiners obtain evidence when crimes are committed over digital platforms. In order to identify and look at intrusions, digital forensics analyzes network traffic (Al-Mousa, 2021). Numerous models have been developed for cybercrime investigation, each having pros and cons of their own. This study looks at some of the difficulties in digital forensic research as well as the features of current models for digital forensic inquiry.

A few of the current digital forensic frameworks in use were examined in this thesis' literature study portion. Here, in the implementation stage, we weigh the advantages and disadvantages of each of these frameworks and provide an alternative for XSS attack research.

As was previously mentioned in the previous chapter, cross-site scripting (XSS) assaults are client-based and, despite their apparent lack of significance in comparison to other types of attacks, their influence should not be underestimated. An XSS attack gives the attackers the ability to take credentials, start a process that the user can complete, and steal session information.

Because web applications are used by 75% of cyberattacks, it is imperative to detect XSS assaults and undertake forensic investigations. This is supported by research conducted by Acunetix. It is also one of the online application vulnerabilities that is tested the most frequently.

In the capacity of a SOC analyst, the conventional methods of identifying a cross-site scripting vulnerability would be as follows ("What is XSS and how to prevent it? | Web Security Academy"):

- Searching for keywords:

The most common way to detect the XSS attack would be to look for keywords such as “alert” and “script” that are commonly used in XSS payloads.

- Learn about commonly used XSS payloads

The perpetrators tend to utilize the same payloads in order to search for vulnerabilities before exploiting an XSS vulnerability. Therefore, a good idea is to familiarize yourself with the most used XSS payloads.

- Check for the use of special characters

Check the data coming from a user to see if any special characters commonly used in XSS payloads, such as greater than (>) or less than (<), are present.

However, in the context of an ongoing investigation, for detecting this kind of attacks, a formal Digital Forensic Investigation needs to be conducted and for this reason and based on the studies presented in the previous chapter, a novel framework has been presented here. This approach may also be applied for performing other kinds of investigations related to web application attacks. However, in this scenario, it has been tailored for Cross-site Scripting attack.

5.1 XSS Framework for Digital Forensic Analysis

The following is an outline of the key steps included in the suggested XSS Digital Forensic framework:

5.1.1 Phase 1: The planning phase

Prior to the actual investigation, this critical stage is conducted with the goal of comprehending the crime, associated activities, and evidence sources. Legal requirements, organizational constraints, and jurisdictional limits must all be adhered to by investigation techniques. In this phase, the required authorizations and search warrants are also acquired.

5.1.2 Phase 2: Identify the threat actor

Finding the attacker should be the main goal of the investigation following an attack. Email addresses and websites could be the attack's sources. Because web server attacks make use of the TCP/IP protocol suite, it is difficult for attackers to spoof TCP connections and change their originating address. As a result, the attacker's real IP address is usually as indicated by the log files' source address. Investigators can trace the attacker's IP address and system by looking through IIS log files on the victim's computer. In the event that the attack comes from an email, the investigators should look up the attacker's machine's IP address in the email header. Several techniques are available to ascertain the attacker's location once the IP address is known. Locating the attacker's location might be aided by tools like the Internet Protocol (IP) Locator and WhoisIP. This stage helps detectives find the actual perpetrator of the attack.

5.1.3 Phase 3: Protecting the compromised machine

This phase involves securing the victim's and attacker's computers to keep the evidence safe and stop any unwanted access. It is essential to conserve all the information or data.

5.1.4 Phase 4: Determining the threat actor

Finding the attacker and protecting the devices of the perpetrator and the victim before moving on to the next critical stage of the investigation is identifying the attacker. It's possible that the attacker used a machine in a public setting to carry out the crime. In order to guarantee that the assailant may be brought before a judge and jury, crime investigation organizations need to pinpoint the real culprit using a variety of information sources.

5.1.5 Phase 5: Investigation strategy and record-keeping

It is necessary to create a search strategy in order to gather and examine digital evidence. Maintaining accurate data records and paperwork is essential since they may be used as evidence in court.

5.1.6 Phase 6: Protecting the scene of the crime

This is crucial to preventing evidence tampering by cybercriminals. This safety measure enables professionals in cyber forensics to retrieve and examine legitimate digital evidence associated with the offense.

5.1.7 Phase 7: Gathering and safeguarding digital evidence

To guarantee data integrity, rigorous scientific methods must be used in its collection. The objective is to locate and collect evidence that is both volatile and non-volatile while maintaining its preservation. Every item and system connected to the perpetrator or victim laptop should be closely inspected in order to collect data, per the rules for proof.

5.1.8 Phase 8: Retrieve pertinent digital evidence

Extraction of pertinent evidence connected to the crime comes next after digital evidence has been gathered and preserved. It is possible to assess digital evidence using a variety of programs, including AccessData, FTK, and Encase.

5.1.9 Phase 9: Examination

Important information about the crime will be taken from the evidence after it has been carefully examined, and this information can be utilized to strengthen the case in court. The identity of the investigator, the date of the examination, and other pertinent information should all be included in the proper paperwork. To make sure the data was not altered after scrutiny, it is also crucial to authenticate it.

5.1.10 Phase 10: Analysis

Investigators must analyze the evidence to draw conclusions. This phase can identify if additional extraction steps are necessary. The analysis aims to achieve more accurate results and must comply with government and judicial laws to ensure court admissibility. After the evidence analysis, a review should be conducted to identify potential improvements.

5.1.11 Phase 11: Correlating findings and preparing the report

Following the completion of the forensic experts' analysis of the evidence acquired from the attacker and victim machines, the next step is to compare the outcomes from both systems. The data can be shown in court if the results indicate that the attack originated from the same attacker machine. If the results match, a report needs to be prepared in accordance with the relevant rules and laws.

5.1.12 Phase 12: Presentation to the appropriate authority

The relevant authorities must be shown the evidence. The evidence has a better chance of being accepted by the appropriate authorities and supporting the case if it is collected, scrutinized, and evaluated according to legal and scientific procedures and the outcomes line up.

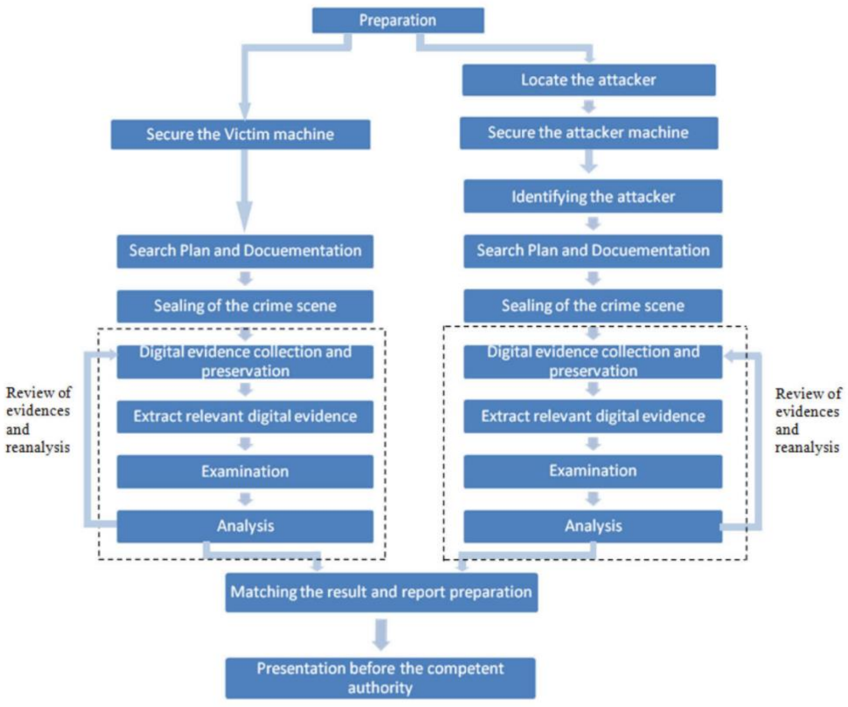

Preparation
Locate the attacker
Secure the Victim machine
Secure the attacker machine
Identifying the attacker
Search Plan and Docuementation
Search Plan and Docuementation
Sealing of the crime scene
Sealing of the crime scene
Digital evidence collection and preservation
Digital evidence collection and preservation
Review of evidences and reanalysis
Extract relevant digital evidence
Extract relevant digital evidence
Review of evidences and reanalysis
Examination
Examination
Analysis
Analysis
Matching the result and report preparation
Presentation before the competent authority

Figure 12: Digital Forensic Model Steps

The process of conducting an investigation, whenever a digital crime is committed, is highlighted in the model that is being provided here. The initial steps involve getting ready, tracking down the assailant, protecting the victim's computer, identifying the assailant, recording the search strategy, securing the crime scene, gathering and preserving digital evidence, extracting digital evidence, analyzing it, and creating a report and presentation for the appropriate authority. The actions listed above can be taken whenever an XSS assault occurs to get to a conclusive result.

The diagram below shows the process of investigation in the form of a flow chart. The investigation would naturally begin with the preparation for the event, then the next step would be locating the attacker through its IP address, logs or other similar resources. After that, the attacker machine would be secured, so that the evidence does not get lost. Then the search plan would be developed. Crime scene would be sealed, to prevent any possible loss of evidence and keep the crime environment intact. After that, the Digital Evidence would be collected, preserved and extracted. Then it would be examined, and its analysis would be performed. Finally, the report would be prepared and presented to the competent authorities.

6 Analysis and Synthesis of Results

The fifth and sixth phases of a systematic review, analysis and synthesis, are covered in this chapter. The application of the research, which involved defining the online storage and online storage applications and then going over the several facets of online storage security—of which the XSS attack in Online storage Application Extensions was the most pertinent—was covered in the preceding chapters. In order to ensure that the investigation of digital crimes follows a methodically defined procedure, the second element of the implementation involved suggesting a forensic framework for the examination of cross-site scripting attacks. We now concentrate on the analysis of the work we have completed in the thesis' implementation section in this chapter.

More specifically, the online storage applications we presented and the case studies we performed related to the Cross-site scripting attack will be analyzed individually and then synthesis of the important findings will be done. The same analysis will be done for the various Digital Forensic frameworks and then the synthesis of results will be done. Structure of chapter . . .

6.1 Empirical Analysis

The analysis methodology used here is Empirical Analysis. For this purpose, a few select extensions from various marketplaces were selected and XSS related testing was performed on them. Now the analysis of this test study is given here.

To determine the frequency of XSS flaws in online storage application extensions, we empirically investigated insecure extensions that were available in the exchanges of three online storage application suites of our choosing. This section explains how we narrowed down the extensions to shortlist them and how we methodically found vulnerabilities in the extensions we chose. Ultimately, we showcase the findings from our examination.

6.2 Extension selection for investigation

For the scope of this study, only the free extensions from the marketplaces were included. The number of these extensions is listed in Table 3. It's crucial to remember that Shopify and Microsoft refer to them as apps and add-ins, respectively. For our analysis, we took three free extensions from each of the three marketplaces, for a total of ten extensions. The following criteria were used to choose these extensions:

Table 2: Vulnerable Add-ons

Marketplace	Available Free	Utilized Add-ons	Exploited Add-on
Microsoft Office 365	> 1100	3	Translator for Outlook
Google Suite	> 1100	3	Form Ranger
Shopify	> 1200	3	Order Printer Pro

Microsoft Office Online

Within the suite, we focused on their several tools extensions. The extensions for the other applications are either not available to domain users or are not present at all. We chose the top 3 extensions based on the quantity of reviews, as the marketplace does not show the total number of users for each extension.

Google Workspace

Individual users can access Google Workspace among other applications. Because Drive and Calendar's extensions just provide additional menus to the program's dashboard and don't include any client-side code, we disregarded them in order to prevent XSS attacks. We choose the three extensions having the most users out of the remaining applications.

Shopify

The "Most installed" option was first used to order the available extensions in the marketplace, and the top 3 extensions were then chosen from the resultant list. It's crucial to remember that the marketplace just provides the rating and the quantity of reviews for each extension—it does not reveal the installation count.

6.3 Analysis Methodology

Black-box testing is used to manually review the selected extensions. Determining if an extension exposes cross-site scripting (XSS) vulnerabilities automatically presents a number of challenges. Present automated techniques identify most XSS vulnerabilities that affect client-side XSS; nevertheless, the type of XSS attacks on online storage-application extensions differs based on the implementation of the host application and extension. For example, vulnerabilities in MS Office Online extensions are always client-side because the attacker's malicious input is submitted via local messaging (i.e., window.postMessage()) and processed by the extension UI. However, vulnerabilities may exist on the client or server side of G Suite and Shopify extensions, depending on whether the extension server processes the data beforehand or returns the attacker's raw input to the client.

Second, the majority of extension functions are only activated by user actions, in contrast to typical web applications where a large percentage of processes can be evaluated simply loading every URL that is available in a browser. Furthermore, there isn't a set way to invoke an extension. While some extensions require additional user inputs before the XSS code may be performed, others can be started by the user clicking and choosing them from a host application's menu.

Third, user login procedures are not a part of the majority of methods for effortless XSS analysis in the literature. To track user data, however, a lot of extensions act as connectors between their host apps and other online services, necessitating account creation and login from users. These registration procedures differ greatly from one another and are difficult to automate without human intervention. We postpone the automatic identification of XSS vulnerabilities in online storage-application extensions until later studies due to these considerations.

6.4 Black box testing

We initially installed and became acquainted with the functionality of each extension. After that, we made the subsequent assessment items for every one of the intended applications:

a- Document editing apps from MS Office Online and G Suite (including Word, Excel, Google Docs, and Google Sheets) were tested using a corresponding document. To ensure that the extension could accept JavaScript code from all data sources, we added brief JavaScript snippets to various portions of the report. For example, we put JavaScript code to the sheet name, some cells in each column, and the heading of each column in a spreadsheet. An example of what is used is the code snippet {\script>console.log("Pwned!!!") \/script>}.

b- An email we sent to ourselves served as the test item for email programs (Outlook, Gmail, etc.). We added scripts to the email content and subject line, just like with document editing software.

c- Initially, we made a test shop for Shopify. Scripts can only be injected into orders and products, as mentioned in Section 4.3.1. As a result, we made a number of test orders and products for the store and added scripts to every place that could be, including the items' names and descriptions.

We looked for workflows that involved rendering the item's content and evaluated every function of each extension on the test item after it was created. In several circumstances, we additionally implemented bespoke JavaScript code because certain extensions only handled data in particular formats. For instance, we had to incorporate our script into the definition of a variable using Doc Variables, which looked like this: ${variable name}. We concluded that the extension was vulnerable to XSS ("Google Docs Editor Extension quickstart | Apps Script") if any of the injected script snippets ran.

The table below includes a list of a number of functions that were used for this investigation, along with the selection criteria that were used, the attack vector that was used to exploit them, and the status of whether or not these extensions had been repaired. Microsoft Office 365 online has been used as the initial marketplace. Three extensions have been chosen after that. Due to time constraints and the study's scope, we were only able to choose three extensions from each marketplace. We could have chosen more, perhaps even fifty or even one hundred. The popularity of an extension, or the quantity of users or downloads, has been the primary criterion used in the selection process. Additionally, only free extensions were chosen, and some of them were chosen at random based on how frequently they were used.

Table 3: Results of Add-ons Study

Marketplace	**Selection Criterion**	**Vulnerable Add-ons**	**Attack Vector**	**Status**
MS Office Online	Popular - # of users	Translator for Outlook	Outside input	Being Fixed
	Popular - # of users	GIGRAPH-Network Visualization	Shared Workspace	No Response
	Random	Excel to JSON	Shared Worksapce	No Response
Google Suite	Popular - # of users	Form Ranger	Shared Workspace	Being Fixed
	Popular - # of users	Doc Appender	Shared Workspace	Being Fixed
	Random	Mail Merge	Shared Workspace	No Response
Shopify	Popular - # of users	Order Printer Pro	Shared Workspace	No Response
	Random	ShipHero Fulfillment	Shared Workspace	Being Fixed
	Random	Ship Systems 3D Box Packing	Shared Workspace	No Response

Microsoft Office 365 extensions investigation

For the case of Microsoft Office 365, three extensions were selected:

- Translator for Outlook
- GIGRAPH – Network Visualization
- Excel to JSON

The first extension was chosen because it was very user-friendly and had received a lot of downloads: Translator for Outlook. The primary function of this extension is to translate the content of the email for the recipient. In case the user is operating in a multi-cultural environment having employees from all over the world, who might not necessarily know how to communicate in English, this feature can help the recipient translate the received email into their desired language. That is the reason for its wide usage and the study being performed for this extension. First of all, this extension was installed and its various features along with its usability was explored. After that, a test item was created for this extension. Since Translator for Outlook is part of Office 365, email was utilized as the test case. Scripts were placed into the subject line and the entire content of the email, which was sent to the user directly.

Following the creation of the test item, all of the extension's functions were tested on it, and any overflows involving the rendering of the item's content were further investigated. In this instance, additional custom JavaScript code was written as well because certain extensions only took into account data in particular forms. It was determined that the extension was susceptible to cross-site scripting if any of the injected scripts were run. After the script was run in this instance, it was discovered that the extension was open to cross-site scripting attacks. Nevertheless, the patch for this extension is still being worked on and has not yet been deployed.

GIGRAPH-Network Visualization was the second Microsoft 365 addon that was chosen for investigation. The selection process also took into account the quantity of downloads and usage. With its strong yet user-friendly visualization capabilities, this plugin contributes to the expansion of MS Excel Online's core features. Among other things, data that describes directed interactions between several parties—like payments or dependencies—fits perfectly for visualization. This extension was installed in order to conduct tests, after which all of its features were investigated. A test item was then made for this extension after that.

Following the development of the test item, all of its features were tested on it, and any overflows involving the rendering of the item's content were then further investigated. Moreover, since this extension supports particular data types, additional JavaScript code was included. This extension was deemed susceptible to Cross-site Scripting if the script that was injected into the add-in runs successfully. Regarding GIGRAPH, there was no discernible reaction to the scripts that were injected.

The third extension Microsoft Office 365 that was selected for study was Excel to JSON. This add-in was also selected based on its usage. It can be found in the Microsoft AppSource store of the MS Office 365. This add-in can convert selected data to JSON. Row and Nested conversions are the two types that are supported in its current state. The first row in a row conversion will be regarded as the header, and the subsequent rows will be regarded as the data. A JSON schema is supplied for nested conversions, and the extension performs the conversion in accordance with the supplied schema. This application extension can also read and make changes to your document and send data over the internet to the requesting parties. For testing this extension, it was installed and all its features were explored. Some JavaScript code was added because it only accepts data in certain formats. If the script that is injected executes successfully, the extension is assumed to be vulnerable to the XSS vulnerability. For this add-in, the injected script gave no response. So, it is not considered vulnerable to the Cross-site vulnerability.

Google Marketplace extensions Analysis:

For the case of Google marketplace, three extensions were chosen for the study: -

- Form Ranger
- Doc Appender
- Mail Merge

Form Ranger is the first Google Marketplace extension chosen for the investigation. With the help of this online storage application extension, you can automatically fill in alternatives for grid, multiple choice, checkbox, and list questions based on data columns in any Google Sheet. Making sure form selections match values in the database of records, students, inventory items, and formulas like COUNTIF, VLOOKUP, and MATCH, among others, is beneficial. Options can also be configured to repopulate upon form submission. Similarly, take a list of open-ended submissions and apply the special formula to it to create a "growing" list of checked selections, leaving the remaining possibilities undefined.

This extension was installed initially, and then its different functions were tested. Following that, a test item for this add-in was made. It was examined for overflows involving the rendering of the item's content. Further custom JavaScript code was written in this instance as well, since certain extensions only took into account a limited set of data formats. Cross-site scripting vulnerability was identified in the extension if any of the injected scripts were run. The XSS vulnerability fix for Form Ranger is still being implemented.

Doc Appender was the second Google Marketplace extension chosen for research. With this plugin, certain Google Documents will have Google Form questions added to the bottom. It was installed first, and then its different features were investigated in order to perform the extension tests. For this extension, a test item has been made. The overflows rendering this item's content were examined. Additional custom JavaScript code was injected since certain extensions only took into account a particular format for data. A script was deemed vulnerable to cross-site scripting if it was executed. The XSS fix for Doc Appender is still being implemented.

Mail Merge was the third add-on chosen for analysis from the Google Marketplace. In addition, the fact that it is free and has received a lot of downloads and usage went into its selection. Gmail, Google Documents, Sheets, Forms, and Slides are all utilized with it. Insert place holders into your document, slide, or email body with ease with this straightforward yet effective Gmail tool for delivering bulk personalized emails. Additional use cases could involve leveraging apps like Zoom, Microsoft Teams, and others to invite people to events like webinars, weddings, job dates, conferences, fairs, or meetups. Moreover, it might also be used to send customized emails upon online submission of a Google Form. In addition, it is possible to apply simultaneously to multiple job offers, create and print envelopes for each recipient, send invoices and payment reminders to a list of clients, and send email notifications for Google Forms, which enable the sending of emails automatically in response to each Google Form submission. It is feasible to notify the form participant and his colleagues. These confirmation emails can help you streamline and de-stress your process.

This extension was initially installed, and its features were investigated, before being tested. Following that, a test item for this add-in was made. It was checked for overflows involving the item's content being rendered. Since some extensions only took into consideration data in a certain format, some JavaScript code was injected into the extension. An XSS vulnerability was identified in this extension if either of the injected scripts was executed. Regarding Mail Merge, this plugin was deemed to be not susceptible to Cross-site Scripting since it did not respond.

Therefore, it was discovered that Form Ranger and DocAppender, two Google marketplace extensions, were susceptible to cross-site scripting. In this investigation, however, it was discovered that the Mail Merge extension was not vulnerable.

Shopify extensions Analysis:

For the case of Shopify, three extensions were selected for the study: -

- Order Printer Pro
- ShipHero Fulfillment
- Ship Systems 3D Box Packing

Order Printer Pro, the first Shopify plugin, allows users to effortlessly personalize their store's logo, colors, and other elements while printing in bulk and sending automated invoices. Simply add your logo to the polished template, or use the code to fully customize it. An excellent, round-the-clock support staff is also available. It also enhances fulfillment by printing forms for returns and orders, among other things. It inserts invoice PDF links automatically. Draft orders may be printed and exported with ease. Supports B2B, translation, and multicurrency.

It was first installed in order to conduct testing, after which its features were investigated. Following that, a test item for this add-in was made. It was examined for overflows involving the rendering of the item's content. The extension only takes in data in a specific format, thus some JavaScript code was injected into it. This extension was thought to be vulnerable to cross-site scripting if the scripts were run. It was determined that cross-site scripting was not present in this instance because it did not respond.

ShipHero Fulfillment is the second extension we have in mind. It is a full-service eCommerce fulfillment solution for customers that come directly to you. You can forget about the hassle of managing your own warehouse when you work with ShipHero Fulfillment. Everything is taken care of by it, including picking, packing, shipping, and putting away merchandise. It offers quicker delivery, volumetric awareness, load balance, inventory control, reporting, and pre- and post-shipment visibility via ParcelView and PostHero.

It was installed initially, then its features were investigated, just as with other extensions. For this add-in, a test item was first made. It was examined for overflows that required the item's content to be rendered. Because the extension can only take data in a specific format, JavaScript code was injected into it. if the script was run. Cross-site scripting could exploit this feature, and a patch is currently being implemented.

Ship Systems 3D Box Packing is the third extension that is being tested in the Shopify marketplace. This plugin determines the smallest box or boxes your customers' orders will fit in by using actual product and package dimensions. After that, it receives real-time shipping quotes based on the packed order's final weight and dimensions. Customers will be shown the actual rates and given the option to select the ones they wish to pay 100% of the actual shipping costs. Their ability to safeguard their profit margins and obtain the desired products at actual pricing enables them to expand their commercial operations and grow more quickly. Additionally, it will demonstrate how to pack the finished order in 3D via an integrated app or your browser. You can obtain the packing instructions without ever leaving the store's admin area thanks to comprehensive Shopify shipping integration.

It was installed and its different functions examined for testing. For this extension, a test item was first made. It was examined for overflows that would have required rendering the content of the item. This extension only takes in a particular format of data, hence JavaScript code injection was done for it. Should the script execute successfully, the extension was deemed susceptible to cross-site scripting attacks. Regarding Ship Systems 3D Box Packing, the script received no answer. It is therefore thought to be immune to XSS threats.

Even though we conducted our study by hand, we believe it covers most, if not all, of the XSS flaws in the chosen extensions. This assurance is based on the fact that extensions are usually simpler and have a smaller feature set than standalone web services. Additionally, there are fewer opportunities for users to inject scripts. In light of this, black-box testing makes it simple to manually comprehend and validate any extension workflow.. Though our methodology produces results that we believe are worthy of wider consideration, it is obviously unfeasible for anyone wishing to perform extensive assessments of online storage-application extensions. In the future, a more thorough and effective approach will be required for such projects.

6.5 Comparative Analysis of Digital Forensic Frameworks

For the second part of thesis, an investigation model has been proposed for XSS threats concerning online storage-based application extensions. Therefore, in the analysis part, a comparative analysis of this framework with the previous versions needs to be done. That will involve comparing the pros and cons of each of these frameworks and then highlighting the proposed framework's importance.

The important part here is to note that these various frameworks presented here are generic and do not apply to any specific kind of attack or vulnerability. In the case of a real-life example of a certain attack, each of these models can be adopted for the specific case.

The table below shows how the suggested Digital Forensic Environment compares to the current frameworks:

Table 4: Comparative Analysis of Digital Forensic Investigation Models

Kruse	**US DOJ**	**ADFM**	**SDFIM**	**IDIP**	**DSCFIPM**	**AIDFF**	**Proposed DFIM**
3 Phases	4 Phases	9 Phases	11 Phases	17 Phases in 5 Groups	10 Phases	3 Phases	12 Phases
Acquisition	Collection	Identification	Preparation	Readiness	Preparation	Smart Acquisition	Preparation
Analysis	Examination	Preparation	Securing scene	Deployment	Strategic Planning & Scope Definition	Smart Analysis	Locate the Attacker
Presentation	Analysis	Approach	Survey & Recognition	Physical Crime Investigation	Crime scene sealing	Smart Presentation	Securing the Attacker/Victim
	Writing Report	Preservation	Documentation	Digital Crime Investigation	Evidence Collection		Identifying the Attacker
		Collection	Communication Shielding	Review	Evidence Preservation		Search Plan and Documentation
		Examination	Evidence Collection		Extraction of Relevant Evidence		Sealing of Crime Scene
		Analysis	Preservation		Evidence Analysis		Digital Evidence Collection and Preservation
		Presentation	Examination		Presentation and findings		Extract Relevant Digital Evidence
		Returning	Analysis		Archiving of case history		Examination
			Presentation		Post-case review		Analysis
			Review				Report Preparation

The first model put forth is the three-phase model of acquisition, analysis, and presentation developed by Kruse. When data is gathered from various sources during the collection phase, the research process starts. These sources include scanning the crime scene for clues and collecting evidence from hardware devices like hard drives and USB drives. The tools required for collection are also used in this phase, and the collected evidence is subsequently stored and kept safe. The second phase is Analysis, in which the collected data and various pieces of evidence are further analyzed for getting meaningful insights from the information. The third phase is presentation, in which the collected data and its post-analysis picture is presented to the audience in the form of Proof of Concept etc. This final form indicates the value of this data. From an analysis perspective, this model lacks a comprehensive approach.

The US Department of Justice's model is the second one put forth here. The collection, examination, analysis, and report writing steps make up this process. Both digital and physical evidence are acquired from the crime site during the first stage. The data is inspected in the second phase, and the most pertinent information is transformed into usable data that is then prepared for analysis. After the analysis phase, the digital forensic report is prepared which is then presented to the audience. From a comparative analysis point of view, this model is more comprehensive than the one proposed by kruse but still lacks the detailed approach and systematic way of digging out the problem.

With nine phases, the third model ADFM contains a great deal more information than the first two. The first step is the identification phase, which entails identifying an incident based on its indicators and subsequently classifying it. The second phase is preparation, which includes setting up instruments, protocols, authorization, permissions for surveillance, and managerial support. The third phase, referred to as the plan of action, aims to reduce the impact of the victim while maximizing the quantity of clean evidence that is obtained. The objective of the fourth phase, preservation, is to safeguard, isolate, and sustain the state of the tangible and digital data. Using established and approved protocols, onlina data is replicated to record the real scene in the fifth step of collecting. Examining the information in relation to the suspected crime is the sixth phase, which involves a thorough and methodical search. Analysis, which takes evidence-based judgments, reconstructs data fragments, and assesses importance, is the seventh phase. A presentation that summarizes and explains the conclusions is the eighth stage. The final stage is Returning Evidence, which entails figuring out what and how criminal data information needs to be deleted as well as ensuring that online and tangible Ownership is given back to the original owner. Compared to earlier models, this one is superior because it establishes uniform and standardized frameworks for the development of digital forensics and offers a way to adapt the same framework to new digital technologies. But there are drawbacks to this paradigm as well. Its categories can be seen too broad for practical use, and Increasing the number of subcategories in the system will only make it harder to utilize in general.

The fourth model, the Comprehensive online Investigation Model, consists of ten phases. There is a preparatory phase before the actual investigation starts. The second stage is keeping the data clean and the crime scene safe from intruders. Phase three has the investigators surveying the situation in order to assess it. In the fourth stage, The scene of the crime needs to be accurately recorded through mapping, drawing, and photography. Shielding communications would be stage five. The gathering of evidence, which comprises both volatile and non-volatile data, would be stage six. The seventh phase, preservation, entails storage, transit, and packaging. Phase eight would be an examination in which experts in forensics would examine the data extracted from the obtained data and uncover information crucial to building the case.. Phase nine is analysis, a technical assessment carried out by a team of investigators based on the evidence's examination findings. Phase Ten: Extracting and analyzing the acquired evidence is part of the presentation process. A wide range of stakeholders, including corporate management, legal experts, law enforcement officers, and technical specialists, may need to hear the outcomes. The model's last phase, Results and Review, is phase eleven. This means reviewing every stage of the study once more and identifying any areas for improvement. Comparing this model to the earlier versions, the former is more thorough and organized. It will help with the dynamics of the evidence and the reconstruction of events by identifying the traits of uniqueness, reliability, reliability, efficacy, validity, flexibility, effectiveness, and norms in the study of computer scams and cybercrimes.

The fifth model IDIP, the Integrated online investigation Procedure, consists of five groups of 17 phases. The phases of an assault scene examination are as follows: implementation, readiness, assessment, online, and actual search for crime scenes. This concept does not regard the computer as a piece of tangible evidence, but rather as a stand-alone crime scene. Instead of seeing the computer as the weapon used in the crime, it sees it more like a body at a murder scene. The body is examined for substances in the bloodstream, indications of past misuse, and an examination of the organs to ascertain the cause of death, including whether the gunshot was lethal, while the pistol is submitted to a lab for testing to ascertain whether it was fired and who handled it. In a similar vein, a body has other evidence for examination, such as chemicals. Computers are investigated with the same level of detail as a body in order to find more evidence and the chronology of events that occurred there. This paradigm helps define the relationship between physical and digital investigations and makes it easier to develop the technical needs for each phase. Because of its abstract nature, it can be used in both corporate and law enforcement environments. The results of digital analyses can be made more credible by using practices and models from the field of physical investigations, as digital evidence is being scrutinized more and more in court.

The Sixth model The six phases of DSCFIPM are as follows: creating standard operating procedures for institutional cyber forensics investigations (SOPs); defining the scope and strategy; sealing the crime scene; having investigating authorities collect evidence; preserving evidence; extracting pertinent evidence; analyzing evidence; archiving case history documents; presenting analysis and conclusions; and, finally, carrying out a post-case evaluation. This model is the most appropriate to date and has been more thorough than the preceding models.

The seventh AIDFF model is the Digital Forensic Framework, which is based on artificial intelligence. Its three unique processes are Intelligent Acquisition, Intelligent Analysis, and Intelligent Presentation. This idea integrates artificial intelligence and machine learning into each stage of the forensic investigative procedure. Machine learning algorithms are used during the phases of data gathering, processing, and display.

The model that has been suggested as the outcome of this investigation and following a comparison of all of these earlier models is the final eight model. This model, which contains 12 steps, is the most thorough one to date. The first stage is preparation, which entails learning about the offense, any associated actions, potential evidence sources, etc. The next step is to identify the attacker by figuring out their IP address, email address, and other details. Examining the email header will allow the investigators to determine the hacker's the machine's IP address. Several tools can be used to locate the attacker's location once the IP address has been determined. In order to protect the evidence, the victim machine must be secured in the third phase, which involves removing any potential entry points. One of the most important steps in the criminal detection approach is locating and identifying the perpetrator. The documentation and search strategy for gathering and examining digital evidence make up the fifth phase. It is also necessary to maintain the data record through documentation. Phase six involves sealing the scene of the crime, which aids cyber forensic specialists in locating and examining reliable digital evidence connected to cybercrime. Phase seven involves gathering and preserving digital evidence, making sure that it is obtained using appropriate scientific techniques to preserve its integrity. The goal is to gather and preserve both volatile and non-volatile evidence. The eighth phase involves utilizing resources like FTK, Encase, and Access data to retrieve pertinent digital evidence connected to the crime. Phase nine is the examination, where appropriate paperwork is required, including the investigator's name, the examination date, and other facts. To demonstrate that the data was not tampered with after the inspection, data authentication is also required. Phase ten is analysis, wherein the investigators examine the data in order to get ready to draw a conclusion. The goal is to obtain more precise outcomes. For the outcomes to be approved by the court later on, it is imperative that the legal and administrative regulations be followed at this point. Report preparation, phase 11, is carried out to obtain more precise results. In this phase, the evidence is initially gathered from the machines of the attacker and the victim, and it is then compared. If the outcomes match, it proves the attacker utilized the same system, and this evidence is admissible in court. If the outcomes are in agreement, a report must be prepared in compliance with all applicable rules and legislation. Presenting the case to the appropriate authority is the twelfth and last step. Out of all the models presented, this one is the most thorough and error-free.

6.6 Synthesis of results

The Synthesis of Results, the fifth stage of the literature review, is covered in this section. The qualitative synthesis in this case is the performed synthesis. The implementation and analysis of the research have already been summarized. The body of evidence's advantages and disadvantages are discussed here, along with a cumulative evaluation of the risk of bias across research. The holes in the earlier studies that were not sufficiently examined come next. This is carried out for the two research components.

We found 4 susceptible extensions out of the total of 9 extensions that were analyzed. This result emphasizes how common XSS vulnerabilities are in the online storage-application extensions of today. The names of these extensions, their attack routes, and whether or not fixes have been applied are shown in Table 4. Of the three marketplaces, G Suite is the only one that offers data on the quantity of users for every extension. Notably, Form Ranger, the most vulnerable extension in the marketplace with the most users, had about 7.8 million users. The most widely used vulnerable extensions in the Shopify and Microsoft Office Online marketplaces were Order Printer Pro and Translator for Outlook, respectively. 371 reviews were left for the latter, compared to 1772 reviews for the former.

In all three marketplaces, the popular extensions appear to have a lower vulnerability rate. This could be explained by the possibility that more seasoned developers create popular extensions. Furthermore, it seems that vulnerabilities in extensions that accept input from outside sources are uncommon. In particular, our analysis found that only one extension exhibited this kind of susceptibility. We hypothesize that extension developers are more used to risks coming from shared workspaces than from external sources, like emails in this case.

We communicated the vulnerabilities to every extension developer we could get in contact with. We also provided instructions on how to fix these security holes (see the previous section's remedies for extension authors). As of this writing, of all the extension teams and developers we reached out to, one has acknowledged the problems with their extension and fixed them, while some developers have acknowledged the weaknesses and are attempting to fix them, the remaining developers have not responded to us. We also discussed with Google the Picker API problem. They acknowledged the problem and said they will consider it when they release their next extension system.

The research's second section presented an approach for online investigation. This approach, with its twelve phases, was significantly more comprehensive than the others. As a result, it is appropriate enough to be used in forensic investigations pertaining to both physical and digital crimes, as this thesis's setting illustrates.

7 Recommendations, Conclusion & Future Work

This chapter discusses the last stage of the systematic review which is the conclusion of all our implemented work. It also includes recommendations for future developers and vendors. Finally, in the conclusion part, all the research questions have also been answered.

7.1 Recommendations

In this section, based on our study in the previous chapters, we outline measures that the online storage application vendors and extension developers can take to protect against XSS attacks resulting from susceptible extensions.

Strategies for extension developers

Since untrusted data may contain dangerous JavaScript code, extension developers should refrain from embedding it into the extension UI as HTML in order to prevent XSS. We outline some recommended practices below for them to adhere to.

Coding practices

The majority of the online safety literature covers the use of secure coding techniques to stop cross-site scripting. One simple solution to stop XSS threats with online storage-application extensions is to handle user input as text instead of HTML at all times. Developers should utilize the textContent and innerText attributes instead of innerHTML. When using jQuery, the.text() function should be utilized rather than the.html() method.

User data is generally not intended to be managed in the same way as online application logic. Nevertheless, if it's essential to consider untrusted HTML in the extension user interface, developers must properly check and escape input beforehand. Most server-side web development platforms include built-in capabilities for these kinds of tasks. Before processing user data, the client can extract dynamic HTML elements and attributes using JavaScript techniques like.toStaticHTML(). for comprehensive guidelines on character escapes to stay away from XSS.

Security Enforcement

Because extensions are web services, developers can strengthen the security of their extensions by implementing a Content Security Policy (CSP). `<script>} tags and inline event handlers are examples of inline scripts that could not be run under strict policies. By using this method, malicious scripts that are successfully injected into the extension user interface will not run. The developers would only allow trustworthy servers in order to import script code that was encapsulated in separate.js files and allowed to run.

It is not always feasible to completely forbid inline scripts, though, as valid inline scripts are frequently chosen for a variety of functions, including event handlers. It is not enough to just copy and paste the code to move these handlers to separate.js files; instead, they must be completely rewritten using DOM APIs. Fortunately, this trouble can be avoided by implementing CSP in a less restrictive manner (Stamm 2010). In particular, policies that are hash- or nonce-based can be applied, allowing inline scripts with pre-registered nonce or hash values to run.

Minimizing the permissions needed for their extensions is something that extension developers ought to strive for. As was the case with the Form Ranger extension in chapter 4, redundant permissions allowed an XSS attacker to obtain all of the victim's storage devices and send emails on their behalf. Additional generic precautions might be taken to prevent XSS. For example, for temporary cookies and any personal cookies, you should always have HTTPOnly set that JavaScript is unable to access. Furthermore, automatic escaping is a feature that a lot of web frameworks provide and should be used whenever practical.

XSS Detection

Extension developers can use the same methods that we used in our empirical analysis to assess if these extensions are vulnerable. They can explicitly create unit tests to generate test items that are similar to what we accomplished and to continuously check for XSS vulnerabilities during the development process. This approach should work well for testing individual extensions, even though it is not very scalable for large-scale testing.

Insights for cloud application providers

We examined the three widely used extension system designs in Section 4 and spoke about the potential effects of an XSS attack on each. The insights we learned from this analysis are presented in this section, with a particular emphasis on design decisions and how they affect extension security. We think that these lessons will be useful for online storage application providers who are developing or enhancing their extension systems.

Harden the extension iframe

The extension's user interface is housed inside an iframe, which by default has the capacity to use browser APIs to request access to local device features like the camera, microphone, and geolocation. The host application needs to set the sandbox and allowance attributes of the iframe in order to restrict the browser functionalities that the iframe extension can access. Currently, host apps can use a new iframe property called csp to enforce a Content Security Policy (CSP) on extensions thanks to an experimental functionality in Opera and Chrome. A policy that must be upheld by the embedded page can be specified using this attribute. If this were to become a common feature of browsers, it would give online storage-application providers control over the CSP policy in their extensions. However, the majority of the examined modifications did not make use of CSP, which makes putting such constraints into practice difficult. A large CSP (allowing only inline scripts with specific hash or nonce values) will likely impede market extensions.

Implement extension logic in the extension server, not in clientside JavaScript

Microsoft Office Online uses cross-origin messaging within the browser to enable quick access from the extension to user data in the main program UI. Unfortunately, because the malicious scripts used by the attackers to exploit Cross-site Scripting are run within the context of the extension, the wide, general permissions controlling this access are useless against them. Extension logic should be applied on the server side, similar to Shopify's and G Suite's methods, to improve defense against XSS. In the online storage application, the extension server acts as an isolation layer between user data and client-side scripts in two ways: It first establishes a purpose-specific, limited interface for all user data access. By implementing business logic, it additionally filters the types of read and write operations that are sent to the online storage service. These methods act as a buffer between the user's information and any potential XSS code contained in the iframe extension.

Scripts for filtering user input

It is important to carefully analyze the types of user input that host application providers must control in a shared workspace and filter out undesired inputs. For instance, Shopify manages relatively structured data while handling shop resources like orders and commodities, so developers may choose which code segments shouldn't use HTML or JavaScript. G Suite and MS Office Online, for example, are applications that deal primarily with documents; they have more challenges because legitimate HTML and scripts can appear in unexpected locations.

Do not share access tokens to delegate all your permissions

As bearer tokens, OAuth 2.0 tokens allow anybody in possession of the token to access the related resources. This may result in careless coding techniques, such as the Picker API in Google Suite, where very strong tokens are issued to unsafe locations. The extension server should be in charge of access control rather than distributing the access token to the client side. They should only provide the bare minimum of client-side permissions when sharing tokens is necessary. For example, while using the Picker API, the extension server ought to supply the user interface (UI) with a restricted token that permits the listing of only particular files from the user's storage device.

Do not request user approval during runtime

Granting extensions access to user data solely on the basis of the user's choice might not be as beneficial as it initially appears. Specifically, requesting approval from the user while the application is still running could cause them to consent to all prompts—even ones that come from malicious code. Therefore, it can be preferable to request user permission through a distinct user interface while installing or updating extensions. The inability to implement document-level access control is a disadvantage of this approach.

Recommendations Regarding Forensic Framework

The Digital Forensic model proposed here comprises of 12 phases and is the most comprehensive model compared to the existing ones in literature. Owing to the detail orientedness of this model, it is recommended that for the future forensic investigations whether they are physical crimes or the digital crimes, this should be adopted because it will unveil more meaningful results and insights into the nature of committed attack. This environment is perfect for conducting a online criminal examination into an online storage application XSS assault.

7.2 Advantages of Conducted Research

The main advantages of this research are as follows:

- This thesis offers fresh perspectives on the security of cloud application extensions and identifies potential vulnerabilities that these extensions may introduce into their host applications.
- It provides guidance on how to prevent XSS attacks for upcoming developers and providers of online storage applications.
- The suggested investigation model serves as a benchmark and reference point for digital forensic development, offering a framework for looking into a variety of computer fraud and cybercrime cases in the constantly changing field of information technology.
- This model will also aid in the creation of generalized solutions that can meet the needs of a rapidly changing and extremely volatile digital environment.

7.3 Reliability of Thesis Work

The research conducted in this thesis is based on the literature review from authentic resources which include Top quality journals, conferences research papers and previous JAMK and other good universities' databases. Moreover, the case study performed on add-ons from Shopify, Microsoft Office 365 and Google suite is for a few select add-ons and not for a wide range of extensions. And these extensions were selected mainly based on number of downloads and the real users that are using those applications. So, the performed work is on the open-source applications in use. Furthermore, to further validate the results, the next steps could be to conduct this study for at least 100 add-ons from each platform and verify the results to confirm what percentage of add-ons were found to be vulnerable and what results can be deduced from the security testing of extensions from each platform. The results gathered in that way would be more authentic and would give more value to the results of empirical security testing of cloud application add-ons.

7.4 Future Work

Given that XSS is a well-known vulnerability, cautious engineering methods have been created to prevent these errors. Developers are aware of how critical it is to filter unsolicited input, and online storage application providers have developed toolchains and platforms to shield their products against the majority of code injection attacks. But the problem still exists. Attackers are always coming up with new ways to get around defenses, and threat analysis and defenders find it difficult to keep up with the speed at which software is developed. The goal of this study is to discuss these changing issues in a vital field of contemporary software development.

Through experiments, we have verified that the vulnerabilities reported in this report are genuine and exploitable. However, there would be certain challenges for an attacker in the real world. The attacker must identify the susceptible extension the victim is using, and the victim must permit the unsecured extension on the shared file. As a result, an effective assault probably needs a weak extension that people regularly use for a variety of documents they are reading or modifying. These requirements might be met by translator and writing assistant extensions. In addition, if the victim comes across a document that has the pertinent issue, they may install extensions that deal with certain data difficulties, such eliminating duplicates.

This thesis highlights another serious issue, in addition to the XSS vulnerabilities in extensions: the Picker API's usage of OAuth 2.0 tokens and the serious flaws that make it possible for XSS attackers to take advantage of them. Developers are advised by Picker API docs to adhere to a design pattern in which the client-side script shares its OAuth 2.0 token with the extension server. Developers may still employ this dangerous software design even if a safer alternative to the Picker API is substituted.

These factors suggest that further steps may be needed to stop access token misuse. One advantage of Google Suite is that its extension server runs on the Google online storage; third-party code only handles the access token in controlled, defined cases. Furthermore, host applications have the ability to easily reject tokens that come from unapproved sources as opposed to the authorized extension server. Bearer tokens are popular among developers because of their convenience and flexibility, but putting these constraints in place would make them less functional. The authors of OAuth 2.0 deliberately left out support for channel binding, which involves assigning a piece of information to a particular client address. In conclusion, while it is possible to mitigate the dangers related to unsafe token delegation to client-side scripts, a complete solution may not always be achieved right once.

Overall, because additional research is obviously needed, we hope that this study will raise awareness of the topic of online storage-application extensions. It's possible that attackers can take advantage of harmless extensions through additional attack channels. Furthermore, examining the risks presented by malicious extensions may be a fascinating area for further investigation.

Additionally, with regards to the forensic framework investigation model, the recently suggested model can be utilized in a multitude of scenarios, including diverse cyber-attacks. Furthermore, the model will need to be updated over time if new restrictions on technological advancement are discovered.

7.5 Conclusion

Since cloud applications are a relatively recent phenomena, there hasn't been much research done on the vulnerabilities associated with them. Therefore, in this work, we examined the security of these extensions and found that their shortcomings could expose their host apps to new security risks. The following research questions were successfully answered during this research-

- Show how do insecure extensions enable XSS threats against users of cloud applications.
- Examining the architecture and design of popular cloud software suites like Microsoft Office Online, Google Suite, Shopify and AWS.

The research found out that cloud application add-ons are vulnerable to XSS attacks since they frequently handle untrusted input carelessly. The threat actors may insert malicious scripts into emails or uploaded files, which are then processed by flimsy extensions. Our research demonstrated that these weak extensions are real and easily exploitable.

- A discussion on secure product development best practices for extension developers to adhere to.

Moreover, it seems that cloud application providers may include extra measures to restrict the activities an attacker can carry out after their XSS code is executed within an extension. These practices include but are not limited to adopting secure coding techniques, security enforcement, unit testing. Hardening of application iframe and filtering user input.

- A framework for Digital Forensics that analyzes XSS attacks connected to extensions for cloud applications.

Finally, after a comparative study of existing Digital Forensic frameworks in practice, a new model of XSS forensic investigation has been proposed here that is more thorough than the previous models and provides a more methodical understanding of the examination process.

8 Ethics

8.1 Research Ethics & Data Protection

The Ethical Guidelines of JAMK University of Applied Sciences, which were previously adopted by the Student Affairs Board in 2018, were followed in the writing of this thesis. This indicates that the research for this project adheres to the standards of accuracy and integrity.

Prior to starting the thesis, an ethical pre-assessment and research agreement were completed. Other researchers' research in related fields is respected and taken into account. References to other researchers' work are appropriately made. The research is designed, executed, and documented in compliance with the standards for scientific data.

Once all parties reached a consensus, fully licensed Microsoft Office products were used for data transfers and savings, and a different tool for communication. Only those involved in the initiative, such as researchers and JAMK representatives, had access to the data.

Each and every party engaged in the thesis reached an agreement that the authorized thesis would be published in the University of Applied Sciences' Theseus Open Repository, making it impossible for it to contain any material that isn't already public knowledge or be used illegally.

All of the materials used in the research are publicly available and open source, with the exception of the ISO standards, which required purchase in order to access the data. In accordance with JAMK rules, a bibliography and citations have been supplied for each source and other researcher. To create the bibliography and designate citations, the citation generator Zotero was utilized.

Persons taking part in the interviews had their personally identifiable information (PII) organized. Name, phone number, and email address were among the details. Following the interviews, the list of personal information was removed and replaced with the name of the organization, fulfilling the requirements of the General Data Protection Regulation (GDPR). The interviews have not been named, and the study data has been kept anonymous.

Overall, from the beginning to the end of the study, the researcher encouraged the proper conduct of research.

9 References

Singh, A., & Chatterjee, K. (2017). Cloud security issues and challenges: A survey. *Journal of Network and Computer Applications*, *79*, 88–115. https://doi.org/10.1016/j.jnca.2016.11.027

Koskinen, P., & Simola, V. (2019). *Self-assessment of security in cloud deployment* [fi=Ylempi AMK-opinnäytetyö|sv=HögreYH-examensarbete|en=Master'sthesis|]. http://www.theseus.fi/handle/10024/166503

Charif, B. (2014). *Security Concerns on Adoption of Cloud Computing*. https://urn.kb.se/resolve?urn=urn:nbn:se:ltu:diva-57677

Hoffman, K. (n.d.). *LibGuides: Systematic Reviews: What are the Steps of a Systematic Review?* Retrieved June 3, 2024, from https://hslguides.osu.edu/systematic_reviews/steps

Vines, R., & Krutz, R. (2010, August). *Cloud Security: A Comprehensive Guide to Secure Cloud Computing | Wiley*. Wiley.Com. https://www.wiley.com/en-us/Cloud+Security%3A+A+Comprehensive+Guide+to+Secure+Cloud+Computing-p-9780470938942

Winkler, V. (2012). Securing the Cloud: Cloud Computer Security Techniques and Tactics. *Computers &amp; Security*. https://www.academia.edu/77450555/Securing_the_Cloud_Cloud_Computer_Security_Techniques_and_Tactics

Aceto, G., Botta, A., De Donato, W., & Pescapè, A. (2013). Survey Cloud monitoring: A survey. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, *57*(9), 2093–2115. https://doi.org/10.1016/j.comnet.2013.04.001

Kolb, S., & Röck, C. (2016). Unified Cloud Application Management. In R. Bahsoon & L.-J. Zhang (Eds.), *IEEE World Congress on Services, SERVICES 2016, San Francisco, CA, USA, June 27—July 2, 2016* (pp. 1–8). IEEE Computer Society. https://doi.org/10.1109/SERVICES.2016.7

Kavis, M. (2014, January). *Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS) | Wiley*. Wiley.Com. https://www.wiley.com/en-us/Architecting+the+Cloud%3A+Design+Decisions+for+Cloud+Computing+Service+Models+%28SaaS%2C+PaaS%2C+and+IaaS%29-p-9781118617618

Mell, P., & Grance, T. (2011, September). *SP 800-145. The NIST Definition of Cloud Computing*. Guide Books. https://dl.acm.org/doi/book/10.5555/2206223

Ramgovind, S., Eloff, M. M., & Smith, E. (2010). Management of security in Cloud computing. *2010 Information Security for South Africa*, 1–7. https://doi.org/10.1109/ISSA.2010.5588290

Ranjan, R. (2014). *A Comparative Study of SaaS, PaaS and IaaS in Cloud*. https://www.semanticscholar.org/paper/A-Comparative-Study-of-SaaS%2C-PaaS-and-IaaS-in-Cloud-Ranjan/4a613de7b3a0ead4a08ed9a3caac79e553edbf46

o365devx. (2022, June 30). *Permissions element in the manifest file—Office Add-ins*. https://learn.microsoft.com/en-us/javascript/api/manifest/permissions?view=common-js-preview

o365devx. (2024, January 19). *Understanding Outlook add-in permissions—Office Add-ins*. https://learn.microsoft.com/en-us/office/dev/add-ins/outlook/understanding-outlook-add-in-permissions

Veijanen, J. (2020). *Implementation of security best practices on AWS Cloud: Case: Vulnerability scanning of EC2 instances and networks* [fi=Ylempi AMK-opinnäytetyö|sv=Högre YH-examensarbete|en=Master's thesis|]. http://www.theseus.fi/handle/10024/343321

Google Cloud projects | Apps Script. (n.d.). Google for Developers. Retrieved June 3, 2024, from https://developers.google.com/apps-script/guides/cloud-platform-projects

Google. What is Google Picker? https://developers.google.com/picker/.

Order Printer Pro: Invoice App - Order Printer Pro – Shopify Invoice App: Deliver Custom Docs | Shopify App Store. (n.d.). Retrieved June 3, 2024, from https://apps.shopify.com/order-printer-pro

Narayana, S. (2022). *Security Analysis of Web Application for Industrial Internet of Things* [fi=Ylempi AMK-opinnäytetyö|sv=Högre YH-examensarbete|en=Master's thesis|]. http://www.theseus.fi/handle/10024/750085

Reining in the web with content security policy | Proceedings of the 19th international conference on World wide web (world). (n.d.). ACM Other Conferences. https://doi.org/10.1145/1772690.1772784

Loukasmäki, H. (2023). *Cyber Incident Response in Public Cloud: Implications of modern cloud computing characteristics for cyber incident response* [fi=Ylempi AMK-opinnäytetyö|sv=Högre YH-examensarbete|en=Master's thesis|]. http://www.theseus.fi/handle/10024/803156

Loaiza Enriquez, R. (2021). *Cloud Security Posture Management /CSPM) in Azure* [fi=AMK-opinnäytetyö|sv=YH-examensarbete|en=Bachelor's thesis|]. http://www.theseus.fi/handle/10024/504136

Pandey, P. (2021). *Security attacks in cloud computing* [fi=AMK-opinnäytetyö|sv=YH-examensarbete|en=Bachelor's thesis|]. http://www.theseus.fi/handle/10024/502878

Tuunainen, T. (2021). *White Hat hacking: System and application security focusing on its fundamentals, malware and Wi-Fi vulnerability* [fi=AMK-opinnäytetyö|sv=YH-examensarbete|en=Bachelor's thesis|]. http://www.theseus.fi/handle/10024/496306

Kyei, K., Zavarsky, P., Lindskog, D., & Ruhl, R. (2013). A Review and Comparative Study of Digital Forensic Investigation Models. *Digital Forensics and Cyber Crime*, 314–327. https://doi.org/10.1007/978-3-642-39891-9_20

Reith, M., Carr, C., & Gunsch, G. (2002). An Examination of Digital Forensic Models. *Int. J. Digit. EVid.* https://www.semanticscholar.org/paper/An-Examination-of-Digital-Forensic-Models-Reith-Carr/c73f47d8385f452dfd25bbaab754874b65594ccd

Reith, M., Carr, C., & Gunsch, G. (2002). An Examination of Digital Forensic Models. *Int. J. Digit. EVid.* https://www.semanticscholar.org/paper/An-Examination-of-Digital-Forensic-Models-Reith-Carr/c73f47d8385f452dfd25bbaab754874b65594ccd

People—CERIAS - Purdue University. (n.d.). Retrieved June 2, 2024, from https://www.cerias.purdue.edu/

Thakar, A. A., Kumar, K., & Patel, B. (2021). Next Generation Digital Forensic Investigation Model (NGDFIM)—Enhanced, Time Reducing and Comprehensive Framework. *Journal of Physics: Conference Series*. https://www.semanticscholar.org/paper/Next-Generation-Digital-Forensic-Investigation-Time-Thakar-Kumar/a17de6d3f46e7eb8ecd10c4c9b03f5019afcbb91

Agarwal, A., & Gupta, M. (2011). Systematic Digital Forensic Investigation Model. *International Journal of Computer Science and Security*, *5*(1). https://www.cscjournals.org/library/manuscriptinfo.php?mc=IJCSS-438

Staff permissions. (n.d.). Shopify Help Center. Retrieved June 3, 2024, from https://help.shopify.com/en/manual/your-account/staff-accounts/staff-permissions

Mukasey, M., & Sedgwick, J. (n.d.). *Electronic Crime Scene Investigation: A Guide for First Responders, Second Edition*. US Department of Justice.

Satti, R., & Jafari, F. (n.d.). Domain Specific Cyber Forensic Investigation Process Model. *Journal of Advances in Computer Networks*, *3*.

Hannousse, A., Yahiouche, S., & Nait-Hamoud, M. C. (2022, May 17). *Twenty-two years since revealing cross-site scripting attacks: A systematic mapping and a comprehensive survey*. arXiv.Org. https://doi.org/10.1016/j.cosrev.2024.100634

Goni, I., Gumpy, J. M., Maigari, T. U., & Mohammad, M. (2020). Cybersecurity and Cyber Forensics: Machine Learning Approach Systematic Review. *Semiconductor Science and Information Devices*, *2*(2), Article 2. https://doi.org/10.30564/ssid.v2i2.2495

Ranta, J. (2023). *Testing AWS hosted Restful APIs with Postman* [fi=Ylempi AMK-opinnäytetyö|sv=Högre YH-examensarbete|en=Master’s thesis|]. http://www.theseus.fi/handle/10024/789489

Al-Mousa, M. R. (2021, January 5). *Analyzing Cyber-Attack Intention for Digital Forensics Using Case-Based Reasoning*. arXiv.Org. https://doi.org/10.30534/ijatcse/2019/92862019

Dimitriadis, A., Ivezic, N., Kulvatunyou, B., & Mavridis, I. (2020). D4I - Digital forensics framework for reviewing and investigating cyber attacks. *Array (New York, N.Y.)*, *5*. https://doi.org/10.1016/j.array.2019.100015

10 Practical scenarios for XSS attacks. (2024, March 15). Pentest-Tools.Com. https://pentest-tools.com/blog/xss-attacks-practical-scenarios

Black Hat USA 2017. (2016, May 2). https://www.blackhat.com/us-17/briefings/schedule/#dont-trust-the-dom-bypassing-xss-mitigations-via-script-gadgets-7567

Stoyanova, M., Nikoloudakis, Y., Panagiotakis, S., Pallis, E., & Markakis, E. (2020). A Survey on the Internet of Things (IoT) Forensics: Challenges, Approaches, and Open Issues. *IEEE Communications Surveys and Tutorials*. https://www.semanticscholar.org/paper/A-Survey-on-the-Internet-of-Things-(IoT)-Forensics%3A-Stoyanova-Nikoloudakis/2d8f2dab26c4b81acebaee1ab4c4cb9f5d8912dc

Bhushan, S. (2022). A Novel Digital Forensic Inspection Model for XSS Attack. In R. Kumar, C. W. Ahn, T. K. Sharma, O. P. Verma, & A. Agarwal (Eds.), *Soft Computing: Theories and Applications* (pp. 747–759). Springer Nature Singapore.

10 Appendices

Appendix 1: Add-ons Shortlisted for Study

Marketplace	Selection Criterion	Vulnerable Add-ons	Attack Vector	Status
MS Office Online	Popular - # of users	Translator for Outlook	Outside input	Being Fixed
	Popular - # of users	GIGRAPH-Network Visualization	Shared Workspace	No Response
	Random	Excel to JSON	Shared Worksapce	No Response
Google Suite	Popular - # of users	Form Ranger	Shared Workspace	Being Fixed
	Popular - # of users	Doc Appender	Shared Workspace	Being Fixed
	Random	Mail Merge	Shared Workspace	No Response
Shopify	Popular - # of users	Order Printer Pro	Shared Workspace	No Response
	Random	ShipHero Fulfillment	Shared Workspace	Being Fixed
	Random	Ship Systems 3D Box Packing	Shared Workspace	No Response