

# USER EXPERIENCE AND ROBUSTNESS IN SOCIAL VIRTUAL REALITY APPLICATIONS

---

A Thesis presented to  
the Faculty of the Graduate School  
at the University of Missouri

---

In Partial Fulfillment  
of the Requirements for the Degree  
Doctor of Philosophy

---

by  
SAMAIKYA VALLURIPALLY  
Dr. Prasad Calyam, Thesis Supervisor  
December 2020

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

USER EXPERIENCE AND ROBUSTNESS IN  
SOCIAL VIRTUAL REALITY APPLICATIONS

presented by Samaikya Valluripally,  
a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

---

Dr. Prasad Calyam

---

Dr. Khaza Anuarul Hoque

---

Dr. Zhihai He

---

Dr. Isa Jahnke

## ACKNOWLEDGMENTS

My Ph.D. journey at Mizzou gave me amazing opportunities to grow professionally in terms of research, industry collaborations, inter-disciplinary efforts, and leadership skills. To achieve this, I am grateful and honored to work with many talented people, amazing professors, and peers. First and foremost, I would like to thank my advisor Dr. Prasad Calyam for his immense support during the pursuit of my Ph.D. Dr. Calyam motivated me towards the goals of my Ph.D. dissertation along with guidance for the career growth opportunities suitable to me. In addition to our academic collaboration, I greatly value Dr. Calyam for believing in me during the tough times of my Ph.D. thesis research. I cannot imagine a better advisor and an example of success, productivity than Dr. Calyam.

I would like to thank the esteemed faculty members – Dr. Khaza Anuarul Hoque, Dr. Henry He, Dr. Reshmi Mitra for giving me an opportunity to work with for my research goals. I would like to express my gratitude to Dr. Khaza Anuarul Hoque, Dr. Henry He, and Dr. Isa Jhanke for their interest and consent to be part of my thesis committee. In addition, a special and heartfelt thanks to my Virtualization, Multimedia, and Networking (VIMAN) Laboratory peers – Aniket Gulhane, Songjie Wang, Sai Swathi, Dmitrii Chemodanov, Yuanxun Zhang, Ronny Bazan Antaquera, Roland Oruche, Arjun Chandrashekara, Shreya Nuguri, Vaibhav Akashe, Chengyi Qu.

Last but not the least, I would like to thank the most special people in my life - my parents who are my strength. You were the only people who believed in my choice of going ahead for a Ph.D. after my Masters's degree. Thank you for making me a good human being and guiding me towards persistence in the pursuit of my career with humility. I also would like to thank my friends (Awesome people, Mizzou seniors, and STARs) who are my family. Mainly, for staying with me through thick and thin and making Columbia, my second home in this awesome journey which I will remember for life!

# TABLE OF CONTENTS

<b>ACKNOWLEDGMENTS</b> . . . . .	<b>ii</b>
<b>LIST OF TABLES</b> . . . . .	<b>vii</b>
<b>LIST OF FIGURES</b> . . . . .	<b>ix</b>
<b>ABSTRACT</b> . . . . .	<b>xiv</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Social Virtual Reality (VR) as a new paradigm . . . . .	1
1.1.1 Novel Application Domains for Social VR . . . . .	2
1.1.2 Applications of Social VR as Learning Environments (VRLE) . . . . .	3
1.2 Robustness and User Immersion Challenges . . . . .	4
1.2.1 Inducing Factors of Cybersickness . . . . .	4
1.2.2 Disruption Factors of User Immersion . . . . .	7
1.2.3 Social VRLE Application Case Study . . . . .	8
1.3 Joint Tuning Approach of Robustness and Immersive Performance based on DevSecOps Paradigm . . . . .	9
1.4 Thesis Organization . . . . .	13
<b>2 Towards Harmonizing Robustness and Performance by Design</b>	<b>15</b>
2.1 Background and Related works . . . . .	15
2.1.1 Characterization of Robustness Factors in social VRLE . . . . .	17
2.1.2 Threat Modeling Approaches using STRIDE . . . . .	19
2.1.3 Attacks in VRLE application use case: . . . . .	20
2.1.4 Statistical Model Checking . . . . .	22
2.1.5 Design principles . . . . .	24
2.2 Quantitative Framework for Security, Privacy Issues . . . . .	24

2.2.1	Formal Modeling of Security and Privacy using Attack Tree (AT) Formalism . . . . .	25
2.2.2	Formal Description of Novel ATs for Security and Privacy issues in VRLE . . . . .	27
2.2.3	Translation of ATs into Stochastic Timed Automata . . . . .	28
2.3	Quantitative Analysis of ATs and Evaluation of Design Principles . . . . .	30
2.3.1	Vulnerability Analysis for Security AT . . . . .	31
2.3.2	Vulnerability Analysis for Privacy AT . . . . .	33
2.3.3	Recommended Design Principles . . . . .	34
2.3.4	Conclusion . . . . .	38
2.4	Social Virtual Reality Attacks inducing Cybersickness . . . . .	39
2.4.1	Background on Social Virtual Reality Attacks: . . . . .	40
2.4.2	Cybersickness in Virtual Reality Environments . . . . .	41
2.4.3	Experimental Validation of Social VR Attacks Inducing Cybersickness . . . . .	43
2.4.4	Impact Analysis of Security, Privacy Issues on Cybersickness . . . . .	47
2.5	Security, Privacy Harmonized Framework for Social VRLE . . . . .	51
2.5.1	Formalization of Security and Privacy Attack-fault Trees (AFTs) . . . . .	54
2.5.2	Formal Description of Novel AFTs for VR attacks inducing cybersickness . . . . .	57
2.5.3	Translation of attack-fault trees into stochastic timed automata . . . . .	60
2.5.4	Attacker Profiling (AP) for BAS . . . . .	63
2.6	Quantitative Analysis of Safety AFT . . . . .	66
2.6.1	Vulnerability Analysis in the safety AFT . . . . .	66
2.6.2	Evaluation of attacker profiles in terms of impact on disruption of safety AFT . . . . .	69
2.6.3	Risk assessment based on vulnerability analysis and attacker profiles . . . . .	70

2.7	Recommended Design Principles . . . . .	72
2.7.1	Implementation of design principles on safety AFT: . . . . .	73
2.8	Conclusion . . . . .	75
<b>3</b>	<b>Threat Intelligence Collection for Critical Anomaly Event De-</b>	
	<b>tection . . . . .</b>	<b>79</b>
3.1	Attack Issues Disrupting Immersion Factors . . . . .	79
3.1.1	Quantifying Immersive User Experience . . . . .	80
3.2	Anomaly Detection Module . . . . .	83
3.3	Anomaly Detection using Machine learning (ML) techniques . . . . .	85
3.3.1	Attack Data Collection in Testbed Setup . . . . .	86
3.3.2	Data Preprocessing and Feature Extraction . . . . .	91
3.3.3	Accuracy Analysis for Suitable ML Model . . . . .	93
3.3.4	Single and Multi- Network Attack Detection Results . . . . .	94
3.3.5	Anomaly Detection using Statistical Analysis . . . . .	98
3.3.6	Attack Data Collection . . . . .	102
3.3.7	Accuracy Analysis of statistical analysis technique . . . . .	103
3.3.8	Application Attack Detection Results . . . . .	104
3.4	Quantitative Metrics for Attack Occurrence based on Attacker Profiling . . . . .	105
3.5	Impact analysis on User Immersion for Detected Attacks . . . . .	107
3.6	Knowledge Base . . . . .	108
3.7	Summary . . . . .	108
<b>4</b>	<b>Adaptive Control of Robustness and Immersive User Experi-</b>	
	<b>ence for Resilience . . . . .</b>	<b>110</b>
4.1	3QS Framework Dynamic Rule Based Approach for Social VRLE . . . . .	110
4.1.1	Closed Loop Mechanism in Social VRLE Case Study . . . . .	111
4.1.2	Anomaly Monitoring Tool . . . . .	112

4.1.3	Dynamic Decision Making for Handling Critical Anomaly Events . . . . .	113
4.1.4	Creation of a Knowledge Base . . . . .	116
4.1.5	Adaptation Control . . . . .	117
4.1.6	Testbed Setup . . . . .	119
4.2	Rule-based Policy Recommendations . . . . .	121
4.2.1	Quantification of Cybersickness . . . . .	121
4.2.2	Risk and Cost Aware Trade-off Analysis . . . . .	123
4.2.3	Priority-based Queuing Model for Cybersickness Control . .	125
4.2.4	Rule-based semantics for Performance and Robustness . . .	128
4.2.5	Summary . . . . .	129
<b>5</b>	<b>Conclusion and Future Works . . . . .</b>	<b>130</b>
5.1	Summary of contributions . . . . .	131
5.2	Future Work . . . . .	132
	<b>BIBLIOGRAPHY . . . . .</b>	<b>133</b>
	<b>Glossary . . . . .</b>	<b>147</b>
	<b>VITA . . . . .</b>	<b>148</b>

## List of Tables

Table		Page
2.1	Mapping of threats on vSocial Server to SPS aspects . . . . .	18
2.2	Overview of the VRLE STRIDE Model . . . . .	21
2.3	Descriptions of leaf nodes in security and privacy attack trees. . . . .	28
2.4	$\lambda$ values for leaf nodes of security & privacy ATs. . . . .	30
2.5	Most vulnerable components considering the individual & combination of leaf nodes. . . . .	33
2.6	Survey questions asked during a CS experiment. . . . .	47
2.7	Description of leaf nodes in safety AT . . . . .	52
2.8	Description of the Intermediate nodes for safety AT . . . . .	52
2.9	Potential Fault-attacks in VRLE . . . . .	54
2.10	Attacker profiles for modeling different BAS. . . . .	63
2.11	Values of $\lambda$ given to the basic attack steps of the safety AFT . . . . .	65
2.12	$\lambda$ values for leaf nodes of security & privacy subtrees in safety AFT. . . . .	65
3.1	Survey questions asked during immersion experiment . . . . .	81
3.2	Attacker profiling with estimated level of impact. . . . .	89
3.3	Performance metrics of our anomaly detection method in terms of detection and classification of anomaly events. . . . .	96
3.4	List of statistical analysis system functions. . . . .	101
3.5	Bruteforce attack detection scenarios in terms of precision and recall metrics . . . . .	103



3.6	Calculation of suspiciousness scores based on multiple combination of attacks. . . . .	106
4.1	Potential adaptation choices for different 3QS anomalies . . . . .	115
4.2	Cost-aware application performance analysis of adaptations chosen for 3QS anomalies . . . . .	124
4.3	Performance metrics of our queuing model . . . . .	127
4.4	Recommendations based on risk level ( $R_l$ ), Cost level ( $C_l$ ), and control on cybersickness ( $\Delta CS$ ) . . . . .	127

## List of Figures

Figure	Page
1.1 Illustration of a cloud-based Social VRLE deployed across high-speed, low-latency network sites. . . . .	2
1.2 Sophisticated attacks in social VRLE . . . . .	4
1.3 Immersive attacks in social VRLE . . . . .	6
1.4 The interplay of the 3Q's : QoE, QoA, and QoS . . . . .	8
1.5 vSocial – Social VRLE application system setup . . . . .	9
1.6 A novel methodology for ensuring robustness and high performance in social VRLE applications . . . . .	10
2.1 Threat model representing formalization and classification of Security, Privacy and Safety threats originating in the vSocial VRLE server hosting the virtual reality content. . . . .	20
2.2 Unauthorized access to VRLE learning content. . . . .	21
2.3 Privacy attack on vSocial application. . . . .	21
2.4 An exemplar STA. . . . .	23
2.5 Proposed framework for security, privacy analysis in social VRLE. . . . .	24
2.6 Formalized security attack tree with threat scenarios disrupting LoI. . . . .	27
2.7 Formalized privacy attack tree with threat scenarios disrupting privacy leakage. . . . .	27
2.8 STA for root node of security AT. . . . .	28
2.9 STA for OR gate and root node of security attack tree. . . . .	29

2.10	TS of security AT where - <i>TS2</i> , <i>TS3</i> , <i>TS4</i> , <i>TS7</i> are the most vulnerable nodes. . . . .	31
2.11	TS of security AT where - <i>TS6*</i> , <i>TS7*</i> are the most vulnerable combination. . . . .	32
2.12	TS of privacy AT where - <i>PTS1</i> , <i>PTS3</i> , <i>PTS4</i> , <i>PTS9</i> are the most vulnerable nodes. . . . .	33
2.13	TS of privacy AT where - <i>PTS1*</i> is the most vulnerable combination. . . . .	34
2.14	Prob. in LoI reduced by 15.85% in security AT due to application of design principles. . . . .	35
2.15	Prob. of privacy leakage reduced by 68% in privacy AT due to application of design principles. . . . .	36
2.16	Prob. of LoI is reduced by 26% in security AT due to application of design principles for a combination of security attack tree nodes. . . . .	37
2.17	Prob. of privacy leakage reduced by 80% in privacy AT due to application of design principles for a combination of privacy AT nodes. . . . .	38
2.18	A before and after scenario showcasing the effect of DoS attack on vSocial causing a server crash. . . . .	43
2.19	vSocial system content affected due to insertion of malicious scripts in the student learning environment. . . . .	44
2.20	Packet Sniffing attack to disclose avatar and host server information. . . . .	45
2.21	A malicious user exposes the IP address of a valid VRLE user (disclosing user physical location) by gaining access to VRLE activity logs. . . . .	45
2.22	Immersion attack to tamper the chaperone file. . . . .	46
2.23	cybersickness virtual questionnaire for the users to respond after each activity in a vSocial experiment . . . . .	48
2.24	Activities and their associated virtual questionnaires used for the immersion survey experiment. . . . .	49

2.25	Results of CS-survey experiments. . . . .	50
2.26	Safety attack tree for VR application case study: vSocial . . . . .	51
2.27	Proposed framework for security and privacy analysis of a social VRLE. . . . .	51
2.28	Formalized attack fault tree with threat scenarios inducing cyber- sickness. . . . .	57
2.29	Formalized attack fault sub-tree with threat scenarios triggering a LoC and/or LoI issues. . . . .	57
2.30	Formalized attack fault sub-tree with threat scenarios triggering a LoA issue. . . . .	58
2.31	Formalized attack fault sub-tree with threat scenarios triggering a privacy leakage issue. . . . .	59
2.32	Framework for translation of attack trees into network of stochastic timed automata. . . . .	60
2.33	STA of cybersickness root node in safety AFT. . . . .	61
2.34	STA of OR gate and signal transition between the root node and the childnode for the LoA subtree in the safety-AFT. . . . .	61
2.35	STA of AND gate for the LOA sub-tree of the safety AFT. . . . .	62
2.36	STA for SAND gate in safety AFT. . . . .	62
2.37	STA for PAND gate in safety AFT. . . . .	63
2.38	STA for leaf node data tampering in LOA subtree of the safety AFT. . . . .	63
2.39	BAS for impersonation. . . . .	64
2.40	BAS for SQL injection. . . . .	65
2.41	TS of safety AFT where - $TS3$ , $TS7$ , $TS8$ are the most vulnerable nodes. . . . .	67
2.42	Combinations of TS of safety AFT where - $TS1$ , $TS4$ , $TS6$ are the most vulnerable nodes. . . . .	68
2.43	Probability of disruption of cybersickness for different attacker pro- files. . . . .	69

2.44	Probability of disruption of response time in safety attack tree for different attacker profiles. . . . .	70
2.45	Risk assessment of threats affecting cybresickness. . . . .	71
2.46	Pr reduced by 24.14% in safety AFT due to application of hardening design principle. . . . .	73
2.47	Pr reduced by 2.6% in safety AFT due to application of redundancy design principle. . . . .	74
2.48	Pr reduced by 28.96% in safety AFT due to application of hardening and principle of least privilege design principles. . . . .	75
2.49	Pr reduced by 25.52% in safety AFT due to application of hardening and redundancy design principles. . . . .	76
2.50	Pr reduced by 3.05% in safety AFT due to application of redundancy and principle of least privilege design principles. . . . .	77
2.51	Pr reduced by 35.18% in safety AFT due to application of hardening, redundancy and principle of least privilege design principles. . .	78
3.1	Formalized immersion attack tree with logical gates . . . . .	80
3.2	Results of UIX survey experiments. . . . .	83
3.3	Anomaly detection method to address SP attacks on VRLE(s). . . .	84
3.4	VRLE architecture implemented in an OpenCloud testbed setup. . .	87
3.5	Packet rate time series comparison for single attacks. . . . .	90
3.6	Five of the most distinguishing extracted features. . . . .	92
3.7	Single-label classifier accuracy comparison. . . . .	94
3.8	Single-label KNN classifier confusion matrix. . . . .	95
3.9	Multi-label K-NN classifier confusion matrix. . . . .	97
3.10	Brute-force password attempt Z-score distribution. . . . .	104
3.11	Relative change in UIX as the Suspiciousness scores increase for different type of SP breaches. . . . .	107

4.1	Proposed rule-based 3QS-adaption framework for a social VRLE system. . . . .	111
4.2	Detected 3QS related anomaly categories in the anomaly event monitoring tool . . . . .	112
4.3	Decision making of a suitable adaptation among the list of solution candidates . . . . .	113
4.4	DynamoDB table that show data collection in terms of cost, resource, solution and time . . . . .	117
4.5	Upgrading Instance Type Graph . . . . .	118
4.6	Enabling Enhanced Networking . . . . .	119
4.7	Overview of our Experimental Testbed Setup . . . . .	120
4.8	Anomaly Data Collection for Simulated Security, QoS, QoA scenarios along with the baseline data . . . . .	120
4.9	Avg. Latency measured (in ms) for QoA anomaly, QoS anomaly scenario in different adaptation scenarios . . . . .	121
4.10	Risk evaluation associated with the best (BA1, BA2), worst (WA1) and combination of adaptations in controlling cybersickness for the given QoA and QoS anomaly event . . . . .	123
4.11	Modelling stages of our proposed rule-based 3QS-adaptation framework as a queue . . . . .	125
5.1	Contributions for Research Goal 1: Harmonized Robustness and Performance by Design . . . . .	131
5.2	Contributions for Research Goal 2: Threat Intelligence Collection for Critical Anomaly Events . . . . .	131
5.3	Contributions for Research Goal 3: Rule-Based Approach for tuning Performance and Robustness for Resilience . . . . .	132

# ABSTRACT

Cloud-based applications that rely on emerging technologies such as social virtual reality are increasingly being deployed at high-scale in e.g., remote-learning, public safety, and healthcare. These applications increasingly need mechanisms to maintain robustness and immersive user experience as a joint consideration to minimize disruption in service availability due to cyber attacks/faults. Specifically, effective modeling and real-time adaptation approaches need to be investigated to ensure that the application functionality is resilient and does not induce undesired cybersickness levels. In this thesis, we investigate a novel ‘DevSecOps’ paradigm to jointly tune both the robustness and immersive performance factors in social virtual reality application design/operations. We characterize robustness factors considering Security, Privacy and Safety (SPS), and immersive performance factors considering Quality of Application, Quality of Service, and Quality of Experience (3Q). We achieve “harmonized security and performance by design” via modeling the SPS and 3Q factors in cloud-hosted applications using attack-fault trees (AFT) and an accurate quantitative analysis via formal verification techniques i.e., statistical model checking (SMC). We develop a real-time adaptive control capability to manage SPS/3Q issues affecting a critical anomaly event that induces undesired cybersickness. This control capability features a novel dynamic rule-based approach for closed-loop decision making augmented by a knowledge base for the SPS/3Q issues of individual and/or combination events. Correspondingly, we collect threat intelligence on application and network based cyber-attacks that disrupt immersiveness, and develop a multi-label K-NN classifier as well as statistical analysis techniques for critical anomaly event detection. We validate the effectiveness of our solution approach in a real-time cloud testbed featuring vSocial, a social virtual reality based learning environment that supports delivery of Social Competence Intervention (SCI) curriculum for youth. Based on our experiment findings, we show that our solution approach enables: (i) identifica-

tion of the most vulnerable components that impact user immersive experience to formally conduct risk assessment, (ii) dynamic decision making for controlling SPS/3Q issues inducing undesirable cybersickness levels via quantitative metrics of user feedback and effective anomaly detection, and (iii) rule-based policies following the NIST SP 800-160 principles and cloud-hosting recommendations for a more secure, privacy-preserving, and robust cloud-based application configuration with satisfactory immersive user experience.



# Chapter 1

## Introduction

### 1.1 Social Virtual Reality (VR) as a new paradigm

Virtual Reality (VR) extends the reality-virtual world continuum for providing immersive access to remote synthetic computer-generated environments [1]. Social VR is a new paradigm of collaboration systems that uses edge computing for novel application areas involving virtual reality learning environments (VRLE) for special education, surgical training, and flight simulators. Cloud applications integrate with social VR via Internet-of-Things (IoT) devices to create intelligent network services interconnecting smart edge devices, cloud/fog resources and for providing high quality of experience [2, 3] in a flexible personal immersive learning environment i.e. social VRLEs [4, 5]. In addition, such VRLEs allows one to be virtually present at a distant location in an immersive manner, and increases accessibility to remote learning materials, instruments and domain experts.

For instance, an exemplar social VRLE e.g., vSocial [4] shown in Figure 1.1 provides access to online VR-based content, web applications and session monitoring to enhance learning effectiveness of geographically dispersed VRLE students/instructors [6]. Networked VRLE components are controlled from a central cloud instance with administrative privileges for data collection/aggregation from dis-

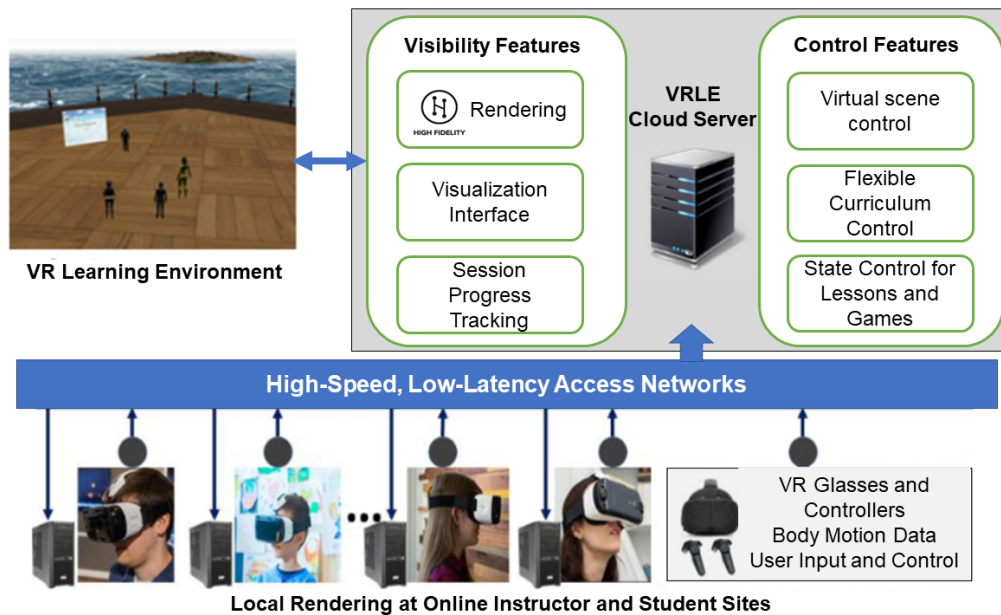


Figure 1.1: Illustration of a cloud-based Social VRLE deployed across high-speed, low-latency network sites.

tributed user/instructor locations. Another example is adopting social VR in the field of emerging services and analytics can enable the visualization techniques and 3D or multi-dimensional computing technologies to be more interactive and accurate [3]. In addition, Social VR hosting in cloud is a great capability for many applications since it provides the storage and computation needs for users with flexibility in cost and performance. With such facilitation of improving user experience, VR can be used: (i) for transforming designs of interactive systems (e.g. AR toolkit released by Apple) [7], (ii) for augmenting social networks with VR (e.g. Facebook own VR social platform named as Spaces) [8] and (iii) for enhancing the capabilities of a digital workplace, adding unprecedented spatial depth and realistic environmental detail (e.g. information and workspaces).

### 1.1.1 Novel Application Domains for Social VR

With the ongoing Covid-19 pandemic, social VR are widely spread into novel application domains as immersive learning environments (VRLEs) for remote learning purposes in education [9], surgical training, flight simulations. Nowadays, people

with mental health issues use therapy sessions via social VR [10]. Technology giants like facebook, mozilla, huwaei create new social VR applications as office workspace environments, social gathering spaces for human presence in the current pandemic. Moreover, instead of the video-conferencing tools (e.g.zoom) which lack in capturing non-verbal cues for effective social communication new social VR applications with a blend of video conferencing are being created. For instance, spatial VR meeting room application [11] details about having life-like realistic virtual avatars (users virtual model in VR) using 3D scans of their faces and an immersive environment where the meetings of a workspace can be conducted similar to a real world office. All such developments make the immersive social VR as the next generation social media application and services due to their impact on human activity.

### **1.1.2 Applications of Social VR as Learning Environments (VRLE)**

Due to the capability of stimulating users' immersive senses of virtual worlds, social VR in education are used as interactive learning environments mainly for remote learning purposes. In such learning environments which are computer generated with the use of data models and algorithms based on the current pose and other input stimuli especially in VR based gaming [12, 13]. Social VR in public safety is used for training personnel in the learning environments simulated for disaster scenarios. For instance, seismic simulations in VRLEs can help in saving more lives by identifying the critical moments that earthquake will happen and training the people accordingly can enable them to prepare for safety [3]. Moreover, manufacturing industries adopt VR as the medium for user communication mainly to analyze and interpret data in an immersive way. For this, they upload their VR software to the cloud for interactive data visualization services which provides a significant edge in the domains of service, training, research and development. With such remote monitoring and control through cloud applications

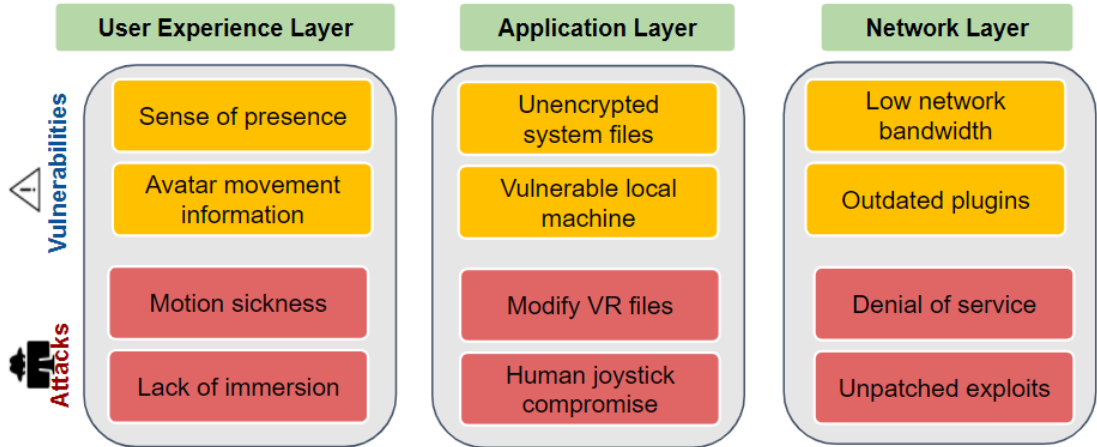


Figure 1.2: Sophisticated attacks in social VRLE

provided by VR can be great tools for communications in the manufacturing and automobile industry [5, 14].

## 1.2 Robustness and User Immersion Challenges

Social VR, has the potential to augment human performance through the integration of Internet-of-Things (IoT) devices (e.g. headsets, sensors, controllers). The content rendering, strategies for effective interaction scenarios are key requirements for the next generation Immersive social media powered by VR [15]. There is a need for more insights on the facilitation of user experience in terms of perception, sense of presence and cognition which are the basic immersion factors in social VR. However, there are certain challenges in terms of robustness issues and user immersive experience that needs to be addressed.

### 1.2.1 Inducing Factors of Cybersickness

Typical VR system applications comprise of interactions that require coordination of diverse user actions from multiple Internet-of-Things (IoT) devices, processing activity data and projecting visualization to achieve cooperative tasks. However, inter-connectivity of the VR devices at the network edge (i.e., VR headsets, emo-

tion recognition devices) and the VR content/monitoring in the cloud instance makes VRLEs vulnerable to both traditional cyber attacks [16] (e.g., unauthorized access, data tampering), as well as new kinds of cyber attack risks (e.g., occlusion attack, Chaperone attack) [17]. In addition, there is a lack of existing knowledge in terms of addressing critical security and privacy (SP) issues, as well as system/network faults that an attacker can use to negatively impact VRLE functionality [18]. Any disruption caused by an attacker on the instructor's VR content or administrator privileges will in turn impact user activities. There is a clear dearth of works on the *quantitative evaluation* for these security and privacy threats in the context of VRLE applications.

Social VRLEs that are not well-designed can be affected by undesirable effects due to e.g., physiological conditions, poor content organization, faults from hardware/software, or cyber attacks - while training students in sessions. The impact of such effects can specifically induce *cybersickness*, which is a set of unpleasant symptoms such as eyestrain, headache, nausea or even vomiting, thus compromising *user safety* in a VR session [19]. Prior work in [20] addresses both social and technical/security aspects to provide immersiveness in VR applications. There are even works that characterize user disruption due to discrepancies and overexposure in the socio-technical systems [21]. However, state-of-the-art in the design of VRLEs to critically ensure user safety against these anomaly events is not mature.

The work in [22] outlines the different threats and vulnerabilities in VR but does not provide any quantitative evaluation. Another notable recent work [17] also showed how cyber attacks (e.g., overlaying images in field of user vision, modifying VR environmental factors, human joystick compromise) can trigger a user safety issue. This issue can disorient users in immersive VR environments (e.g., force them to hit walls or physical objects) through security analysis of HTC Vive and Oculus Rift based systems. Further, detection of anomaly events that disrupt user safety in VRLEs requires use of causal event analysis techniques [23, 24, 25] that correlates both cyber and human/social component data points.

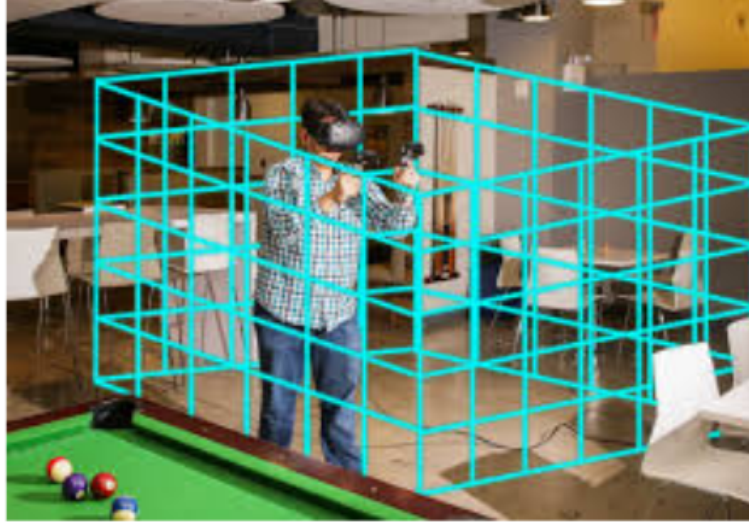


Figure 1.3: Immersive attacks in social VRLE

Thus we are motivated to explore the inter-relationship between SPS concerns for a harmonized robustness and performance social VRLE system design with reduced impact of cybersickness. *Failure to address these SPS issues in a social VRLE application may result in poor user experience (e.g., poor student engagement, disruption of learning/collaboration, reputation loss for the institution hosting the VRLE [26], and disruption of user safety (e.g., physical harm, eye strain, cybersickness).*

In this thesis, we characterize robustness factors as security, privacy and safety issues in social VRLE and user immersive performance issues as 3Q: Quality of Application, Quality of Service, Quality of Experience. For this, we define *security* as the robustness of the VR system against various attacks, *privacy* as the protection and secrecy over data sensitivity, and *safety* as the disruption in the system that compromises the user's overall well-being and a *fault-attack* as an abnormal condition or defect at the component, equipment, or sub-system level which may lead to a failure.

### 1.2.2 Disruption Factors of User Immersion

User in VR would feel amazed due to the immersiveness, but later on vast majority of issues occur when the user tries to interact with objects and get the same level of return for their efforts as they would from a usable mobile application. These issues are often quite severe to the point where the user is frustrated and confused and quickly loses the motivation to continue. Also, disruption in such application functionalities harm the user safety.

For providing immersive experience, VRLE applications need to render the content to multiple output devices. In addition, maintenance of seamless the user-system communication (i.e. IoT wearables) for accurate localization and tracking of the controllers needs to be considered. With ubiquity of IoT wearables (i.e. headsets and controllers) along with the demand for network and storage capabilities in VRLE creates a set of unique performance issues related to immersiveness.

For instance, perceived delay in visualization of user emotion data that is collected from the IoT sensors in the VRLE application can cause the disruption in monitoring the student performance thereby can compromise the learning outcomes. In other words, an attacker can take such network delay issues as vulnerabilities and use it to trigger a cyber attack (malicious packet drop) which further can disrupt the user in session.

Similarly, a network packet drop during the content transfer over the network can cause a degradation of the quality of the content (audio/video) which further disrupts the user experience in VRLE. Such scenarios occur possibly, when a VRLE session is going on. Based on that real-time applications, factors that can potentially affect include insufficient bandwidth, frame-rate of a content rendered, network quality, packet loss. To elucidate, in such an application scenario, user experience is more likely to be effected which can be due to bandwidth consumption or sufficient image quality factors. In addition to that, there should a strategy selection based on suitability for the issues (e.g., bandwidth conditions) should be emphasized. In this thesis we provide a characterization study of a remote learning

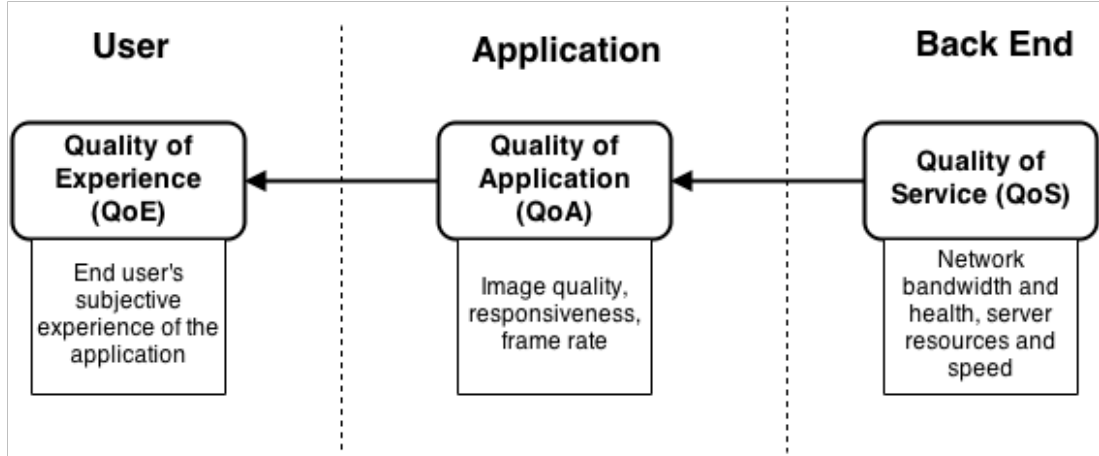


Figure 1.4: The interplay of the 3Q's : QoE, QoA, and QoS

application in a virtual reality infrastructure. To the best of our knowledge, this work is the first to characterize data processing latency issues affected by multiple factors, and investigates appropriate system architecture configurations to satisfy user QoE requirements in a social VRLE.

### 1.2.3 Social VRLE Application Case Study

In this thesis, our social VRLE case study will involve vSocial [4], which is developed for training youth with Autism Spectrum Disorder (ASD). The vSocial application is used for delivering a Social Competence Intervention (SCI) curriculum via a system as shown in Figure 1.5 that consists of modules such as: VR content rendering, web applications and classroom portal with instructional content hosted as web pages. Our multi-modal VRLE system as shown in Figure 1.5 uses the High Fidelity platform [27], and renders 3D visualizations based on the dynamic human computer interactions with an edge cloud i.e., vSocial Cloud Server. The vSocial Cloud Server shown in Figure 1.5 delivers the VRLE content to the users in these virtual classrooms and stores the user engagement information and activities in real-time. The vSocial server provides functionalities such as user access control, session content management, and student progress tracking. Given that the vSocial server is a critical system component, it is an



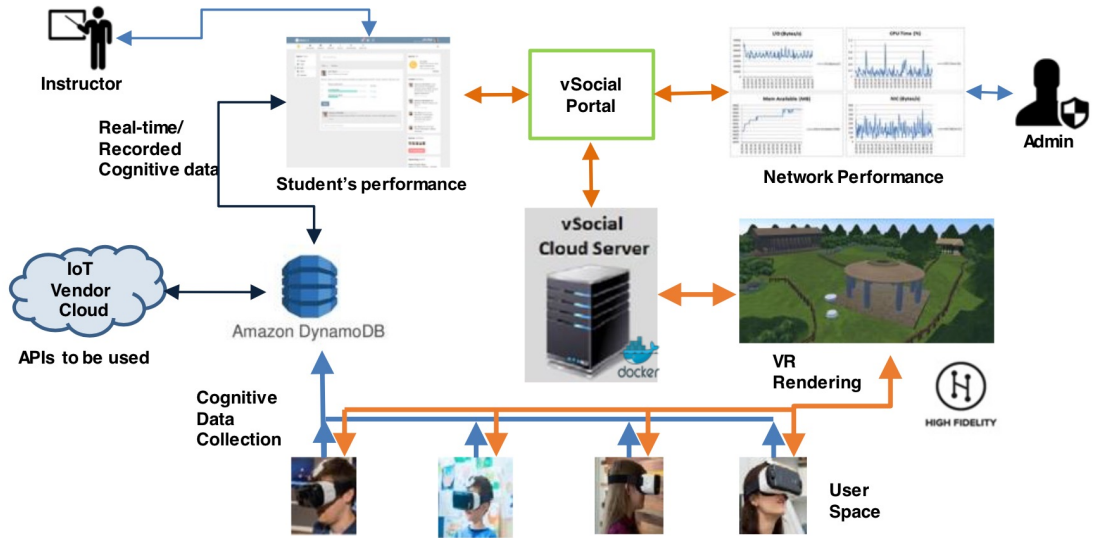


Figure 1.5: vSocial – Social VRLE application system setup

attractive target for an attacker. Our VRLE setup includes: VR headset devices (HTC Vive), hand-held controllers, and base stations for accurate localization and tracking of controllers [4].

### 1.3 Joint Tuning Approach of Robustness and Immersive Performance based on DevSecOps Paradigm

In this thesis, we propose a novel joint methodology that tunes both user immersive performance and robustness factors jointly such that the user experience is facilitated in terms of increased immersion and cybersickness control for a social VRLE application case study (i.e. vSocial). Our joint tuning methodology will feature “towards a harmonized robustness and performance by design” (i.e. Formal modeling and analysis) and an “adaptive control of performance and robustness” as shown in Figure 1.6 to enable immersive experience for VRLE users within future VRLE systems.

The main goal of our methodology is to control the cybersickness along with

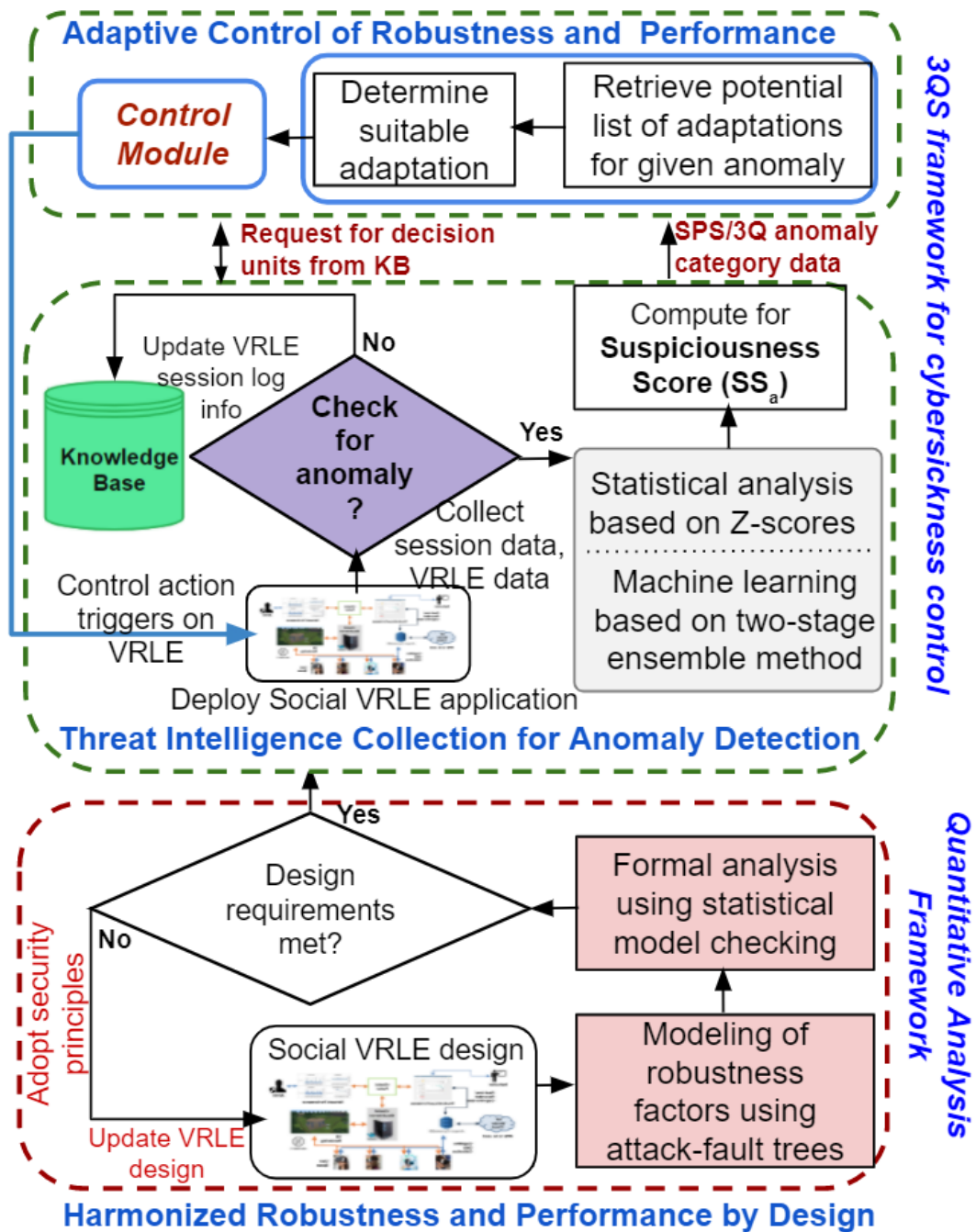


Figure 1.6: A novel methodology for ensuring robustness and high performance in social VRLE applications

maintaining high immersive experience for the users participating in VRLE sessions. The joint tuning approach requirements are motivated by the consequences of ill-suited VRLE design which makes it vulnerable to security breaches and privacy leakage attacks that can significantly impact sensitive users (e.g., special education students, patients). Till date, state-of-the-art in social VR focused on technology advancements, software, application and collaboration architectures with little attention to SPS concerns. In addition, there are no well-defined threat models for the design of trustworthy learning environments with a co-operative setting of geographically distributed users and IoT devices. Moreover, there is a knowledge gap that provides insights into how cyber-attacks influence the immersion factors that directly cause cybersickness.

Our joint tuning approach features will supplement the current social VR systems to facilitate immersive, collaborative user interactions and ensure the safety of the user. The solution approach in Figure 1.6 features anomaly detection module, control and decision making module for real-time adaptive control along with development of a harmonized VRLE design. This thesis addresses the challenges of joint tuning performance and robustness in social VRLEs that will advance user experience research related to online learning frameworks, cloud platforms, formal methods. In addition, my thesis outcome brings new insights to education/special education, public safety, telehealth application areas using social VR. In detail, the contributions of my thesis can be summarized as follows:

## 1. Harmonized Robustness Design and Performance

- *Theoretical contributions:* Explored about the inter-relationship of immersion and SPS issues inducing cybersickness (see Chapter 2). Specifically, my inter-relationship analysis is built upon Attack Fault Tree formalism (AFT) as novel threat models for social VRLEs. Using formal verification techniques (i.e. statistical model checking), we determine the probability of occurrence of events (i.e. inherent attack vectors and performance issues) leading to cybersickness and disruption of UIX in

current VR systems. To my knowledge, this is the first SPS and 3Q modeling technique for identifying cybersickness factors and provide formal guarantee on the VRLE system and user behavior.

- *Design Contributions:* For a trustworthy social VRLE design, we developed a quantitative framework that integrates relevant security principles following the National Institute of Standards and Technology (NIST) SP800-160 guidelines [28]. Such adoption of security principles is mainly for reduction of cybersickness caused due to either direct immersion attacks or inherent attack scenarios (i.e. consequences of cyber-attacks).

## 2. Threat Intelligence Collection for Critical anomaly event detection

- *Algorithm Contributions:* Novel ML-enabled algorithms for detection and collection of single and combination of attack/fault scenarios that can be used as threat intelligence collection via a mature knowledge base (See Chapter 3). To elucidate, we develop a multi-label K-NN classifier for detection of multi-attack/fault scenarios (i.e. combination/consequences of single network attacks) before the user experience gets disrupted (VRLE server crash occurrence). This is mainly, due to the lack of accuracy for classifying multi-attack using a single labeled KNN. For this reason, we extend the model to a multi-label KNN classifier to obtain the accurate anomaly detection. Moreover, for detection of attacks with sparse data (i.e. application attacks such as unauthorized access), developed a Z-score based statistical analysis for flagging malicious anomaly event behavior. With the experimental results, we compare the performance of each algorithm for a given anomaly event. In addition, novel attack detection metrics such as suspiciousness score for attack occurrence and user immersive experience (UIX) score are also used.

- *Design Contributions:* Novel anomaly detection module that captures the threat intelligence using novel quantitative metrics for cyber-attack occurrence (i.e. network-based and application-based attacks) into a Knowledge base for detecting future anomaly events in real-time before the user safety gets compromised.

### 3. Adaptive Control of Robustness and Performance for Resilience

- *Theoretical contributions:* Novel rule-based approach based on adaptive control for resilience in social VRLEs (See Chapter 4). My work is the first to investigate adaptations as attack/fault mitigation strategies with NIST guidelines to control cybersickness
- *Algorithm Contributions:* Dynamic Decision Making built on decision trees and their associated metrics for determining the suitability of the list of adaptation for a given anomaly event inducing undesired cybersickness. Novel Closed loop mechanism for determining the impact of the mitigation strategies for a given anomaly event. In addition, we perform characterization of cybersickness quantitative parameters for social VRLE application users.
- *Design Contributions:* Developed a novel 3QS framework that relies on decision metrics implemented in a VRLE application setting for real-time adaptations. In addition, performed modeling as a priority queue for the user-system events along with objective assessment of CS and UIX during VRLE session.

## 1.4 Thesis Organization

The remainder of thesis is organized as follows: In Chapter 2, we firstly introduce our quantitative framework for formal modeling and analysis of SPS and immersion issues. Then, we explain each component of our framework in details. Lastly,

we evaluate our framework for attack datasets; and provide the case studies to demonstrate the possible usage of NIST principles for cybersickness mitigation. In Chapter 3, we describe our anomaly detection module by developing a multi-label KNN model and statistical analysis technique to detect the cyber-attacks disrupting user experience. We also introduce novel assessment techniques for user immersive experience and attack occurrence metrics based on different attacker profiling. In Chapter 4, we introduce our adaptive control 3QS framework by proposing a rule-based dynamic decision making approach. Finally, we summarize this thesis in chapter 5.

# Chapter 2

## Towards Harmonizing Robustness and Performance by Design

### 2.1 Background and Related works

We present a summary of SPS issues in emerging technologies such as IoT as discussed in [29], which in particular, comprises of an exhaustive compilation of potential security and privacy threats and challenges. Another comprehensive study about this topic is presented in [30]. A typical VRLE application is susceptible to similar vulnerabilities along with human well-being, which has not been previously addressed in existing works. It is important to note that we are not exploring the trustworthiness, but rather focusing on the effect of VR usability due to potential security and privacy attacks.

In VRLE, the instructor plays a critical role by changing the pace of curriculum delivery and educational VR content in the virtual classroom based on continuous student evaluation (e.g., tokens, passes or strikes). The VRLE administrator also occupies an elevated position as he or she controls the user data of several sessions. Any disruption due to masquerading or spoofing by attacker with malicious intents [31] on the instructor's VR content or administrator privileges will compromise the learning activities in such a collaborative (virtual) space.

Authors in [32] discuss challenges in security and privacy for Augmented Reality (AR) applications and explore opportunities for securing AR systems without much discussion about the safety aspects. Although threats in AR and other IoT systems are also relevant in VR applications, VR threats differ from AR threats because of the complete user immersion in a virtual world. Multiple attacker models in [33] for threats in security and privacy for distributed IoT systems focus on threats such as DoS, physical damage, eavesdropping, etc. They suggest countermeasures but without evaluation studies. A survey in [34] classifies the security and privacy attacks in IoT systems, and discusses security issues in different layers i.e., in application, network, transport and perception.

Previous research has examined privacy issues in AR, fog and mobile computing. The AR browser in [35] examines in depth about the vulnerabilities and requirements for mobile devices without any significant evaluation of proposed approaches. Works about privacy attacks for fog computing [36] discuss issues such as trust and authentication, data storage, location and usage privacy. A thorough survey on mobile computing privacy threats [37] highlights the trade-off between functionality and privacy. Some major threats which are also applicable to our VRLE include: lack of transparency, tracking, leaks from (mobile) sensor, among others. They highlight the need to go beyond specific components such as network, hardware or application, and propose end-to-end solutions that consider system and data vulnerabilities [38]. An observation given the above state-of-art is that there are very few scholarly works on the quantitative evaluation for these security and privacy threats in the context of VR applications.

A seminal work on safety issues for virtual environments [39] established that human performance efficiency is affected by task and user characteristics. Existing works such as [40], compared a virtual environment in a display monitor with Head-Mounted Display (HMD) to establish its correlation with cybersickness. Most recently, works such as [41] highlight the problems about overexposure and cybersickness in VRLEs for training youth with ASD. Considering the exist-



ing SPS research in such VRLE applications, our work not only considers security and privacy, but also safety threats in a single comprehensive risk assessment framework.

### 2.1.1 Characterization of Robustness Factors in social VRLE

The focus of our work is to categorize threats based on three orthogonal aspects in VR applications: security, privacy and safety (SPS) which we consider as the robustness factors. We define *security* as the robustness of the VR system against various attacks, *privacy* as the protection and secrecy over data sensitivity, and *safety* as the disruption in the system that compromises the user's overall well-being.

To facilitate collaboration among students distributed over multiple locations, social VRLE applications such as vSocial [4] demand continuous and secured interoperability with entities (e.g., network-connected edge cloud) that requires large-scale capture, processing, and visualization of sensor data streams. These inherent challenges demand novel techniques for threat modeling of SPS factors in such complex multi-modal systems. In our proposed work, we provide threat formalization and risk assessment of the virtual reality learning environments using vSocial as a case study. We consider the potential threats pertaining to the vSocial server as the critical attack target, along with the respective SPS factors as shown in Fig. 2.1. We use these threat-to-system component mappings to design our attack trees, which ultimately quantify the risk score for each of the VRLE application system modules.

In the following paragraphs, various SPS attacks on server components are summarized:

**1) Security:** All VRLE systems are open to security attacks that can compromise integrity and performance. Session failure results from malicious activities carried out by attacker to crash VR sessions such as Denial of Service (DoS) attacks [33]. Threats such as Elevation of Privilege (EOP) provide attackers elevated

Table 2.1: Mapping of threats on vSocial Server to SPS aspects

Threat	Security	Privacy	Safety
Insertion of Malicious Code	✓		
Location Inference Attack		✓	✓
DDoS/DoS	✓		✓
Elevation of Privilege (EOP)	✓		
Tampering	✓	✓	
Eavesdropping	✓	✓	
Improper Disposal of User Data		✓	
Extended Sessions			✓
Shoulder Surfing		✓	
Session Failure	✓		✓
Informed Consent		✓	
Network Discrepancies	✓		✓
Bugs in VR			✓
Impersonation	✓	✓	

access to sessions and activities that enable them to modify system contents or add malicious files. A typical VRLE system collects large amounts of sensitive data, which are susceptible to manipulation through data tampering performed via unauthorized channels. Furthermore, data integrity can be compromised by insertion of malicious code to modify session entities or data, or even change system configuration or access policies. Also, network attacks, such as DoS or DDoS result in system crashes and data unavailability. Moreover, impersonation attacks can occur when impostors login with stolen credentials, and access sensitive user information.

**2) Privacy:** Threats to privacy impact VRLE data confidentiality. Attacks such as eavesdropping allow an attacker to access confidential information via packet sniffing. Also through shoulder surfing, attacker can gain access to user authorization information through screen or hand movement observation in VR sessions. Furthermore, data security breaches including tampering with static and transit data allows the attacker to gain access to user credentials and real-time data. Additionally, tampering can indirectly lead to other privacy threats; for instance, the manipulation of instructor information enable attackers to impersonate instructors

and lead courses, which is extremely harmful to students. This can be particularly detrimental to vSocial, as sensitive user data such as emotion recognition, student information and other personal information is constantly collected. Informed consent is also a concern if users are not notified about what data is being collected from them. Improper disposal of data can also compromise privacy due to deleted information still residing in the server, putting the users confidentiality at risk.

**3) Safety:** We consider safety threats to be factors that directly impact user well-being. For instance, session takeover allows an attacker to control the VR rendering, impacting user activity in a VR session. Moreover, any network discrepancy initiated by an attacker can cause sudden changes in VR rendering leading to user disorientation. Unintentional activities can also cause safety issues. For example, computer bugs can cause glitches within the VR rendering software, resulting in sudden differences in visuals inside the headset. Extended sessions can cause cybersickness [40] due to an individual being forced to stay in a VR session for an extended period of time.

### 2.1.2 Threat Modeling Approaches using STRIDE

A threat model is defined as a framework which details internal and external vulnerabilities, as well as objectives and countermeasures [42]. Threat models are utilized across various disciplines such as cloud computing [43], health records [44], and storage [45]. Threat model for a multi-modal system such as VRLE can provide a systematic analysis of possible threats and help identify any module that is highly vulnerable to attacks.

Rather than exploring threats in the entire vSocial system, we consider the vSocial server as a critical target (i.e., the trusted computing base) as it executes several functionalities such as: rendering controls, visualization, web applications, storage, as shown in Fig. 1. The session permissions refer to users' ability to access session resources. The storage deals with transit data, which is any data collected in real-time such as emotion data or network performance measurements, and static data,

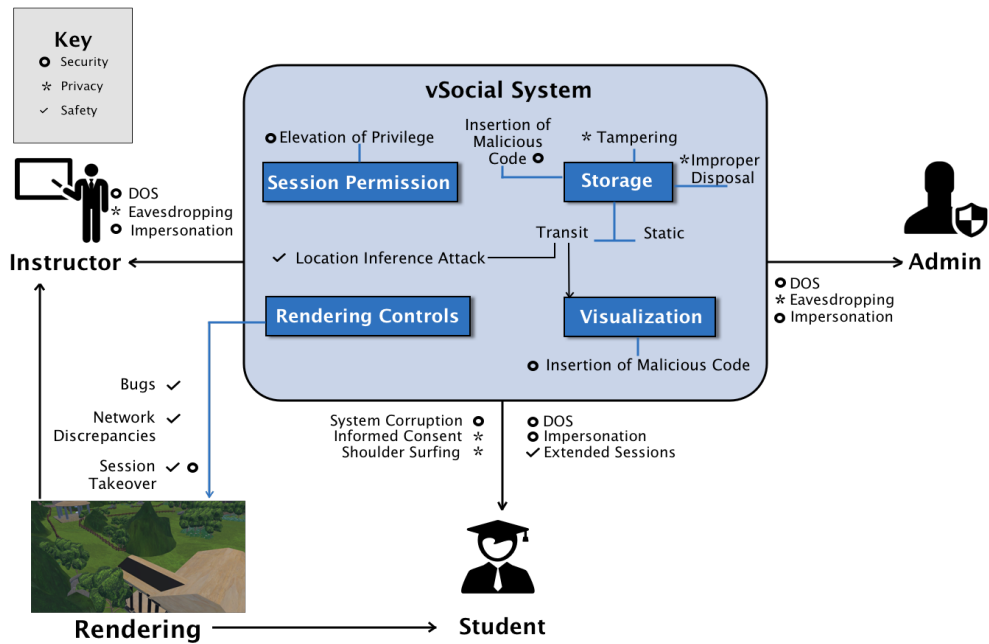


Figure 2.1: Threat model representing formalization and classification of Security, Privacy and Safety threats originating in the vSocial VRLE server hosting the virtual reality content.

which refers to user data and progress reports. The visualization functionality allows for display of real-time data so that the instructor or administrator can view it on a webportal. The rendering controls enable the instructor to invite students (other users) to join the VR class. A compromise in the security of these modules can leak confidential information or could compromise the integrity of the entire VRLE system.

### 2.1.3 Attacks in VRLE application use case:

The users in a social VRLE are networked and geographically distributed, which creates a series of potential attack scenarios. Using vSocial shown in Figure 1.5 as a social VRLE case study, herein we demonstrate exemplar security and privacy attacks that can affect the VRLE application sessions.

**Security Attacks:** An attacker can gain unauthorized access to either tamper any confidential information (user, network, VRLE components) by impersonating as a valid user, or disclose compromised confidential information. To elucidate, the instructional content in a vSocial application is in a web-enabled presentation

Table 2.2: Overview of the VRLE STRIDE Model

STRIDE Categories	Examples
Spoofing	An attacker can assume the identity of a student to track other users movement or disturb the user learning experience.
	An unauthorized access to the storage in VRLE will expose the stored visualized data to the attacker.
Tampering of Data	EEG data which is collected through muse headsets can be tampered if not transited on a secured network to storage or for visualization.
	Unauthorized access as an instructor can tamper the VRLE content, modules or change of different user permissions.
	Data retrieved from the storage can be tampered via SQL injection attack.
	Tampering in configuration or firmware of microcontrollers in VR device can disrupt user learning experience but can also, impact visual content in VRLE.
Reputation	An attacker can change the authorization/validation actions which is part of data collection/ visualization phases in VRLE and can log incorrect information.
	A malicious insider tampers the stored data (EEG data) and erases the logs recorded.
Information Disclosure	User emotion data at data collection/ visualization can be disclosed via sniffer attack or unauthorized access to the network.
	Data stored in the storage component can be disclosed using the logs information if it is not secured.
	Confidential data such as user ID, session information etc., can be disclosed if the data is not sent over a secured network.
	User avatar information can be disclosed by the attacker through packet spoofing which can eventually impact user VRLE experience.
Denial of Service	VRLE component muse which collects user emotion data would be unavailable due to a denial of service attack.
	Sending multiple Sync flood requests to VRLE server interrupts the data visualization resources/ access to storage.
	Loss of availability for VR rendering due to Denial of Service (DoS) for users.
	False user emotion data is displayed to the instructor due to intermittent network discrepancies (low bandwidth conditions).
Elevation of Privilege	A malicious user might elevate his or her privilege level to compromise the confidential user data in VRLE.
	An attacker impersonates as an instructor for higher user/ access rights than his/her original authorization levels (admin) in order to change user accounts, tamper the active directory.
	Unauthorized access to the instructor side can result in elevation of access to users. An attacker can give more access to users in VRLE which can result in a negative impact on learning outcome.



Figure 2.2: Unauthorized access to VRLE learning content.

format using the features present in High Fidelity. To guide the students through activities in the vSocial learning environment, the instructor will have privileged access to control the learning content settings such as e.g., editing the slides, and rewarding the students based on their performance. Gaining *Unauthorized access* to the instructor account as shown in Figure 2.2 can lead to *disclosure of user information*, and tampering of the learning content in vSocial to negatively impact the users' (students') learning experience.

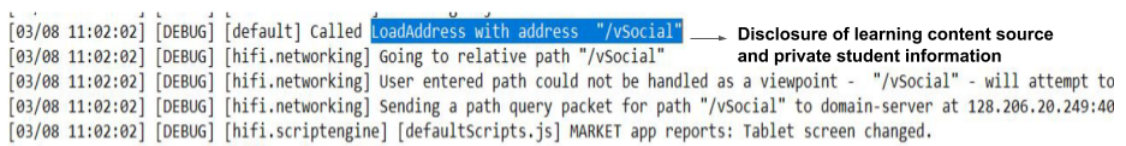


Figure 2.3: Privacy attack on vSocial application.

**Privacy Attacks:** A user privacy breach can involve an intruder entering a VRLE world with fake credentials to snoop into the virtual classroom conversations. The attacker can then disrupt an ongoing VRLE session by obstructing the view of the users in their learning sessions and can even disorient the content. Disorientation can possibly lead to a user running into a wall and getting physically hurt. Privacy attacks can also involve *packet tampering* that was demonstrated in [46], where an attacker performs illegal packet capture in order to extract sensitive information (*packet sniffing attack*). A potential privacy breach can occur when the attacker *discloses the confidential information* obtained from the captured packets as shown in Figure 2.3. Using this packet sniffing attack, the avatar (user virtual information) can also be disclosed with private location information and student credentials.

#### 2.1.4 Statistical Model Checking

Statistical model checking (SMC) is a variation of the well-known classical model checking [47] approach for a system that exhibits stochastic behavior. The SMC approach to solve the model checking problem involves simulating (Monte Carlo simulation) the system for finitely many runs, and using hypothesis testing to infer whether the samples provide a statistical evidence for the satisfaction or violation of the specification [48].

**Stochastic timed automata:** Stochastic timed automata (STA) is an extended version of timed automata (TA) with stochastic semantics. A STA associates logical locations with continuous, generally distributed sojourn times [49]. In STA, constraints on edges and invariants on locations, such as clocks are used to enable transition from one state to another [50].

**Definition 1** (Stochastic timed automata). *Given a timed automata which is equipped with assignment of invariants  $\mathcal{I}$  to locations  $\mathcal{L}$ , we formulate an STA as a tuple  $T = \langle \mathcal{L}, l_{init}, \Sigma, \mathcal{X}, \mathcal{E}, \mathcal{I}, \mu \rangle$ , where  $\mathcal{L}$  is a finite set of locations,  $l_{init} \in \mathcal{L}$  is the initial location,  $\Sigma$  is a finite set of actions,  $\mathcal{X}$  is the finite set of clocks,*

$\mathcal{E} \subseteq \mathcal{L} \times \mathcal{L}_{clk} \times \Sigma \times 2^x$  is a finite set of edges, with  $\mathcal{L}_{clk}$  representing the set of clock constraints,  $\mathcal{I}: \mathcal{L} \rightarrow \lambda$  is the invariant where  $\lambda$  is the rate of exponential assigned to the locations  $\mathcal{L}$ ,  $\mu$  is the probability density function ( $\mu_l$ ) at a location  $l \in \mathcal{L}$ .

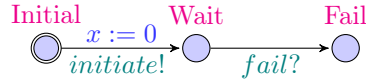


Figure 2.4: An exemplar STA.

An exemplar STA is shown in Figure 2.4 that consists of the locations  $\{\text{Initial}, \text{Wait}, \text{Fail}\}$ . Herein, the *Initial* location represents the start of execution of an STA and a *clock*  $x$  is used to keep track of the global time. The communication in an STA exists between its components using message broadcast signals in a bottom-up approach. The STA is activated by broadcasting *initiate!* signal, which transitions to wait location and waits for the *fail* signal. In an STA, time delays are governed as probability distributions (used as invariants) over the locations. The Network of Stochastic Timed Automata (NSTA) is defined by composing all component automaton to obtain a complete stochastic system satisfying the general compositionality criterion of TA transition rules [49, 51].

**UPPAAL SMC:** UPPAAL SMC is an integrated tool for modeling, validation, and verification of real-time systems modeled as a network of stochastic timed automata (NSTA) extended with integer variable, invariant, and channel synchronizations [51]. In SMC, the probability estimate is derived using an estimation algorithm and statistical parameters, such as  $1 - \alpha$  (required confidence interval) and  $\epsilon$  (error bound) [52]. For instance, if we indicate goal state in the STA of *Top\_event* as *Fail*, then the probability of a successful occurrence within time  $t$  can be written as:  $Pr[x \leq t](\langle \rangle \text{Top\_event.Fail})$  where,  $\langle \rangle$  represents the existential operator ( $\diamond$ ) and  $x$  is a clock in the STA to track the global time.

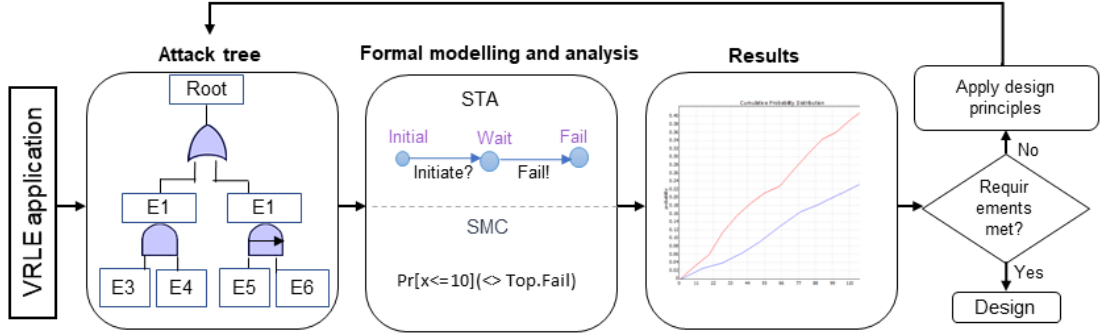


Figure 2.5: Proposed framework for security, privacy analysis in social VRLE.

## 2.1.5 Design principles

To build a trustworthy VRLE system architecture which ensures security and privacy, integration of design principles in the life cycle of edge computing interconnected and distributed IoT device based systems is essential [53]. We adapt the following three design principles from NIST SP800-160 [28, 53] such as: (i) *Hardening* – defined as reinforcement of individual or types of components to ensure that they are harder to compromise or impair, (ii) *Diversity* – defined as the implementation of a feature with diverse types of components to restrict the threat impact from proliferating further into the system, and (iii) *Principle of least privilege* – defined as limiting the privileges of any entity, that is just enough to perform its functions and prevents the effect of threat from propagating beyond the affected component.

## 2.2 Quantitative Framework for Security, Privacy Issues

In this section, we present details of our proposed framework that uses preliminary results from section 2.1.3 for security and privacy analysis (i.e., to identify the most vulnerable attacks) of social VRLE applications. An overview of the steps followed in our framework is shown in Figure 2.27. Firstly, we outline the threat scenarios in a social VRLE application [4] using traditional approaches. Secondly, we use an attack tree formalism for the modeling procedures. Following this, each attack



tree is translated into an equivalent STA to form an NSTA, which is input into the UPPAAL SMC tool. Lastly, we use the quantitative assessment from the tool to determine if the probability of disruption is higher than a set threshold (user specified requirements). Based on this determination, we subsequently prescribe the design principles such as: *hardening*, *diversity* and *principle of least privilege* that can be adopted in VRLE deployments. Overall, our framework steps help us to investigate potential security, privacy attack scenarios and recommend the design alternatives based on design principles for securing an edge computing based VRLE application.

### 2.2.1 Formal Modeling of Security and Privacy using Attack Tree (AT) Formalism

We formalize the outlined security, privacy threat scenarios from stride model described above using attack tree formalisms. Attack trees are hierarchical models that show how an attacker goal (root node) can be refined into smaller sub-goals (child/intermediate nodes) via gates until no further refinement is possible such that the basic attack steps (BAS) are reached. BAS represents the *leaf nodes* of an attack tree [54]. The *leaf nodes* and the *gates* connected in the attack trees are termed as *attack tree elements*. To explore dependencies in attack surfaces, attack trees enable sharing of subtrees. Hence, attack trees are often considered as directed acyclic graphs, rather than trees [50].

There are several tools to develop attack trees such as ADTool, SecurITree [55], among others which formalize attack trees that can be used for risk assessment. SecurITree specifically shows its flexibility by considering several input parameters such as: rates, weights, and counter measure nodes at every level of the tree. We also can perform risk assessment by using Frequency rates and duration. Frequency rates refer to the number of times an attack occurs in a specific time period, and the duration is the timespan of the attack on the VRLE system. Other functionalities such as counter measure node could also be considered within a

VRLE application setup context. We use the attack trees concept from [56] and derive graphical models that provide a systematic representation of various attack scenarios.

**Definition 2** (Attack trees). *An attack tree  $A$  is defined as a tuple  $\{N, Child, Top\_event, l\} \cup \{AT\_elements\}$  where,  $N$  is a finite set of nodes in the attack tree;  $Child: N \rightarrow N^*$  maps each set of nodes to its child nodes;  $Top\_event$  is an unique goal node of the attacker where  $Top\_event \in N$ ;  $l$ : is a set of labels for each node  $n \in N$ ; and  $AT\_elements$ : is a set of elements in an attack tree  $A$ .*

**Attack tree elements:** Attack tree elements aid in generating an attack tree and are defined as a set of  $\{G \cup L\}$  where,  $G$  represents gates;  $L$  represents leaf nodes. Following are the descriptions of each of the AT elements.

**Attack tree gates:** Given an attack tree  $A$ , we formally define the attack tree gates  $G = \{OR, AND, SAND\}$ .<sup>1</sup> An AND gate is disrupted when all its child nodes are disrupted, whereas an OR gate is disrupted if either of its child nodes are disrupted. Similarly, SAND gate is disrupted when all its child nodes are disrupted from left to right using the condition that the success of a previous step determines the success of the upcoming child node. The output nodes of the gates using these gates  $G$  in an attack tree  $A$  are defined as *Intermediate nodes (I)*, which will be located at a level that is greater than the leaf nodes.

**Attack tree leaves:** An attack tree *leaf node* is the terminal node with no other child node(s). It can be associated with *basic attack steps (BAS)*, which collectively represent all the individual atomic steps within a composite attack scenario. To elucidate, for an attacker to perform intrusion, the prospective BAS can include: (i) identity spoofing, and (ii) unauthorized access to the system depending on the attacker profile. Thus, every BAS appears as an implicit leaf node of the attack tree. We assume the attack duration to have an exponential rate and model the equation as :  $P(t) = 1 - e^{-\lambda t}$  where,  $\lambda$  is the rate of exponential distribution. We

---

<sup>1</sup>We limit our modeling to these three gates, however attack trees can adopt any other gates from the static/dynamic fault trees.

use this exponential distribution because of its tractability and ease of handling, and also because it is defined by a single parameter.

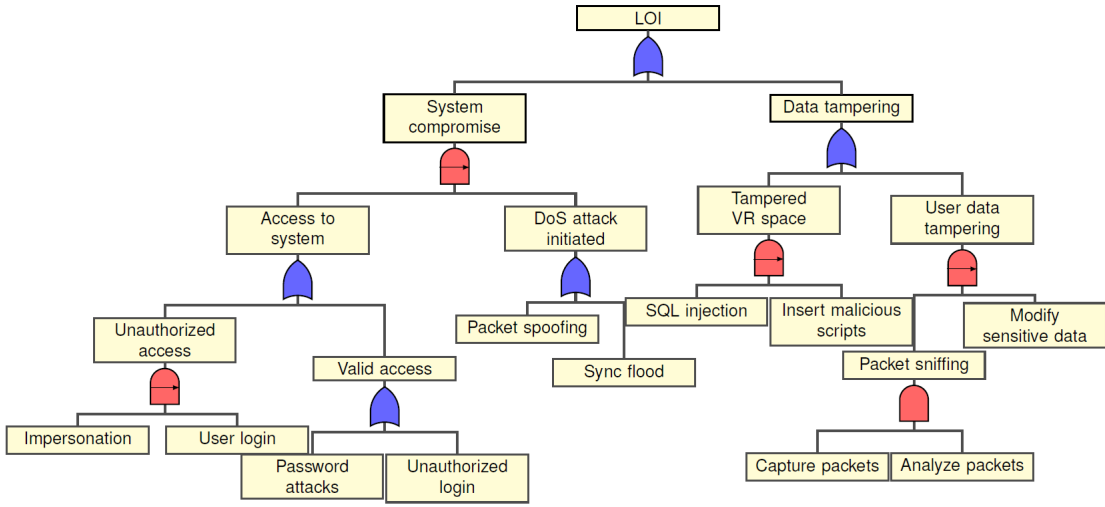


Figure 2.6: Formalized security attack tree with threat scenarios disrupting LoI.

## 2.2.2 Formal Description of Novel ATs for Security and Privacy issues in VRLE

Using experimental evidence from our work [26], we model threat scenarios in the form of a security attack tree and privacy attack tree as shown in Figures 2.6 and 2.31, respectively. The descriptions of the leaf nodes are listed in Table 2.3. Each of the threat scenarios relevant to Loss of Integrity (LoI) issue are outlined in security AT and privacy leakage issues in the privacy AT.

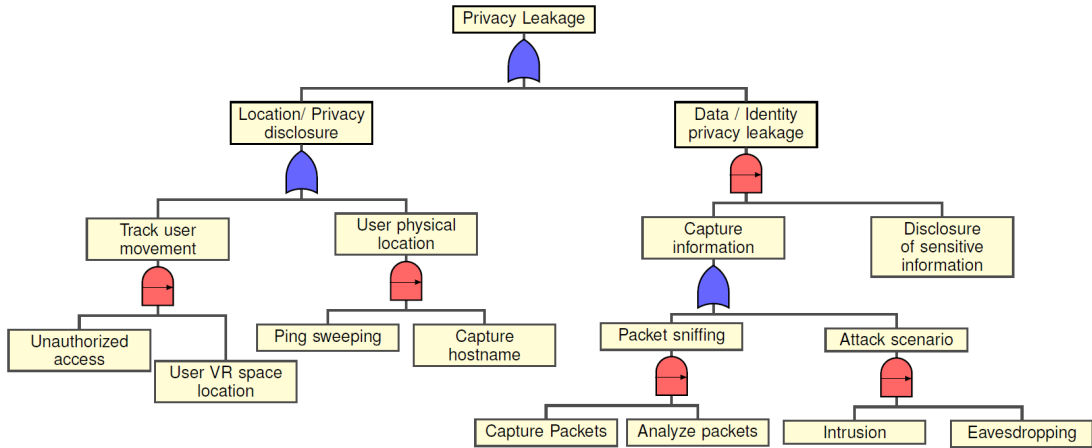


Figure 2.7: Formalized privacy attack tree with threat scenarios disrupting privacy leakage.

Table 2.3: Descriptions of leaf nodes in security and privacy attack trees.

Security Attack Tree		Privacy Attack Tree	
Leaf Node Components	Description of Leaf Nodes	Leaf Node Components	Description of Leaf Nodes
Impersonation	Attacker successfully assumes the identity of a valid user	Unauthorized Access	Attacker gains access to VR space
Packet Spoofing	Spoofing packets from a fake IP address to impersonate	User VR space location	Attacker determines the user location in VR space
Sync Flood	Sends sync request to a target and direct server resources away from legitimate traffic	Ping sweeping	Attacker sends pings to a range of IP addresses and identify active hosts
SQL Injection	Attacker injects malicious commands in user i/p query using GET and POST	Capture packets	Attacker uses packet sniffer to capture packet information
Insert Malicious Scripts	Attacker successfully adds malicious scripts in VR	Analyze packets	To identify erroneous packets to tamper
Capture Packets	The attacker uses a packet sniffer to capture packet information	Intrusion	Attacker performs an unauthorized activity on VR space
Analyze Packets	Attacker identifies erroneous packets to tamper	Eavesdropping	Attacker listens to conversations in VR space
Modify Sensitive Data	To modify any sensitive information by eavesdropping	Disclosure of sensitive information	Attacker maliciously releases any captured sensitive data
User Login	User login into VRLE	Capture hostname	With IP address obtained, attacker can capture the hostname in the VRLE application
Unauthorized Login	Attacker gains access into VRLE by unauthorized means		
Password Attacks	Attacker recovers password of a valid-user		

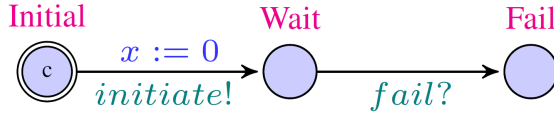


Figure 2.8: STA for root node of security AT.

Moreover, the listed attack trees are useful when they are concerned with user privacy and safety in a VRLE system. To elucidate, critical VRLE applications such as flight simulations, military training exercises and vSocial (developed for children with ASD) are sensitive to information disclosure attacks that can cause significant disruption for the stakeholder participants. If an attacker compromises such sensitive information, it can be used to harm the participants in the form of e.g., a chaperone attack (to make a user run into walls) and other physical safety attacks detailed in [38].

### 2.2.3 Translation of ATs into Stochastic Timed Automata

In this section, we generate STA from the corresponding security and privacy attack trees shown in Figures 2.6 and 2.31. In our translational approach: (i) each of the leaf nodes in these attack trees is converted into an individual STA. The intermediate events, which are basically the output of the logic gates used at different levels are converted imperatively into STA; (ii) the generated STAs are composed in parallel by including the root node; (iii) the obtained NSTA is then used for statistical model checking in order to verify the security and privacy properties formalized as SMC queries.

To demonstrate the translation of an attack tree into an STA, we consider the

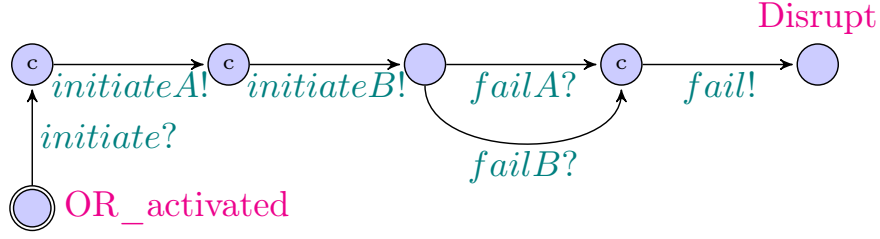


Figure 2.9: STA for OR gate and root node of security attack tree.

security attack tree as shown in Figure 2.6. As part of the translation, each of the security AT element (leaf and gates) input signals are connected to the output signal of child nodes. The generated network of STA communicates using signals. *initiate* - indicates activation signal of attack tree element. This signal is sent initially from the root node to its children. *fail* - indicates disruption of that attack tree element. This signal is sent to the parent node from its child node to indicate an STA disruption. The scope of the above signals can also be extended by special symbols such as: i) '?' (e.g., *initiate?*) means that the event will wait for the reception of the intended signal, ii) '!' (e.g., *initiate!*) implies output signal broadcasts to other STA in the attack tree.

**Illustrative example:** For instance, we show the conversion of LoI i.e., root node (*Top\_event*) into STA as shown in Figure 2.33. The converted STA of the LoI is equipped with *initiate!* and *fail?* signals. The root node is the OR gate output for the two child nodes: (A) “System Compromise” and (B) “Data Tampering”. Top OR gate sends an initiate signal and activates its child nodes “System compromise” and “Data Tampering” as shown in Figure 2.9 by broadcasting *initiateA!* and *initiateB!* signals. After initialization, if either of the nodes (A) OR (B) are disrupted, then a *fail!* signal is sent to the *Top\_event*, which forces a transition to *Disrupt* state, representing LoI in the system. The clock  $x$  is a UPPAAL global variable to keep track of the time progression as mentioned in Section Preliminaries. Similarly, STAs for the AND gate, SAND gates and the leaf nodes are also developed. Moreover, STAs for leaf nodes such as *impersonation* in the security attack tree are instantiated with  $\lambda$  (rate of exponential) values. For the given  $\lambda$  values to the leaf nodes, the probability of occurrence is calculated.

Table 2.4:  $\lambda$  values for leaf nodes of security & privacy ATs.

Security AT		Privacy AT	
Security threats	$\lambda$	Privacy threats	$\lambda$
Impersonation	0.006892	Unauthorized access	0.006478
User login	0.0089	User VR space location	0.0094
Password attacks	0.008687	Capture hostname	0.004162
Unauthorized login	0.008687	Ping sweeping	0.002162
Packet spoofing	0.0068	Capture packets	0.00098
SYNC flood	0.0068	Analyze packets	0.0048
SQL injection	0.00231788	Disclosure of sensitive info	0.0009298
Insert malicious scripts	0.008	Intrusion	0.006628
Capture packets	0.000 98	Eavesdropping	0.08
Analyze packets	0.0048	–	–
Modify sensitive data	0.002642	–	–

This value then propagates upward in the tree to calculate the probability of LoI. As mentioned earlier, the developed STAs are composed using the parallel composition [49] technique to form an NSTA, which is then used for SMC by the UPPAAL tool [51].

## 2.3 Quantitative Analysis of ATs and Evaluation of Design Principles

In this section, we present the results obtained using our proposed framework. As mentioned in section 2.2, the threat scenarios we consider are: *LoI* and *privacy leakage* for security and privacy attack trees (AT), respectively. In the following analysis, we assume that our design requirement is to keep the probability of LoI and privacy leakage below the threshold of 0.25. For evaluation purposes, we use arbitrary values of  $\lambda$  as parametric input to the leaf nodes as shown in Table 2.12 obtained from [57], [58]. Note, after providing  $\lambda$  values as parameters to the leaf nodes, we utilize the SMC queries as explained in Section to find the respective probabilities of LoI and privacy leakage. Any other user specified threshold values can also be used in our framework. This is due to the fact that the model checking approach takes the user specified values at the beginning of an experiment. For our

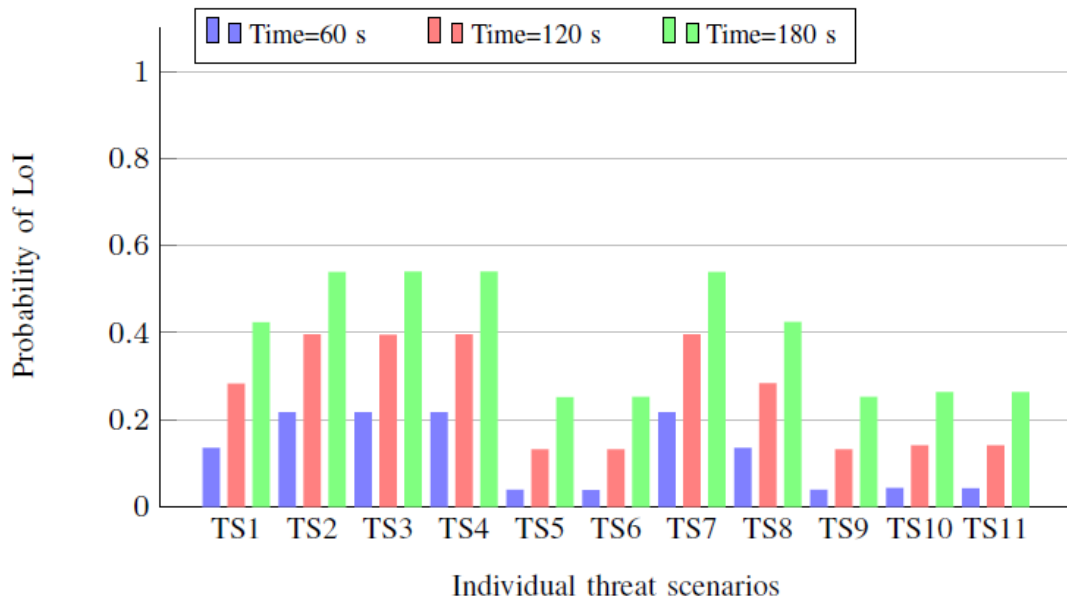


Figure 2.10: TS of security AT where -  $TS2$ ,  $TS3$ ,  $TS4$ ,  $TS7$  are the most vulnerable nodes.

experiment purposes, we consider LoI (security attack tree) and privacy leakage (privacy attack tree) as goal nodes. In the following set of experiments, we present the obtained probability of the goal nodes with respect to the time window used by the attacker.

### 2.3.1 Vulnerability Analysis for Security AT

We assign the values of  $\lambda$  shown in Table 2.12. However, when assigning a  $\lambda$  value to a leaf node in the attack tree, we consider a very small positive constant ( $K$ )  $\approx 0.002$  for the remaining leaf nodes. This is because, in real time systems, multiple attack scenarios can happen.

To identify a vulnerability in a security attack tree, we analyze: (i) individual leaf nodes, and (ii) combinations of leaf nodes, to determine their effect on the probability of LoI occurrence.

*i) Individual leaf node analysis:* In Figure 2.10, we show the probability of LoI over multiple time windows for each leaf node in the security attack tree. We perform a thorough analysis of leaf nodes in the security attack tree for threat scenarios across different time intervals i.e.,  $t = \{0, 60, 120, 180\}$ . For the individual

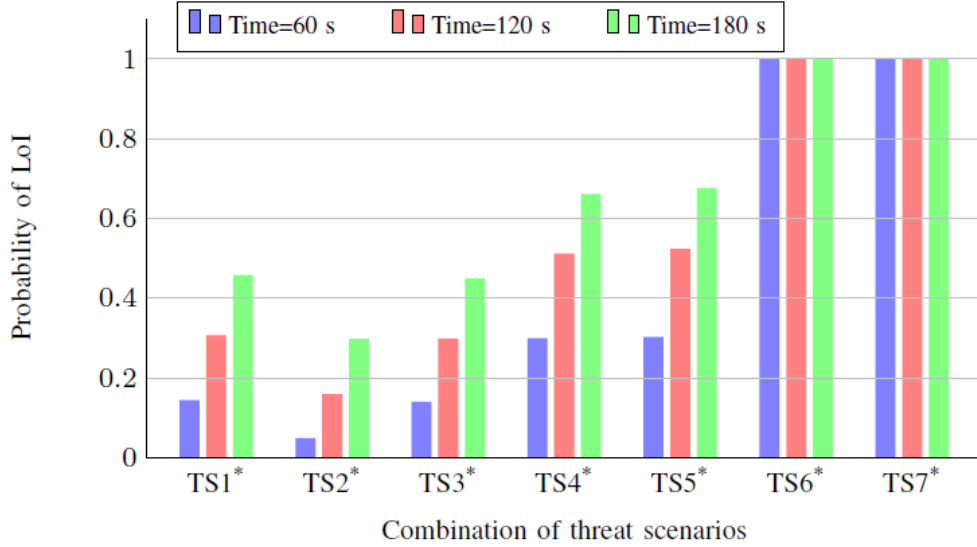


Figure 2.11: TS of security AT where –  $TS6^*$ ,  $TS7^*$  are the most vulnerable combination.

leaf node analysis, the considered threat scenarios (TS) shown in Figure 2.10 are termed as:  $TS1$  – insert malicious scripts,  $TS2$  – packet spoofing,  $TS3$  – unauthorized login,  $TS4$  – password attacks,  $TS5$  – modify data,  $TS6$  – analyze packets,  $TS7$  – Sync flood,  $TS8$  – SQL injection,  $TS9$  – capture packets,  $TS10$  – impersonation,  $TS11$  – user login. As shown in Figure 2.10, the leaf nodes  $TS3$  and  $TS4$  (for unauthorized access) as well as  $TS2$  and  $TS7$  (for DoS attack) are the most vulnerable in the security attack tree with the probability of 0.53.

*ii) Analysis using combination of leaf nodes:* Herein, we consider combinations of leaf nodes to identify their impact on LoI. For these experiments, we explore two scenarios: In the first scenario, we consider combinations of leaf nodes that belong to the same sub-tree, and in the second scenario, we consider leaf nodes from different sub-trees. The considered combination of threat scenarios are enlisted as:  $TS1^*$  – {impersonation, SQL injection},  $TS2^*$  – {impersonation, modify data},  $TS3^*$  – {SQL injection, capture packets},  $TS4^*$  – {pwd attacks, SQL injection},  $TS5^*$  – {impersonation, packet spoofing},  $TS6^*$  – {packet spoofing, unauthorized login},  $TS7^*$  – {unauthorized login, Sync flood}. As shown in Figure 2.11,  $TS6^*$  and  $TS7^*$  are the most vulnerable combination of threat scenarios with a probability of 1 for an LoI event. As part of further analysis in Section



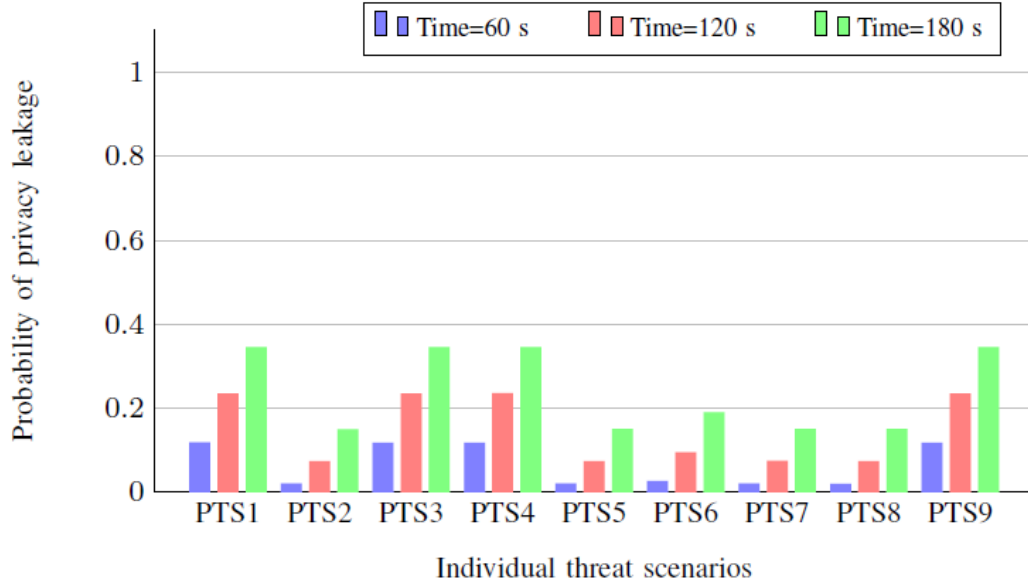


Figure 2.12: TS of privacy AT where - *PTS1*, *PTS3*, *PTS4*, *PTS9* are the most vulnerable nodes.

Table 2.5: Most vulnerable components considering the individual & combination of leaf nodes.

Level in attack trees	Analysis on security AT		Analysis on privacy AT	
	Different Scenarios	Identified vulnerable components in security AT	Different Scenarios	Identified vulnerable components in privacy AT
Individual leaf nodes	Leaf nodes where probability of disruption in LoI at ( $t \leq 180$ ) = 0.53	Unauthorized login	Leaf nodes where probability of disruption in privacy leakage at ( $t \leq 180$ ) = 0.34	Unauthorized access
		Packet spoofing		User VR space location
		Sync flood		Ping sweeping
Combination of leaf nodes	Leaf nodes where probability of disruption in LoI at ( $t \leq 180$ ) = 1	{Unauthorized login, Packet spoofing},	Leaf nodes where probability of disruption in privacy leakage at ( $t \leq 180$ ) = 1	{Unauthorized access, user VR space location}
		{Unauthorized login, Sync flood}		

Design-principles, we discuss about the potential candidates for design principles to apply on these leaf nodes such that the VRLE application resilience against security threats is enhanced.

### 2.3.2 Vulnerability Analysis for Privacy AT

We analyze the privacy attack tree similarly for: (i) individual leaf nodes, and (ii) combinations of leaf nodes. For the considered individual leaf node analysis in the privacy attack tree, the threat scenarios are termed as: *PTS1* – unauthorized access, *PTS2* – capture packets, *PTS3* – user VR space location, *PTS4* – ping sweeping, *PTS5* – analyze packets, *PTS6* – disclosure of sensitive information,

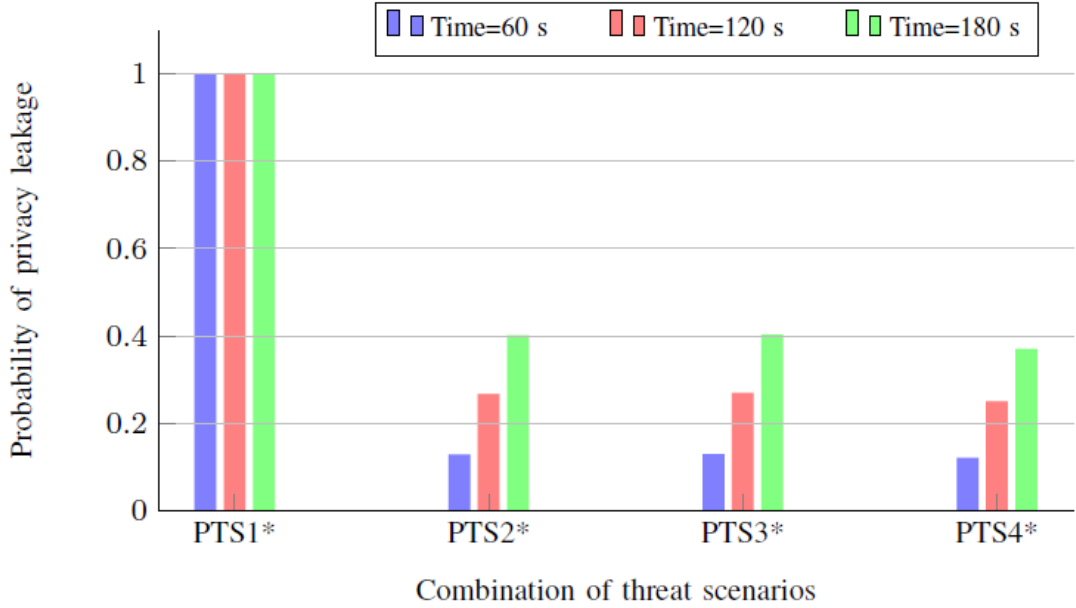


Figure 2.13: TS of privacy AT where –  $PTS1^*$  is the most vulnerable combination.  $PTS7$  – intrusion,  $PTS8$  – eavesdropping,  $PTS9$  – capture hostname. As shown in Figure 2.12, the most vulnerable leaf nodes are:  $PTS1$ ,  $PTS3$ ,  $PTS4$ ,  $PTS9$  with the highest probability of privacy leakage of 0.34.

For the analysis of combination of leaf nodes, we refer to the combination of threat scenarios as:  $PTS1^*$  – {unauthorized access, user VR space location},  $PTS2^*$  – {capture packets, disclosure of sensitive information},  $PTS3^*$  – {unauthorized access, disclosure of sensitive information},  $PTS4^*$  – {capture packets, analyze packets}. As shown in Figure 2.13,  $PTS1^*$  is the most vulnerable combination of threats for privacy leakage with a probability of 1. In summary, we can conclude that the above numerical analysis shown in Table 2.5 on both security and privacy attack trees can help in identifying the LoI and privacy leakage concerns that need to be addressed in the social VRLE design.

### 2.3.3 Recommended Design Principles

In this section, we are examining the effect of applying various design principles to the most vulnerable components identified in the Sections 2.3.1, and 2.3.2. Existing works such as NIST SP800-160 document [28], [53] suggest that the services

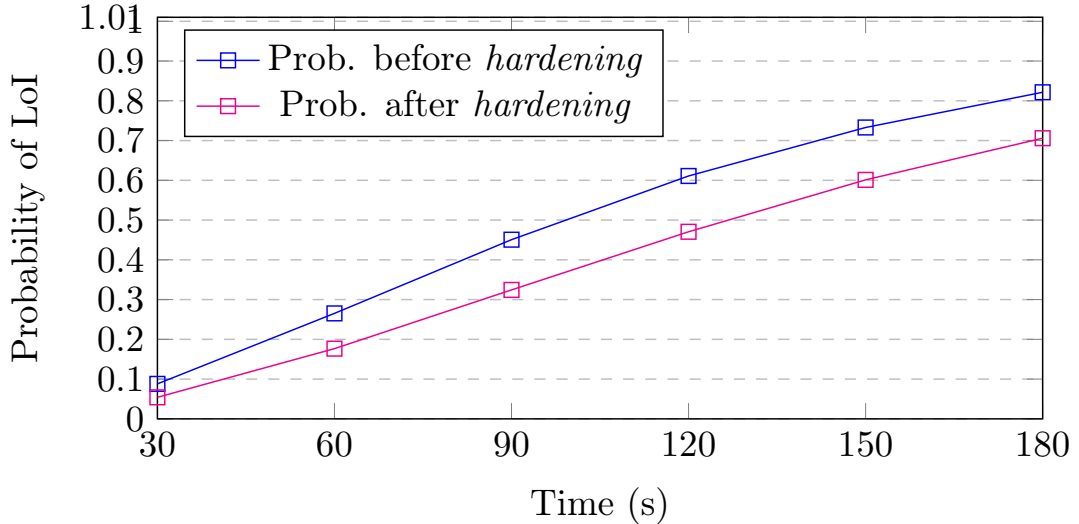


Figure 2.14: Prob. in LoI reduced by 15.85% in security AT due to application of design principles.

for safeguarding security and privacy are critical for successful operation of current devices and sensors connected to physical networks as part of IoT systems. As mentioned in Section 2.1.5, these design principles are essential to construct a trustworthy edge computing based system architecture. The goal is to apply a combination of design principles at different levels of abstraction to help in developing effective mitigation strategies. We adopt a selection of design principles such as *hardening*, *diversity* and *principle of least privilege* among the list of principles available in NIST document [28], and [53]. In the following, we demonstrate their effectiveness by showing that there is a reduction in the probability of disruption terms after adopting them in our VRLE system design.

***Implementation of design principles on security attack tree:*** In this section, we apply design principles on one of the identified vulnerable nodes of the security attack tree as shown in Section 2.3.1. For instance, we incorporate *hardening* design principle on the password attacks, to study its effects on the security metric LoI as shown in Figure 2.14. As part of the *hardening* principle, we added new nodes such as a firewall and a security protocol in the security attack tree. Our results show that the probability of disruption of LoI is reduced from 0.82 to 0.69 (15.85%), with the given attacker profile. The decrease in the disruption of

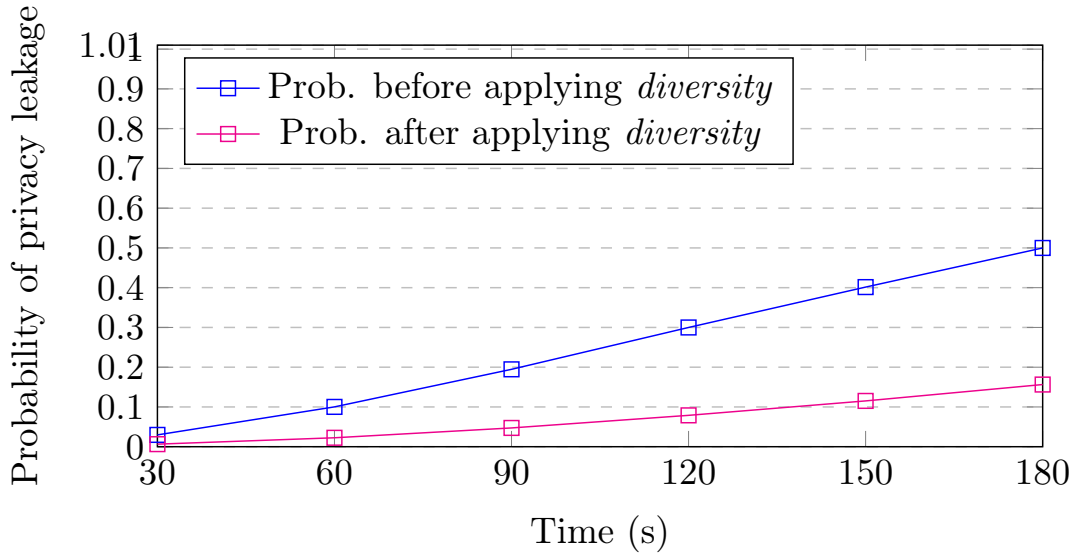


Figure 2.15: Prob. of privacy leakage reduced by 68% in privacy AT due to application of design principles.

LoI is due to the rise in additional resources that are required by the attacker to compromise such a VRLE application system which is incorporating the *hardening* principle. Similarly, we apply the *principle of least privilege* on the security attack tree, which under-provisions privileges intentionally. This in turn reduced the probability of disruption of LoI from 0.82 to 0.79 (3.66%).

***Implementation of design principles on privacy attack tree:*** Using the similar approach mentioned in design principles on the security attack tree, we apply *diversity* design principle on one of the identified vulnerable nodes (unauthorized access) in the privacy attack tree. After adding multi-factor authentication procedures as part of the *diversity* principle, the probability of disruption on privacy leakage is reduced significantly from 0.5 to 0.16 (68%) as shown in Figure 2.15. Similarly, we apply the *principle of least privilege* by under-provisioning privileges on the privacy attack tree where the probability of disruption of privacy leakage is slightly reduced from 0.5 to 0.48 (4%). Thus, from the above implementation of individual design principles, we conclude that *hardening* and *diversity* are more effective in reducing the disruption of LoI and privacy leakage, respectively. Thus, our findings shows that some security principles are more effective

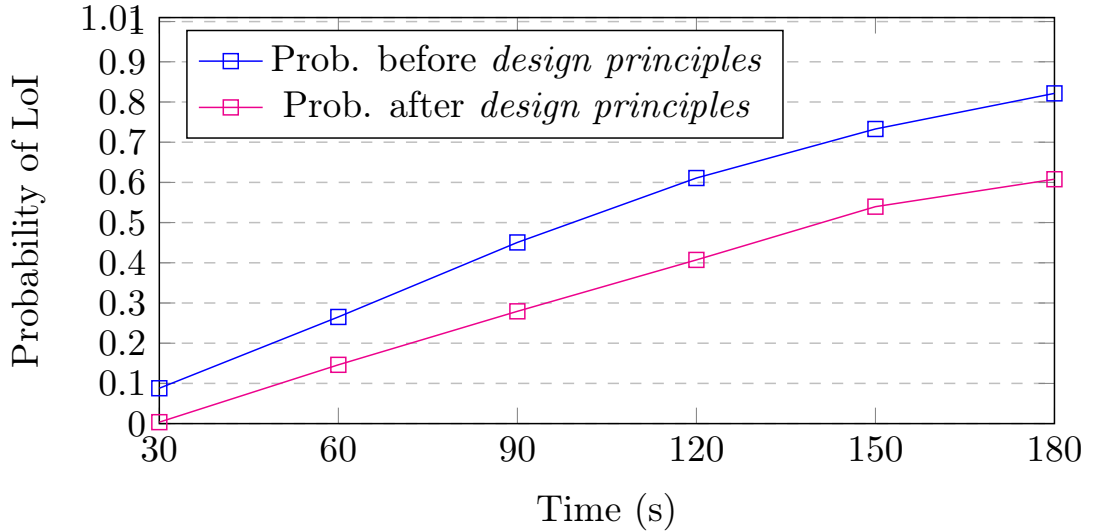


Figure 2.16: Prob. of LoI is reduced by 26% in security AT due to application of design principles for a combination of security attack tree nodes.

than others. In addition, our results emphasize the benefits in implementing a combination of design principles in both security and privacy attack trees to overall improve the attack mitigation efforts.

To study the effect on disruption of the LoI and privacy leakage, we adopt a combination of design principles such as: (i) for the security attack tree:  $\{hardening, principle\ of\ least\ privilege\}$ , and (ii) for the privacy attack tree:  $\{diversity, principle\ of\ least\ privilege\}$ . We observe that there is a significant drop in the probability of disruption of LoI from 0.81 to 0.6 (26%), and 0.5 to 0.1 (80%) for privacy leakage as shown in Figures 2.16 and 2.17, respectively.

From the above numerical analysis, we can conclude that incorporating relevant combination of standardized design principles and their joint implementation have the potential to better mitigate the impact of sophisticated and well-orchestrated cyber attacks on edge computing assisted VRLE systems with IoT devices. In addition, our above results provide insights on how the adoption of the design principles can provide the necessary evidence to support a trustworthy level of security and privacy for the users in VRLE systems that are used for important societal applications such as: special education, surgical training, and flight simulators.

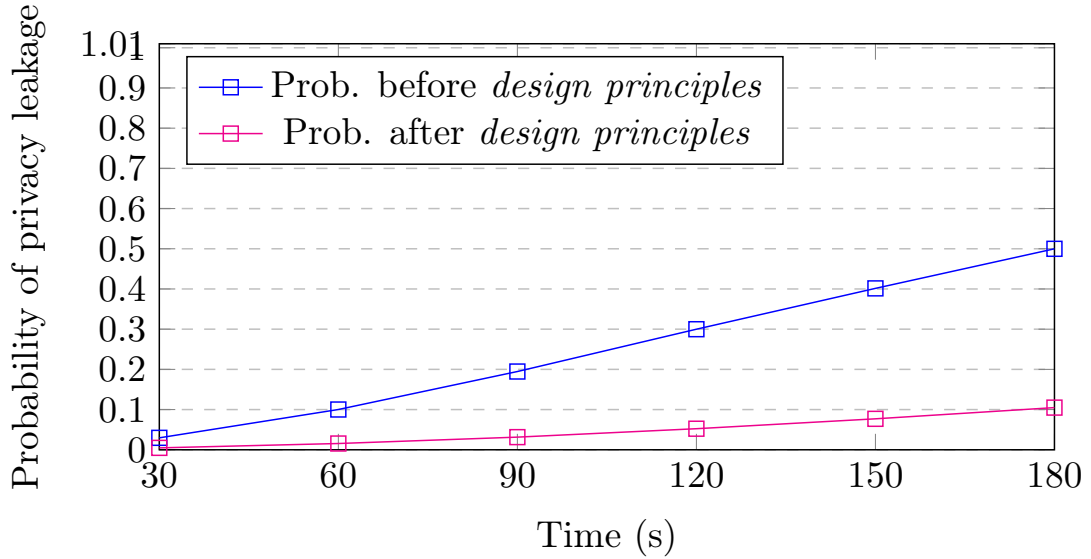


Figure 2.17: Prob. of privacy leakage reduced by 80% in privacy AT due to application of design principles for a combination of privacy AT nodes.

### 2.3.4 Conclusion

Social Virtual Reality Learning Environments (VRLEs) are a new form of immersive VR applications, where security and privacy issues are under-explored. To address security, privacy issues we presented a novel framework that quantitatively assesses such threat scenarios for a social VRLE application case study viz., vSocial. Specifically, we explored different threat scenarios that possibly cause LoI (e.g., unauthorized access) and privacy leakage (e.g., disclosure of sensitive user information) in a set of social VRLE application session scenarios. We utilized the attack tree formalism to model the security and privacy threats. Specifically, we developed relevant attack trees and converted them into stochastic timed automata and then performed statistical model checking using the UPPAAL SMC tool. Furthermore, we illustrated the effectiveness of our framework by analyzing different design principle candidates. We showed a ‘before’ and ‘after’ performance comparison to investigate the effect of applying these design principles on the probability of LoI and privacy leakage occurrence. The highlights from our experiments with realistic social VRLE application scenarios indicate that some security principles are more effective than others. However, combining them can result in a more effective mitigation mechanism. For instance, among the design

principle candidates, (i) *{hardening, principle of least privilege}* is the best design principle combination for enhancing security, and (ii) *{diversity, principle of least privilege}* is the best design principle combination for enhancing privacy.

In future, we plan to explore the effect of fault and attacks as a combination using the attack-fault tree formalism [50] for VRLE applications. This will allow us to reason about the safety metrics and study the safety, security and privacy trade-offs. Since, different components in a typical social VRLE application go through different maintenance actions, we also plan to explore the impact of various maintenance strategies on the reliability metric of social VRLE applications using the fault maintenance tree formalism [59].

## 2.4 Social Virtual Reality Attacks inducing Cybersickness

Recent works [17, 29, 30] highlight the importance of security and privacy (SP) issues in VR applications. However, they lack in evaluation of critical SP, system/network fault issues that impact VRLE applications' functionality and user experience. For instance, an intruder can gain unauthorized access (e.g., using fake credentials) to tamper the VRLE content which constitutes as a security breach. The unauthorized access can also lead to a privacy breach through eavesdropping during a VRLE session. Similarly, an intentional network fault can be triggered by an adversary by launching a Denial of Service (DoS) attack. As a result of such an intentional cyber-attack which we also define as a fault-attack, the VRLE content can be rendered unavailable to VRLE users. In addition, a faulty VRLE component (e.g., infrared sensor, gyroscope) can be used for distorting the view of a VRLE user during a session. The impact of such security and privacy breaches can induce cybersickness, which is a set of unpleasant symptoms such as eyestrain, headache, nausea or even vomiting, thus compromising user safety in a VR session [19, 60, 61].

To motivate the impact of security and privacy issues on user safety, we consider a VRLE application viz., vSocial [4] shown in Figure 1.5 that is designed for youth with autism spectrum disorder (ASD). This multi-modal VRLE system uses the High Fidelity platform [27], and renders 3D visualizations based on the dynamic human computer interactions with an edge cloud i.e., vSocial Cloud Server. Owing to the inherent inter-connectivity of the network-edge and the core cloud in the VRLE setup, the VR application is vulnerable to novel attacks known as *immersion attacks*. To elucidate, using SP issues as the vulnerabilities, an attacker can: (i) cause defacement of VRLE content with offensive images known as *overlay attack* [17], (ii) obstruct the user view or trigger noise attenuation during VRLE sessions known as *occlusion attack* [17], and (iii) create application issues i.e., reduction of graphical content or delays between both user and avatar movement. Failure to address such security, privacy and resulting safety (SPS) issues in VRLE results in alteration of instructional content, compromise of learning outcomes, abuse of access privileges leading to confidential student information disclosure and/or poor student engagement due to cybersickness. Hence, it is imperative to model the inter-relationship between security, privacy and user safety for identification of cybersickness events and for creation of relevant defense mechanisms to increase the resilience of social VRLE systems.

#### 2.4.1 Background on Social Virtual Reality Attacks:

**i) Network-based attacks:** There have been several prior studies that highlight the importance of security and privacy threats on Augmented Reality (AR) devices, and edge computing. A recent study [22] on challenges in AR and VR discusses the threat vectors for educational initiatives, however this study does not characterize related attack impacts. Survey articles such as [29, 30, 36, 62, 63] are significant for understanding the concepts of threat taxonomy and attack surface area of sensors and fog computing applications. They highlight the need to go beyond specific components such as network, hardware or user interface, and



propose end-to-end solutions that consider system and data vulnerabilities [38]. An observation from the above state-of-art is that - there is a dearth of scholarly works on the quantitative evaluation for security and privacy threats in the context of VR applications.

**ii) Immersion attacks:** Existing works [17, 38] discuss about the immersion attacks that can cause physical harm and disrupt the user experience in a virtual environment [17]. For instance, authors in [17] detail about generating a *disorientation attack* by changing the translation and yaw, creation of *physical collision attack* by tampering the SteamVR [64] chaperone file and an *overlay attack* to display unintended images during the session. The work in this study [17] serves as a fundamental source for our attack data i.e., to study unique cyber attack patterns relevant to VRLEs that can potentially induce cybersickness for users. In addition, the works in [39, 65, 66] discuss the security, privacy challenges in AR and VR applications. However, they do not study the impact of such security, privacy challenges on the VRLE user safety. Furthermore, the authors in the work [67] discuss about how the participants can get disoriented due to the exposure in a virtual environment even for a 20 minutes session. We build our work based on these prior works, and explore potential novel attack surfaces related to security, privacy, safety issues in VRLE applications and their associated impact on inducing user cybersickness.

## 2.4.2 Cybersickness in Virtual Reality Environments

**i) Potential factors inducing cybersickness** Several works define cybersickness as a form of motion/simulation sickness due to several physiological factors of the user in correlation with immersion and presence in VRLE. For instance, the work in [60], [61] detail that the sensory conflict between the vestibular and visual senses causes motion sickness in an unfamiliar virtual environment. Another work [68], attributes cybersickness as a form of disorientation due to the user's view point. In contrast, the work in [69] notes the difference between motion

sickness and cybersickness in terms of the individual causes and their associated impacts on the users using motion sickness susceptibility questionnaire (MSSQ) via experimental simulation. In addition, the work in [70] found that that graphical quality in virtual environments significantly affects immersion and induces cybersickness. Our work builds on these existing works, but uniquely identifies potential factors that induce cybersickness and thus impact user safety within social VRLE application sessions.

**ii) Measuring cybersickness in virtual reality environments** Several works [69, 71] devise questionnaires related to cybersickness in relation to motion sickness factors. The work in [71] adapts the simulator sickness questionnaire (SSQ) to quantify the physiological effects. Their survey methodology considers factors related to general discomfort, fatigue, eyestrain, difficulty focusing, headache, fullness of head, blurred vision, dizziness with eyes closed, and vertigo. Effects such as cybersickness, simulator sickness, and motion sickness have been found to be correlated in [69]. Similarly, the work in [72] explores the subjective aspects of the virtual environments i.e., presence, engagement, immersion and their relation to cybersickness symptoms (i.e., consequences on user experience). In addition, there have been studies that examined the effect on user immersion by implementing such questionnaires within a virtual environment [73]. The authors in [73] found that users in the virtual environments prefer such deployment of virtual questionnaires over traditional paper survey formats. However, we remark that this finding is dependent on the integration of the questionnaires such that the immersion is preserved during the virtual sessions for the users. We adapt these works in devising our questionnaires that are seamlessly integrated in the VRLE sessions to help us determine the potential factors of cybersickness.



Figure 2.18: A before and after scenario showcasing the effect of DoS attack on vSocial causing a server crash.

### 2.4.3 Experimental Validation of Social VR Attacks Inducing Cybersickness

To understand the potential cyber-attacks/fault-attacks and their impact on cybersickness, we perform an experimental behavior analysis on a vSocial instance. For this, we simulate exemplar security, privacy and safety attacks based on my works [16, 26] through our experimental validations. We validate the SP attacks of vSocial application using simulation tools (such as Clumsy 0.2 [74] and Wireshark [75]). For the purposes of this paper, we define: (a) *security* – as a condition that ensures a VR system to perform critical functions with the establishment of confidentiality, integrity, and availability [28], (b) *privacy* – as a property that regulates the IoT data collection, protection, and secrecy in interactive systems [28], (c) *safety* – as the disruption in the system that compromises the user’s overall well-being [28].

**i) Security Concerns Disrupting User Experience in VRLEs:** Social VRLEs use distributed head-mounted displays and wearable devices when connecting to virtual classrooms. Consequently, user experience in social VRLE applications is highly sensitive to Distributed Denial of Service (DDoS) attacks. To elucidate, a malicious user simulates a DoS attack scenario on the vSocial environment, via *packet tampering*, *packet duplication*, and *packet drop* which results in a VRLE server crash as shown in Figure 2.18. Based on our validation experiments, a

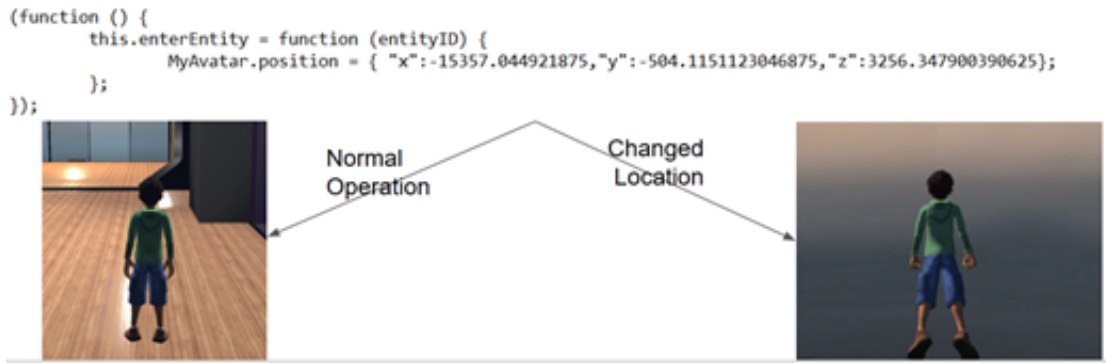


Figure 2.19: vSocial system content affected due to insertion of malicious scripts in the student learning environment.

packet drop of 80% disrupts the communication between the user and VRLE server thereby impacting the learning experience [26]. Similarly, a tamper rate of 20% is sufficient to crash the VRLE server for all the connected VRLE users as shown in Figure 2.18. In case of packet tampering, a man-in-the-middle attack scenario can reveal confidential information as discussed in [26].

Another security attack scenario can involve an attacker gaining unauthorized access by impersonating as a valid user. *Gaining unauthorized access* to the instructor account can trigger a privacy attack with threats such as: *disclosure of confidential user information*, and *tampering* of the learning content, edge computing and network devices. For instance, an intruder in social VRLE inserts malicious scripts to initiate data tampering of the VRLE application content as shown in Figure 2.19. With this, the attacker can modify the boundaries of the users' virtual view that can lead to a user running into a wall and getting physically hurt. *Failure to address such security attack scenarios can lead to an immersive attack i.e., overlay attack [76]) that can impact the VRLE user experience.*

## ii) Privacy Concerns Disrupting User Experience in VRLEs

A user privacy breach can involve an intruder who enters into a VRLE world by gaining unauthorized access (e.g., using fake credentials) and tries to eavesdrop the virtual classroom conversations. The attacker can gain access to the virtual location of the user and can disrupt the orientation of the user's virtual object (avatar). Privacy attacks can also involve *packet tampering* that was demonstrated

```

GET /ozan/dev/avatars/invisible_avatar/invisible_avatar.fst HTTP/1.1
User-Agent: Mozilla/5.0 (HighFidelityInterface)
Connection: Keep-Alive
Accept-Encoding: gzip, deflate
Accept-Language: en-US,*
Host: hifi-content.s3.amazonaws.com

```

**Host server information for VR rendering**

```

HTTP/1.1 200 OK
x-amz-id-2: Y7BJClhR++9PYY3fftBpsRwvP79krxMHYGHyoMYiyX+Otu91T9Kmw6nQjnjqM7beyfw8k5UmSU0=
x-amz-request-id: A3F506C04B96C83B
Date: Wed, 18 Jul 2018 18:08:15 GMT
Last-Modified: Fri, 20 Oct 2017 21:03:58 GMT
ETag: "81e2c1c96cf0937f1de19f76fba9b51a"
x-amz-version-id: RymAkXYpAaScbXfFkswh9g6zWMT7Dkq2
Accept-Ranges: bytes
Content-Type: image/vnd.fst
Content-Length: 4414
Server: AmazonS3

```

```

name = invisible_avatar
type = body+head
scale = 1
filename = invisible_avatar/invisible_avatar.fbx
texdir = invisible_avatar/textures
joint = jointLeftHand = LeftHand
joint = jointEyeLeft = LeftEye
joint = jointLean = Spine

```

**Attacker is able to acquire avatar information in real time**

Figure 2.20: Packet Sniffing attack to disclose avatar and host server information.

in [26], where an attacker performs an illegal packet capture to extract sensitive information (*packet sniffing attack* as shown in Figure 2.20).

Another form of privacy breach can occur when the attacker *discloses confidential user information* via packet sniffing attack to gain access to users' physical location information and credentials as shown in Figure 2.21. Such privacy attacks can create: (a) loss of confidentiality (LoC) when sensitive information is disclosed, and (b) loss of integrity (LoI) when the attacker tampers with the VRLE content.

```

[03/08 11:00:40] [DEBUG] [hifi.networking] Found metaverse API account information for https://metaverse.highfidelity.com
[03/08 11:00:40] [DEBUG] [hifi.interface.deadlock] DEADLOCK WATCHDOG WARNING: lastHeartbeatAge: 673957 elapsedMovingAverage:
[03/08 11:00:40] [DEBUG] [hifi.interface.deadlock] DEADLOCK WATCHDOG WARNING: lastHeartbeatAge: 673957 elapsedMovingAverage:
[03/08 11:00:40] [DEBUG] [hifi.networking] QHostInfo lookup result for "stun.highfidelity.io" with lookup ID 1 is "54.67.22.2
[03/08 11:00:40] [DEBUG] [hifi.networking] Sending intial stun request to stun.highfidelity.io
[03/08 11:00:40] [DEBUG] [hifi.audio] Processing sound file "sample.wav"
[03/08 11:00:40] [DEBUG] [hifi.networking] Waiting for inital public socket from STUN. Will not send domain-server check in.
[03/08 11:00:40] [DEBUG] [hifi.interface] Created Display Window.
[03/08 11:00:40] [DEBUG] [hifi.networking] New public socket received from STUN server is 104.166.194.109:59395
[03/08 11:00:41] [DEBUG] [hifi.audioclient] Starve detected, 2 new unfulfilled reads
[03/08 11:00:41] [DEBUG] [hifi.interface.deadlock] DEADLOCK WATCHDOG WARNING: lastHeartbeatAge: 1675029 elapsedMovingAverage:

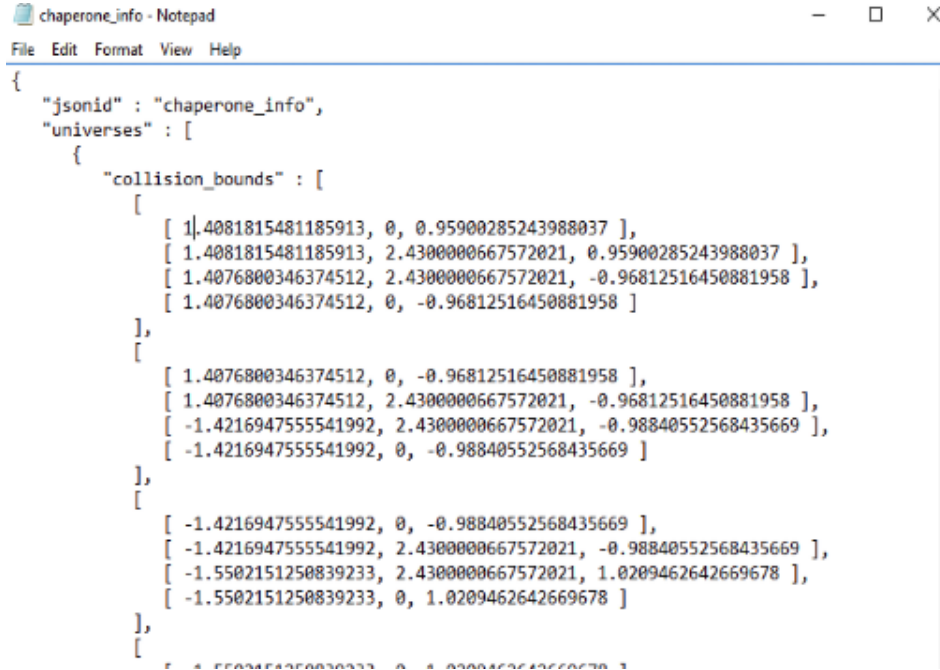
```

IP Address	Country	Region	City
104.166.194.109	United States 🇺🇸	Missouri	Columbia
ISP	Organization	Latitude	Longitude
Mediacom Communications Corp	Not Available	38.9517	-92.3341

Figure 2.21: A malicious user exposes the IP address of a valid VRLE user (disclosing user physical location) by gaining access to VRLE activity logs.

Failure to address such privacy attack scenarios can disrupt the UIX in an ongoing VRLE session by obstructing the view of the users in their learning sessions i.e., occlusion attack, or by creating a noise attenuation issue or by causing disorientation of the content to disrupt the UIX.

### iii) Safety Concerns Disrupting User Experience in VRLEs



```

{
  "jsonid" : "chaperone_info",
  "universes" : [
    {
      "collision_bounds" : [
        [
          [ 1.4081815481185913, 0, 0.95900285243988037 ],
          [ 1.4081815481185913, 2.4300000667572021, 0.95900285243988037 ],
          [ 1.4076800346374512, 2.4300000667572021, -0.96812516450881958 ],
          [ 1.4076800346374512, 0, -0.96812516450881958 ]
        ],
        [
          [ 1.4076800346374512, 0, -0.96812516450881958 ],
          [ 1.4076800346374512, 2.4300000667572021, -0.96812516450881958 ],
          [ -1.4216947555541992, 2.4300000667572021, -0.98840552568435669 ],
          [ -1.4216947555541992, 0, -0.98840552568435669 ]
        ],
        [
          [ -1.4216947555541992, 0, -0.98840552568435669 ],
          [ -1.4216947555541992, 2.4300000667572021, -0.98840552568435669 ],
          [ -1.5502151250839233, 2.4300000667572021, 1.0209462642669678 ],
          [ -1.5502151250839233, 0, 1.0209462642669678 ]
        ],
        [
          [ 1.5502151250839233, 0, 1.0209462642669678 ],
          [ 1.5502151250839233, 2.4300000667572021, 1.0209462642669678 ],
          [ 1.4076800346374512, 2.4300000667572021, -0.96812516450881958 ],
          [ 1.4076800346374512, 0, -0.96812516450881958 ]
        ]
      ]
    }
  ]
}

```

Figure 2.22: Immersion attack to tamper the chaperone file.

Based on prior work in [17], we perform safety attacks that can potentially disrupt the UIX in a vSocial instance. For instance, we assume that a *XSS browser attack* can occur, where the web entities in vSocial are targeted to hook the browser for hijacking the VRLE content. Immersion attacks e.g., *overlay attack* overlays and replaces visuals with offensive content on a user’s local machine which can partially compromise the VRLE [17]. Performing an overlay attack along with an SQL injection can create more impact on user privacy as the confidential information can be captured via overlay entities. By modifying the boundaries in a SteamVR chaperone file as shown in Figure 2.22 via TFTP [64], an attacker can disorient the user avatar or can lead a user to run into walls or physical objects. With this immersion attack, a VRLE user can experience light-headedness, a potential cybersickness factor. In addition, a *network fault-attack* that is triggered by low

Table 2.6: Survey questions asked during a CS experiment.

<b>Label</b>	<b>CS Questions</b>	<b>Related Works</b>
Nausea	How often do you feel nauseous ?	[77, 78]
Discomfort	What level do you feel discomfort or disoriented during the sessions ?	[77]
Vection	How often do you have the feeling of self-movement ?	[77, 78]
Eyestrain	What level do you feel eye strain or light-headed ?	[77]
Task Completion	How likely are you able to finish the tasks in the session ?	[77, 78]
Mental Engagement	How mentally engaged are you in the environment ?	[72, 77]

bandwidth events (*e.g.*, *packet loss scenarios*) can disrupt the VRLE content, thereby inducing cybersickness for a user who is remotely participating in a VR session.

Using our experimental validations, we next describe a preliminary study on the outlined security, privacy threat scenarios and their respective impact on the cybersickness. Towards this aim, we perform a systematic study to identify the potential cybersickness factors that rely on both the simulator sickness as well as user experience factors adapted from relevant prior works [19, 60].

#### **2.4.4 Impact Analysis of Security, Privacy Issues on Cybersickness**

In this section, we perform an experimental behavior analysis of the above simulated SP attack scenarios and their impact on potential cybersickness factors in VRLEs. For this, we adopt several CS factors based on the existing works [72, 77, 78] that include *e.g.*, the Virtual Sickness Questionnaire (VSQ). To measure the impact on each of these CS factors during the VRLE users session, we organize the CS factors under 6 questions as shown in Table 3.1. These questions are

**Cybersickness Assessment**

Survey Number

1	2	3	4	5
---	---	---	---	---

**CS Questions**

Please respond to the following questions with 1 being not at all and 5 being very much

How often do you feel nauseated?

1	2	3	4	5
---	---	---	---	---

On a scale of 1-5, what level of discomfort/disorientation do you feel?

1	2	3	4	5
---	---	---	---	---

How often do you have the feeling of self-movement?

1	2	3	4	5
---	---	---	---	---

On a scale of 1-5, what level of headache/eyestrain do you feel?

1	2	3	4	5
---	---	---	---	---

How likely were you were able to finish the task?

1	2	3	4	5
---	---	---	---	---

How mentally engaged are you in the environment?

1	2	3	4	5
---	---	---	---	---

Clear

Figure 2.23: cybersickness virtual questionnaire for the users to respond after each activity in a vSocial experiment

integrated as virtual questionnaires (VQs) and post-session survey for feedback collection such that they do not cause any disruption to the VRLE users [79]. In addition, based on our prior work in [26]: (i) we simulate a *security attack*, *privacy attack* and *combination of security, privacy attacks* across three different user activities as shown in Figure 2.24 and, (ii) subsequently quantify CS using a set of VQs. In this context, we set up a vSocial [4] instance with activities that rely on fine movement, visual clarity, and clear audio, all of which are disrupted by SP attacks as shown in Figure 2.24.

Firstly, we collect baseline data (i.e., benign behavior) using VQs at the end of



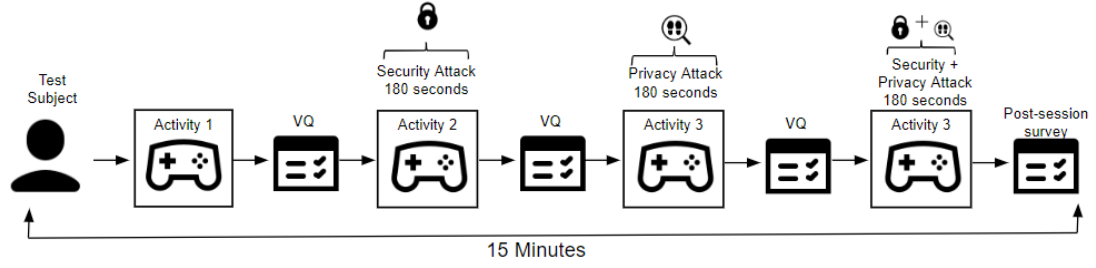


Figure 2.24: Activities and their associated virtual questionnaires used for the immersion survey experiment.

activity1. Next, during activity 2, we simulate malicious packet drops to disrupt VRLE content rendering as a security attack and then collect user feedback. Similarly, in activity 3 we simulate a privacy attack to trace the user’s virtual location and then play a distracting noise to disrupt the user experience. This disruption is stopped approximately halfway through activity 3, where the user response is recorded. For the rest of activity 3, a combination of security and privacy attacks (i.e., packet tampering + disclosure of user location), are simulated after which the user exits the vSocial environment to give feedback via a post-session paper survey.

To maintain uniformity across the collected feedback, we used the same VQ as shown in Figure 3.2. For this survey, 15 VRLE user participants provided their feedback for each of the questions in our CS survey on a scale of 1 to 5, where 1 (very poor), 2 (poor), 3 (Moderate), 4 (good), 5 (excellent). This collected data allows us to quantify the aggregated impact value of CS score due to SP attacks in the VRLE. We stop the attack simulations until the checkpoint (attack duration is 180 seconds), to avoid further discomfort in relation to the time spent by the user in vSocial.

**Results of the Impact analysis of SP attacks on CS factors:** Using the collected data for the SP versus CS factors analysis, we outline the data points for each cybersickness factor versus the average rating given by the users as shown in Figure 3.2. The cybersickness factors listed in Table 3.1 are represented as N,D,V,E,T,M on x-axis and the average of the user responses on y-axis as shown

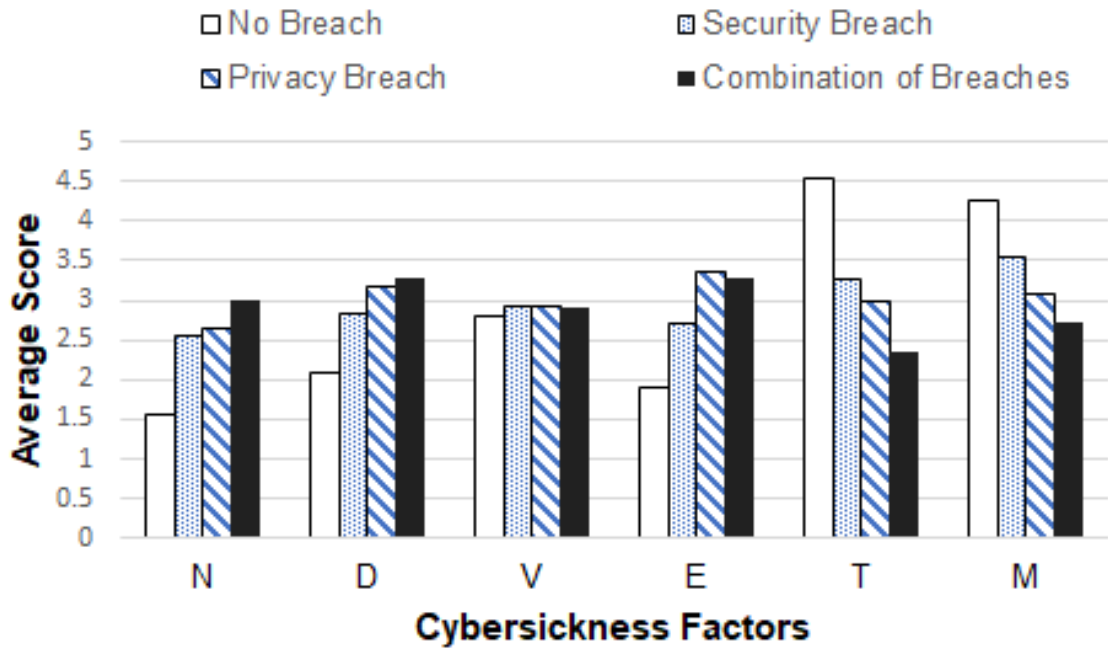


Figure 2.25: Results of CS-survey experiments.

in Figure 3.2. Based on the results, we observe that security, privacy, and combination of SP attacks have significant impact the cybersickness factors. To elucidate, the combination of security, privacy attacks causes significant dizziness and nausea, the primary factors that induce cybersickness. From these results, we understand that SP attacks do certainly induce cybersickness by disrupting users and also the functionality of the VRLE. With this analysis, we further explore the potential security and privacy, fault-attack in detail (single and combination of threats) that could induce CS in a VRLE.

Based on the above impact analysis, we will focus on the causes of cybersickness for a Virtual Reality Learning Environment (VRLE) case study: vSocial [4] which was developed for youth with autism spectrum disorder (ASD). Since every VR users in such applications are susceptible to cybersickness in a different manner, we refine our problem scope to the following research questions:

- What components (sensors, controllers, headsets, etc.,) in VR are vulnerable and what type of cyber-attacks on those identified components may lead to cybersickness?
- What are the VR applications issues (network, storage, etc.) that might also

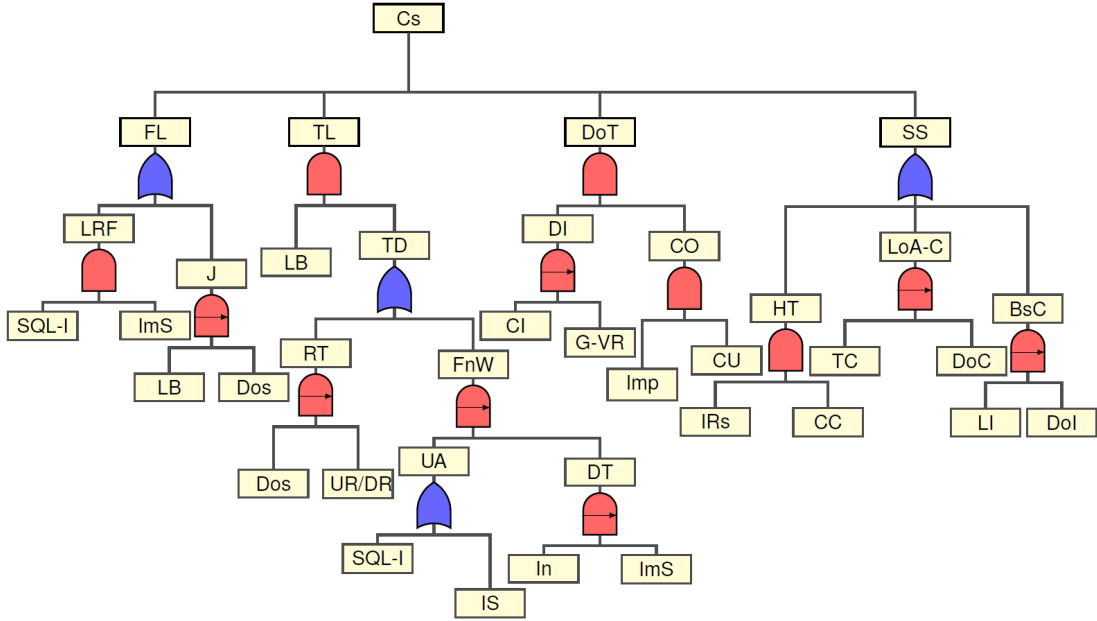


Figure 2.26: Safety attack tree for VR application case study: vSocial

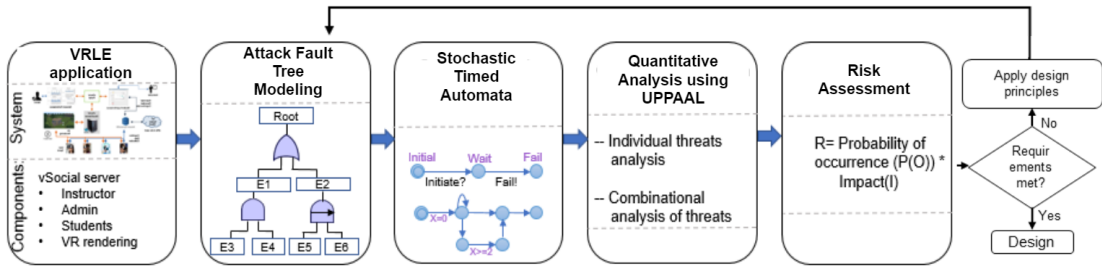


Figure 2.27: Proposed framework for security and privacy analysis of a social VRLE.

triggers cybersickness?

## 2.5 Security, Privacy Harmonized Framework for Social VRLE

In this section, we present our proposed framework to identify the most vulnerable attacks/fault components inducing cybersickness by performing security, privacy and safety analysis of social VRLE applications. An overview of the approach followed in our framework is shown in Figure 2.27. Firstly, we model the SP attack vectors inducing cybersickness by using an attack-fault tree formalism based on our preliminary results from Section 2.4.4. Secondly, each attack-fault tree is

Table 2.7: Description of leaf nodes in safety AT

Leaf node components	Description of leaf nodes	Leaf node components	Description of leaf nodes
SQL Injection (SQL-I)	Attacker injects malicious commands in user input query using GET and POST to disclose information such as user credentials	Impersonation (Imp)	Attacker pretends to be a valid user and tries to influence user experience in VR environment.
Identity Spoofing (IS)	Attacker pretends to be a legitimate user and tries to disrupt the VR experience.	Control of user (CU)	After impersonating to VR space attacker can control the user activity.
Intrusion (In)	Attacker gets unauthorized access to user VR location.	IR sensors (IRs)	Attacker tampers the IR sensors in VR headset to disrupt the visual in VR.
Insert Malicious Scripts (ImS)	Attacker successfully adds malicious scripts in the VR environment to compromise visual or change the contents.	Change coordinates (CC)	Attacker tries to disrupt the cybersickness by changing the coordinates.
Denial of service (Dos)	Loss of availability in of user features in environment to compromise the visual or change the contents.	Track coordinates (TC)	Attacker determines the coordinates of the controllers.
Upload rate/download rate (UR/DR)	Intermittent discrepancies in network can reduce the overall bandwidth resulting in bad VR experience.	DoS of controller (Dos)	Attacker tracks the coordinates to disrupt the of the controllers.
Collect information (CI)	Information such as ip or any other confidential information of the user can be disclosed to disrupt user privacy.	Locate information (LI)	Locate the base station information which includes the information about controllers, sensors and headset information.
Goto VR space (G-VR)	Go to the user's VR space location.	Disclosure of information (DI)	Attacker discloses the component's information such as ip to disrupt the cybersickness of the controllers or change the contents.

Table 2.8: Description of the Intermediate nodes for safety AT

Node	Description	Node	Description	Node	Description
Cs	Cybersickness	UA	Unauthorized Access	IRs	Infrared sensors
FL	Flicker	DT	Data Tampering	CC	Change coordinates
LRF	Low Refresh Rate	IS	Identity Spoofing	TC	Track coordinates
SQL-I	SQL Injection	In	Intrusion	DoC	DoS of controller
ImS	Insertion of malicious scripts	DoT	Disruption of userin task	LI	Locate information
J	Jitter	DI	Disclosure of information	DoI	Disclosure of data
LB	Low Bandwidth	CO	Control User	CU	Control of user
DoS	Denial of Service	CI	Collect Information	HT	Tampering of headset
TL	Timelag	G-VR	GoTo VR Space	LoA-C	Loss of availability of controller
LB	Low Bandwidth	Imp	Impersonation	BsC	Compromise base station
TD	Transfer Delay	FnW	Features not Working	-	-
RT	Response Time	UR/DR	Upload Rate/Download Rate	-	-

translated into an equivalent STA to form an NSTA, as input into the UPPAAL SMC tool. Thirdly, we use the quantitative assessment from the tool to determine if the probability of disruption is higher than a set threshold determined as part of VRLE design requirements. Lastly, we perform risk assessment of the identified attack vectors to determine the potential impact on VR application functionality and inducing cybersickness. Based on this determination, we subsequently prescribe the design principles such as: *hardening*, *diversity*, *redundancy* and *principle of least privilege* that can be adopted in VRLE deployments. Overall, our framework steps help in the investigation of potential cyber attacks and fault-attacks. Further, they help in recommendations of VRLE application design alternatives based on design principles.

**Fault-attacks in social VRLEs:** Using the attack tree formalism we model the potential attacks inducing cybersickness as shown in the Figure 2.26. Note, the dynamic user interactions in a social VRLE application can create new security, privacy attacks and/or faults surfaces (individual and combination of attacks). For instance, an intentional network fault can be triggered by an adversary by launching a Denial of Service (DoS) attack. As a result of such an intentional cyber-attack which we also define as a fault-attack, the VRLE content can be rendered unavailable to VRLE users. In addition, a faulty VRLE component (e.g., infrared sensor, gyroscope) can be used for distorting the view of a VRLE user during a session. The impact of such security and privacy breaches can induce cybersickness, which is a set of unpleasant symptoms such as eyestrain, headache, nausea or even vomiting, thus compromising user safety in a VR session [19, 60, 61]. Based on the effectiveness of attack trees for successfully modeling the security and privacy scenarios, we further explore the potential fault-attacks leading to cybersickness as shown in the table 2.9.

To study such potential security, privacy and safety issues causing cybersickness from different aspects of a VRLE application pertaining to (i) *different phases of the VRLE workflow (services)* — this includes the threat scenarios for different

Table 2.9: Potential Fault-attacks in VRLE

Types of faults in VRLE	Example fault scenarios
Hardware / physical components in VR	A faulty sensor in VRLE can cause not to display the VRLE content on the headset.
	Detection of an issue in the application can occur due to a component failure such as base stations in the range
	Failure of a Infrared sensor (used for eye tracking), accelerometer and gyroscope can result in a distorted view of VR content to the user in VRLE
Software and communication	Data is not retrieved due to a discrepancy in the code and causes the workload and makes the retrieval process slow or fail
Network faults	Such faults occur in the VRLE server which is hosted in a cloud environment.
	Occurs due to packet loss or corruption, congestion, failure of the destination node or link, etc. Factors like Low bandwidth, packet loss, can result in crashing VRLE application
Permanent Faults	Crash faults that cause the system components to completely stop functioning or remain inactive during failures (e.g., power outage, hard disk crash)
Transient Faults	Byzantine faults occur in VRLE system components to behave arbitrarily or maliciously during failure, causing the system to behave unpredictably incorrect.
Process faults	Such faults occur in the VRLE workflow processes due to lack of resources for processing the EEG data, storing the user data

phases in VRLE workflow such as: data Collection, data in transit, data visualization, data storage. (ii) *VRLE resources (components)* — this includes the VRLE components such as: VR headset, controllers as well as different sensors in the architecture, and (iii) *different user interactions relevant to the VRLE application workflow management* — which includes user interactions that occur in the VRLE such as: user-user, user-instructor, instructor-admin. In order to explore the relationship of fault, and security and/or privacy threats leading to cybersickness, we utilize the Attack Fault Tree (AFT) formalism [50].

### 2.5.1 Formalization of Security and Privacy Attack-fault Trees (AFTs)

To analyze the interplay of faults, security and privacy threat scenarios that trigger cybersickness, the dependencies are modeled using the attack fault tree (AFT) formalism [50] using the identified threats and faults outlined in Table 2.2, Table 2.9. Specifically, the focus is on modeling the security issues pertaining to the Confidentiality, Integrity and Availability (CIA) triad such as: Loss of Integrity (LoI), Loss of confidentiality (LoC), Loss of Availability (LoA) leading to cybersickness. On the other hand, the specific privacy threats (privacy leakage) and

fault scenarios (network faults, transient faults) are focused to study the causes of cybersickness in VRLE. For instance, availability of VR contents or data on the instructor/admin web application can be compromised if an attacker performs DoS which can lead to both privacy and security issues. Such complex inter-dependencies can aid in modeling more realistic scenarios, and hence covers a wider attack surface area.





Attack Fault trees (AFTs) outlines the logical steps to trigger an attack or a fault scenario in comparison with the traditional threat models. AFTs are hierarchical models that show how an attacker goal (root node) can be refined into smaller sub-goals (child/intermediate nodes) via gates until no further refinement is possible such that the lead node is reached. A leaf node can be a terminal node or a basic attack steps (BAS) [54]. To explore dependencies on VRLE attack surfaces, attack-fault trees enable sharing of subtrees. Hence, attack-fault trees are often considered as directed acyclic graphs, rather than trees [50].

**Definition 3** (Attack-Fault Trees (AFTs)). *An attack-fault tree  $A$  is defined as a tuple  $\{N, Child, Top\_event, l\} \cup \{AFT\ elements\}$  where,  $N$  is a finite set of nodes in the attack tree;  $Child: N \rightarrow N^*$  maps each set of nodes to its child nodes;  $Top\_event$  is an unique goal node of the attacker where  $Top\_event \in N$ ;  $l$ : is a set of labels for each node  $n \in N$ ; and  $AFT\ elements$ : is a set of elements in an attack-fault tree  $A$ .*

**Definition 4** (AFT elements). *AFT elements aid in generating the attack-fault tree and are defined as a set of  $\{G \cup BE \cup IE\}$  where,  $G = \{OR, AND, SAND, SOR, PAND\}$  represents the set of gates used for modeling the multi-threat and fault-attack scenarios in AFTs, and  $BE$  is the set of basic events  $\{BAS \cup BCF\}$  and  $IE$  is the set of intermediate events.*

**Attack-fault tree elements:** Attack-fault tree elements aid in generating an attack-fault tree and are defined as a set of  $\{G \cup L\}$  where,  $G$  represents gates;  $L$  represents leaf nodes. Following are the descriptions of each of the AFT elements.

**Attack-fault tree gates:** Given an attack-fault tree  $A$ , we formally define the attack-fault tree gates  $G = \{OR, AND, SAND, PAND\}$ .

-  **OR:-** An OR gate is disrupted if either of its child nodes are disrupted.
-  **AND:-** An AND gate is disrupted when all its child nodes are disrupted.
-  **SAND:-** A Sequential AND (SAND) gate is disrupted in the order of left to right only when its leftmost child node is disrupted. To elucidate, the gate is disrupted using the condition: the success of previous step determines the success of the upcoming child node.
-  **PAND:-** A Parallel AND (PAND) gate model the order dependent disruption (i.e., left to right) but activates its child nodes all at once.

We limit our attack-fault tree modeling to these gates, however attack-fault trees [50] can adopt any other gates from the static/dynamic fault trees. The output nodes of the gates  $G$  in an attack-fault tree  $A$  are defined as *Intermediate nodes* ( $I$ ), which will be located at a level that is greater than the leaf nodes.

**Basic attack step (BAS):** A BAS collectively represents all the individual atomic steps within a composite attack-fault scenario. Each of the BAS also represents the *leaf nodes* of an AFT [54].

**Attack-fault tree leaves:** Given an attack-fault tree  $A$ , we formally define the *attack tree leaves*  $L_{node} = \{BAS \cup leaf\ nodes\}$ .

In other words,  $L_{node}$  is the terminal node with no other child node(s) which is either modeled as BAS or a simple leaf node (modeled with exponential distribution) of the attack-fault tree. To elucidate, for an attacker to impersonate as a valid user, the prospective BAS can include: (i) spoofing attack, and (ii) session hijacking to the system depending on the attacker profile. For an attack-fault tree  $A$ , we assumed the attack duration to have an exponential rate and model



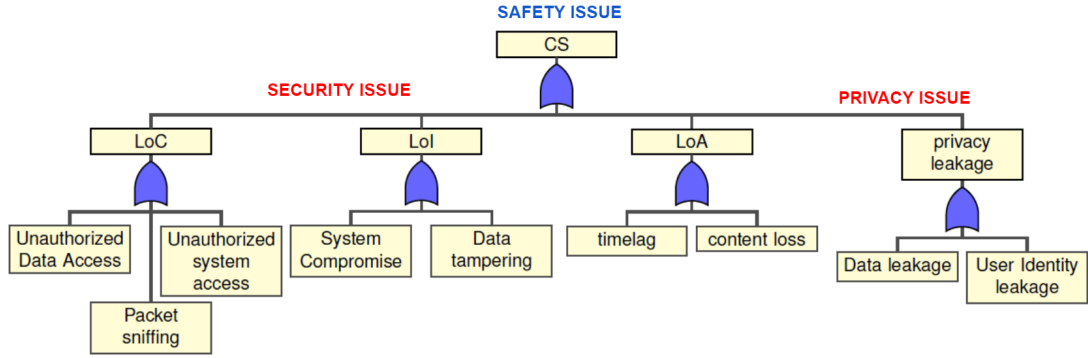


Figure 2.28: Formalized attack fault tree with threat scenarios inducing cybersickness.

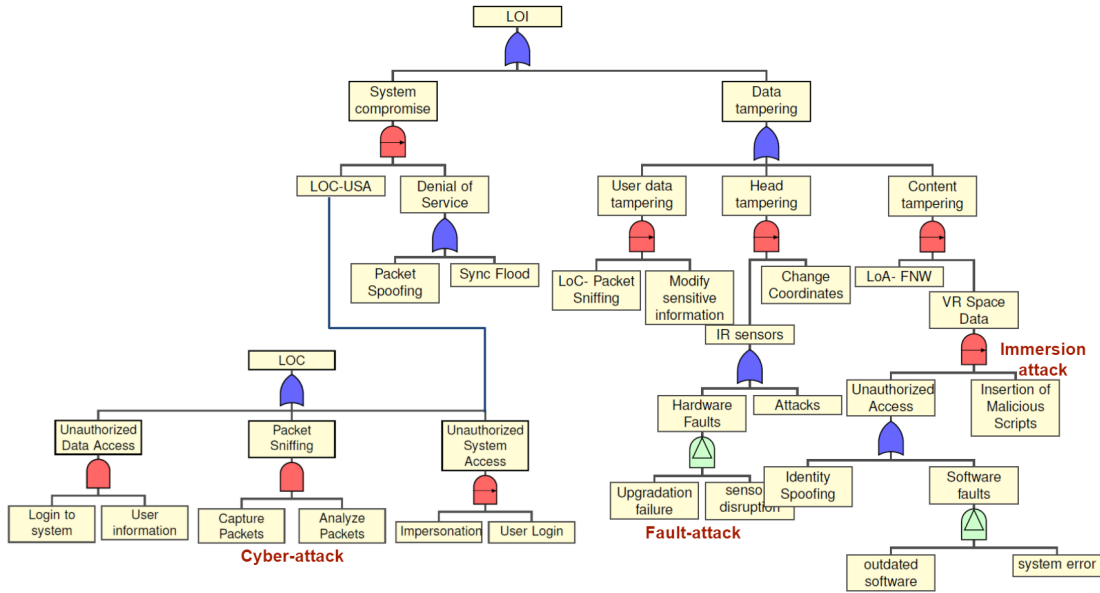


Figure 2.29: Formalized attack fault sub-tree with threat scenarios triggering a LoC and/or LoI issues.

the equation as :  $P(t) = 1 - e^{-\lambda t}$  where,  $\lambda$  is the rate of exponential distribution [16]. We use the exponential distribution because of its tractability and ease of handling, since they are defined by a single parameter.

## 2.5.2 Formal Description of Novel AFTs for VR attacks inducing cybersickness

Based on the results discussed in Section 2.1.4 and experimental evidence from our prior work [16], we model potential VRLE security, privacy and fault-attack scenarios that induce cybersickness in the form of an attack-fault tree as shown in

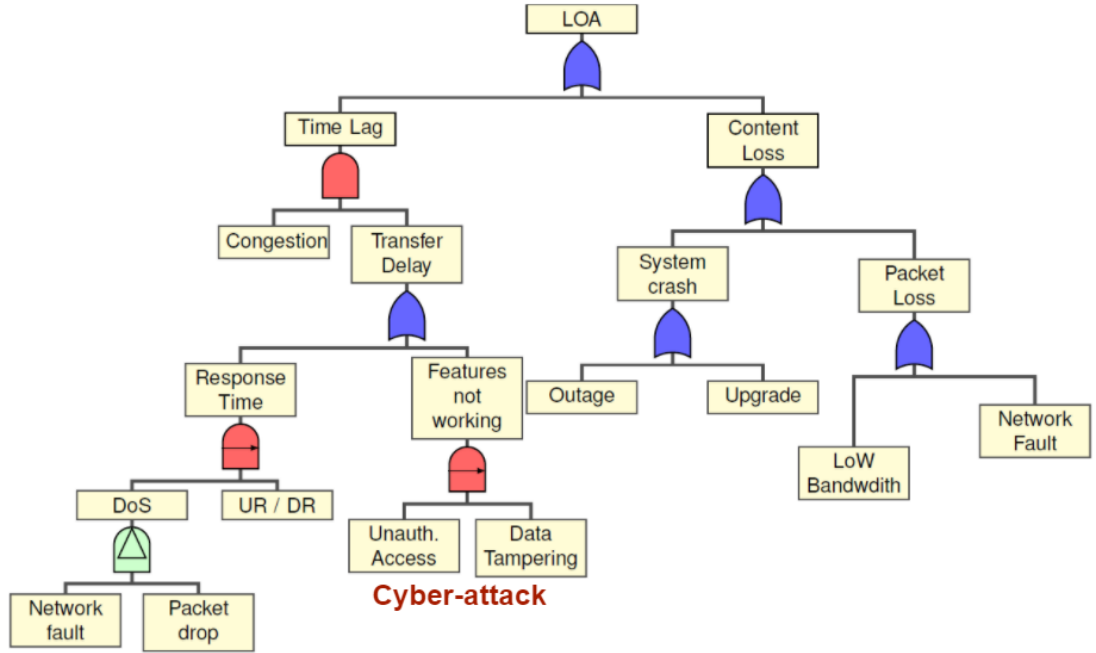


Figure 2.30: Formalized attack fault sub-tree with threat scenarios triggering a LoA issue.

Figure 2.28. As part of our framework approach, we consider cybersickness as the primary goal for an attacker. We model our main AFT as shown in Figure 2.28 using different security, privacy, fault-attacks in the form of sub-trees as shown in Figures 2.29, 2.30, 2.31. Exploring the security aspect in CIA triad of {Confidentiality, Integrity, Availability} results in an enormous number of leaf nodes in the AFT. For the purposes of our work, we term the main generated AFT that is covering SPS attack scenarios inducing cybersickness as *Safety-AFT*.

To elucidate, the safety AFT contains root node as cybersickness (CS) which branches out to intermediate nodes such as security issues relevant to CIA triad i.e., {Loss of Confidentiality (LoC), Loss of Integrity (LoI), Loss of Availability (LoA)}. This branching results in an enormous number of leaf nodes as shown in Figures 2.29 and 2.30. The intermediate nodes serve the purpose of establishing the relationship of root node to leaf nodes which are basically the basic attack steps to disrupt cybersickness. We continue the branching of intermediate nodes until no further division is possible, i.e., a case that terminates as leaf nodes. Similarly, the other sub-tree whose root node is privacy leakage address the privacy threat

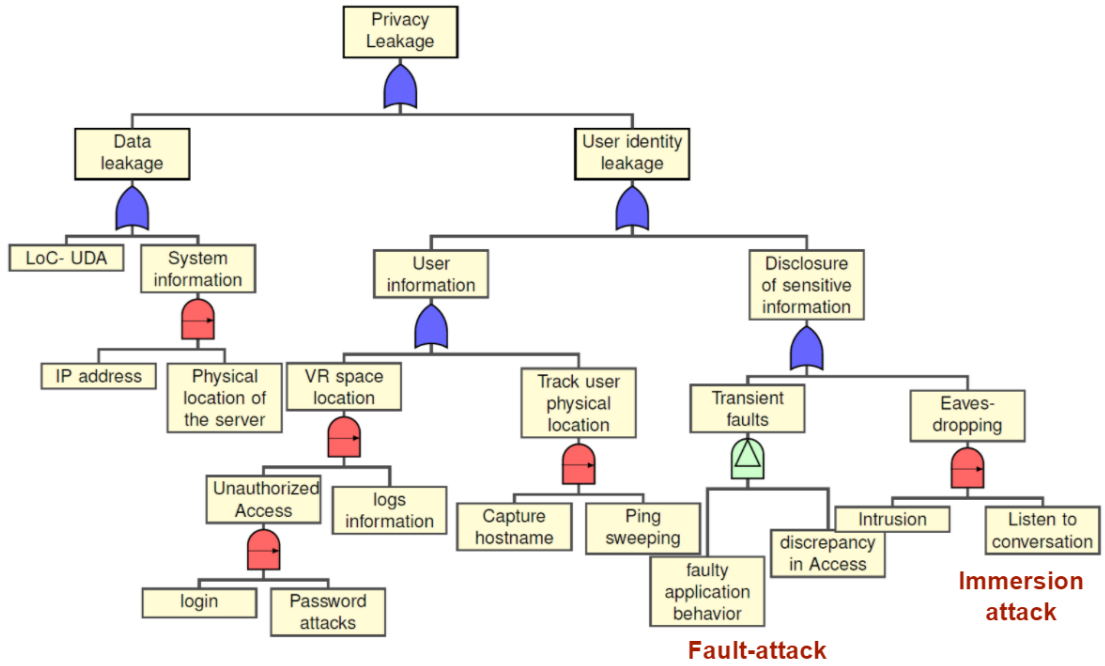


Figure 2.31: Formalized attack fault sub-tree with threat scenarios triggering a privacy leakage issue.

scenarios that induce cybersickness for a social VRLE user.

In our generated safety-AFT, we also explore the temporal dependencies in terms of sharing subtrees where the cause and effect relationship is outlined. To elucidate, an intermediate node in a sub-tree can be used as leaf node in a different sub-tree which can act as a sharing node. For example, the unauthorized system access (USA) which is an intermediate node in the LoC subtree is used as a leaf node represented as LoC-USA in the LoI subtree as shown in the Figure 2.29. Similarly, LoA subtree shares its intermediate node “Features not working (FNW)” as a leaf node which is represented as LoA-FNW in the LoI subtree. Moreover, the LoC subtree intermediate node i.e., Unauthorized data access (UDA) acts as the leaf node LoC-UDA in the privacy leakage subtree as shown in Figure 2.31.

Using this generated safety attack-fault tree, we discuss the process of converting an AFT into STA to perform stochastic model checking in the next section.

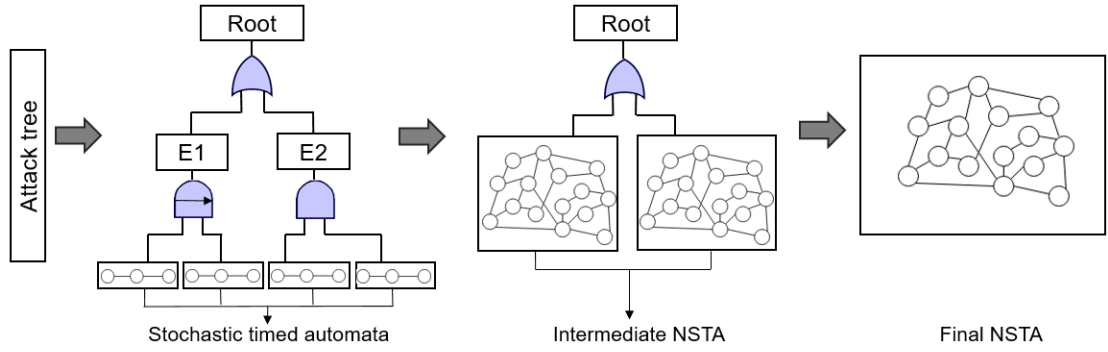


Figure 2.32: Framework for translation of attack trees into network of stochastic timed automata.

### 2.5.3 Translation of attack-fault trees into stochastic timed automata

In this section, we generate STA for each of the sub-trees in Figures 2.29, 2.30 and 2.31 that are part of the safety-AFT in Figure 2.28. An overview of our translation approach is shown in Figure 2.32 where: (i) each of the leaf nodes in these AFTs are converted into individual STAs. The intermediate events, which are basically the output of the logic gates that are used at different levels are converted imperatively into STA; (ii) the generated STAs are composed in parallel by including the root node; (iii) the obtained NSTA is then used for statistical model checking in order to verify the security, privacy and fault-attack properties formalized as SMC queries. As mentioned earlier, these obtained STAs are used for performing model checking to verify the cybersickness metrics formalized as SMC queries.

To demonstrate the translation of an AFT into an STA, we first consider the safety AFT shown in Figure 2.28. As part of the translation, each of the security AFT element (leaf and gates) input signals are connected to the output signal of child nodes. The generated network of STAs (i.e., NSTA) communicates using  $\{initiate, fail\}$  signals. *initiate* - indicates activation signal of attack tree element. This signal is sent initially from the root node to its children. *fail* - indicates disruption of that attack tree element. This signal is sent to the parent node from its child node to indicate an STA disruption. The scope of the above signals also includes special symbols such as:  $i'?$  (e.g.,  $initiate?$ ) means that

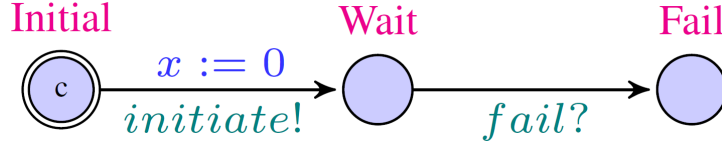


Figure 2.33: STA of cybersickness root node in safety AFT.

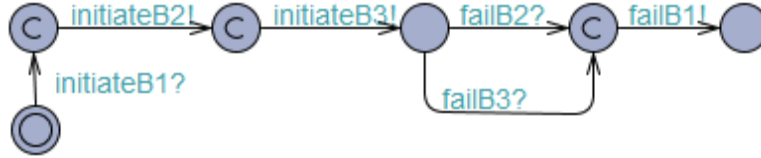


Figure 2.34: STA of OR gate and signal transition between the root node and the childnode for the LoA subtree in the safety-AFT.

the event will wait for the reception of the intended signal, ii) ‘!’(e.g., initiate!) implies output signal broadcasts to other STA in the attack-fault tree.

**Illustrative example:** In the subsequent paragraph, we illustrate our translational approach by converting the safety AFT into an exemplar NSTA. For instance, we show the conversion of root node (i.e., cybersickness) (*Top\_event*) into equivalent STA as shown in Figure 2.33. Here, the STA broadcasts *initiate* signal and waits for the *fail* signal from the child nodes to disrupt the cybersickness node. In addition, the clock  $x$  is a UPPAAL global variable where we declare  $x = 0$  to keep track of the time progression as mentioned in Section 2.1.4.

An example of OR gate in the safety AFT involves the child node *Transfer Delay* of the LoA sub-tree shown in Figure 2.30. The OR gate of the *Transfer Delay* is disrupted, when any of its child nodes *Features not working*, *Response Time* sends a *fail* signal as shown in Figure 2.34. This *fail!* signal is sent to the *Top\_event* which forces a transition to *Disrupt* state, representing LoA in the system thereby disrupting cybersickness. Similarly, STAs for the AND gate, SAND gates and the leaf nodes are also developed.

For instance, we consider the sub-tree LoA in Figure 2.30 whose intermediate node *Timelag* is converted into an STA representation with AND gate as shown in Figure 2.35. For this, the STA for intermediate node *Timelag*, waits for the disrupt (*fail*) signal from both *Traffic Congestion*, *Transfer delay* as shown in

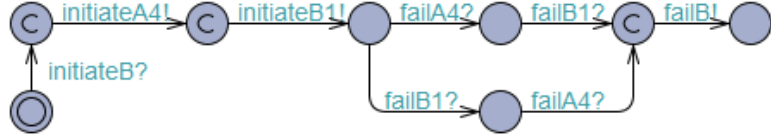


Figure 2.35: STA of AND gate for the LOA sub-tree of the safety AFT.

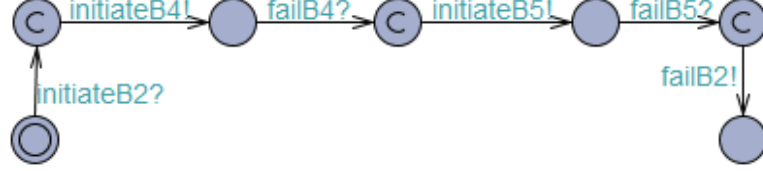


Figure 2.36: STA for SAND gate in safety AFT.

Figure 2.35. The *Timelag* node gets disrupted once the fail signal is received from its child nodes. Similarly, the node *Features not working* with a SAND gate is disrupted only if its child nodes *Unauthorized access*, *Data tampering* send the *fail* signal in a sequential manner (i.e., left to right order) as shown in Figure 2.36.

Next, we explain PAND gate which we have used to model the faults in NSTA as shown in Figure 2.37. In case of PAND gate for the node DoS in the LoA sub tree, we convert the equivalent STA where its children (i) DoS (ii) UR are initialized at the same time via *initiateA8* and *initiateA9* signals. The disruption of the PAND gate occurs once its both children are disrupted in a sequential manner as shown in Figure 2.37.

Moreover, the leaf node *Data Tampering* of the LOA subtree is converted to STA as shown in Figure 2.38. Here, the STA gets activated after receiving *initiateA10* signal form its parent node *FNW* and disrupts the gate by sending a disrupt signal. Each of these converted STA leaf nodes are instantiated with  $\lambda$  (rate of exponential) values. For the given  $\lambda$  values to the leaf nodes, the probability of occurrence is calculated. This value then propagates upward in the tree to calculate the probability of LoA thereby can be used to determine the probability of the main root node of the safety AFT i.e., cybersickness.

Using this translational approach, we convert all the nodes including the leaf nodes and their associated BAS in our safety AFT into equivalent STAs. These developed STAs are composed using the parallel composition [49] technique to

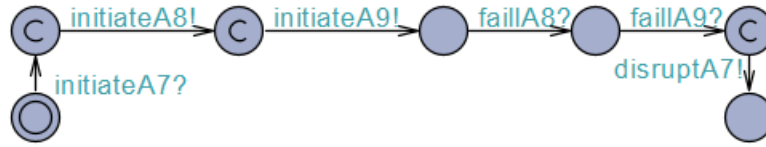


Figure 2.37: STA for PAND gate in safety AFT.



Figure 2.38: STA for leaf node data tampering in LOA subtree of the safety AFT. form an NSTA, which is then used for SMC by the UPPAAL tool [51].

#### 2.5.4 Attacker Profiling (AP) for BAS

The level of cybersickness induced due to each of these threat factors is also dependent on the considered attacker profiles (AP). For this, we develop AP based on prior works [54, 80] with attributes such as – (i) *time* taken to execute an attack, (ii) *cost* incurred to perform the attack, (iii) *skill* level of the attacker, and (iv) *resources* required to perform a cyber/immersion attack [81, 82] as shown in Table 2.10. To explain, we enlist the APs required to perform cyber-attacks/immersion attacks (e.g., DoS, Impersonation, SQL injection, control of users) as shown in the Table 2.10. With these generated AP, we showcase the logical steps taken by an attacker (i.e., BAS) to disrupt a leaf node. Each of these BAS are equipped with exponential distribution, discrete probabilities [50] in the

Table 2.10: Attacker profiles for modeling different BAS.

Type of Attack	Attacker	Skills	Resources
DoS Attack	Attacker1	High	Low (ping sweeping),
	Attacker2	Low	High (sync flood)
Impersonation	Attacker1	High	Low (spoofing attack),
	Attacker2	Low	High (Session Hijack)
SQL injection	Attacker1	High	High (Fraud. access),
	Attacker2	Low	Medium (Alter Data)
Insert malicious scripts	Attacker1	High	Low (Add URL),
	Attacker2	Low	Medium (spooky vis.)

safety AFT. In addition, we incorporate these APs for quantitative evaluation (i.e., in-depth analysis) to identify the most vulnerable components in the generated safety AFT of social VRLE applications. These APs can be further extended depending on the attributes considered for profiling.

**Modeling of basic attack step (BAS)** In each of the BAS, to evaluate the disruption values over various paths (i.e., attack steps) taken by attacker, we assign the probability weights and rate of exponential over the edges and nodes, respectively. For every attack scenario in BAS, the steps taken by the attacker are different depending on the attacker profile. To explain our approach in modeling a BAS, we consider the leaf nodes – *Impersonation*, *SQL injection*, *Insert malicious scripts* and *Denial of service* in our safety AFT. For instance, the STA

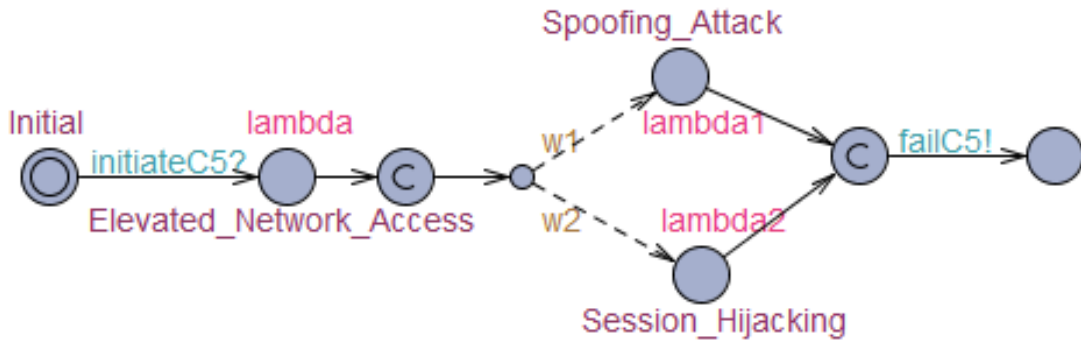


Figure 2.39: BAS for impersonation.

of the leaf node *impersonation* gets activated after receiving *initiate* signal from the parent node as shown in the BAS representation Figure 2.39. The BAS for impersonation node in Figure 2.39 is equipped with probability weights ( $w_1$ ,  $w_2$ ), rate of exponential represented as  $\lambda$ . After getting elevated access to the network, the attacker chooses a path (i.e., by performing a spoofing or session hijacking attack) with probabilities  $w_1/w_1 + w_2$ ,  $w_2/w_1 + w_2$ , respectively. After exploiting one of the paths as shown in the Figure 2.39, *fail* signal is sent to the parent node representing disruption of the impersonation node in the given attack-fault tree.

Another example of BAS of SQL injection is shown in Figure 2.40, where the logical steps taken by the attacker are outlined. To elucidate, the attacker can perform SQL injection in following ways: i) login as a fraudulent user, ii)



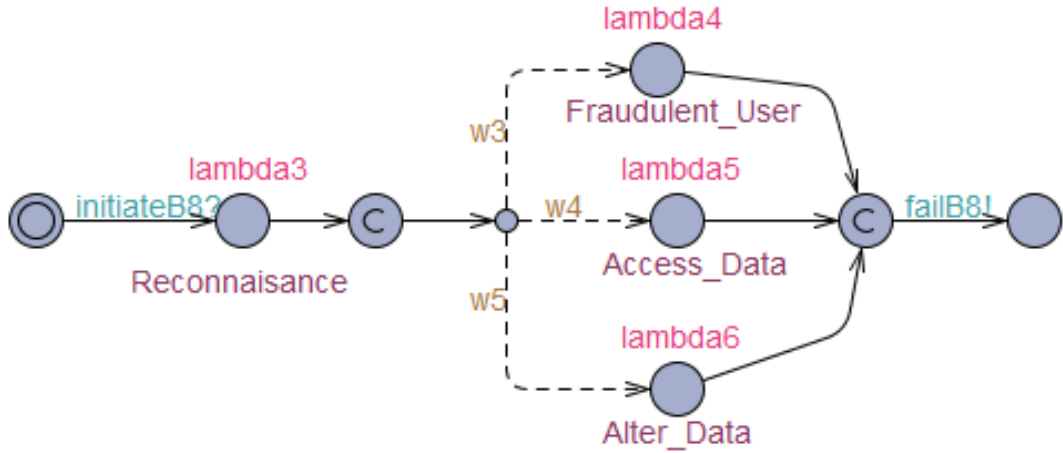


Figure 2.40: BAS for SQL injection.

Table 2.11: Values of  $\lambda$  given to the basic attack steps of the safety AFT

Basic attack steps of safety AFT	
Threat scenarios	$\lambda$
Impersonation	0.00004925, 0.00005466
SQL injection	0.0000502, 0.0000854, 0.0000492
Insert malicious scripts	0.0005389, 0.0006899
Denial of service	0.0003638, 0.0003084

access data, and iii) alter data, without having authorized access. In addition, the probability of taking different paths is described based on the probability weights i.e.,  $w_3$ ,  $w_4$  and  $w_5$  distributed over the edges. After disruption, *fail* signal is sent to the parent node representing disruption of the respective event.

Table 2.12:  $\lambda$  values for leaf nodes of security & privacy subtrees in safety AFT.

LoC subtree		LoI subtree		LoA subtree		Privacy leakage subtree	
Threat scenario	$\lambda$	Threat scenario	$\lambda$	Threat scenario	$\lambda$	Threat scenario	$\lambda$
Login to system	0.0089	Packet spoofing	0.0068	Network traffic	0.0000113	Ping sweeping	0.002162
User data	0.004162	SYNC flood	0.0068	Upload/down-load rate	0.0001	User physical location	0.0000078
Capture packets	0.00098	Modify sensitive data	0.002642	Unauthorized access	0.006478	Password attacks	0.08687
Analyze packets	0.0048	Change coordinates	0.002642	Low Bandwidth	0.000121	Capture Hostname	0.004162
Impersonation	0.006892	Identity spoofing	8.1219E-06			Intrusion	0.006628
User login	0.0089	Insert malicious scripts	0.008			Listen to conversation	0.08

## 2.6 Quantitative Analysis of Safety AFT

In this section, we present the quantitative results obtained from experiments that use our proposed framework. We quantitatively assess individual as well as combinations of leaf nodes in the safety AFT, to study how multiple SPS threats associated to cyber-attacks/fault-attacks affect the cybersickness occurrence. For this, we consider the sub-trees pertaining to threat scenarios – *LoI*, *LoC*, *LoA* and *privacy leakage* for the outlined safety AFT shown in Figure 2.28. In the following analysis, we assume that our design requirement is to keep the probability of cybersickness below the threshold of 0.25. For evaluation purposes, we use arbitrary values of  $\lambda$  as parametric input to the leaf nodes as shown in Table 2.12 obtained from [57, 58]. In addition, we give the  $\lambda$  value based on the weights of probability of distributions for each BAS as shown in Table 2.11.

Using these  $\lambda$  values as parameters to the leaf nodes, we utilize the SMC queries as explained in Section 2.1.4 to analyze and find the probability of cybersickness for the generated STAs. For our experimental purposes, we consider cybersickness (i.e., root node of safety AFT) as the goal node. Any other user-specified threshold values for different applications can also be used in our framework. This is due to the fact that the model checking approach takes the user-specified values at the beginning of an experiment. In addition, we also consider an error bound  $\epsilon$  value of 0.01 and 95% confidence interval for the calculation of probability of disruption. In the following set of experiments, we present the obtained probability of the goal nodes with respect to the time window used by the attacker.

### 2.6.1 Vulnerability Analysis in the safety AFT

With this quantitative analysis, we identify the most vulnerable components that can act as inducing factors for cybersickness in a VRLE session. For this, we assign the values of  $\lambda$  for the leaf nodes shown in Table 2.12. For the remaining leaf nodes in the safety AFT, we consider a very small positive constant ( $K$ )  $\approx$

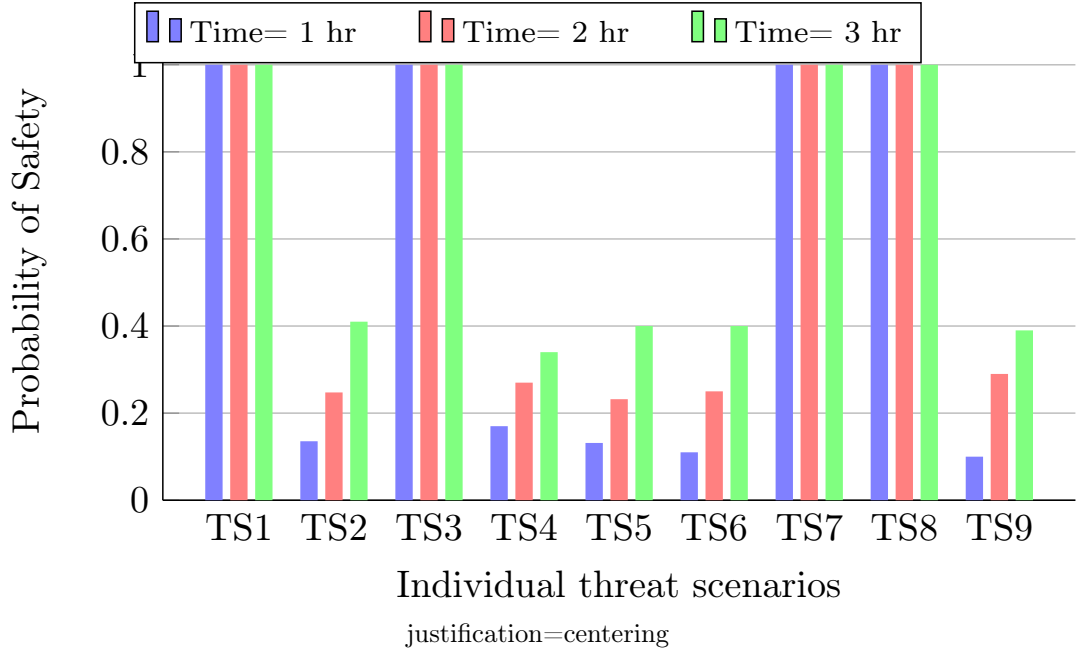


Figure 2.41: TS of safety AFT where -  $TS3$ ,  $TS7$ ,  $TS8$  are the most vulnerable nodes.

0.00001. This is because, in real world systems, multiple attack scenarios can happen. To identify a vulnerability in the safety AFT, we analyze: (i) individual leaf nodes, and (ii) combinations of leaf nodes, to determine their effect on the probability of CS occurrence.

### Individual leaf node analysis

In Figure 2.41, we show the probability of CS over multiple time windows for each leaf node in the safety AFT. We perform a thorough analysis of leaf nodes in the safety AFT for threat scenarios across different time intervals i.e.,  $t = \{1 \text{ hr}, 2 \text{ hr} \text{ and } 3 \text{ hr}\}$ . For the individual leaf node analysis, the considered threat scenarios (TS) shown in Figure 2.41 are termed as:  $TS1$  – Impersonation,  $TS2$  – User login,  $TS3$  – User physical location,  $TS4$  – Capture packets,  $TS5$  – Ping sweeping,  $TS6$  – Intrusion,  $TS7$  – Upgrade,  $TS8$  – Low bandwidth,  $TS9$  – Network fault. As shown in Figure 2.41, the leaf nodes  $TS1$  and  $TS8$  (for causing a DoS attack),  $TS3$  and  $TS7$  (for sensitive data leakage) are the most vulnerable in the safety AFT with the probability of 1. In addition, we also determine  $TS6$  that is considered as the next vulnerable node in the safety AFT.

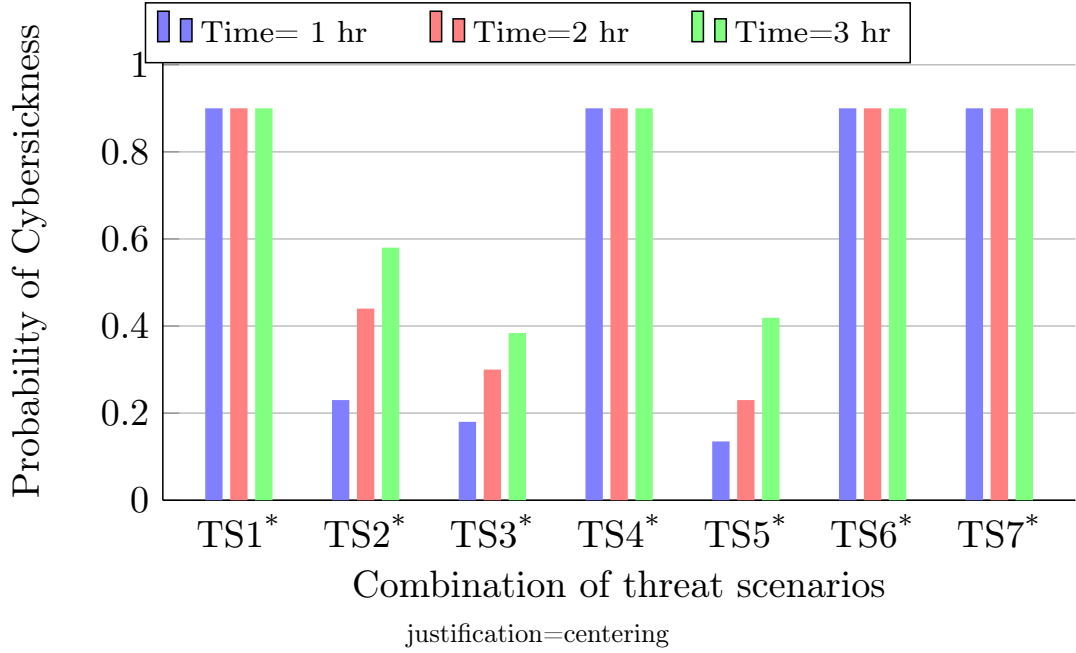


Figure 2.42: Combinations of TS of safety AFT where -  $TS1^*$ ,  $TS4^*$ ,  $TS6^*$  are the most vulnerable nodes.

### Combination of leaf node analysis

Herein, we consider combinations of leaf nodes to identify their impact on CS. For these experiments, we explore two scenarios: In the first scenario, we consider combinations of leaf nodes that belong to the same sub-tree (e.g., LoI). In the second scenario, we consider leaf nodes from different sub-trees (e.g., LoA and privacy leakage). The considered combination of threat scenarios are enlisted as:  $TS1^*$  - {Capture hostname, Ping sweeping},  $TS2^*$  - {Data tampering, user information},  $TS3^*$  - {Data tampering, faulty application behavior},  $TS4^*$  - {Low bandwidth, unauthorized access},  $TS5^*$  - {Sync flood, ping sweeping},  $TS6^*$  - {Intrusion, Listen to conversation},  $TS7^*$  - {Impersonation, Packet Spoofing}. As shown in Figure 2.42  $TS1^*$ ,  $TS4^*$ ,  $TS6^*$  and  $TS7^*$  are the most vulnerable combination of threat scenarios with a probability of 0.9 for a CS event. As part of further analysis in Section 2.7, we discuss about the potential candidates for design principles to apply on these leaf nodes such that the VRLE application resilience is enhanced against security threats.

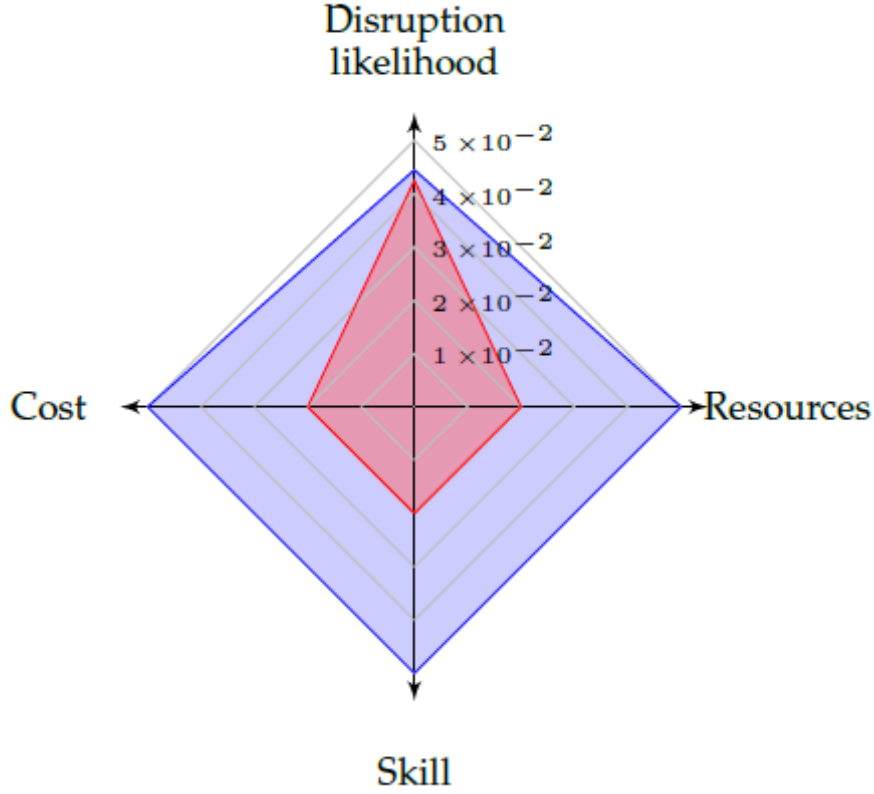


Figure 2.43: Probability of disruption of cybersickness for different attacker profiles.

### 2.6.2 Evaluation of attacker profiles in terms of impact on disruption of safety AFT

In this section, we analyze how APs impact the disruption of an intermediate node, root node (cybersickness) for the safety AFT shown in Figure 2.28. For instance, the Figure 2.43 considers the APs enlisted in Table 2.10 along with the AP attributes such as cost, resources, skills required and their associated likelihood of the cybersickness disruption in the safety AFT. From the graphical analysis, we determine that for  $time \approx 3$  hours, the disruption of likelihood of CS for *Attacker 1* is  $4.44 \times 10^{-2}$  and for *Attacker 2* is  $4.25 \times 10^{-2}$ . Similarly, in Figure 2.44, we also analyze the impact on disruption of an intermediate node  $\{Response\ Time\}$  in the LoA subtree of the safety AFT. From our results in Figures 2.43 and 2.44, we determine this change of probability of disruption between the considered APs is due to the change in the profile attributes i.e., skill, cost, time, resources. With this analysis, we consider both the attacker profiles in determining the vulnerable

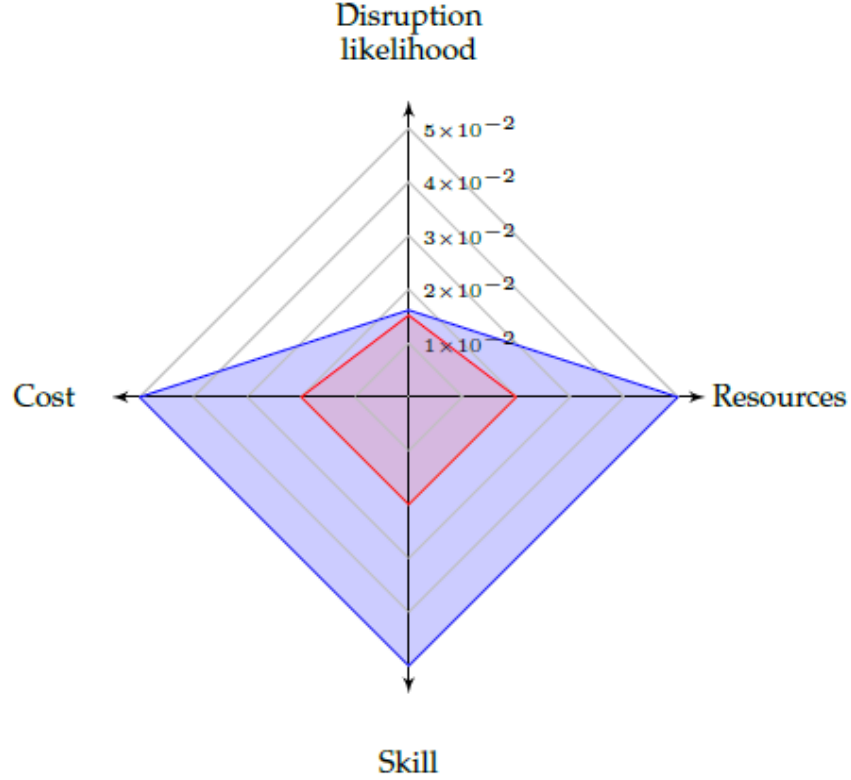


Figure 2.44: Probability of disruption of response time in safety attack tree for different attacker profiles.

components for a given social VRLE application design.

### 2.6.3 Risk assessment based on vulnerability analysis and attacker profiles

In this section, we perform risk assessment on the identified vulnerability components outlined in Section 2.6.1 for the safety AFT. To determine the severity of these identified threat vectors in a VRLE, we adopt a widely accepted NIST-based risk assessment procedure [28, 83]. Taking the vulnerability results into account, we calculate their associated risk values based on the formulation:

$$R = P(O) \times I \quad (2.1)$$

where,  $P(O)$  is the probability of occurrence,  $I$  is the level of impact in the event of attack occurrence and  $R$  is the associated risk.

In addition, we use the semi-quantitative scale based on the NIST SP 800-

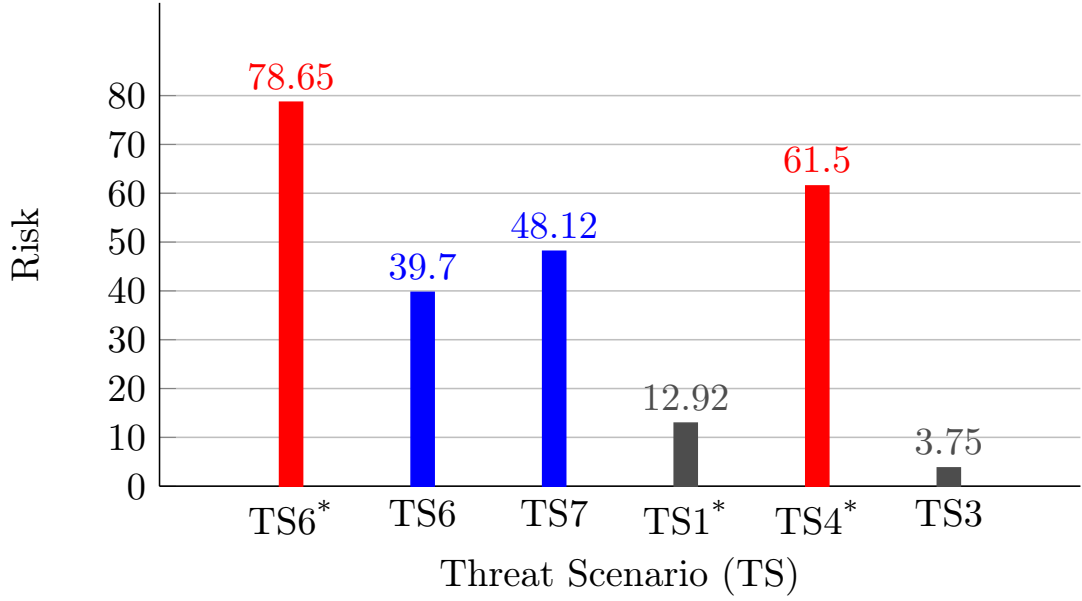


Figure 2.45: Risk assessment of threats affecting cybersickness.

53 [28] to categorize the risk levels as ‘High’, ‘Moderate’, ‘Low’ based on the  $I$  and  $P(O)$  values for a given threat scenario. For this, we determine the  $P(O)$  from our vulnerability analysis of the STAs related to the safety AFT discussed in Section 2.6.1. Similarly, we estimate the  $I$  using NIST guidelines in terms of impact on cybersickness based on our preliminary experiments results on CS levels as shown in Figure 3.2. The rationale behind such a calculation is to get the most conservative estimate of the critical components in the VRLE application. To categorize into the appropriate risk level for each of the threat scenarios, we first calculate the risk% using the Equation 2.1. To elucidate, the calculated risk% is assigned the risk level using a percentage scale from 0 to 100%, where  $\geq 100\%$  (*very high*),  $> 70\%$  (*high*),  $> 50\%$  (*moderate*),  $> 20\%$  (*low*),  $< 20\%$  (*very low*).

The risk assessment shown in Figure 2.45 details the risk value of each of the most vulnerable components considered as threat scenarios in Section 2.6.1. These threat scenarios –  $TS3$ ,  $TS6$ ,  $TS7$ ,  $TS1^*$ ,  $TS4^*$ ,  $TS6^*$  are enlisted on the X-axis and the Y-axis represents the associated risk%.

Based on our risk assessment results, we identify that  $TS6^*$  has a higher risk level with value 78.65%. Similarly,  $TS4^*$  also falls under high risk category when compared to  $TS3$ ,  $TS6$ ,  $TS7$  and other combinations. In addition, the risk levels

for  $TS6$ ,  $TS7$  are higher than the combination  $TS1^*$ , which is a privacy breach scenario. The low risk level we can observe for  $TS1^*$  can be considered as a potential risk factor if the hostname of the user is obtained for an attacker with higher AP. From our risk analysis, we can also determine that - among the considered threat scenarios, a *DoS* and a *man-in-the-room attack* cause higher risk for occurrence of cybersickness. Based on the above, we can see how SP attacks can be used to trigger novel attack scenarios such as an immersion attack, to induce undesired cybersickness levels. Thus from our above quantitative results, we can conclude that along with physiological conditions, the security and privacy related threats can act as major factors for inducing cybersickness in a VRLE session.

## 2.7 Recommended Design Principles

In this section, we examine the effect of applying various design principles to the most vulnerable components identified in the Section 2.6.1 for our safety AFT. Existing works such as NIST SP800-160 [28], [53] suggest that the services for safeguarding security and privacy are critical for successful operation of current devices and sensors connected to physical networks as part of edge computing systems. As mentioned in Section 2.1.5, these design principles are essential to construct a trustworthy edge computing based system architecture. The goal is to apply a combination of design principles at different levels of abstraction to help in developing effective mitigation strategies. We adopt a selection of design principles such as *hardening*, *diversity*, *Redundancy* and *principle of least privilege* among the list of principles available in NIST SP800-160. In the following, we demonstrate their effectiveness by showing that there is a reduction in the probability of disruption terms after adopting them in a VRLE design.



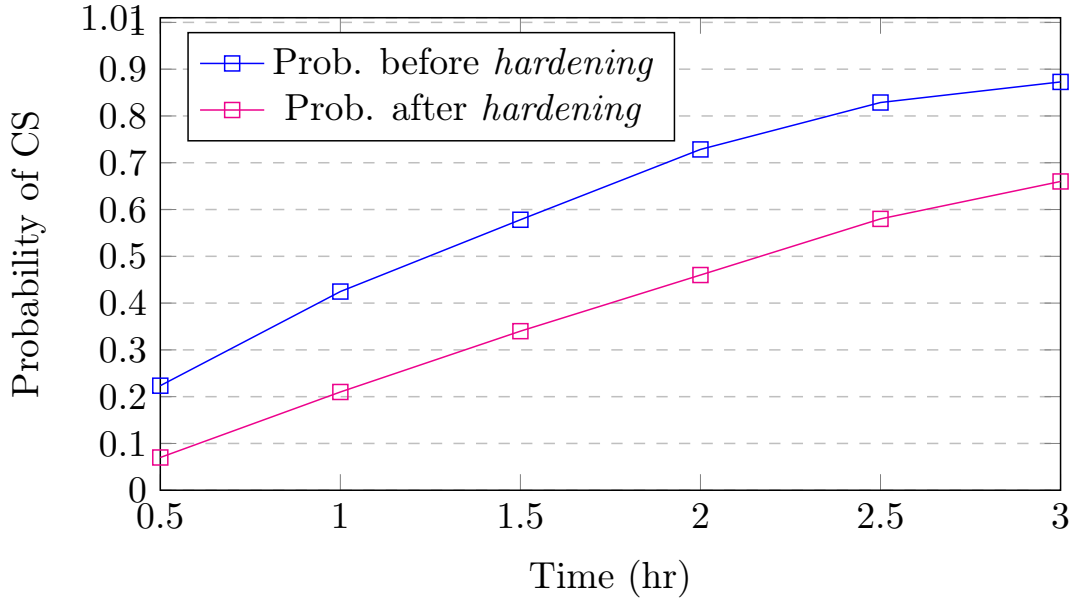


Figure 2.46: Pr reduced by 24.14% in safety AFT due to application of hardening design principle.

### 2.7.1 Implementation of design principles on safety AFT:

To study the effect of design principles on the cybersickness occurrence, we incorporate *hardening*, *redundancy* as shown in Figures 2.46 and Figure 2.47 relating to the safety AFT. As part of *hardening* principle, we added new nodes such as a firewall and a security protocol to analyze the impact of the most vulnerable nodes triggering a DoS attack in the safety AFT. Our results show that the probability of CS disruption is reduced from 0.87 to 0.66 (24.14%), with the given attacker profile.

The decrease in the disruption of CS is due to the increase in the resource requirement for an attacker to compromise a VRLE application which is incorporating the *hardening* principle. Similarly, we apply: (i) *redundancy* on the safety AFT that adds more components with similar functionalities intentionally which in turn reduces the probability of disruption of cybersickness by 2.6% as shown in Figure 2.47, and (ii) *principle of least privilege* where the probability of cybersickness in the safety AFT is reduced by 2.61%.

Thus, from the above implementation of individual design principles, we can

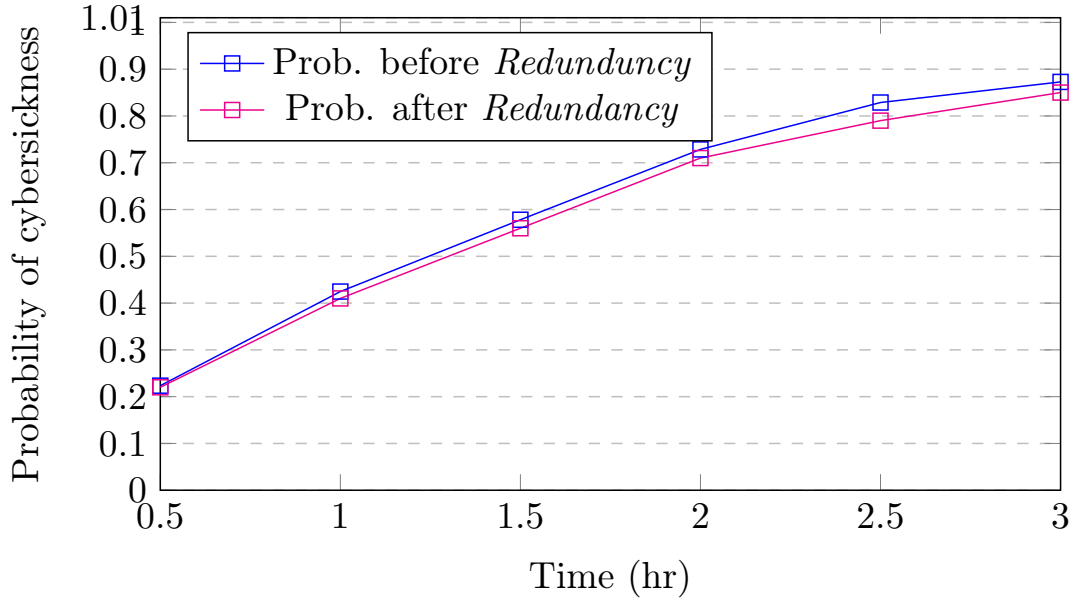


Figure 2.47: Pr reduced by 2.6% in safety AFT due to application of redundancy design principle.

observe that *hardening* is more effective in reducing the disruption of cybersickness compared to any of the other design principles. In addition, our results demonstrate the benefits in implementing a combination of design principles in safety AFT to overall improve the attack mitigation efforts.

To study the effect on disruption of the cybersickness, we adopt a combination of design principles for the safety AFT such as: (i)  $\{hardening, principle\ of\ least\ privilege\}$ , (ii)  $\{hardening, redundancy\}$ , and (ii)  $\{Redundancy, principle\ of\ least\ privilege\}$ . We observe that there is a significant drop in the probability of disruption of cybersickness from 0.87 to 0.62 (28.96%) due to  $\{hardening, principle\ of\ least\ privilege\}$  as shown in Figure 2.48. Similarly, Figure 2.49 shows the effect of combination of  $\{hardening, redundancy\}$  that results in reducing the disruption of CS with 25.2% in a safety AFT. In addition, we apply the  $\{redundancy, principle\ of\ least\ privilege\}$  combination, which results in a reduction of cybersickness by 3.05% as shown in Figure 2.50.

Finally, we evaluate the effect of all the combined design principles on safety AFT. The Pr of the cybersickness for the safety AFT is reduced by 35.18% as shown in Figure 2.51. From the above numerical analysis, we can conclude that in-

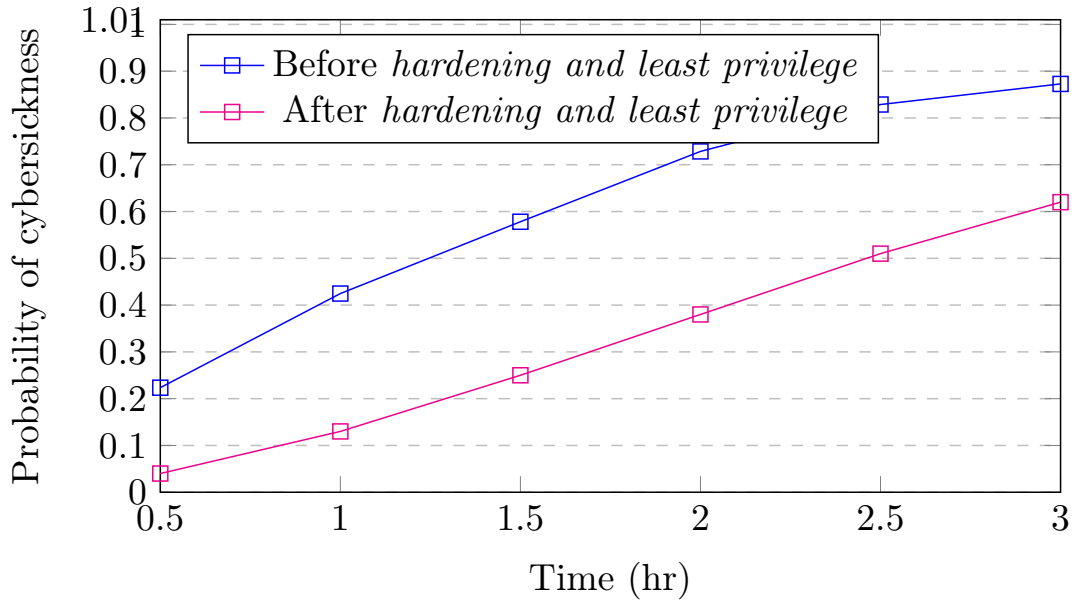


Figure 2.48: Pr reduced by 28.96% in safety AFT due to application of hardening and principle of least privilege design principles.

corporating relevant combination of standardized design principles and their joint implementation have the potential to better mitigate the impact of sophisticated and well-orchestrated cyber attacks on edge computing assisted VRLE systems with wearable devices. In addition, our above results provide insights on how the adoption of the design principles can provide the necessary evidence to support a trustworthy level of security, privacy and safety for the users in VRLE systems that are used for important societal applications such as: special education, surgical training, and flight simulators.

## 2.8 Conclusion

Social Virtual Reality Learning Environments (VRLE) provide immersive experience by delivering online content to distributed users. Currently, VRLE applications are being adopted in various application domains, however the related security, privacy and user safety (SPS) issues are under-explored. In this paper, we present a novel quantitative framework to analyze potential security, privacy issues that induce *cybersickness* (a safety issue) in a VRLE session. With our preliminary

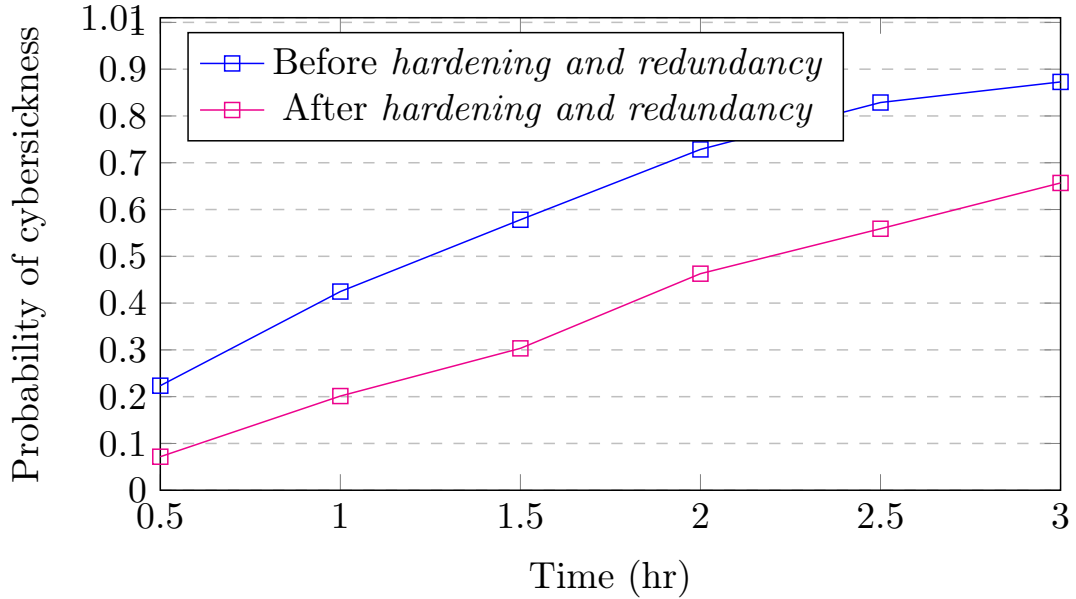


Figure 2.49: Pr reduced by 25.52% in safety AFT due to application of hardening and redundancy design principles.

experiments that considered a social VRLE application case study viz., vSocial, we demonstrated the disruptive effects of various cyber-attacks/fault-attacks based SPS issues (i.e., DoS, eavesdropping via man-in-the-room) on cybersickness levels. We utilized attack-fault tree formalism to model the security issues (i.e., LoC, LoI and LoA scenarios) and privacy issues (i.e., privacy leakage) inducing cybersickness in a VRLE session. Specifically, we developed relevant attack-fault trees and converted them into stochastic timed automata and then performed model checking using the UPPAAL SMC tool.

Using our proposed framework, we determined causes of: *DoS attack*, *data leakage*, *man-in-the-room attack* and *unauthorized access* as the most vulnerable components that can induce higher level of cybersickness in VRLE sessions. By using the NIST SP800-16 risk assessment method, we determined the severity of these identified threat scenarios (i.e., critical issues) in terms of impact on cybersickness and degradation of application functionality. Furthermore, we illustrated the effectiveness of our framework by analyzing different design principle candidates. We showed a ‘before’ and ‘after’ performance comparison to investigate the effect of applying suitable design principles to reduce the probability of cy-

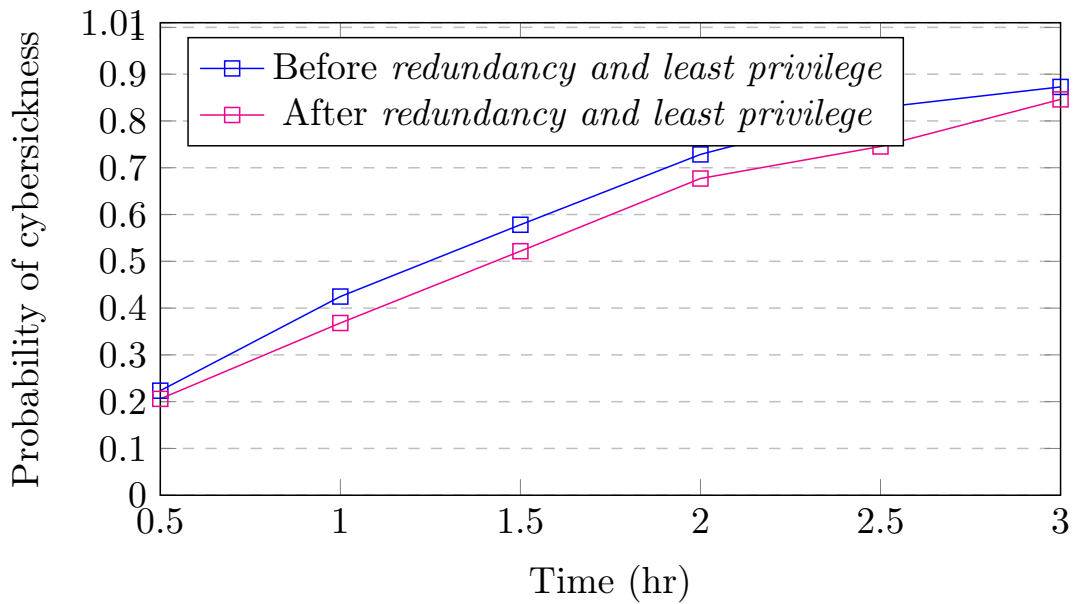


Figure 2.50: Pr reduced by 3.05% in safety AFT due to application of redundancy and principle of least privilege design principles.

bersickness occurrence. From our experiments with the vSocial application, we determined that the choice of a suitable design principle pertaining to the most vulnerable threat components is significant for use in an attack mitigation strategy. Specifically, we observed that implementing a combination of design principles can result in a more effective mitigation strategy. Among the design principle candidates, we found: (i) *{hardening, principle of least privilege}*, (ii) *{hardening, redundancy}* were the most suitable combinations for reducing the probability of CS occurrence with an average of 29%.

As part of future work, evaluation of different attack-defense strategies and potential performance adaptations can be performed. This can help to assess their effectiveness in mitigating the cybersickness occurrence, to ultimately ensure a safe, trustworthy and high-performing VRLE to the application users.

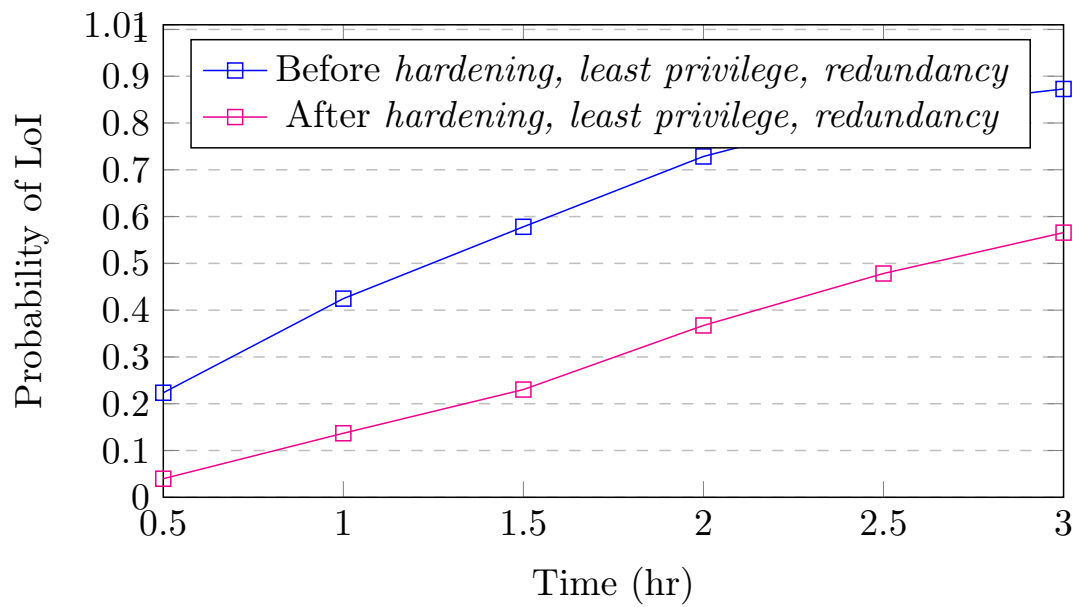


Figure 2.51: Pr reduced by 35.18% in safety AFT due to application of hardening, redundancy and principle of least privilege design principles.

# Chapter 3

## Threat Intelligence Collection for Critical Anomaly Event Detection

### 3.1 Attack Issues Disrupting Immersion Factors

The adoption of virtual reality in IoT-based infrastructure merges the physical and the digital world to create a new form of an interactive environment [84]. We collect the SP attack data related to potential vulnerabilities in VRLE via different subjective assessments. In this section, we formulate our problem by stating the relationship between potential SP attacks and UIX factors in two phases: (i) modeling of potential SP issues in VRLE that disrupts users' UIX, and (ii) analyzing the relationship of the impact of SP attacks on UIX due to the vulnerabilities in VRLE.

Using the attack tree formalism discussed in our works [16], [26] we model the potential SP attacks that disrupts UIX factors as leaf nodes and the root node respectively as shown in Figure 3.1. We generate this *SP-UIX attack tree* whose root node (i.e., disruption in UIX) is the goal of an attacker, intermediate nodes are the potential attacks and the leaf nodes as threats causing such SP attacks. The attacks listed in this SP-UIX attack tree can be generated in various ways based on the attacker profile. In our study context, we generate this SP-UIX

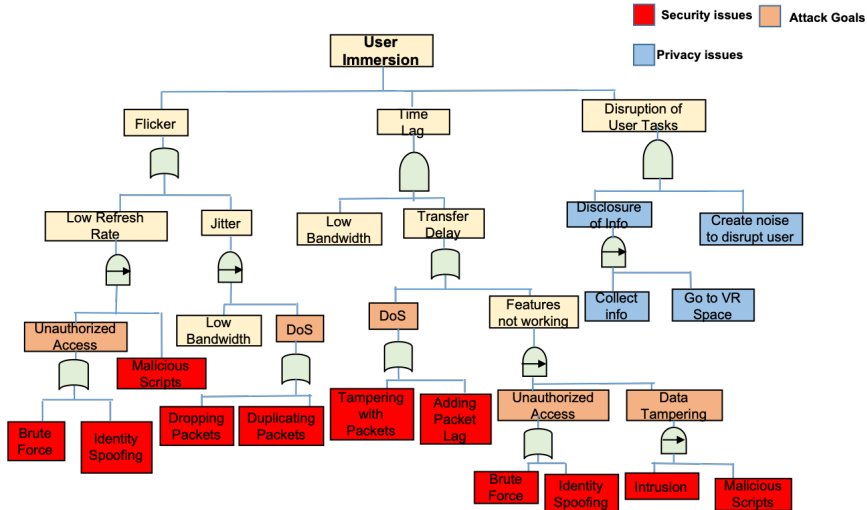


Figure 3.1: Formalized immersion attack tree with logical gates attack tree by validating the potential threat scenarios from our prior work in [85] and penetration testing on the vSocial application as explained in Section 3.4. Moreover, the SP-UIX attack tree shows how an attacker can take advantage of a vulnerability (i.e., leaf node) to trigger an anomaly event (e.g., packet spoofing, tampering packets), which can in turn lead to an attack scenario (i.e., intermediate nodes such as e.g., Denial of Service) as shown in Figure 3.1.

### 3.1.1 Quantifying Immersive User Experience

In order to understand the impact on UIX factors due to the listed potential SP attacks (individual attacks and combination attacks), we set up a vSocial instance with three different activities for users to perform as shown in Figure 2.24. These activities are part of the learning curriculum of the vSocial environment [85] and rely on fine movement, visual clarity, and clear audio, all of which are disrupted by simulated attacks. We perform an experimental behavior analysis based on one of our earlier works [85] to assess significant factors that disrupt UIX in a VRLE. We simulate three attacks across the activities and quantify UIX using a set of virtual questionnaires (VQs) as shown in Figure 2.24.

After the orientation of the vSocial, the test subject participates and their feedback is collected at the end of activity 1 which is the baseline data for a nor-



Table 3.1: Survey questions asked during immersion experiment

Label	Immersion Question	Original Survey	Related Works
Mental Engagement	How mentally engaged are you in the virtual environment?	ITQ	[86][72]
Comfort	Do you feel comfortable in the virtual environment?		[87]
Smoothness of Movement	How Smooth is movement in the virtual environment?	PQ	[86][72]
	<b>Usability Question</b>		
Accuracy of Controls	Are the controls sensitive to your movement and capable of performing tasks?	PQ	[86][72]
Visual Clarity	Are you able to see the games clearly?	PQ	[86][72]
Sound Clarity	Are you able to hear the sounds in the virtual environment?	PQ	[86][72]

mal functioning VRLE without any attack scenario. Next, the subject proceeds to activity 2, where a security attack is simulated with a data packet drop that disrupts the rendering of the VRLE [26], after which user feedback is collected using a VQ. The user then proceeds to activity 3, where a privacy attack is simulated to capture the user’s virtual location and a distracting noise is played such that the user’s learning experience is disrupted. We stop this disruption for the VRLE user approximately halfway through activity 3, where the user response is recorded using another VQ. For the rest of activity 3, we simulate a combination of security and privacy attack (packet tampering and disclosing the user location), after which the users exit the vSocial environment to submit their feedback via a post-session survey.

We run the attacks until the next checkpoint is reached i.e., if the attack duration reaches 180 seconds, we stop the attack simulation to avoid further discomfort to the user in relation to their time spent in the VRLE. The surveys conducted for this attack behavior analysis using VQs and post-session paper format are used to measure each of the UIX factors i.e., usability of the application and sense of immersiveness which the VRLE users feel. These VQs are integrated into the VRLE such that they do not cause any disruption to the users [79]. We model the questions in these VQs based on existing works and surveys [72] [86], such as the Presence Questionnaire (PQ) and Immersive Tendencies Questionnaire (ITQ).

For the purpose of our data analysis, we use usability and immersiveness based questions adapted from [72, 86, 87]. To maintain uniformity across the collected feedback data, we used the same questions related to UX factors in every VQ as shown in Figure 3.2. We recruited 15 human subjects to participate in this UX survey and obtained their feedback for each of the questions present in our UX survey on a scale of 1 to 5, where 1 (very poor), 2 (poor), 3 (Moderate), 4 (good), 5 (excellent). This collected data allows us to quantify the aggregated value of UX scores in the VRLE.

### **Results of the analysis about the impact of SP attacks on UX factors**

To investigate the impact of SP attacks on UX factors, we first study whether our chosen usability and immersion related questions are dependent on each other. Based on the definition of UX factors (usability of the application and sense of immersiveness), we perform several statistical correlation tests (i.e., Pearson, Kendall and Spearman) for the data collected using these set of questions. From our statistical correlation analysis, we observe that the correlation coefficient for different tests between usability and immersion is high  $\geq 0.50$ . For the purpose of the statistical analysis, we consider a null hypothesis as follows: *the usability and immersion data are independent of each other*. However, the obtained p-values are very low, which supports the rejection of our proposed null hypothesis, thereby concluding that immersion and usability factors are indeed correlated.

Based on our data collected for the SP versus UX factors analysis, we outline the data points for each question versus the average rating given by the users as shown in Figure 3.2. We observe that the security, privacy, and combination of SP attacks have significantly impacted both the immersion and usability factors. From these results, we understand that SP attacks do certainly affect UX factors and cause disruption in the functionality of the VRLE. With this analysis and our modeled SP-UX attack tree, we further explore the threat signatures to use for detection of such attack scenario before they disrupt a user's UX in a VRLE.

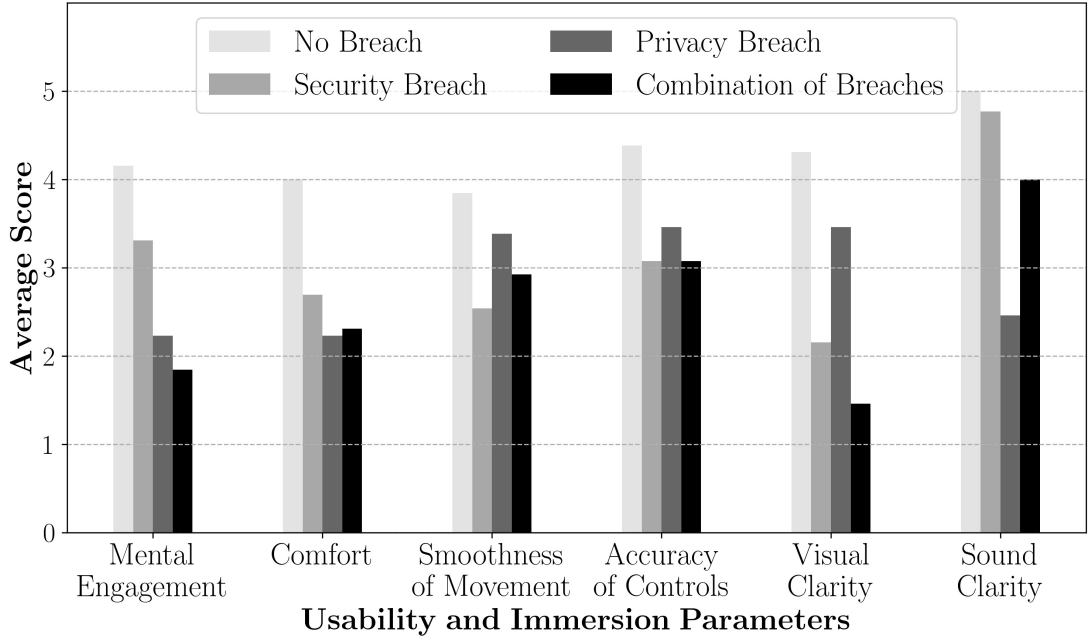


Figure 3.2: Results of UX survey experiments.

### 3.2 Anomaly Detection Module

Due to the dynamic interactions in real-world VRLE applications, the SP attacks (individual and combination of attacks) can create diverse attack signatures/-patterns. In order to continuously learn from such VRLE system behavior and mitigate the impact of SP attacks on UX factors, we propose a novel anomaly detection method. Our approach is based on DevSecOps [88] principles of diverse solutions for change (i.e., attack, fault) detection in resilient systems. Although there are several existing anomaly detection methods, our methodology considers events in human-computer dynamic interactions that are specific to VRLEs involving spatially distributed users. These events can be caused due to a new set of attack surfaces that are not seen in traditional network systems. For instance, a concurrent/consequent multi-attack scenario can uniquely disrupt UX and user safety in a real-time VRLE application. An example of a *multi-attack* scenario can be seen in a case where sensitive user information has been disclosed (*privacy attack*) by gaining *unauthorized access* to the data (*security attack*) in a social VRLE application. In this section, we present an overview of our novel anomaly detection method shown in Figure 3.3 that is applicable to VRLEs and

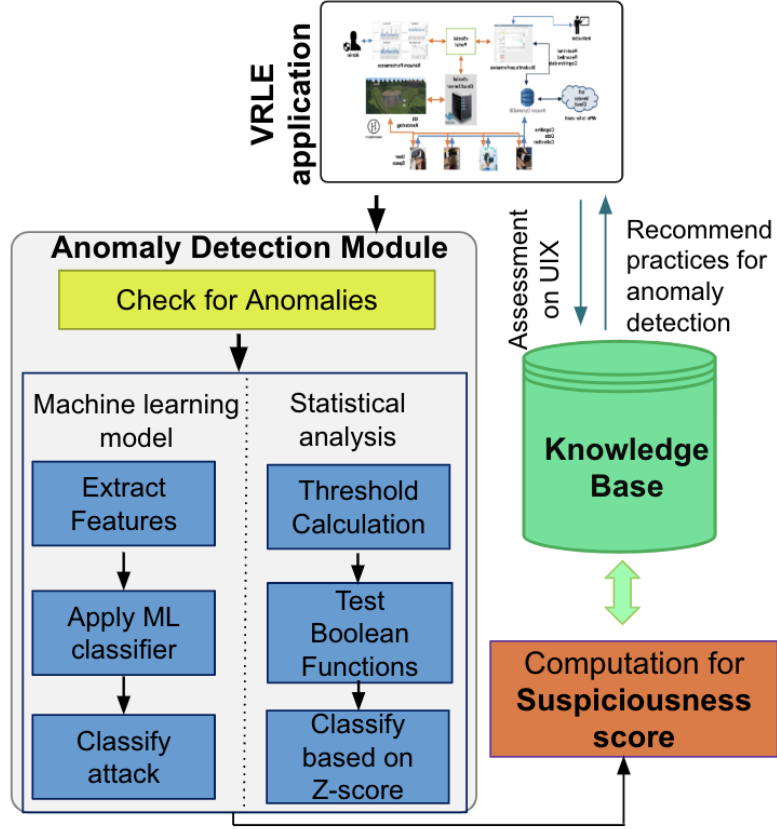


Figure 3.3: Anomaly detection method to address SP attacks on VRLE(s). addresses security and privacy attack events that uniquely impact human and system interactions. We categorize our approach into two main modules: i) *Anomaly Detection module* to check and classify for anomaly events in the incoming data, ii) *Computation of Suspiciousness Score*  $SS_{attack}$  for the detected attacks in the VRLE data.

The primary goal of our anomaly detection method is to enable the creation of a secure and reliable IoT based VRLE that can: (a) identify vulnerabilities against cyber-attacks, and (b) avoid the disruption of UIX caused due to anomaly events during operational use. We highlight our attack detection shown in Figure 3.3 as part of a monitoring system that analyzes relevant data (user, system information) to detect any cyber-attack events. Our anomaly detection is based on DevSecOps [88] principles of diverse solutions and thus utilizes an ML classifier and statistical analysis for classifying the detected anomalies into specific attack types. In a VRLE application setting, our anomaly detection approach runs the

ML model to analyze the incoming data from VRLE sessions, whereas the statistical analysis continuously monitors the VRLE to identify and recursively trace the origin of the attack.

### 3.3 Anomaly Detection using Machine learning (ML) techniques

Once an anomaly is detected, our attack detection module uses ML techniques in order to classify the type of network-based attacks. Incorporating ML techniques can aid in automating the detection process and security analysis (e.g., malware detection, network log analysis, vulnerability assessment) in VRLE applications [89]. We utilize ML techniques mainly to deal with the continuous input data (VRLE session, network information) to analyze anomaly events in VRLEs and correlate them to user and system behaviors. In addition, ML helps us parse through large sets of data logs that are generated from multiple IoT devices (headsets, controllers, emotion headbands) during user-system interactions in VRLEs. Further, ML can be integrated into functionalities such as network performance and user emotion monitoring, and anomaly events tracking across VRLE sessions. Based on these motivations, we implement a ML classifier in our proposed anomaly detection module that can accurately differentiate normal (benign) behavior from a number of network-based attacks (e.g., DoS). Even though these network attacks generate a lot of data, these attacks share common traits in features [90] as shown in Figure 3.6, which in turn suggests a good fit for applying ML techniques.

Our attack detection ML technique shown in Figure 3.3 mainly consists of two steps: (i) pre-processing, and (ii) detection of the attack type. To detect and classify the network-based attacks, the pre-processing step considers the packet data and the user data for building an ML classifier. Based on pattern analysis, our ML technique can categorize the baseline data to attack type. The *baseline data* refers to the data collected from the VRLE components and from the user

interactions in vSocial without any attack scenarios (benign behavior). In addition, SP issues related to application-based attacks evolve with varied behavior patterns and diverse features which can make the labeling of the data infeasible in real-time. To address this different feature dimension issue, we focus our ML classifier only for network-based attacks with common feature traits.

Next, we establish the effectiveness of our proposed anomaly detection method. We evaluate the performance of our two mechanisms (ML classifier, Z-score analysis) in the detection module using both numerical simulations and event-driven experimental testbed evaluations based on the vSocial use case. Our evaluation goals are: (i) validation of different security and privacy attacks (individual and combination), (ii) detection of network and application based attacks (iii) calculation of suspiciousness score for each of the detected network and application-based attacks, and their impact on UX scores using our proposed anomaly detection method in realistic settings. To demonstrate our anomaly detection effectiveness for each targeted attack type, we start by describing our testbed configuration, followed by the data collection efforts for the various attack experiments, and finally conclude with discussion of the obtained results.

### **3.3.1 Attack Data Collection in Testbed Setup**

#### **Testbed setup**

The realistic, Open Cloud [91] testbed that we used in our anomaly detection experiments is shown in Figure 3.4. The testbed contains two software-defined networking (SDN) switches, a root switch, and a slave switch. The slave switch is attached to nodes (users and attackers) and a connection to the root switch. Moreover, the root switch is connected to an elastic virtual machine (VM), which could serve as a candidate for hosting the target application (i.e., the vSocial portal) that could be compromised by the attackers. All switches are connected to a unified SDN controller located in the cloud service provider domain, which directs

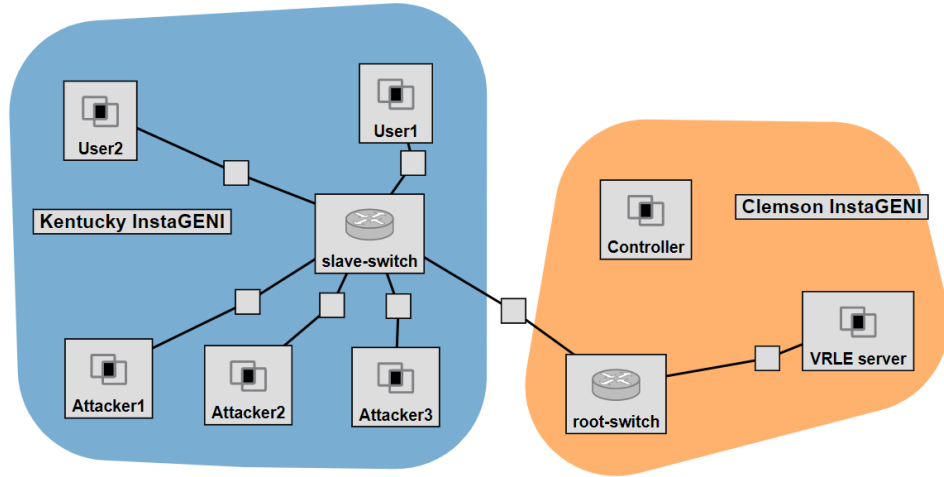


Figure 3.4: VRLE architecture implemented in an OpenCloud testbed setup. the policy updates. We can extend this VRLE setup to many more switches and devices as necessary. In our testbed setup, the three attackers try to disrupt the server functionality, hosted on the VM connected to the root switch. We use Frenetic [92] to link the nodes together and subsequently implement a `slowhttpstest` script in order to simulate a DoS attack in the Open Cloud testbed. Before illustrating all the simulated SP attacks, we first discuss the tools used as part of performing attack scenarios on the vSocial application client.

### Tools Used

As part of simulating several SP attacks, we used a Kali Linux [93] VM specifically with a few built-in tools such as Metasploit for port scanning and running attacks using the CVE database [94], and vulnerabilities related to unauthorized access found in Trivial File Transfer Protocol (TFTP) [95]. In order to simulate a DoS attack on vSocial, we used Clumsy0.2 [74] and Wireshark [75] to capture the packet rates, where the attacks were run against a vSocial client and the corresponding vSocial server. Using the above-specified tools and the testbed setup, we generate the attack data for network-based attacks (i.e., DoS) and application-based attacks (i.e., Unauthorized access).

## Attack simulations

In order to generate the attacks listed in the SP-UIX tree shown in Figure 3.1 related to VRLE application, we perform SP attacks. Using the leaf nodes as vulnerabilities in the SP-UIX attack tree, we generate SP attacks such as DoS (Packet Drop, Packet Tampering, Packet Duplication), Unauthorized access (to obtain local files).

**a) Unauthorized access attack:** As part of gaining unauthorized access, a password attack is executed using a Brute-force method. This attack can impact the integrity of the VRLE if the attacker gains administrator level access and discloses the user information or tampers the content in the VRLE. Our proposed anomaly detection module detects this form of unauthorized access to avoid potential privacy breaches as detailed in Section 3.2. Moreover, an attacker requires considerable amount of resources to gain unauthorized access via brute-force. In addition, the likelihood of a server being highly vulnerable would be low. In order to obtain the local files from such a server, we determine based on the existing work [85], the amount of resources required to go from medium to high.

## Attacker Profiling for attack simulations

Based on the work discussed in [17], we perform a *XSS browser attack*, where the web entities in vSocial are targeted to hook the browser for hijacking the VRLE content. In order to simulate this attack, the attacker needs to be highly skilled and requires significantly more resources as it concerns with the VRLE content in web entities. Another type on SP attack is to *overlay* and replace visuals with offensive content on a user's local machine which can partially compromise the VRLE [17]. Performing an overlay attack along with an SQL injection can create even more impact on user privacy as the confidential information can be captured via overlay entities. Another form of privacy attack, packet sniffing can create a privacy breach if the captured packets can be decrypted. The probability of success of such SP attacks are dependent on the attacker profile.



Table 3.2: Attacker profiling with estimated level of impact.

Attack	Resources/ Skill	Level of Impact <sup>ϕ</sup>	Impact Scale <sup>ϕ</sup>
Unauthorized Access	High	5	1. Normal System Functionality 2. Minor Decrease in System Functionality 3. Unusable System Functionality 4. Partially Compromised System 5. Fully Compromised System
Obtaining Local Files	Medium	4	
Dropping Packets <sup>1</sup>	Medium	3	
Duplicating Packets <sup>2</sup>	Medium	3	
Tampering Packets <sup>3</sup>	Low	5	
XSS Browser Attack	High	3	
Overlay Attack	Medium	4	
Packet Sniffing	High	4	
Malicious Script Execution	High	3	
Modification of Chaperone	Medium	5	
<sup>1</sup> 80% Drop Rate <sup>2</sup> 20x Rate, 40% Likelihood <sup>3</sup> 20% Tamper Rate			

Using the observations (level of impact, resources required) through our SP attack simulations, we develop the attacker profiles shown in Table 3.2 for every SP attack simulated as part of the SP-UIX tree validation. In this study, we adapt the definition of attacker profile from several existing works [81] as a collection of relevant characteristics of an attacker (such as e.g., skills, resources, motivations/-goals) and the associated level of attack impact. We estimate the level of impact using a uniform scale of 1 to 5 which is categorized based on the number of VRLE components that get disrupted during an SP attack scenario. These attacker profiles contribute primarily to the calculation of the suspiciousness score discussed in Section 3.4. However, these attacker profiles can be extended for risk management and deploying defense mechanisms on VRLEs in the future. Based on the simulated SP attack data (available at [96]), using the attacker profiles presented in Table 3.2 and the modeled SP-UIX attack tree, we illustrate the evaluation of our detection approach employed in the vSocial application.

With the collected data from different SP attacks generated, our proposed anomaly detection uses a two-stage ensemble learning scheme from [85]. The first stage includes attack (anomaly event) detection whereas, the second stage is about the classification of these events into specific attack types. We collect a

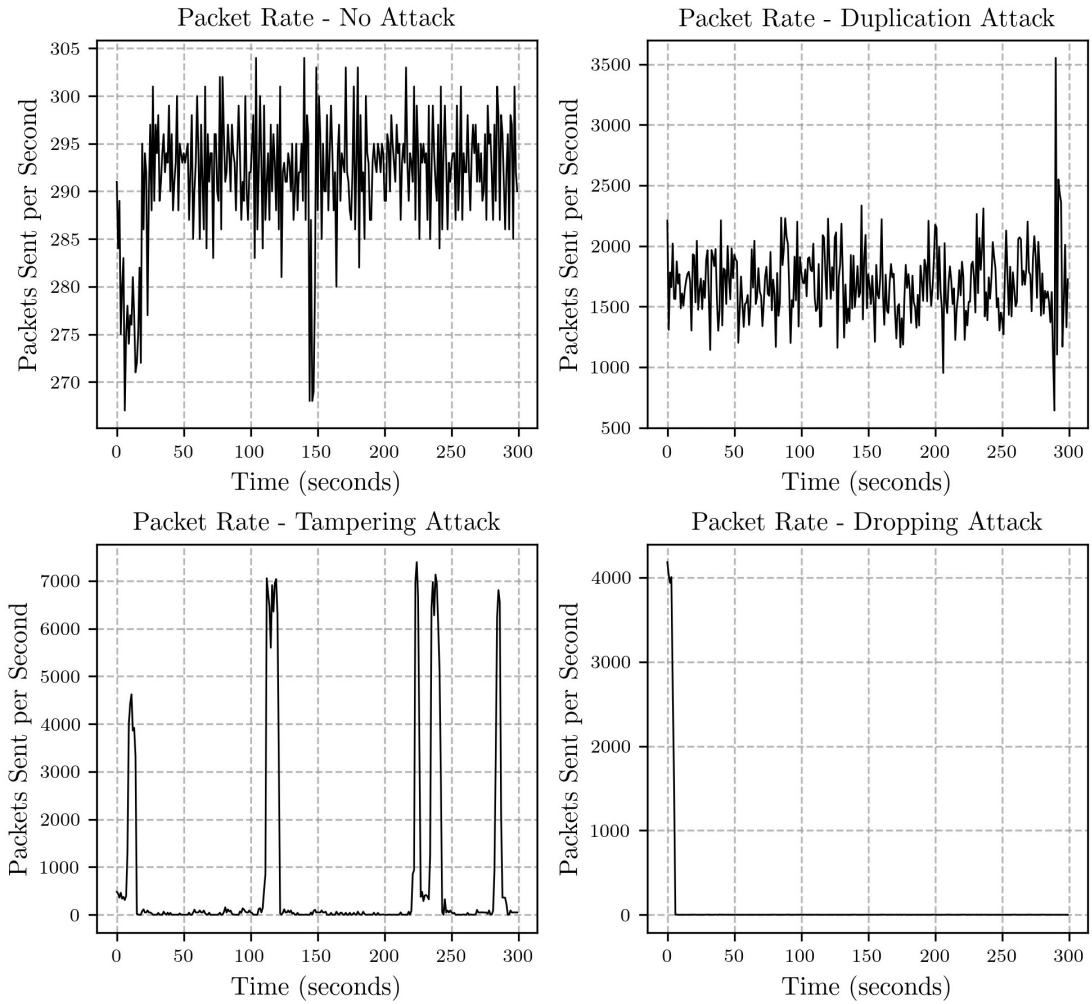


Figure 3.5: Packet rate time series comparison for single attacks.

significant amount of data to utilize in our two-stage ensemble learning scheme for attack detection and classification effectively. For evaluation purposes, our proposed approach detects and classifies DoS attack (network-based attack) using a machine learning classifier, Unauthorized access (application-based attacks) by performing Z-score statistical analysis. We also illustrate the potential privacy breach that is caused due to unauthorized access in vSocial. We then present results from these two different implemented techniques by considering single SP attack scenario, and a combination of several SP attack scenarios to show how our anomaly detection approach can be used in real-time.

## ML on a single label network-based attack dataset

Herein, we detail the pre-processing of the data collected, feature extraction and the subsequent attack detection steps for detection of single and multi-labeled network-based attacks using ML techniques.

### 3.3.2 Data Preprocessing and Feature Extraction

*Data Pre-Processing for our ML-based approach:* To identify and accurately differentiate benign data from a number of selected network attacks, our proposed anomaly detection method aims to develop a pertinent machine learning classifier. We generate training and testing datasets by sampling different types of network-based attacks (i.e., DoS), and also the causes of DoS attack (i.e., packet tampering, packet drop, packet duplication). In this study, we use Clumsy0.2 for generating these network attacks where we consider that these anomaly events occur with malicious intent. On the other hand, `scikit-learn` [97] and `scikit-multilearn` [98] are used for implementing the classifier models.

In our pre-processing step, we opt for a low-demand data collection approach. As part of data collection, we monitor network activity during: (i) normal time periods, and (ii) while performing each of the three DoS attacks such that all this data is updated into the Knowledge Base. We calculate the number of packets sent per second over a span of time by capturing the raw data associated with the timestamp of each packet. With this, we construct a dataset of packet rates captured over a time span for each attack type, thereby initializing a time series classification problem. In this study, we consider that the samples are collected in a uniform distribution such that each class is distributed uniformly.

The considered data set includes four classes: Normal (benign) behavior class and the remaining three classes are DoS attack events (packet duplication, packet tampering, and packet dropping). For each class, we collect 40 samples of 15-second sequences, where each sequence contains a packet rate at 1-second intervals.

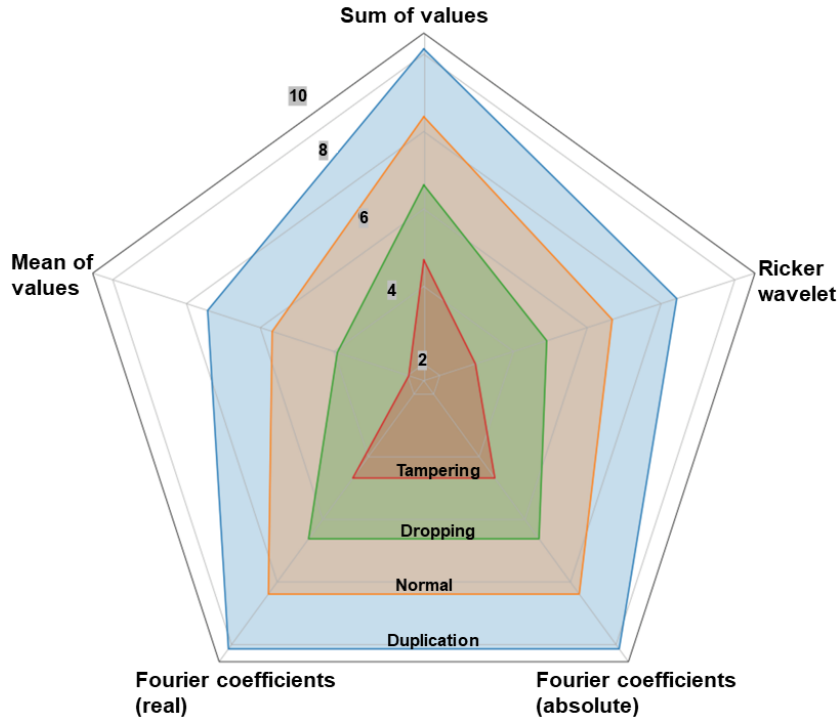


Figure 3.6: Five of the most distinguishing extracted features.

Due to data collection at equally spaced time intervals (1 second) i.e., packets rate per second ( $PRS$ ), we can model the data effectively as time series data, where each class has its own individual sequence. We support our statement by analyzing how the time series of the packet rates for each class contains distinguishing features as shown in Figure 3.5. This motivates our work to use this time series data to train an effective machine learning model that can take packet rate data at a sequence of time as an input and accurately label the data as one of the four attack data classes.

As part of feature extraction, we use `tsfresh` [99], a time series feature extraction tool that can be used for translating the time series data to a data format that can be utilized for training traditional machine learning models. Using `tsfresh` on the packet rate data, 212 relevant statistical features are generated. The five most distinguishing features among the 212 relevant returned features that clearly differentiate classes from one another are outlined in Figure 3.6. We use these distinguishing five features – *Sum of Values*, *Mean of Values*, *Ricker Wavelet*, *Fourier Coefficient (real)*, and *Fourier Coefficient (absolute value)* for training a ML classifier.

### 3.3.3 Accuracy Analysis for Suitable ML Model

Before we implement a ML classifier, we perform several tests based on quantifiable metrics that suits our data. We repeat the feature extraction step on multiple ML classifiers to aid in developing an algorithm for an accurate detection approach. We implement the following classifiers i.e.,  $\{Decision\ Tree, Random\ Forests, K\ Nearest\ Neighbors, Ensemble\ Voting\ Classifier\}$  [97] by adapting from the `sklearn` python machine learning library for the best-fit analysis. For each of the classifiers, we run 1000 tests, each of which included a random train-test split designed to prevent overfitting. Each test consists of training the model with 112 samples and testing it on 48 samples. There are several performance evaluation metrics that can be used for a classification exercise. We specifically use performance metrics such as *average time*, *precision* and *recall* that are computed to decide the suitable classifier for our data collected in order to classify a sample as an attack or a benign data class. We calculate the considered performance metrics as follows:

$$Recall = \frac{TruePositives}{FalseNegatives + TruePositives}$$

$$Precision = \frac{TruePositives}{FalsePositives + TruePositives}$$

The detailed comparison in terms of the performance metrics (*precision* and *recall*) and time needed for each model to classify a sample between ML classifiers is shown in Figure 3.7. Based on our analysis shown in Figure 3.7, we determine that the KNN with the K-parameter set to 2 is most suitable with 97.8% average *precision* and 97.6% average *recall*. We compare the KNN performance for different K-parameters ranging from 2 to 20 where, the KNN performs at its best with the K-parameter set to 2. On an average this KNN takes 7 milliseconds to classify a sample. Although the decision tree is able to run faster for classification of a sample in 1.52 milliseconds, we give priority to the *precision* and *recall* factors for accuracy in classifying a sample. Moreover, we expected the KNN + Random

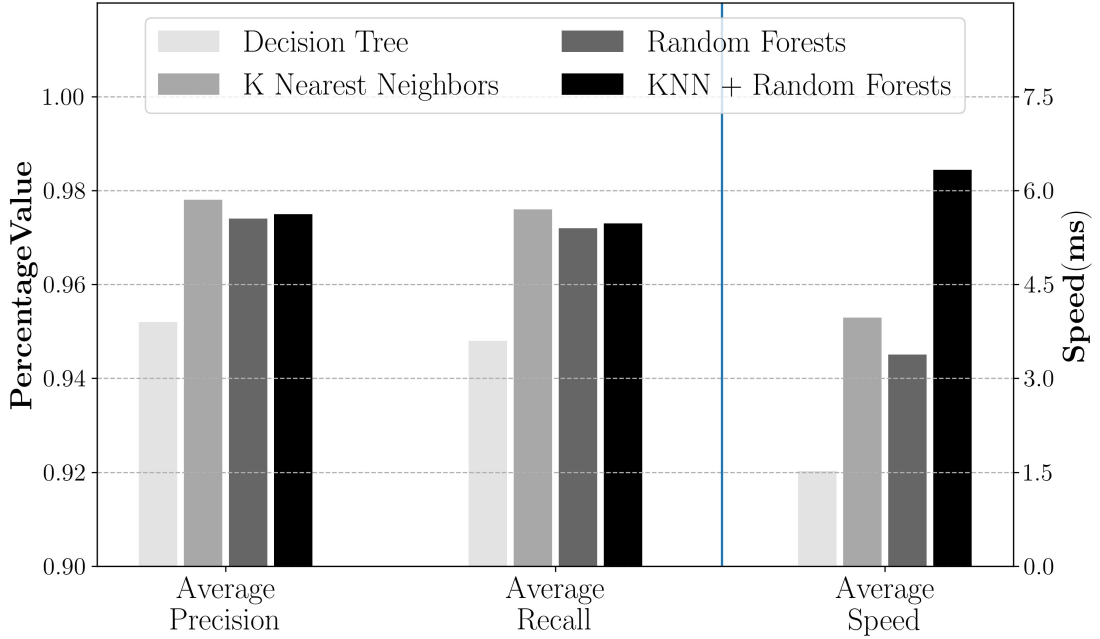


Figure 3.7: Single-label classifier accuracy comparison.

forest (voting classifier) to be the winner among the considered classifiers. But, the KNN was most suitable due to the huge difference in running the classifier in terms of the considered metrics of performance.

### 3.3.4 Single and Multi- Network Attack Detection Results

To determine the most suitable classifier in terms of faster attack detection along with higher accuracy in classification, we use average speed (i.e. time taken to classify a single sample) as another metric. Based on the graphical analysis shown in Figure 3.7, we use the most suited classifier *KNN* and use it for attack detection and classification for single labeled data. We term the time taken to recognize the attack initiation as *Time to detect* an attack. We calculate the attack detection time by monitoring the network packet rate as it transitions from normal (benign) behavior to any of the events related to DoS attacks. To elucidate, we calibrate the average time (in seconds) from the start of a DoS attack to the identification of such anomaly behavior by the ML classifier in the anomaly detection scenarios as shown in Table 3.3.

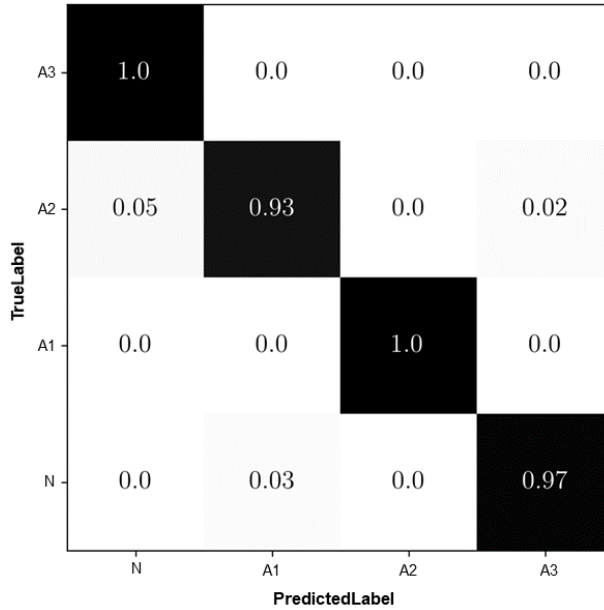


Figure 3.8: Single-label KNN classifier confusion matrix.

We determine the average time taken by our anomaly detection approach applying KNN classifier for detection of network-based attacks (i.e., DoS) as 3.2 seconds. Based on our preliminary work [85], we highlight that the time taken to detect an attack is lower than the time taken for a DoS attack to cause a VRLE server crash. With this detection result, it is evident that our anomaly detection method works effectively and can avoid any VRLE server crash before the user gets interrupted or before an attack event causes a significant impact on VRLE sessions. The Listed anomaly types ( $A_i$ ) in the table 3.3 are the set of:  $\{A_1 - \text{Dropping}, A_2 - \text{Duplication}, A_3 - \text{Tampering}\}$  simulated single attack scenarios and  $N$  represents a ‘no breach scenario’ as shown in Figure 3.8.

Figure 3.8 shows a normalized confusion matrix generated for attack classification using KNN classifier. The row values in the matrix represent the true label (actual values) and the column represents the predicted label (predicting the categories of the values). Moreover, the diagonal elements represent the degree of accurate prediction of the values (assigning on what level our model was able to identify and classify the attacks from the given data). The non-diagonal elements represent the falsely classified (confused to categorize with other classes in the dataset) in the given confusion matrix.

Table 3.3: Performance metrics of our anomaly detection method in terms of detection and classification of anomaly events.

Type of ML Model	Stage 1: Detection - time to detect network attacks (in seconds) for each Anomaly Type ( $A_i$ ) and accuracy metric								Stage 2: Classification		
	$A_1$	$A_2$	$A_3$	$A_4$	$A_5$	$A_6$	$A_7$	Avg. detection time (in seconds)	Avg. accuracy for detection	Avg. time to classify into n/w attacks (in milli seconds)	Avg. accuracy to classify into network attacks
	Multi labeled KNN (Multi attack scenario)	4.0	2.0	2.8	2.6	1.2	3.6	3.6	2.82	99.5%	1.3
Single labeled KNN (single attack scenario)	4.4	2.4	2.8	-	-	-	-	3.2	98.8%	7	97.5%

### Multi-label network based attack detection and classification using multi-label KNN

Based on the results shown in Figure 3.7 and in Figure 3.8, we justify the potential of employing a suitable classifier in our proposed anomaly detection method to identify any network-based anomaly events from a sample of time-series based packet data. However, considering the dynamic interactions in a VRLE, we determine with several experiments that a multi-attack scenario (combination of attacks) can occur in a real-time VRLE application. The above discussed single-label KNN classification classifier will not have the capability to handle a multi-attack scenario. Although the single labeled model can determine the occurrence of an attack, it has no capability to identify which specific attacks are acting in a combination.

In order to address such multi-attack scenarios, our anomaly detection method adapts multi-label KNN classification based on the existing works [100]. The single-label KNN classifier in our anomaly detection method uses adaptations (i.e., the `skmultilearn.adapt` module) for multi-label classification with changes in cost/decision functions. For demonstration purposes, we use an exemplar ML model that focuses only on DoS attacks. Our approach can be extended beyond the scope of this paper for non-DoS attacks affecting the VRLE application.

In our development of a multi-label ML classifier, we collect the attack data similar to the single-label classifier data. We collect the packet rate in a time-series format while a combination of two or more DoS attacks is simulated on the VRLE application. By observing the time-series data, we proceed to develop our multi-label KNN classifier by adapting from the works in [98, 100]. Our developed multi-label KNN classifier is able to detect the combination of network



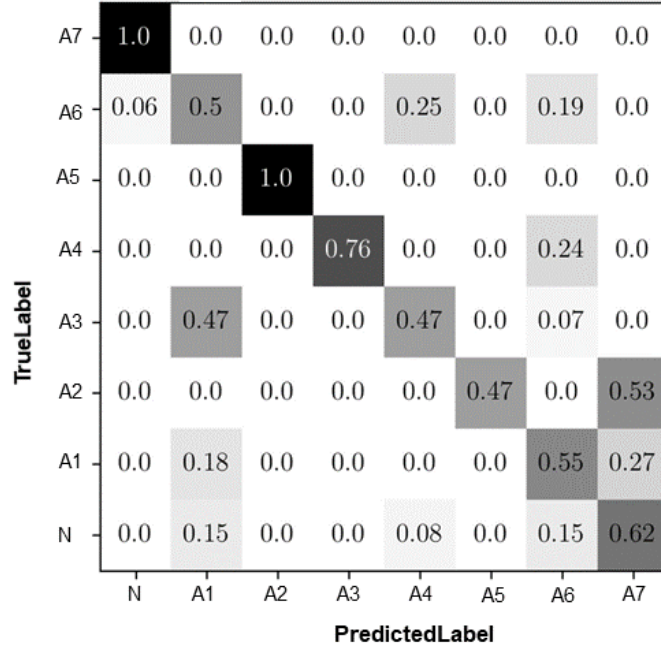


Figure 3.9: Multi-label K-NN classifier confusion matrix.

attack events from the normal behavior (benign) class of data with an average accuracy of 99.5% when our model was run for 1000 tests. We compare these results to check the capability of our model with the existing works[101], where their optimal detection model discusses the potentiality to exceed 99% accuracy in detecting various DoS attacks from normal (benign) behavior data.

Moreover, our multi-label KNN classifier accurately identifies if any attack behavior is observed in the VRLE sessions. In addition, the multi-label KNN classifier is successful in labeling the data samples with the appropriate class/-classes with an average accuracy of 87.5% for 1000 trials. Although this classification accuracy is significantly lower than our single-label KNN classifier, these results can be used as a preliminary model considering the complexity in determining a multi-attack scenario (multi-class labels). We also compare our multi-label KNN classifier with existing works that discuss classifier chains and label power-set model. We observed that the average accuracy of these two models failed to exceed 60% which is lower than our developed multi-labeled KNN classifier.

We tabulate all the different attack scenarios including different attack combinations, where we compute the average time taken by our multi-label KNN classifier to detect an attack is 2.8 seconds as shown in Table 3.3. We generate a

confusion matrix for the multi-labeled classifier as shown in Figure 3.9 with the correct classifications and the incorrectly labeled predictions. The listed anomaly event types shown in the Table 3.3 includes the combination of  $\{A_4\text{- Duplication} + \text{ Dropping}, A_5\text{- Dropping} + \text{ Tampering}, A_6\text{- Duplication} + \text{ Tampering}, A_7\text{- Tampering} + \text{ Duplication} + \text{ Dropping}\}$  multi-attack scenarios simulated and  $N$  represents a no breach scenario as shown in Figure 3.8. Our algorithm does not fail in classifying a single attack (Duplication) that is part of a combination attack (Dropping + Duplication). This shows that our multi-label solution can identify and classify an anomaly event (in combination scenario), and at the very least could specify one attack out of a combination of attacks that are present in the VRLE. Thus, our anomaly detection method detects different DoS attack scenarios (single and combination) efficiently using the most suitable KNN classifier. We enlist all the detection results for the network attacks considered in different scenarios in terms of different performance metrics (time taken to detect and classify, accuracy) as shown in Table 3.3.

### 3.3.5 Anomaly Detection using Statistical Analysis

Our proposed anomaly detection method involves performing statistical Z-score analysis [102] to detect application-based attacks with sparse data and unique features. For instance, in the case of unauthorized access, database logs information on privacy breaches cannot provide enough quantifiable data. Testing such sparse data presents additional new challenges for training ML classifiers in terms of feature extraction. We identify and classify the application-based attacks that are not detected with our ML technique using the statistical analysis as the alternative technique. Our statistical Z-score analysis technique utilizes flag conditions that are further separated into threshold functions and boolean conditions.

We adapt the concept of threshold values for this statistical analysis from the work in [103]. However, due to the single-dimensional nature of the data considered for the statistical analysis, we calculate Z-scores instead of principal com-

ponent analysis as discussed in [103]. To determine the deviation of attack data from normal (benign) VRLE application behavior, Z-score allows us to calculate the thresholds for each attack encountered. The Z-score is computed using the standard deviations from a mean, where standard deviation alone only indicates variance. Based on the existing literature [104, 105], we adapt the formulations of standard deviations (Equation 3.1) and Z-score (Equation 3.2) for sample size  $n$  as follows:

$$SD = \sqrt{\frac{\sum (X - \bar{X})^2}{n-1}} \quad (3.1)$$

$$z = \frac{(X - \mu)}{\sigma} \quad (3.2)$$

where  $X$  is individual value,  $\mu$  is mean,  $SD$  is standard deviation,  $\bar{X}$  is sample mean and  $n$  is the sample size. To determine the threshold values for application-based attacks, our anomaly detection approach calculates the standard deviation for each sample of *baseline data*. Based on the deviation from the mean using the formulations in Equations 3.1 and 3.2, the threshold values are calculated.

Once the Z-scores for *baseline data* are calculated, our anomaly detection method proceeds to calculate the Z-scores for the incoming data and determines an anomaly event based on the deviation from the Z-scores of *baseline data*. The standard deviation, mean used for the calculation of Z-score of the baseline data are represented as  $\sigma$  and  $\mu$ , respectively. Our anomaly detection method identifies anomaly event patterns in VRLEs based on the outlined categories in Equation 3.3. The pseudocode in Algorithm 1 presents the implementation of the statistical analysis technique employed in our proposed anomaly detection approach. As part of calculating the threshold values for the incoming log data, our anomaly detection approach utilizes the Equation 3.3 and determines the Z-scores for baseline data and incoming application data.

$$f(z) = \begin{cases} \textit{Baseline data}, & \text{if } z \in [x, y] \\ \textit{Anomaly}, & \text{otherwise} \end{cases} \quad (3.3)$$

The threshold values represent the Z-scores for baseline data. Moreover, the

threshold values are applied to any data that follows standard distribution as discussed in Section 3.2. Once the respective Z-scores are calculated for both baseline data and input data, we determine whether an anomaly event is occurring or not occurring.

More specifically, to identify the anomaly events, we compare Z-scores of the incoming data to the baseline data in the Threshold function listed in line 3 of Algorithm 1. If the Z-scores of the incoming data are out of the threshold range of the baseline data, then we determine an anomaly event has occurred. In addition to this, the Threshold function in Algorithm 1 also calculates the suspiciousness score  $SS_{attack}$  of the identified anomaly event. On the other hand, we utilize a Boolean function (see line 15 of Algorithm 1) to validate the boolean conditions especially for other form of application-based attacks such as e.g., data tampering. If the boolean conditions are true, then the data is flagged as a malicious event and the IP of that user is retrieved to calculate the suspiciousness score  $SS_{attack}$  of that specific anomaly event. Thus, the application-based attacks are detected based on the calculation of Z-scores using the Equations 3.1, and 3.2 in order to determine the threshold and boolean conditions. Such a determination helps us to flag malicious events as detailed in Table 3.4 and Algorithm 1.

Application-based attacks such as unauthorized access, and disclosure of confidential information are not feasibly detected by an ML classifier. To elucidate, each of these application based attacks generates unique threat signatures or not enough data which would be difficult to train and test the heterogeneous data using an ML classifier. For overcoming this problem, our proposed anomaly detection method employs statistical techniques for identifying these kinds of application-based attacks. An example of an attack generating no/less data is when a VRLE session starts and the attacker tries to gain access to the system and seeks to modify the files after the session start timestamp. In this case, there are no quantifiable parameters to determine file modification (i.e., data tampering). As a solution, the pseudocode in Algorithm 1 listed in Section 3.2 implements the boolean conditions

---

**Algorithm 1:** Statistical analysis pseudocode

---

```
Input:  $D_i$ : Input application data (user, session info) from logs
Output: Return suspiciousness score value
begin
  Function SA ():
    Function Threshold ():
      Function Z-scores ():
        Calculate Z-scores:  $Z_b$ , SD :  $\sigma_b$  for baseline data using Eqn. 3.2;
        for each  $y \in D_i$  do
           $SD_i = \sqrt{\frac{(x-\mu_b)^2}{n-1}}$ 
           $Z_i = \frac{x-\mu_b}{SD_b}$ 
        end
      end Function
      if  $Z_i \notin \text{range}(Z_b)$  then
        | Suspiciousness score();
      end
      else
        | return as benign data;
      end
    end Function
    Function Boolean ():
      if  $Bcon\_attackisTrue$  then
        | Suspiciousness score();
      end
    end Function
    Function Suspiciousness score ():
       $SS = Est.Impact * Attack$ 
      return as SS;
    end Function
  end Function SA
end
```

---

to flag malicious events and thresholding equations to perform Z-score analysis.

We illustrate our experimental evaluation of our statistical technique by calculating the thresholds for the application-based attacks listed in Table 3.4. The Algorithm 1 computes the threshold function of each of the application-based attacks by considering the Z-score of the baseline data as the threshold. Next, the Algorithm 1 compares the Z-scores of the input data to the Z-score of baseline data as mentioned in Equation 3.3. If the input log data is determined as anomaly,

Table 3.4: List of statistical analysis system functions.

<b>Function</b>	<b>Description</b>
<i>Booleans</i>	
SQLInjection()	Scan inputs for illegal characters causing SQL Injection
TFTP()	Check for open ports, new files sent from IPs over TFTP, a sample vulnerability for unauthorized access
checkUsernameIP()	Checks for a username login from an unfamiliar IP
<i>Thresholds</i>	
bruteforce()	Uses Z-scores to calculate when a number of login attempts exceeds the normal threshold
bruteforceUsername()	Uses Z-scores to calculate when no. of login attempts for a specific username exceeds the normal threshold

the *Suspiciousness Score* is calculated for that specific attack. This calculation of *Suspiciousness Score* can aid in alerting the VRLE server administrator about the attack. This Algorithm 1, can be extended to the other application-based attack scenarios that are not included in this study along with several other existing vulnerabilities that can cause attacks considered in this study.

### 3.3.6 Attack Data Collection

In order to test the effectiveness of our Algorithm 1, our anomaly detection method collects the data by simulating an unauthorized access on the VRLE application. In order to perform a brute force attack (a form of unauthorized access), we parsed a database of 1,000,000,000 common passwords [106], and categorized a set of passwords as good passwords (possible errors a user can make while typing the password). This good set of password data consists of the passwords that, are less than two characters different and within 2 characters of the correct password. We calculate the ratio of good passwords to the total set of passwords as  $\frac{10}{4958}$  approximately.

As part of data collection, the tests are run within a reasonable amount of time to determine the normal data set (benign user) and malicious data set (attacker patterns). For this, we consider the fraction  $\frac{10}{4958}$ , where the numerator value denotes a good password dataset of 10 passwords. On the other hand, the denominator value can be denoted as an attacker database of 4958 passwords. We use 123456 as the sample correct password of a user for testing different user scenarios. In order to collect data of a non-malicious user logging into the system, we run this test by removing the good passwords until the sample correct password is identified. Our data collection approach runs the test 1000 times to record the data that determines the number of attempts taken to identify the correct password. To collect the data related to unauthorized access (malicious user trying to login), we run the same test 1000 times using the attacker database of passwords to simulate a Brute-force attack scenario onto a VRLE system. The

Table 3.5: Bruteforce attack detection scenarios in terms of precision and recall metrics

<b>Statistic</b>	<b>Good password dataset</b>	<b>Attacker database</b>
<i>Precision</i>	99.95%	99.7%
<i>Recall</i>	99.92%	99.93%

data is recorded to identify the number of attempts taken by a malicious user to guess the correct password.

### 3.3.7 Accuracy Analysis of statistical analysis technique

The data collected in terms of normal (benign) user attempts and malicious (attacker) attempts are analyzed using Algorithm 1 for calculating the deviations from the mean. With this, the Z-scores are rounded to the nearest whole number due to the computational requirements. We were able to run only 1000 tests due to system limitations and computation time for the statistical analysis. Using the Boolean and threshold functions listed in Table 3.4, we generate histogram to show the distribution of Z-score frequency for normal (benign) user data on login attempts and for malicious user (unauthorized access) login attempt data as shown in Figure 3.10.

The Z-score data related to benign user shown in Figure 3.10, follows a Gaussian distribution along with thresholds created with a sensitivity value of 2. Any data that falls outside the range of these thresholds can be categorized as anomaly events (malicious attacks). We run the Algorithm 1 associated with the Z-score analysis for 100 times on both the benign user data and malicious user data to calculate *precision* and *recall* scores for each test. The average *precision* and *recall* scores are calculated for 100 tests on both the good password and attacker database as shown in Table 3.5 where we identify that these both statistics are with higher levels of 99% in terms of *precision* and *recall* factors.

Our statistical analysis technique considers other forms of unauthorized access for determining the efficiency of our Z-score based analysis. For simplicity, we run tests related to an attacker database of 100 passwords instead of the 4958

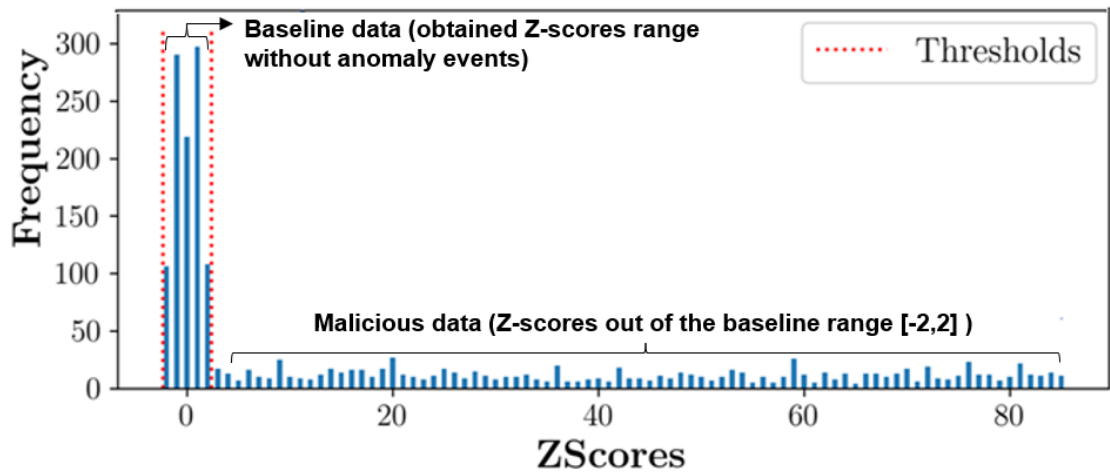


Figure 3.10: Brute-force password attempt Z-score distribution. passwords. This considered 100 set of passwords, represents the scenario where an attacker tries to gain access using other forms of sniffing methods to narrow down to the correct password to a small number. For uniformity, we perform the same number of tests similar to the tests performed for determining the number of login attempts to Brute-force a VRLE system. We tabulate the statistical parameters *precision* and *recall* for other forms of attack data as shown in Table 3.5 to understand the efficiency of our Z-score based analysis.

### 3.3.8 Application Attack Detection Results

Although the password data considered for the other forms of unauthorized access is significantly reduced from the password data considered for the brute force attack, our Algorithm 1 listed in Section 3.2 demonstrates very promising results with above levels of 99% in terms of *precision* and *recall* factors. The y-axis in the bar graphs shown in Figure 3.10 represents the frequency of the data and the x-axis represents the Z-score value ranges. The thresholds for the normal data (benign user) are in the form of the dotted line on the extreme left as shown in Figure 3.10. The malicious attack data is detected and flagged, as the Z-score distribution falls out of the threshold range. This case is annotated as malicious data in the bar graph shown in the Figure 3.10. Thus, using statistical analysis, we identify the application-based attacks (i.e., unauthorized access) with the data



threshold shown in Figure 3.10.

### 3.4 Quantitative Metrics for Attack Occurrence based on Attacker Profiling

Based on the attack classification results, our proposed anomaly detection method involves calculation of the suspiciousness score ( $SS_{attack}$ ) for each of the attacks detected. The  $SS_{attack}$  for a specific attack pattern is calculated based on the estimated level of impact on the UIX and functionality of the VRLE as mentioned in the Equation 3.4 and Algorithm 1. The suspiciousness score calculated for each detected attack in VRLE is as follows:

$$SS_{attack} = \sum_{i=1}^n (AttackScenario_i * EstLevelOfImpact_i) \quad (3.4)$$

The  $SS_{attack}$  calculation takes into account the detected attacks from both the ML technique and the Z-score based statistical analysis. This  $SS_{attack}$  serves as a quantitative analysis metric of the risk associated with an attack or a specific user in a VRLE. The pseudocode in Algorithm 1, consists of a callable, unique dictionary entry and function for each detected attack is based on the ML classifier and Z-score statistical analysis that iteratively adds to the overall suspiciousness score.

#### Calculation of Suspiciousness Scores ( $SS_{attack}$ )

Based on the approach shown in Figure 3.3 that is described in Section 3.2, our anomaly detection module calculates the suspiciousness score  $SS_{attack}$  for a specific detected attack. We utilize a sandbox technique for the calculation of suspiciousness score  $SS_{attack}$  based on the parameters: ‘Estimated level of impact of an attack on the UIX’. In addition, for the attacks that affect privacy, it is difficult to get quantifiable indicators such as in the case of log files being stolen or browsers being hooked. We include all these attacks to build an attacker profile [81]. We remark that these can be extended for future works to determine quantifiable

Table 3.6: Calculation of suspiciousness scores based on multiple combination of attacks.

Type of attack	Detected attack scenarios	Impact value from AP	Total Suspiciousness Scores ( $SS_{attack}$ )
No breach (baseline data)	Benign Data (non-malicious)	1	0
Single network attack scenario (DoS attack)	Duplication	3	0.17
	Dropping	3	0.17
	Tampering	5	0.40
Multi network attack scenario (combination of DoS attacks)	Duplication + Dropping	6	0.50
	Tampering + Duplication	8	0.70
	Tampering + Dropping	8	0.70
	Tampering + Dropping + Duplication	11	1
Single non-network based attack	Brute Force attack	5	0.40
	Unauthorized access (other forms)	5	0.40

parameters for the application-based attacks.

Our anomaly detection module calculates these  $SS_{attack}$  based on Equation 3.4 which is mentioned in the Algorithm 3.2. Our anomaly detection method normalizes the estimated level of impact such that the  $SS_{attack}$  values are normalized on a scale from 0.0 to 1.0; where 1.0 represents the maximum score of  $SS_{attack}$ . There are a few attacks that have similar indicators in case of a single attack or a multiple attack scenario. To elucidate, a recently modified system file can cause an overlay attack (where a default image file has been overridden) or a Chaperone file modification attack [17]. On the other hand, unauthorized access encompasses any kind of brute-force attack or administrator login from a non-administrator user role. The  $SS_{attack}$  for each detected attack is assigned based on Equation 3.4, where we calculate the estimated level of impact of each attack on the UIX as follows:

$$est.level_I = \frac{(x - min_{impact})}{max_{impact} - min_{impact}} \quad (3.5)$$

The  $SS_{attack}$  gets updated by the above formulated estimated level of impact as shown in Table 3.6 using the Equations 3.4 and 3.5. Our proposed anomaly detection will generate the normalized  $SS_{attack}$  scores so that they can be used for

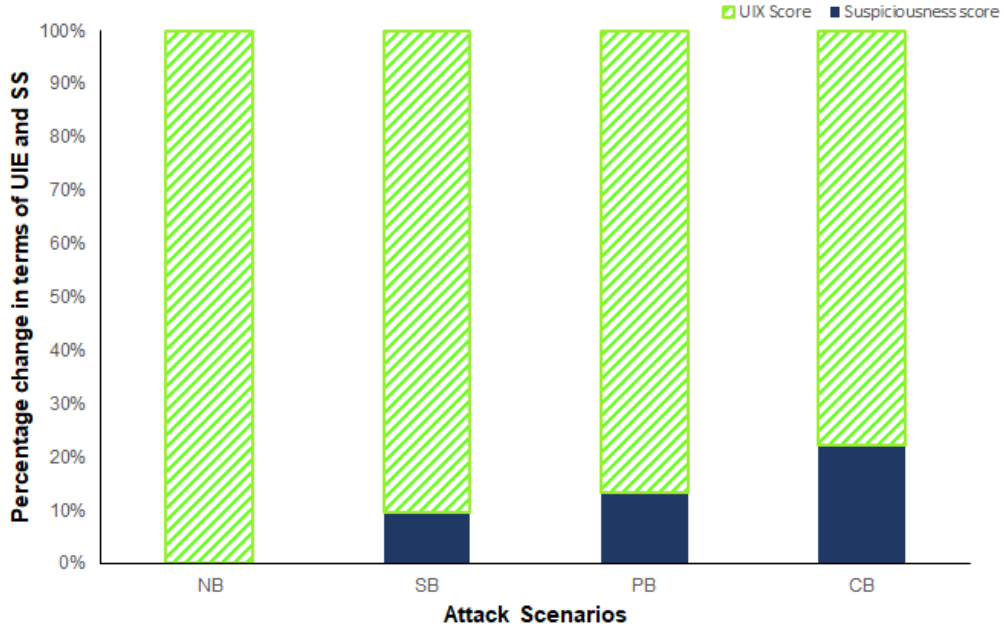


Figure 3.11: Relative change in UIX as the Suspiciousness scores increase for different type of SP breaches.

triggering an alert. These  $SS_{attack}$  scores can also be used for defense mechanisms when an attack is detected.

### 3.5 Impact analysis on User Immersion for Detected Attacks

With the calculated  $SS_{attack}$  for each of the detected attack, we also evaluate the impact on overall UIX score (aggregated numerical value of UIX factors considered) for different types of scenarios (attack and benign user), as shown in the Figure 3.11. The x-axis in Figure 3.11 denotes the breaches occurred where, NB – *No breach*, SB – *Security breach*, PB – *Privacy breach* and CB – *Combination of breaches* in the vSocial case study. The y-axis in Figure 3.11, represents the percentage change value that occurs in both UIX and  $SS_{attack}$  parameters. We observe that for every security, privacy breach scenario in the VRLE, the UIX score is significantly reduced compared to the benign user behavior (NB scenario). On the other hand, the  $SS_{attack}$  is increased for every type of breach included in this analysis. We also highlight that as  $SS_{attack}$  increases, the overall UIX score

is reduced significantly for the attack scenarios considered in this validation of impact analysis. With this analysis, we highlight the fact that  $SS_{attack}$  (that tells about the attack occurrence) and UIX score (usability of the application and sense of immersiveness) can be used as salient quantitative parameters in our proposed anomaly detection method for notifying attack events before they occur in real-world VRLE systems. We store all this historic data on detected attack patterns,  $SS_{attack}$  score and impact on UIX score in the knowledge base, to train models for zero-day attack anomaly events detection by continuously learning the dynamic interactions in a VRLE.

### 3.6 Knowledge Base

Our anomaly detection method stores the baseline data into a database, created to serve as a Knowledge Base for training the models employed in our detection approach. This knowledge base actively stores VRLE session information, detected attack patterns (qualitative descriptions of each attack) along with the associated  $SS_{attack}$ , and user data. This historic data can be used to further train models and can help in detection of zero-day attack anomaly events with continuous learning of dynamic interactions in a VRLE.

### 3.7 Summary

With our experimental survey on several test subjects, we demonstrated the disruptive effects of various application-based and network-based SP attacks on UIX factors. Building upon these established observations, we proposed a novel anomaly detection method that: (a) effectively identifies an attack event or a combination of attack events and, (b) correspondingly classifies the relevant anomaly events into a specific attack category recorded in a knowledge base. We utilize attack trees to model the SP attacks that are caused due to the vulnerabilities in

VRLEs that can potentially disrupt UIX. Our proposed anomaly detection method involved two major techniques: (i) ML-based KNN classifiers for detection of network-based attacks, and (ii) Z-score based analysis for detection of application-based attacks. Using a real-world VRLE application case study viz., vSocial, we successfully detected single network attack scenarios (e.g., security attack) within 3.2 seconds and classified the attack with an accuracy of 97.5%. Moreover, we adapted a multi-label KNN classifier model to detect multi-attack scenarios (i.e., combinations of security and privacy attacks) within 2.82 seconds, and classified the attack categories with an accuracy of 87.5%. Additionally, our anomaly detection method identified unauthorized access via a Z-score based statistical analysis with an average *precision* of 99.7%. We also generated an attack-specific ‘Suspiciousness Score’ metric normalized on a scale of 0-1 for both network-based and application-based attacks in order to serve as a quantifiable parameter for future attack events. We utilized different attacker profiles generated from our experiments in our extensive attack simulations on a vSocial testbed to demonstrate our proposed anomaly detection method’s efficacy in terms of accuracy, precision and recall. In the same context, we created a knowledge base that stores detected attack patterns along with a history of calculated Suspiciousness Scores and UIX scores. The created knowledge base can thus be used to extend our proposed anomaly detection method to identify zero-day attack events on VRLE systems.

## Chapter 4

# Adaptive Control of Robustness and Immersive User Experience for Resilience

### 4.1 3QS Framework Dynamic Rule Based Approach for Social VRLE

With the dynamic user-system interactions for content rendering, VRLEs are a target for an attacker to trigger security, privacy attacks [16, 26]. In addition, the work in [107] details about the performance issues that can disrupt the social VRLE user experience. However, prior works lack in the knowledge to address both performance and security issues that can impact the user experience and even user safety in VRLE sessions. Failure to address such impediments can lead to deface attacks on the VR content with offensive images [38] that can hamper user experience. They can also lead to application latency issues that degrade performance. Based on prior works in VRLE and other IoT applications [108], [109] we adopt the following definitions of various performance (3Q) factors: Quality of Application (QoA) – a measure of the application performance; Quality of Service (QoS) – a measure of network resources such as network bandwidth and jitter; Quality of Experience (QoE) – a measure of the perceived satisfaction or annoy-

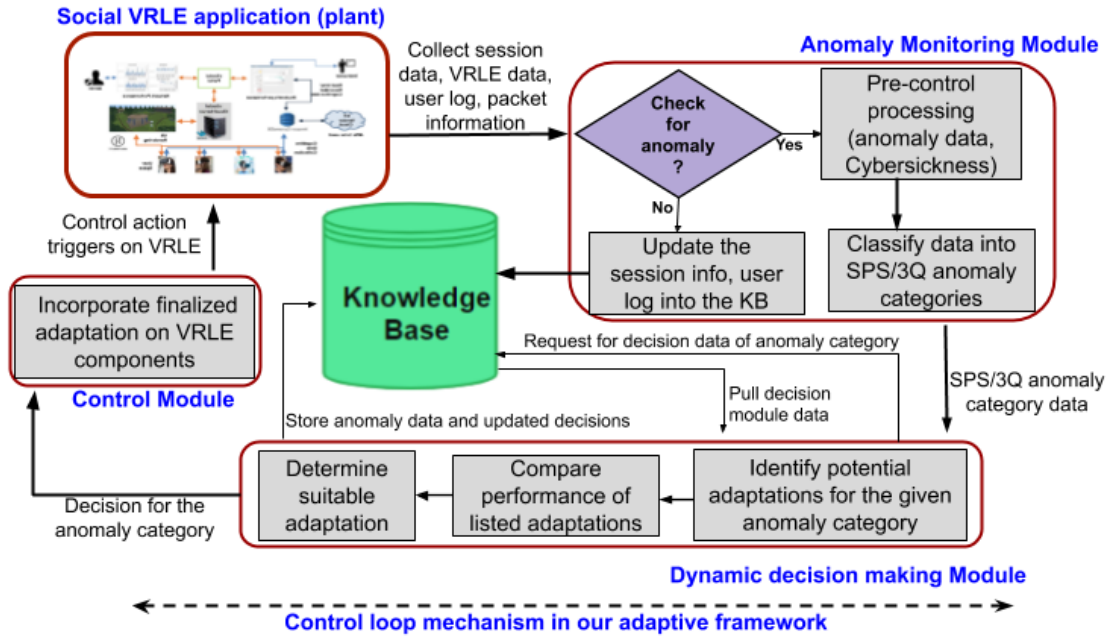


Figure 4.1: Proposed rule-based 3QS-adaption framework for a social VRLE system.

ance of a user’s experience. Together, such performance and security issues can induce “cybersickness” (e.g., eyestrain, nausea, headache, disorientation of user movement) [19, 60]. Hence, there is a need to study methods to mitigate impact of performance and security anomaly events that induce cybersickness.

#### 4.1.1 Closed Loop Mechanism in Social VRLE Case Study

In this section, we present details of our novel rule-based 3QS-adaptation framework to control the impact of cybersickness levels induced due to potential anomaly events caused by 3QS issues. An overview of our 3QS-adaptation framework is shown in the Figure 4.1. Firstly, we detail the monitoring process for detecting anomaly events in the collected network data during a VRLE session. Given such anomaly event data, we next describe how our decision module determines suitable adaptations relevant to the anomaly event. Following this, we explain how the decision outcome is incorporated on an affected VRLE component such that the cybersickness level is reduced. Lastly, we describe how we update session information and the impact of the adaptation into a knowledge base.

<b>Anomalies Detected</b>						
<b>Type</b>	<b>Time</b>	<b>Source</b>	<b>Destination</b>	<b>Protocol</b>	<b>Length</b>	<b>Info</b>
SPS	557.369417	128.206.20.46	128.206.20.43	UDP	92	58222 > 62054 Len=50
SPS	421.962237	128.206.20.43	128.206.20.46	UDP	89	40102 > 58222 Len=47
SPS	553.712925	128.206.20.46	128.206.20.43	UDP	90	58222 > 62054 Len=48
SPS	113.666930	128.206.20.46	128.206.20.43	UDP	78	58222 > 62054 Len=36
SPS	421.248308	128.206.20.43	128.206.20.46	UDP	83	62058 > 58222 Len=41
QoA	82.558665	3.20.240.238	192.168.10.68	UDP	128	48001 > 52766 Len=86
SPS	223.134959	128.206.20.46	128.206.20.43	UDP	90	58222 > 62054 Len=48
SPS	556.242194	128.206.20.46	128.206.20.43	UDP	92	58222 > 62054 Len=50

Figure 4.2: Detected 3QS related anomaly categories in the anomaly event monitoring tool

### 4.1.2 Anomaly Monitoring Tool

To identify any potential 3QS issues in a social VRLE, we developed an anomaly event monitoring tool [110] to observe the network behavior changes, and user activity trends during the VRLE session. We create alarms to trigger when an anomalous behavior pattern is identified in the vSocial application. The anomaly event types include: QoA issues (e.g., visualization delay due to network lag), QoS issues (e.g., packet loss), and security issue (e.g., DoS attack, unauthorized access). Next, we collect this anomaly event data as shown in Figure 4.1 in order to calculate the corresponding impact on the cybersickness level for the session user(s). Following this, we classify the collected anomaly event data into specific 3QS categories.

To identify the traces of 3QS anomaly events in the collected network data during a VRLE session, we developed a web-based anomaly monitoring tool using the Flask micro framework with Python3 [111]. Our anomaly monitoring tool uses the AWS CloudWatch alarms to create triggers based on a threshold condition for every 3QS anomaly. For instance, a QoS alarm is triggered if the threshold condition *if ([No. of packets out] < 7280 packets/second)* fails. Similarly, for a QoA alarm, we use a threshold condition if the (CPU Utilization %) > 8%. Next, the anomaly monitoring tool will pass the collected anomaly data as shown in



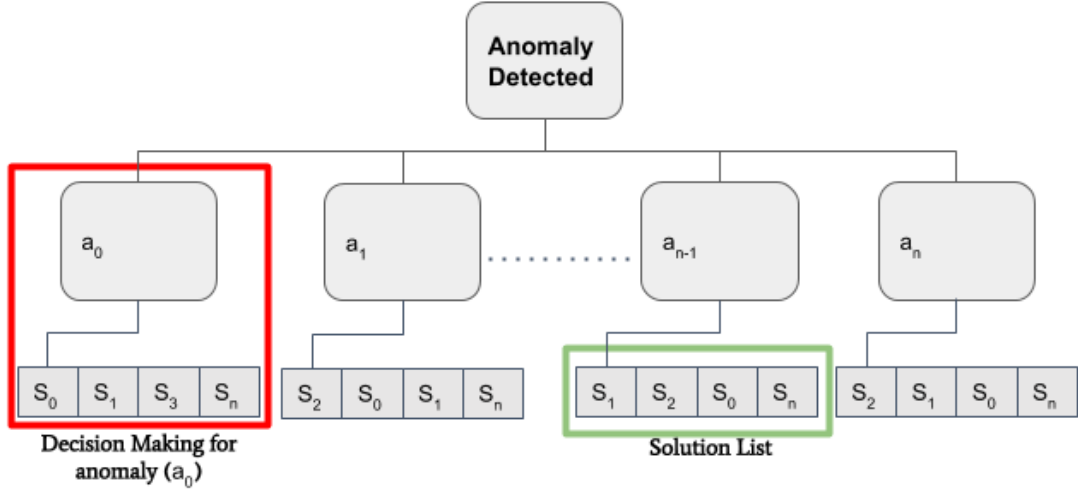


Figure 4.3: Decision making of a suitable adaptation among the list of solution candidates

Figure 4.2 to the decision module of our 3QS-adaptation framework as detailed in Section 4.1.3. We store this detected anomaly data into a AWS S3 bucket [112], which is further interfaced with DynamoDB [112], the knowledge base.

### 4.1.3 Dynamic Decision Making for Handling Critical Anomaly Events

The anomaly event data is classified based on 3QS issue categories. Given anomaly event, our 3QS-adaptation framework activates a decision module that has knowledge of potential adaptations for the specific event as shown in Figure 4.1. The decision module knowledge allows it to compare an anomaly event in a particular category with a set of relevant decision units, where each decision unit is generated and trained on how to deal with a specific type of anomaly event as described in Algorithm 1. The decision units contain a list of potential candidate adaptations that are retrieved from the knowledge base module as shown in Figure 4.3.

Each of the decision units contain a list of defined tuples. These tuples are of the form  $\{A_n, Ct, I\}$ , where  $A_n$  represents the adaptation name,  $Ct$  represents the history of adaptation in terms of number of times that specific choice was implemented, and  $I$  represents the impact on cybersickness level after the adaptation was implemented for a given anomaly event. As and when such decision units are

---

**Algorithm 2: Build Decision Units**

---

**Input:** Adaptation list, Anomaly type list  
**Output:** Relevant Decision units

```
begin
  Function BuildFramework ():
    for each AnomalyType ∈ AnomalyTypeList do
      Function BuildDecisionUnits ():
        Let TupleList = []
        for each Adaptation ∈ TupleList do
          | Let Tuple = (An, Ct, I) TupleList.append(Tuple)
        end
        return DecisionUnit(AnomalyTypei, TupleList)
      end Function
    end
    return AdaptiveFramework(DecisionUnit0, ..., DecisionUnitsk)
  end Function
end
```

---

---

**Algorithm 3: Find Suitable Adaptation**

---

**Input:** Anomaly, Adaptive Framework  
**Output:** Solution

```
begin
  Function FindAdaptation ():
    for each DecisionUnit ∈ AdaptiveFramework do
      if Anomaly = DecisionUniti.AnomalyType then
        Function ProcessAnomaly ():
          Let SortedList = AdaptationList
          for each Adaptation ∈ AdaptationList() do
            if Adaptationk is suitable() then
              | adaptationk.ct + 1
              | return adaptationk.adaptation
            end
          end
        end
      end
      Function AdaptationList ():
        for each Adaptation ∈ AdaptationList do
          | SortedList[] = AdaptationList.sort(C, I, Rt, Ct)
          | AdaptationList.update(SortedList)
        end
        return AdaptationList
      end
    end
  end
end
```

---

created, our decision modules updates the knowledge base to use them for training the decision making relevant to diverse set of future anomaly events.

In the process of determination of a suitable adaptation, our decision module traverses through a list of candidate adaptations enlisted in the matched decision unit. For this, the attributes in the tuples are used as “decision metrics” along with the response time taken by a specific adaptation. To elucidate, the decision module compares a list of candidates using the decision metrics, and determines the *suitability* in the order of *I*, *Ct* and reduced response time in the system. Next, this specific decision unit will be sorted, where the head of the list represent the most suitable adaptation for a given anomaly event. With every iteration of handling anomaly events, the *Ct* value related to the considered adaptation gets updated

Table 4.1: Potential adaptation choices for different 3QS anomalies

Anomaly Issue	Specific Category	Adaptation Name
<b>QoA</b>	High CPU Utilization	Upgrading Instance Type (A1)
		Higher Resources (A2)
		Modifying Instance Volume (A3)
<b>QoS</b>	Low Network Bandwidth	Enabling Enhanced Networking (A4)
		Higher Network Bandwidth (A5)
<b>Security</b>	Denial of Service	Amazon Rout 53 (A6)
		AWS GuardDuty (A7)
<b>Privacy</b>	Unauthorized Access	Blacklist IP via third-party app (A8)

into the knowledge base. Thus, using the decision module, our 3QS-adaptation framework facilitates dynamic decision making for a suitable adaptation to reduce the induced cybersickness level for a given anomaly event.

With the categorized anomaly data, the decision module look up for the relevant decision unit as described in Section 4.1.3. For this, we detail the sample list of potential adaptations for a specific decision unit as shown in the Table 4.1. For example, once a QoA anomaly is identified, the related adaptation candidates are shown in Table 4.1 that include: (i) ‘upgrading the instance type’ (A1), (ii) ‘scaling up the resources’ (A2), and (iii) ‘modifying instance volume’ (A3).

```
[{name: A1, Impact: 0.5, count: 1, Rs: 0.54},
{name: A2, Impact: 0.5, count: 1, Rs: 300},
{name: A4, Impact: 1, count: 4, Rs: 60},
{name: A5, Impact: 0.5, count: 2, Rs: 300}]
```

Listing 4.1: Sample generated decision unit matrix for a given QoS anomaly event.

Listing 4.1 provides details about the decision matrix (unit) retrieved by our decision module where ‘enhanced networking’ (A4) is determined as the suitable solution for a QoS anomaly event. Based on these comparison results, the decision module will sort the candidates list where the suitable adaptation (enhanced networking (A4)) will be at the head of the list. In case, A4 is not effective in reducing the cybersickness for a QoS anomaly event, the decision module will look up the next suitable solution e.g., one that involves allocation of ‘higher network bandwidth’ (A5). With use of such metrics, the decision outcome is generated for

each of the identified anomaly categories. Next, the decision related to an anomaly event is incorporated as detailed in Section 4.1.5 and its relevant information is updated into the knowledge base module to train for future anomaly events.

#### 4.1.4 Creation of a Knowledge Base

Our proposed 3QS-adaptation framework stores the baseline data of benign application behavior into knowledge base for handling future anomaly events in a social VRLE. The knowledge base actively stores VRLE session information, detected anomaly event patterns along with the associated user data. In addition to this, our knowledge base stores the data related to potential adaptations for a specific anomaly category as explained earlier. The anomaly event traces in the knowledge base can be helpful to a network/system administrator to determine the causes of the detected anomalies, and improve the effectiveness of the adaptations. Moreover, the knowledge base can be used as a medium for threat intelligence collection that aids in training of our decision module for mitigation of zero-day anomaly events that arise in an individual and/or combination of 3QS anomaly event scenarios.

In our 3QS-adaptation framework, a knowledge base has been created using a DynamoDB service. To facilitate periodic updates from each of the modules in our framework into DynamoDB, we use the Amazon S3 service along with AWS Lambda functions. We use these both storage systems as our decision module that is hosted on a Jupyter notebook instance that takes only CSV data as input. The full capability of our Knowledge Base can be extended to other applications and can be utilized for employing additional adaptations in VRLE systems.

Using the S3 as the medium, we upload all our session data, adaptation results, decision module results into DynamoDB, thus creating a Knowledge Base as shown in Figure 4.4.

Scan: [Table] solutions: id ^ Viewing 1 to 5 items

Scan [Table] solutions: id ^

+ Add filter

Start search

<input type="checkbox"/>	id ⓘ	cost	resource	solution	time
<input type="checkbox"/>	1	1	128	AWS GuardDuty	513
<input type="checkbox"/>	2	0.199	128	Upgrading Instance Type	553.066
<input type="checkbox"/>	3	0.199	128	Higher Resources	300
<input type="checkbox"/>	4	0.1	128	Enhanced Networking	900.067
<input type="checkbox"/>	5	0.1	128	Higher Network Bandwidth	300

Figure 4.4: DynamoDB table that show data collection in terms of cost, resource, solution and time

### 4.1.5 Adaptation Control

The adaptation control is the control module in our 3QS-adaptation framework enacts suitable adaptations for a given anomaly event category. Once the decision outcome (i.e., suitable adaptation) is obtained, the control module first calculates the risk level associated to a choice of adaptation, along with the cost incurred to control the induced cybersickness anomaly event. Next, the control module invokes an action using an alarm (using e.g., AWS CloudWatch) for the relevant functionality of the determined adaptation. In addition, the risk and cost aware decision outcome implementation is evaluated for the feedback (e.g., control on cybersickness level, user satisfaction). If the anomaly event is successfully handled, then this session information along with the control module data is updated into the knowledge base for handling similar future anomaly events. Thus, the anomalies are monitored continuously, and we perform dynamic decision making to invoke the suitable control actions iteratively for on-demand resource provisioning that delivers satisfactory user experience and controls cybersickness levels in a social VRLE.

Using the decision outcome, the control module implements the adaptation based on the risk and cost aware analysis detailed in Section 4.2.2. In this section, we show a before-and-after implementation scenario for the adaptations listed in

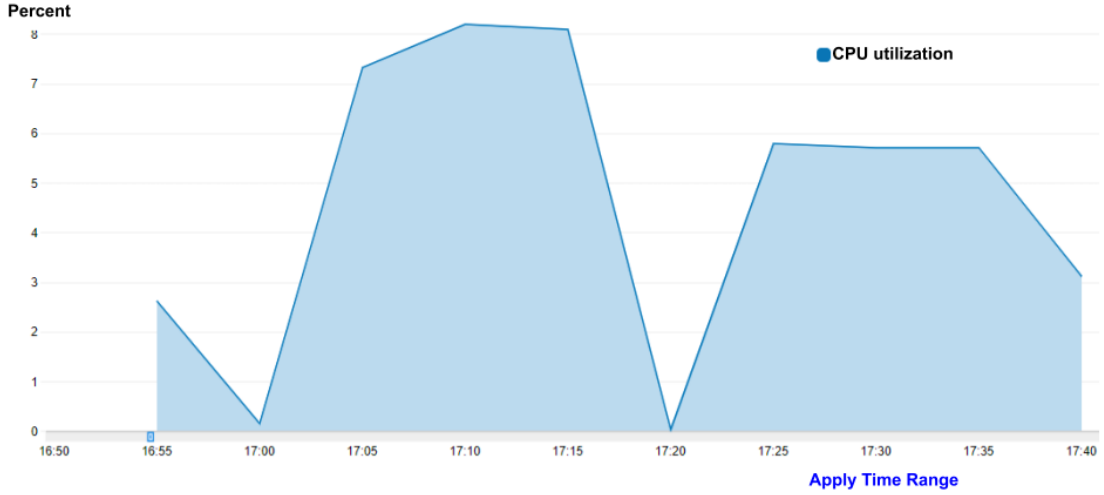


Figure 4.5: Upgrading Instance Type Graph

Table 4.1 in the event of 3QS anomalies within our vSocial application.

For instance, when an AWS alarm relevant to an QoA anomaly is triggered, the control module invokes an action for the suitable adaptation which is to upgrade the instance type (A1) keeping in mind the decision outcome, risk and cost factors. To elucidate, the control module upgrades the current instance (t2.micro) to c4.large based on the risk and cost aware analysis detailed in Section 4.2.2. After incorporating the adaptation A1, we observe there is a 4% decrease in CPU% as shown in Figure 4.5 which is less than the threshold value. We can also see that until the end of the VRLE session, the new instance maintains the application functionality i.e., CPU% values range between 4% and 5%. Using this adaptation, our adaptive framework can ensure the on-demand service of the vSocial application. or can even aid in determining the attacker node connecting to the server.

Similarly, for a QoS anomaly, after implementing the suitable adaptation “enhancing the network conditions” (A4), the packet rate is maintained within a threshold range (i.e., 7280 Packets/Second) beyond which application is non-functional. In case of a security issue, we utilize the adaptation Blacklist IP (A8) to block unauthorized access based on the threshold condition (number of login attempts  $> 5$ ). Based on these implementations for different anomaly categories, we measure the “performance metrics” {response time, Threshold measures}, cost

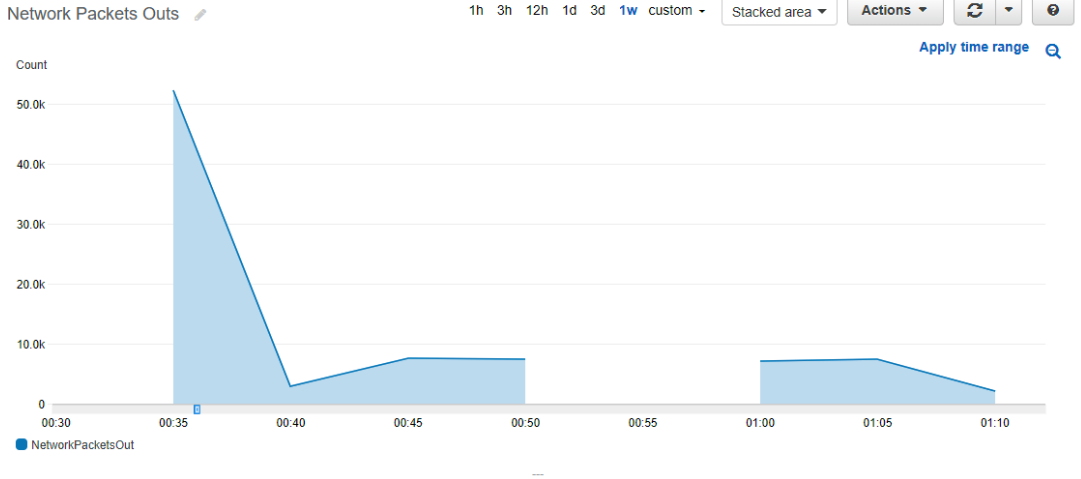


Figure 4.6: Enabling Enhanced Networking

incurred and its impact on controlling the cybersickness level in a VRLE session as shown in Table 4.2.

#### 4.1.6 Testbed Setup

In this section, we first describe the experimental testbed setup based on the vSocial application case study [4]. We use this testbed for the evaluation of our rule-based 3QS-adaptation framework. Next, we discuss the results from experiments to perform a risk and cost aware analysis to finalize suitable adaption choices. Following this, we present the benefits of our priority queuing model in mitigating cybersickness for a given anomaly event. Lastly, based on our experimental results, we enlist suitable practices for handling the identified anomaly events in mitigating cybersickness within social VRLEs.

We setup our experimental testbed in a public cloud i.e., Amazon Web Services (AWS) [112] as shown in Figure 4.7. In this testbed, we host our vSocial application [4] on an Amazon Elastic Compute Cloud (EC2) instance [112] to render the VRLE content to the users. We also host a controller node on another EC2 instance to: (i) capture network data using Amazon CloudWatch [112], and (ii) monitor the network data using our anomaly monitoring tool alongside a decision module hosted on a separate Jupyter notebook instance [113]. In ad-

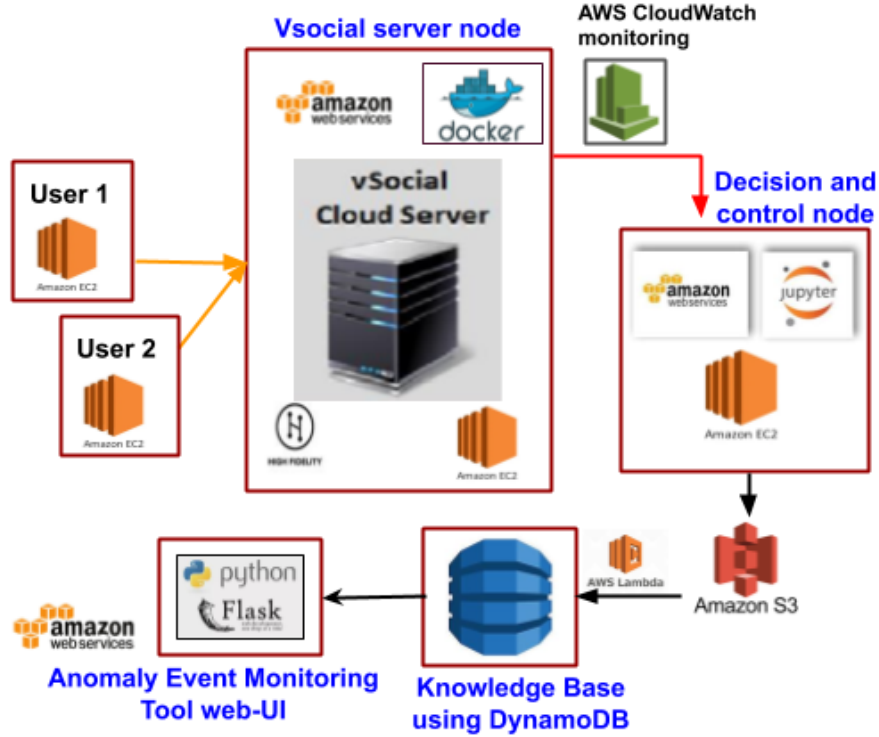


Figure 4.7: Overview of our Experimental Testbed Setup

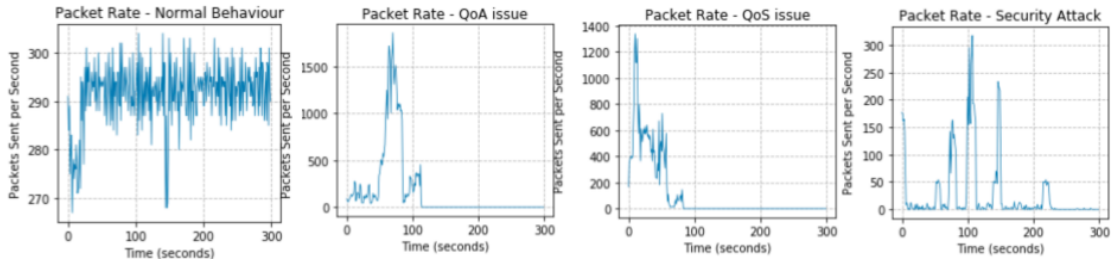


Figure 4.8: Anomaly Data Collection for Simulated Security, QoS, QoA scenarios along with the baseline data

dition, we store the captured and processed network data in the controller node into a DynamoDB [112] service. This DynamoDB service serves as a knowledge base for future anomaly events. We also connect our knowledge base to Amazon S3 [112] service using the Amazon Lambda [112] service in order to provide seamless interaction between the decision module and anomaly monitoring tool. Before illustrating our experimental scenarios, we first detail the tools used for anomaly data collection required for our framework.

As part of anomaly data collection, we simulate a QoS issue (packet drop), QoA issue (packet drop + lag), Security issue (DoS), Privacy (duplication + tampering) in our vSocial application setup. We calculate the packet rate by capturing the



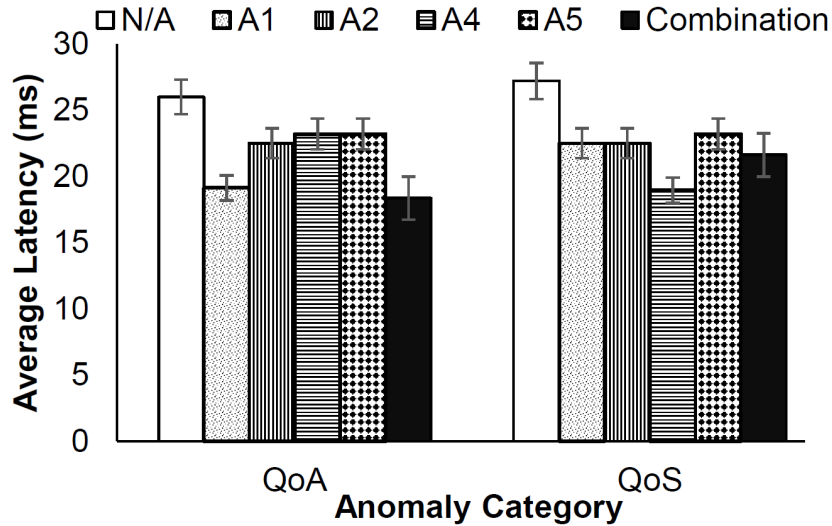


Figure 4.9: Avg. Latency measured (in ms) for QoA anomaly, QoS anomaly scenario in different adaptation scenarios

raw data associated to the timestamp of each packet for each of the simulated 3QS issues along with the baseline data (of benign behavior) of the vSocial application. To simulate a DoS attack on vSocial, we used Clumsy 0.2 [114], a windows based tool to control networking conditions such as lag, drop, throttle, or tamper of live packets. To see the impact on our VRLE application performance, we specifically drop a certain percentage of live packets. Using the Wireshark [115] tool, we capture packets being sent to-and-from our VRLE server in order to demonstrate possible data loss resulting from the packet capture. With the above specified tools and the experimental testbed setup, we collect the anomaly data relevant to 3QS issues in VRLE sessions.

## 4.2 Rule-based Policy Recommendations

### 4.2.1 Quantification of Cybersickness

In this section, we objectively measure the induced cybersickness level for a given set of anomaly events i.e., visualization delay due to network lag (QoA issue), packet loss (QoS issue), and DoS attack (security attack). Existing works [60, 116], attribute cybersickness to physiological conditions (e.g., nausea, eyestrain). How-

ever, the works in [116] study the quantifying effects of latency as the objective parameter to assess cybersickness. Based on these findings, we measure the latency as the primary objective metric of cybersickness for different 3QS anomalies simulated in different network conditions [26]. We also measure the latency in milli seconds (ms) during the normal functionality of a VRLE session, which is 23.5 ms that is used as a baseline data for cybersickness.

The graphical results in Figure 4.9 detail the control of cybersickness (i.e., latency metric) for a few adaptations (i.e., upgrading instance (A1), scaling of higher Resources (A2), enhancing networking (A4), network bandwidth capacity (A5)) listed in Table 4.1. We also consider a no-adaptation (NA) scenario to study the adverse impact on cybersickness control. Moreover, in real-world applications such as vSocial, there is a possibility that one adaptation action might not be enough to mitigate the anomaly impact, and an adaptation should consider the combination of performance and security issues inducing cybersickness [16]. To address such an case, we include a combination of adaptation scenarios to analyze the impact on cybersickness control for a given anomaly event. From the results in Figure 4.9, we observe that for a QoA anomaly, adaptations A1, A2 reduce the cybersickness by 26.43% and 13.46% respectively. In case of a QoS anomaly, the adaptation A4 reduces the cybersickness significantly by 30.28%. In addition, A1 and A2 reduce cybersickness by 17.28% making them the next suitable choice for a QoS anomaly as shown in Figure 4.9. We also note that the combination of best adaptations i.e., A1 and A4 reduces cybersickness by 29.39% for a QoA anomaly and 20.48% for a QoS anomaly event as shown in Figure 4.9. However the choice of combination can vary based on the considered list of potential candidates that can further impact the control of cybersickness levels in a VRLE session.

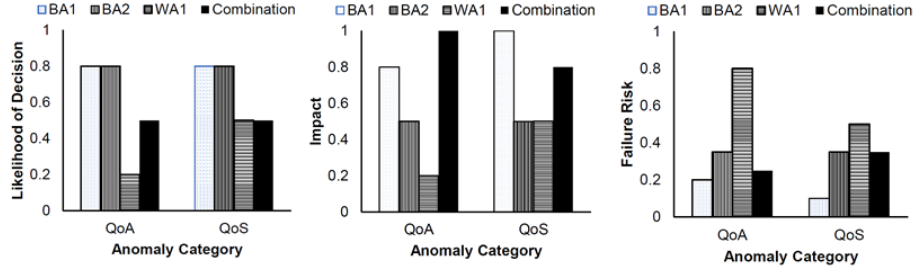


Figure 4.10: Risk evaluation associated with the best (BA1, BA2), worst (WA1) and combination of adaptations in controlling cybersickness for the given QoA and QoS anomaly event

## 4.2.2 Risk and Cost Aware Trade-off Analysis

### Risk calculation for the adaptations

To calculate the risk associated with each decision outcome, we adopt the NIST SP800-30 [28] based risk assessment method [83]. For this, we term the risk as “failure risk” which is a likelihood value of an associated adaptation that can fail in controlling the cybersickness for a given anomaly event. For the assessment, we use two basic variables  $L(D)$  and  $I$  where,  $L(D)$  is the likelihood of decision of a specific adaptation of an anomaly event;  $I$  represents the Impact factor of an adaptation in controlling the cybersickness level. We estimate the  $L(D)$  based on the decision metrics, where the order of priority is cybersickness level, count and then system response time.

We measured  $I$  based on the % of cybersickness level control as discussed in Section 4.2.1. Using these both  $L(D)$  and  $I$ , we calculate the failure risk as  $R_f = 1 - f(L(D), I)$  where,  $f(L(D), I)$  is the average function adopted from existing works [83]. We use a pre-defined semi-quantitative scale of 0-1 as guided by NIST for the impact/likelihood event assessments, with 1 indicating very high, 0.8 indicating a high, 0.5 indicating a moderate, 0.2 indicating a low, and 0 indicating very low levels of impact. Using these parameters, the final failure risk value is calculated for each of the best adaptations (BA1, BA2), worst adaptations (WA1) and combination of adaptation choices as illustrated in Figure 4.10. We considered these adaptations as best/worst based on the latency measurement results shown

Table 4.2: Cost-aware application performance analysis of adaptations chosen for 3QS anomalies

Anomaly Event	Adaptation name	Cost (in \$/hr)	Threshold Metric	$R_{at}$ (in seconds)
QoA	$A1$	0.23	CPU utilization rate is decreased to 4%	0.54
	$A2$	2.4		300
QoS	$A4$	0.10	Packet rate at 7280 packets/second	1
	$A5$	0.10		300
DoS	$A7$	0.33	Packet data measure	0.51
Unauthorized access	$A8$	0.02	Number of login attempts <5	Varies based on number of users

in Figure 4.9 for each anomaly event.

subsubsection Cost aware application performance analysis The results in the previous section discussed the risk associated with some of the best and worst adaptations for a given set of anomaly categories in VRLEs. In order to fully understand the effects of these adaptations on cost and application performance, we calculate the response time taken for implementing the adaptation choice. For instance, we consider  $A1$  and  $A2$  as the best practices for a QoA anomaly due to their control on cybersickness as shown in Figure 4.9. Similarly, for a QoS anomaly, we consider the adaptation choices  $A4$ ,  $A5$  and for a security attack, we consider the  $A7$ ,  $A8$  choices as detailed in Table 4.1.

Each of these adaptations once implemented using the control action detailed in Section 4.1.5, the system response time is measured using the AWS CloudWatch. In addition, we also measure the impact on the application performance using threshold metrics (e.g., CPU utilization rate, network packet rate) as detailed in Section 4.1.2. With this, we highlight the functionality of our framework that takes dynamic decisions to control the cybersickness and maintain satisfactory application functionality. Based on our experimental results (i.e., cost-performance and risk evaluation), we enlist suitable rules (i.e., best practices) to adopt for future anomaly events.

### 4.2.3 Priority-based Queuing Model for Cybersickness Control

In our 3QS-adaptation framework, we use a model to capture the pattern of VRLE application performance, especially with regards to response time in addressing the anomaly events inducing cybersickness. The behavior of anomaly event data processing represents a queue, and thus we model our framework into a  $M/M/1/K$  finite queuing system. This analytical model is based on the embedded Markov Chain, featured by states, events, transitions. The requests that enter into the queue are anomaly events caused by 3QS issues, which are processed on a priority basis i.e., in the order of events with ability to cause higher cybersickness levels.

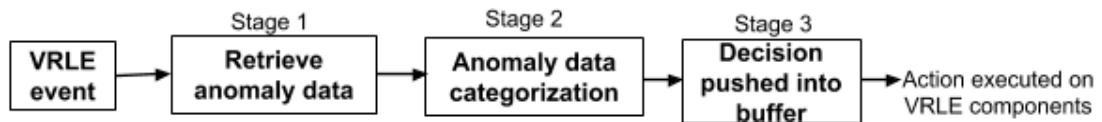


Figure 4.11: Modelling stages of our proposed rule-based 3QS-adaptation framework as a queue

The processing of an incoming request includes three stages: *stage 1* (pre-processing of anomaly event data), *stage 2* (categorization into anomalies caused by 3QS issues), and *stage 3* (anomaly event data pushed into the decision module) as shown in Figure 5.3. After *stage 3*, the processed event record leaves the queue, where the anomaly data is sent to the decision module to determine the suitable action on the corresponding VRLE component. Each stage described in Figure 5.3, has a different average service rate, represented as  $\mu_1$ ,  $\mu_2$ , and  $\mu_3$ . Thus, the overall response time of the system in processing one data record can be computed by solving the similar markov chain transition model as described in one of our works[107]. In this process, the execution of the three stages is mutually exclusive, which means that the second record will not be processed until the previous one is completed. We assume the processing times at each stage is exponentially distributed, and the data records follow a Poisson arrival with an expected rate of  $\lambda$ .

Given the above explanation, the system can be represented as a Markov model

with a state space:  $S = \{(k, n), 0 \leq k \leq K, 0 \leq n \leq 3\}$ , where,  $k$  represents the number of events in the queue,  $K$  represents the maximum size of the queue, and  $n$  is the number of stages in processing the events. The mean response time ( $RT_q$ ) to process an anomaly event in the queue can be obtained by using the Little's formula [107].

The wait time of the queue  $W_q$  is derived based on the number of events in the queue  $L_q$  and arrival rate ( $\lambda$ )

$$W_q = \frac{L_q}{\lambda} \quad (2)$$

$W_t$  is the wait time of the overall system that can be derived using the processing rates  $\mu$ ,  $W_q$

$$W_t = W_q + \frac{1}{\mu} \quad (3)$$

$\bar{X}$  is the sum of the mean service time for all three stages, and can be written as -

$$\bar{X} = \sum_{n=1}^3 1/\mu_n \quad (4)$$

We use the above analytical model in the performance evaluation experiments to determine the waiting delays that might occur in processing the anomaly events inducing cybersickness. To elucidate, a low cybersickness inducing anomaly trigger can be delayed, while a severe threat posing anomaly trigger can be urgently handled by allowing it to experience lower wait times in the triggers handling queue. To achieve such a handling, we use our priority queue model as a Binary-Heap [117] to perform reheapficiation of the events in the queue once an anomaly event is deleted from the queue as detailed earlier.

The priority queuing model used in our framework processes events in the sorted order of cybersickness levels to reduce the the waiting delay in processing higher cybersickness inducing anomaly events. However, this form of a linear priority queue (LPQ) can lag in terms of the overall execution time of the events

Table 4.3: Performance metrics of our queuing model

No. of events in queue	$W_q$ (in sec)	$\bar{X}$ (in sec)	$R_s$ (in sec)	No. of processed severe anomalies
10	2400.48	0.146	3300	4
20	5700	0.204	6303.98	5
30	8700	0.28	9304.15	7
40	11700	0.36	12304.32	11

Table 4.4: Recommendations based on risk level ( $R_l$ ), Cost level ( $C_l$ ), and control on cybersickness ( $\Delta CS$ )

IF		THEN				ELSE			
Anomaly	Scenario in VRLE session	$A_i$	$R_l$	$C_l$	$\Delta CS\%$	$A_i$	$R_l$	$C_l$	$\Delta CS\%$
QoA	Increasing number of users; To improve application run time	A1	L	L	26.43%	A2	M	M	13.46%
QoS	Lower latency in VRLE content	A4	L	L	30.28%	A1+A4	L	M	20.48%
UA	Only valid users in VRLE session	A8	L	L	20.7%	A7	M	H	-
DoS	Avoid loss of content availability	A1+A6	M	M	36.1%	A1+A7	M	H	-

in a queue when compared to the state-of-the-art queuing system (i.e., FIFO) [118]. To overcome such performance overheads in terms of waiting delays and overall response time, our priority queue is implemented using a binary heap [117] which we term as "Binary Heap Priority Queue" (BHPQ). Although, the processing discipline of the events in BHPQ is similar to a LPQ, but BHPQ will address the waiting delays caused in an LPQ by reheapifying the next anomaly once a processed event is deleted from the queue.

Using the formulation detailed in Section 5.3, we calculate the overall system response time ( $R_s$ ) as Sum ( $RT_q, R_{at}$ ), where  $T_q$  is the response time in queue and  $R_{at}$  is the time taken for an adaptation to implement as shown in Table 4.3. In addition, we also enlist the number of processed severe anomaly events (i.e., with high cybersickness level) for a given number of anomaly events in the queue as shown in Table 4.3. Moreover, in case of a traditional FIFO-based queuing model [107], the number of severe events being processed would be much less, thereby delaying the control of the cybersickness level.

#### 4.2.4 Rule-based semantics for Performance and Robustness

Based on our initial experimental evaluation of our framework to control cybersickness level using the listed adaptations in Table 4.1, we recommend rule-based practices as shown in Table 4.4. These practices are expressed in a semantic form i.e., we enlist Event-Condition-Action (ECA) rules with a typical form of *IF – THEN – (ELSE)* [119] to adopt for future VRLE systems. From the results shown in Table 4.4, for a QoA anomaly with the given scenario in VRLE, we recommend adaptation A1 due to the low cost incurred and high impact on cybersickness control when compared to adaptation A2. Moreover, for a QoS anomaly, we recommend adaptation A4 for the significant impact on cybersickness over adaptation A5. In addition, as the next suitable rule-based practice for QoS anomaly, we recommend a combination of A1 and A4 whose risk level is low and the good impact on cybersickness.

Similarly, for an Unauthorized Access (UA), we recommend adaptation A8 over A7 due to the incurred cost and also the lack of control with the GuardDuty service in A7 as shown in Table 4.4. Moreover, the implementation of security recommendations in Table 4.4 are aligned with the NIST security principles [28]. Specifically, mitigation strategies in: (i) A1 (e.g., addition of firewall rules, creation of security groups [120]) corresponds to the *hardening* principle, and (ii) A7 (e.g., addition of multiple components to increase impairment tolerance) corresponds to both hardening and diversity principles. In addition, our recommendations can range from ideas of checking for malware and updating security groups to extreme actions such as terminating the instance altogether.

Using such rule-based adaptations, we showcase the benefit of our proposed framework that controls the cybersickness level induced by the 3QS related anomalies. However, these rules can be updated with an updated trained list of potential adaptations, which we plan to explore as part of our future work. Moreover, there are other forms of rule-based mechanisms such as corrective and enhancing



rules [121], fuzzy rules [122] that can also be adapted to a VRLE application.

#### 4.2.5 Summary

We Proposed a novel rule-based 3QS-adaptation framework that focused on the control of cybersickness induced by performance issues (e.g., visualization delay, packet drop, time lag) and security issues (e.g., unauthorized access, DoS attack). We quantified the cybersickness metric objectively using a latency metric for a simulated anomaly event scenario. We utilized a priority-based queuing model that handles anomaly events in the order of highest cybersickness inducing levels. To determine the suitable adaptation for handling a given anomaly event type, our approach involves performing risk and cost aware analysis for each decision outcome. Once a suitable adaptation is incorporated for a given anomaly event type, cybersickness measurements are updated and used as feedback to determine the impact on the anomaly event.

Our validation results show that the real-time adaptations suggested by our rule-based framework: (i) reduce the cybersickness level by 26.43% for a QoA anomaly and the same for a QoS anomaly event by 30.28%, and (ii) maintains the application functionality within the threshold limit (beyond which an application is non-functional) along with low system response times. Based on these key findings, we enlisted suitable practices for prevention of 3QS issues based on NIST SP800-160 guidelines.

# Chapter 5

## Conclusion and Future Works

Social VRLEs provide immersive experience to the users via remote learning capabilities. The current lack in the state-of-the-art to address potential performance/security issues can disrupt users' experience and potentially induce cybersickness. Failure to address such issues can endanger the user safety by causing physical harm e.g., by obstructing the view of the users, forcing users to run into walls. In this paper, we proposed a novel rule-based 3QS-adaptation framework that focused on the control of cybersickness induced by performance issues (e.g., visualization delay, packet drop, time lag) and security issues (e.g., unauthorized access, DoS attack). We quantified the cybersickness metric objectively using a latency metric for a simulated anomaly event scenario. We utilized a priority-based queuing model that handles anomaly events in the order of highest cybersickness inducing levels. To determine the suitable adaptation for handling a given anomaly event type, our approach involves performing risk and cost aware analysis for each decision outcome. Once a suitable adaptation is incorporated for a given anomaly event type, cybersickness measurements are updated and used as feedback to determine the impact on the anomaly event.

Our validation results show that the real-time adaptations suggested by our rule-based framework: (i) reduce the cybersickness level by 26.43% for a QoA anomaly and the same for a QoS anomaly event by 30.28%, and (ii) maintains the

application functionality within the threshold limit (beyond which an application is non-functional) along with low system response times. Based on these key findings, we enlisted suitable practices for prevention of 3QS issues based on NIST SP800-160 guidelines.

## 5.1 Summary of contributions

Challenges	My Novel Approach	Contributions Description	Thesis
Inducing Cybersickness	Towards Harmonized Robustness & Performance by design	<u>Theoretical contributions</u> <ol style="list-style-type: none"> <li><b>Novel Threat Models</b> using Attack Fault Tree (AFT) formalism for inter-relationship of SPS and Immersion issues inducing Cybersickness</li> <li>Probability of <b>inherent attack vectors and performance issues inducing cybersickness and disruption of UIX</b></li> </ol> <p><b>First to have a SPS and 3Q modeling technique for cybersickness factors in social VRLEs and provide formal guarantee to the observed VRLE system and user behavior</b></p>	2.3, 2.4, 2.5
		<u>Design Contributions</u> <p><b>Quantitative framework</b> -- security principles for meeting design requirements and risk assessment for a risk aware design</p>	2.6

Figure 5.1: Contributions for Research Goal 1: Harmonized Robustness and Performance by Design

Challenges	My Novel Approach	Contributions Description	Thesis
Disruption of User Immersion	Threat Intelligence Collection	<u>Algorithmic contributions</u> <ol style="list-style-type: none"> <li><b>Novel multi-label K-NN classifier model</b> for detection of multi-attack/fault scenarios (i.e. network attacks) before the VRLE server crash occurrence</li> <li><b>Z-score based statistical analysis Technique</b> for detection of sparse attack data (i.e. application attacks), by flagging malicious anomaly event behavior</li> </ol> <p><b>New ML-enabled algorithms for detection and collection of single and combination of attack/fault scenarios that can be used as threat intelligence collection via a mature knowledge base</b></p>	3.3, 3.4
		<u>Design Contributions</u> <ol style="list-style-type: none"> <li><b>Creation of knowledge base</b> for collection of threat intelligence along with quantitative detection metrics</li> <li>Cybersickness and User Immersive Experience subjective <b>assessment methods</b></li> </ol>	3.5

Figure 5.2: Contributions for Research Goal 2: Threat Intelligence Collection for Critical Anomaly Events

Challenges	My Novel Approach	Contributions Description	Thesis
Disruption of User Immersion	Rule-based Approach for Cybersickness control	<u>Theoretical contributions</u> 1. Novel Rule-based approach based on adaptive control for resilience in social VRLEs <b>First to investigate the suitable adaptations as attack/fault mitigation strategies controlling cybersickness following NIST SP 800-160 guidelines</b>	4.1
		<u>Algorithmic contributions</u> 1. Dynamic Decision Making scheme built upon Decision Trees, suitability metrics and for choosing an adaptation to control cybersickness for a given anomaly event 2. Closed loop mechanism for determining the impact of the mitigation strategies for a given anomaly event 3. Characterization of cybersickness quantitative parameters (i.e. latency metric) in social VRLE	4.3
		<u>Design Contributions</u> <b>Developed a novel 3QS framework</b> that relies on decision metrics implemented in a VRLE application setting	4.4

Figure 5.3: Contributions for Research Goal 3: Rule-Based Approach for tuning Performance and Robustness for Resilience

## 5.2 Future Work

Moreover, our rule-based adaptive control can be applied to different domains (such as public safety, workforce training)

1. As part of future work, our joint tuning approach can be adopted for room-based Immersive VR applications and even Mixed Reality (XR) applications.
2. Moreover, our rule-based adaptive control can be applied to different domains (such as public safety, workforce training)
  - Automate our approach using machine learning models for dynamic decision making about anomaly event mitigation strategies
  - Investigate about modeling of our 3QS framework into queue with multiple servers
  - Research about extensive adaptations in real-time which could feature more refined rules for dynamic decision making especially when VRLEs are scaled to handle more number of session users.
3. Can explore issues around identifying zero-day anomalies in social VRLEs and update the threat intelligence collection.

# Bibliography

- [1] (2019) Cloud ar/vr whitepaper: Gsma future networks. Last accessed 2020-10-18. [Online]. Available: <https://www.gsma.com/futurenetworks/wiki/cloud-ar-vr-whitepaper/>
- [2] F. Biocca and M. R. Levy, *Communication in the age of virtual reality*. Routledge, 2013.
- [3] V. Chang, “An overview, examples, and impacts offered by emerging services and analytics in cloud computing virtual reality,” *Neural Computing and Applications*, pp. 1243–1256, 2018.
- [4] C. Zizza, A. Starr, D. Hudson, S. S. Nuguri, P. Calyam, and Z. He, “Towards a social virtual reality learning environment in high fidelity,” in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2018, pp. 1–4.
- [5] (2019) The future of vr communications is on the cloud. Last accessed 2020-10-18. [Online]. Available: <https://realworld-one.com/the-future-of-vr-communications-is-on-the-cloud/>
- [6] Z. Pan, A. D. Cheok, H. Yang, J. Zhu, and J. Shi, “Virtual reality and mixed reality for virtual learning environments,” *Computers & graphics*, vol. 30, no. 1, pp. 20–28, 2006.
- [7] Y. Li and W. Gao, “Muvr: Supporting multi-user mobile virtual reality with

- resource constrained edge cloud,” in *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2018, pp. 1–16.
- [8] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, “Service entity placement for social virtual reality applications in edge computing,” in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 468–476.
- [9] Z. Merchant, E. T. Goetz, L. Cifuentes, W. Keeney-Kennicutt, and T. J. Davis, “Effectiveness of virtual reality-based instruction on students’ learning outcomes in k-12 and higher education: A meta-analysis,” *Computers & Education*, vol. 70, pp. 29–40, 2014.
- [10] J. Dascal, M. Reid, W. W. IsHak, B. Spiegel, J. Recacho, B. Rosen, and I. Danovitch, “Virtual reality and medical inpatients: A systematic review of randomized, controlled trials,” *Innovations in clinical neuroscience*, vol. 14, no. 1-2, p. 14, 2017.
- [11] Zoom, but in vr: Why spatial’s free meeting app feels like a leap forward. Last accessed 2020-10-18. [Online]. Available: <https://www.cnet.com/news/zoom-but-in-vr-why-spatial-free-meeting-app-feels-like-a-leap-forward/>
- [12] K. Boos, D. Chu, and E. Cuervo, “Flashback: Immersive virtual reality on mobile devices via rendering memoization,” in *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, 2016, pp. 291–304.
- [13] R. Zhong, M. Wang, Z. Chen, L. Liu, Y. Liu, J. Zhang, L. Zhang, and T. Moscibroda, “On building a programmable wireless high-quality virtual reality system using commodity hardware,” in *Proceedings of the 8th Asia-Pacific Workshop on Systems*, 2017, pp. 1–7.
- [14] H. Qiu, F. Ahmad, R. Govindan, M. Gruteser, F. Bai, and G. Kar, “Augmented vehicular reality: Enabling extended vision for future vehicles,” in

*Proceedings of the 18th International Workshop on Mobile Computing Systems and Applications*, 2017, pp. 67–72.

- [15] M. D. Lytras, W. Al-Halabi, J. X. Zhang, M. Masud, and R. A. Haraty, “Enabling technologies and business infrastructures for next generation social media: Big data, cloud computing, internet of things and virtual reality.” *J. UCS*, vol. 21, no. 11, pp. 1379–1384, 2015.
- [16] S. Valluripally, A. Gulhane, R. Mitra, K. A. Hoque, and P. Calyam, “Attack trees for security and privacy in social virtual reality learning environments,” in *2020 17th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2020.
- [17] P. Casey, I. Baggili, and A. Yarramreddy, “Immersive virtual reality attacks and the human joystick,” *IEEE Transactions on Dependable and Secure Computing*, 2019.
- [18] J. Mirkovic, P. Reiher, S. Fahmy, R. Thomas, A. Hussain, S. Schwab, and C. Ko, “Measuring denial of service,” in *Proceedings of the 2nd ACM workshop on Quality of protection*. ACM, 2006, pp. 53–58.
- [19] L. Rebenitsch and C. Owen, “Review on cybersickness in applications and visual displays,” *Virtual Reality*, vol. 20, no. 2, pp. 101–125, 2016.
- [20] E. Paja, F. Dalpiaz, and P. Giorgini, “Modelling and reasoning about security requirements in socio-technical systems,” *Data & Knowledge Engineering*, vol. 98, pp. 123–143, 2015.
- [21] N. J. Glaser and M. Schmidt, “Usage considerations of 3d collaborative virtual learning environments to promote development and transfer of knowledge and skills for individuals with autism,” *Technology, Knowledge and Learning*, pp. 1–8, 2018.

- [22] B. Fineman and N. Lewis, “Securing your reality: Addressing security and privacy in virtual and augmented reality applications,” Available at: <https://er.educause.edu/articles/2018/5/securing-your-reality-addressing-security-and-privacy-in-virtual-and-augmented-reality-applications>
- [23] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors),” *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.
- [24] J. H. Friedman, “Greedy function approximation: a gradient boosting machine,” *Annals of statistics*, pp. 1189–1232, 2001.
- [25] C. W. Granger, “Investigating causal relations by econometric models and cross-spectral methods,” *Econometrica: Journal of the Econometric Society*, pp. 424–438, 1969.
- [26] A. Gulhane, A. Vyas, R. Mitra, R. Oruche, G. Hoefler, S. Valluripally, P. Calyam, and K. A. Hoque, “Security, privacy and safety risk assessment for virtual reality learning environment applications,” in *16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–9.
- [27] High fidelity. [Online]. Available: <https://highfidelity.com>
- [28] R. Ross, M. McEvilley, and J. Oren, “Systems security engineering: Considerations for a multidisciplinary approach in the engineering of trustworthy secure systems,” Tech. Rep., 2018. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-160.pdf>
- [29] W. Zhou, Y. Jia, A. Peng, Y. Zhang, and P. Liu, “The effect of iot new features on security and privacy: New threats, existing solutions, and challenges yet to be solved,” *IEEE Internet of Things Journal*, vol. 6, pp. 1606–1616, 2018.



- [30] K. Fu, T. Kohno, D. Lopresti, E. D. Mynatt, K. Nahrstedt, S. N. Patel, D. J. Richardson, and B. Zorn, “Safety , security , and privacy threats posed by accelerating trends in the internet of things,” 2017.
- [31] W. Yu, X. Wang, P. Callyam, D. Xuan, and W. Zhao, “On detecting camouflaging worm,” in *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*. IEEE, 2006, pp. 235–244.
- [32] F. Roesner, T. Kohno, and D. Molnar, “Security and privacy for augmented reality systems,” *Commun. ACM*, vol. 57, pp. 88–96, 2014.
- [33] R. Roman, J. Zhou, and J. López, “On the features and challenges of security and privacy in distributed internet of things,” *Computer Networks*, vol. 57, pp. 2266–2279, 2013.
- [34] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, “A survey on security and privacy issues in internet-of-things,” *IEEE Internet of Things Journal*, vol. 4, pp. 1250–1258, 2017.
- [35] R. McPherson, S. Jana, and V. Shmatikov, “No escape from reality: Security and privacy of augmented reality browsers,” in *WWW*, 2015.
- [36] S. Yi, Z. Qin, and Q. Li, “Security and privacy issues of fog computing: A survey,” in *WASA*, 2015.
- [37] M. H. Mughees, H. Haddadi, and P. Hui, “Privacy leakage in mobile computing: Tools, methods, and characteristics,” *ArXiv*, vol. abs/1410.4978, 2014.
- [38] P. Casey, I. Baggili, and A. Yarramreddy, “Immersive virtual reality attacks and the human joystick,” 2019.
- [39] K. M. Stanney, R. R. Mourant, and R. S. Kennedy, “Human factors issues in virtual environments: A review of the literature,” *Presence*, vol. 7, pp. 327–351, 1998.

- [40] M. S. Dennison, A. Z. Wisti, and M. D’Zmura, “Use of physiological signals to predict cybersickness,” *Displays*, vol. 44, pp. 42–52, 2016.
- [41] N. Glaser and M. J. Schmidt, “Usage considerations of 3d collaborative virtual learning environments to promote development and transfer of knowledge and skills for individuals with autism,” *Technology, Knowledge and Learning*, pp. 1–8, 2018.
- [42] A. Marback, H. Do, K. He, S. Kondamarri, and D. Xu, “A threat model-based approach to security testing,” *Software: Practice and Experience*, vol. 43, no. 2, pp. 241–258, 2013.
- [43] A. Gholami and E. Laure, “Advanced cloud privacy threat modeling,” *arXiv preprint arXiv:1601.01500*, 2016.
- [44] A. Almulhem, “Threat modeling for electronic health record systems,” *Journal of medical systems*, vol. 36, no. 5, pp. 2921–2926, 2012.
- [45] R. Hasan, S. Myagmar, A. J. Lee, and W. Yurcik, “Toward a threat model for storage systems,” in *Proceedings of the 2005 ACM workshop on Storage security and survivability*, 2005, pp. 94–102.
- [46] A. Gulhane, A. Vyas, R. Mitra, R. Oruche, G. Hoefler, S. Valluripally, P. Calyam, and K. A. Hoque, “Security, privacy and safety risk assessment for virtual reality learning environment applications,” *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pp. 1–9, 2018.
- [47] E. M. Clarke Jr, O. Grumberg, D. Kroening, D. Peled, and H. Veith, *Model checking*. MIT press, 2018.
- [48] H. L. Younes, M. Kwiatkowska, G. Norman, and D. Parker, “Numerical vs. statistical probabilistic model checking,” *International Journal on Software Tools for Technology Transfer*, vol. 8, no. 3, pp. 216–228, 2006.

- [49] P. Ballarini, N. Bertrand, A. Horváth, M. Paolieri, and E. Vicario, “Transient analysis of networks of stochastic timed automata using stochastic state classes,” in *QEST*, 2013.
- [50] R. Kumar and M. Stoelinga, “Quantitative security and safety analysis with attack-fault trees,” *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pp. 25–32, 2017.
- [51] A. David, K. G. Larsen, A. Legay, M. Mikucionis, and D. B. Poulsen, “Up-paal smc tutorial,” *International Journal on Software Tools for Technology Transfer*, vol. 17, pp. 397–415, 2014.
- [52] P. Bulychev, A. David, K. G. Larsen, A. Legay, G. Li, and D. B. Poulsen, “Rewrite-based statistical model checking of wmtl,” in *International Conference on Runtime Verification*. Springer, 2012, pp. 260–275.
- [53] A. Laszka, W. Abbas, Y. Vorobeychik, and X. D. Koutsoukos, “Synergistic security for the industrial internet of things: Integrating redundancy, diversity, and hardening,” *2018 IEEE International Conference on Industrial Internet (ICII)*, pp. 153–158, 2018.
- [54] G. Norman, D. Parker, and J. Sproston, “Model checking for probabilistic timed automata,” *Formal Methods in System Design*, vol. 43, pp. 164–190, 2013.
- [55] university news. Securitree for attack tree analysis. Last accessed 2020-10-18. [Online]. Available: <https://www.amenaza.com>
- [56] B. Schneier. (1999) Attack trees. [Online]. Available: <http://www.drdoobbs.com/attack-trees/184411129>
- [57] P. Saripalli and B. Walters, “Quirc: A quantitative impact and risk assessment framework for cloud security,” *2010 IEEE 3rd International Conference on Cloud Computing*, pp. 280–288, 2010.

- [58] M. Kiani, A. J. Clark, and G. M. Mohay, “Evaluation of anomaly based character distribution models in the detection of sql injection attacks,” *2008 Third International Conference on Availability, Reliability and Security*, pp. 47–55, 2008.
- [59] N. Cauchi, K. A. Hoque, A. Abate, and M. Stoelinga, “Efficient probabilistic model checking of smart building maintenance using fault maintenance trees,” in *BuildSys@SenSys*, 2017.
- [60] J. J. LaViola, “A discussion of cybersickness in virtual environments,” *SIGCHI Bull.*, vol. 32, no. 1, p. 47–56, Jan. 2000. [Online]. Available: <https://doi.org/10.1145/333329.333344>
- [61] S. Davis, K. Nesbitt, and E. Nalivaiko, “A systematic review of cybersickness,” in *Proceedings of the 2014 Conference on Interactive Entertainment*, 2014, pp. 1–9.
- [62] R. Roman, J. Lopez, and M. Mambo, “Mobile edge computing: a survey and analysis of security threats and challenges,” *Elsevier Future Gen. Computer Systems*, 2016.
- [63] M. A. Khan and K. Salah, “Iot security: Review, blockchain solutions, and open challenges,” *Future Generation Comp. Syst.*, vol. 82, pp. 395–411, 2017.
- [64] (2019) Steamvr. Last accessed 2019-09-30. [Online]. Available: <http://store.steampowered.com/steamvr>
- [65] “Securing your reality: Addressing security and privacy in virtual and augmented reality applications.” [Online]. Available: <https://tinyurl.com/yxlpqqe>
- [66] J. A. de Guzman, K. Thilakarathna, and A. Seneviratne, “Security and privacy approaches in mixed reality: A literature survey,” *ArXiv*, vol. abs/1802.05797, 2018.

- [67] Virtual reality headsets could put childrens health at risk. Last accessed 2020-04-28. [Online]. Available: <https://www.theguardian.com/technology/2017/oct/28/virtual-reality-headset-children-cognitive-problems>
- [68] Y. Farmani and R. J. Teather, “Viewpoint snapping to reduce cybersickness in virtual reality,” in *Proceedings of the 44th Graphics Interface Conference*. Canadian Human-Computer Communications Society, 2018, pp. 168–175.
- [69] A. Mazloumi Gavgani, F. R. Walker, D. M. Hodgson, and E. Nalivaiko, “A comparative study of cybersickness during exposure to virtual reality and “classic” motion sickness: are they different?” *Journal of Applied Physiology*, vol. 125, no. 6, pp. 1670–1680, 2018.
- [70] A. Tiiro, “Effect of visual realism on cybersickness in virtual reality,” *University of Oulu*, vol. 350, 2018.
- [71] H. K. Kim, J. Park, Y. Choi, and M. Choe, “Virtual reality sickness questionnaire (vrsq): Motion sickness measurement index in a virtual reality environment,” *Applied ergonomics*, vol. 69, pp. 66–73. [Online]. Available: <https://doi.org/10.1016/j.apergo.2017.12.016>
- [72] K. Tcha-Tokey, E. Loup-Escande, O. Christmann, and S. Richir, “A questionnaire to measure the user experience in immersive virtual environments,” in *Proceedings of the 2016 Virtual Reality International Conference*. ACM, 2016, p. 19.
- [73] G. Regal, R. Schatz, J. Schrammel, and S. Suetterle, “Vrate: A unity3d asset for integrating subjective assessment questionnaires in virtual environments,” in *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 2018, pp. 1–3.
- [74] (2019) clumsy0.2. Last accessed 2019-09-30. [Online]. Available: <https://jagt.github.io/clumsy/download.html>

- [75] (2019) Wireshark. Last accessed 2019-09-30. [Online]. Available: <https://www.wireshark.org/>
- [76] university news. (2019) University of new haven researchers discover critical vulnerabilities in popular virtual reality application. Last accessed 2020-04-12. [Online]. Available: <https://tinyurl.com/u87l2oq>
- [77] H. K. Kim, J. Park, Y. Choi, and M. Choe, “Virtual reality sickness questionnaire (vrsq): Motion sickness measurement index in a virtual reality environment.” *Applied ergonomics*, vol. 69, pp. 66–73, 2018.
- [78] A. Tiiro, “Effect of visual realism on cybersickness in virtual reality,” *University of Oulu*, 2018.
- [79] G. Regal, R. Schatz, J. Schrammel, and S. Suetterle, “Vrate: a unity3d asset for integrating subjective assessment questionnaires in virtual environments,” in *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 2018, pp. 1–3.
- [80] M. Rocchetto and N. O. Tippenhauer, “On attacker models and profiles for cyber-physical systems,” in *ESORICS*, 2016.
- [81] “Attacker classification to aid targeting critical systems for threat modelling and security review.”
- [82] “Profiles of cyber-criminals and cyberattackers,” last accessed 2020-01-22. [Online]. Available: <https://tinyurl.com/rstn7df>
- [83] M. Dickinson, S. Debroy, P. Calyam, S. Valluripally, Y. Zhang, R. B. Antequera, T. Joshi, T. A. White, and D. Xu, “Multi-cloud performance and security driven federated workflow management,” 2018.
- [84] Impact of iot on augmented and virtual reality. Last Accessed 2020-08-13. [Online]. Available: <https://www.allerin.com/blog/impact-of-iot-on-augmented-and-virtual-reality>

- [85] A. Gulhane, A. Vyas, R. Mitra, R. Oruche, G. Hoefler, S. Valluripally, P. Calyam, and K. A. Hoque, "Security, privacy and safety risk assessment for virtual reality learning environment applications," in *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–9.
- [86] B. G. Witmer and M. J. Singer, "Measuring presence in virtual environments: A presence questionnaire," *Presence*, vol. 7, no. 3, pp. 225–240, 1998.
- [87] T. Iachini, Y. Coello, F. Frassinetti, and G. Ruggiero, "Body space in social interactions: a comparison of reaching and comfort distance in immersive virtual reality," *PloS one*, vol. 9, no. 11, 2014.
- [88] H. Myrbakken and R. C. Palacios, "Devsecops: A multivocal literature review," in *SPICE*, 2017.
- [89] M. Rege and R. Mbah, "Machine learning for cyber defense and attack," in *DATA ANALYTICS 2018 : The Seventh International Conference on Data Analytics*, 2018.
- [90] J. Zhao, S. Shetty, and J. W. Pan, "Feature-based transfer learning for network security," in *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*. IEEE, 2017, pp. 17–22.
- [91] M. Berman, J. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "Geni: A federated testbed for innovative network experiments," *Elsevier Computer Network*, vol. 61, no. 14, pp. 5–23, 2014. [Online]. Available: <https://ieeexplore.ieee.org/document/8392768>
- [92] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: a network programming language," in *ICFP*, 2011.
- [93] (2019) Kali linux. Last accessed 2019-09-30. [Online]. Available: <https://www.kali.org/>

- [94] (2019) Metasploit. Last accessed 2019-09-30. [Online]. Available: <https://www.metasploit.com/>
- [95] (2019) Solarwinds tftp server. Last accessed 2019-09-30. [Online]. Available: <https://www.solarwinds.com/free-tools/free-tftp-server/>
- [96] Code for attack simulation. Last accessed 2020-08-13. [Online]. Available: <https://github.com/boonakij/Packet-Rate-Monitoring>
- [97] (2019) Scikit-learn. Last accessed 2019-09-30. [Online]. Available: <https://scikit-learn.org/stable/>
- [98] (2019) Scikit-multilearn. Last accessed 2019-09-30. [Online]. Available: <http://scikit.ml/>
- [99] (2019) Ts-fresh. Last accessed 2019-09-30. [Online]. Available: <https://tsfresh.readthedocs.io/en/latest/>
- [100] M.-L. Zhang and Z. W. Zhou, “A k-nearest neighbor based algorithm for multi-label classification,” *2005 IEEE International Conference on Granular Computing*, vol. 2, pp. 718–721 Vol. 2, 2005.
- [101] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, “Practical real-time intrusion detection using machine learning approaches,” *Computer Communications*, vol. 34, no. 18, pp. 2227–2235, 2011.
- [102] (2019) Z-score: Definition. Last accessed 2019-09-30. [Online]. Available: [https://stattrek.com/statistics/dictionary.aspx?definition=z\\_score](https://stattrek.com/statistics/dictionary.aspx?definition=z_score)
- [103] Y. Zhang, S. Debroy, and P. Callyam, “Network-wide anomaly event detection and diagnosis with perfsonar,” *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 666–680, 2016.
- [104] M. Barde and P. Barde, “What to use to express the variability of data: Standard deviation or standard error of mean?” in *Perspectives in clinical research*. PMC, 2012, p. 113–116.



- [105] P. Driscoll, F. Lecky, and M. Crosby, “An introduction to estimation—1. starting from z,” *Emergency Medicine Journal*, vol. 17, no. 6, pp. 409–415, 2000. [Online]. Available: <https://emj.bmj.com/content/17/6/409>
- [106] (2019) Bruteforce-database. Last accessed 2019-09-30. [Online]. Available: <https://github.com/duyetdev/bruteforce-database/>
- [107] S. Wang, S. Valluripally, R. Mitra, S. S. Nuguri, K. Salah, and P. Calyam, “Cost-performance trade-offs in fog computing for iot data processing of social virtual reality,” in *2019 IEEE International Conference on Fog Computing (ICFC)*. IEEE, 2019, pp. 134–143.
- [108] M. H. Ghahramani, M. Zhou, and C. T. Hon, “Toward cloud computing qos architecture: Analysis of cloud systems and cloud services,” *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 6–18, 2017.
- [109] B. Mukherjee, R. L. Neupane, and P. Calyam, “End-to-end iot security middleware for cloud-fog communication,” in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. IEEE, 2017, pp. 151–156.
- [110] M. Vassell, O. Apperson, P. Calyam, J. Gillis, and S. Ahmad, “Intelligent dashboard for augmented reality based incident command response co-ordination,” in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 976–979.
- [111] Flask: A web-development guide. Last accessed 2020-09-01. [Online]. Available: <https://pypi.org/project/Flask/>
- [112] Amazon web services. Last accessed 2020-09-01. [Online]. Available: <https://aws.amazon.com/>
- [113] Jupyter notebook. Last accessed 2020-09-01. [Online]. Available: <https://jupyter.org/>

- [114] “clumsy 0.2”, 2018. Last accessed 2020-09-01. [Online]. Available: <https://jagt.github.io/clumsy>
- [115] Wireshark tool. Last accessed 2020-09-01. [Online]. Available: <https://www.wireshark.org/>
- [116] J. J. LaViola Jr, “A discussion of cybersickness in virtual environments,” *ACM Sigchi Bulletin*, vol. 32, no. 1, pp. 47–56, 2000.
- [117] R. Bhagwan and B. Lin, “Fast and scalable priority queue architecture for high-speed network switches,” in *Proceedings IEEE INFOCOM*, vol. 2. IEEE, 2000, pp. 538–547.
- [118] S. S. Sameer, “Simulation: Analysis of single server queuing model,” *International Journal on Information Theory (IJIT)*, vol. 3, no. 3, pp. 47–54, 2014.
- [119] R. Müller, U. Greiner, and E. Rahm, “Agentwork: a workflow system supporting rule-based workflow adaptation,” *Data & Knowledge Engineering*, vol. 51, no. 2, pp. 223–256, 2004.
- [120] Amazon guardduty. Last accessed 2020-09-01. [Online]. Available: <https://aws.amazon.com/guardduty/>
- [121] I. Lanese, A. Bucchiarone, and F. Montesi, “A framework for rule-based dynamic adaptation,” in *International Symposium on Trustworthy Global Computing*. Springer, 2010, pp. 284–300.
- [122] M. Zeeshan and S. A. Khan, “A novel algorithm for link adaptation using fuzzy rule based system for wideband networking waveform of sdr,” *AEU-International Journal of Electronics and Communications*, vol. 69, no. 9, pp. 1366–1373, 2015.

# Glossary

<b>Term or Abbreviation</b>	<b>Explanation</b>
<b>Accessibility</b>	Related to characteristic needs of the users, technologies involved and their associated user roles in VR application
<b>Online Learning</b>	Virtual Reality is used for distance learning purposes in the field of education especially during pandemic
<b>Usability</b>	We define the context of usability of the application as a combination of effectiveness, user satisfaction, security and efficiency
<b>User Experience</b>	In the context of VRLE, we define user experience as the combination of cyber sickness and user immersive experience during a VRLE session
<b>User Immersive Experience (UIX)</b>	Experience pertaining to the user i.e., to be submerged in a simulated experience generated by a VR technology
<b>Cybersickness (CS)</b>	A Form of motion induced sickness for a user while using the VR application

## VITA

Samaikya Valluripally received her MS degree in Computer Science from University of Missouri-Columbia in 2016 and Bachelor of Technology degree in Computer Science from Jawaharlal Nehru Technological University, India in 2014. She is currently a 5th year Ph.D. student in the department of Electrical Engineering and Computer Science at University of Missouri-Columbia.

Her PhD research is focused on cybersecurity for cloud based applications such as Virtual Reality, Health-Information Sharing. During her PhD studies, she has published over 10 peer-reviewed publications in reputed conference venues and journals including IEEE TSC, IEEE TCC, IEEE Cloud, IEEE CCNC, Springer and others. She also did a Summer internship at Centene Corporation on areas related to Big Data specifically Enterprise Data Ware House, ETL and Data Security in 2018. Her current research interests include Cloud Computing, Cloud Security for AR/VR and IoT applications, Big Data Analytics.