Detection of Security and Privacy Attacks Disrupting User Immersive Experience in Virtual Reality Learning Environments

Samaikya Valluripally, Benjamin Frailey, Brady Kruse, Boonakij Palipatana, Roland Oruche, Aniket Gulhane, Khaza Anuarul Hoque, *Senior Member, IEEE*, and Prasad Calyam, *Senior Member, IEEE*

Abstract—Virtual Reality Learning Environments (VRLEs) are a new form of immersive environments which are integrated with wearable devices for delivering distance learning content in a collaborative manner in e.g., *special education*, *surgical training*. Gaining unauthorized access to these connected devices can cause security, privacy attacks (SP) that adversely impacts the user immersive experience (UIX). In this article, we identify potential SP attack surfaces that impact the application usability and immersion experience, and propose a novel anomaly detection method to detect attacks before the UIX can be disrupted. Specifically, we apply: (i) machine learning techniques such as a *multi-label KNN classification* algorithm to detect anomaly events of network-based attacks that include potential threat scenarios of *DoS (packet tampering, packet drop, packet duplication)*, and (ii) statistical analysis techniques that use a combination of boolean and threshold functions (*Z-scores*) to detect an anomaly related to application-based attacks (*Unauthorized access*). We demonstrate the effectiveness of our proposed anomaly detection method using a VRLE application case study viz., vSocial, specifically designed for teaching youth with learning impediments about social cues and interactions. Based on our detection results, we validate the impact of network and application based SP attacks on the VRLE UIX.

Index Terms—Virtual reality, security and privacy, user immersive experience, anomaly detection, machine learning

1 Introduction

The adoption of virtual reality using an integrated cloud-based infrastructure merges the physical and the digital world to create a new form of an interactive environment [1], [2]. This allows us to create distributed collaborative environments on a large scale in the form of virtual reality learning environments (VRLEs) for several application domains such as defense (military training, flight simulations), medicine (surgical training), and education (virtual classrooms). Existing works [3], [4], [5] discuss the integration of real-world smart objects with virtual world objects (user avatars) to create virtual environments, where the objects or entities can interact in a real-time manner. The VRLE systems render immersive content from the network-connected wearable devices to the users and enhance human cognition [4]. Although current VRLE systems provide such inherent benefits in user experience and accessibility, they however lack in addressing critical security and privacy (SP) issues that can impact the functionality of the VRLE in terms of user experience.

To understand the SP issues, an exemplar VRLE architecture collects and aggregates data from distributed user/instructor locations to a central cloud storage instance using networked devices. This inter-connectivity of the network edge and the core cloud that is necessary in a VRLE setup makes it vulnerable to new kinds of attack risks. To elucidate, user privacy can be compromised in terms of data confidentiality by an attacker [6]. For instance, malicious users can gain access to sensitive information related to entities in the VRLE without any consent or authorization [7], which can create a loss of confidentiality (LoC) issue. An example of a security attack can be seen in the case where an attacker captures confidential data to tamper the VRLE content that can disorient an avatar (user in the virtual world), which creates Loss of Integrity (LoI) issues. These SP attacks can also cause defacing of content with offensive images (*overlay attack* [8]), obstructions in the view of the users or causing noise attenuation in their learning sessions *occlusion attack [8]*, and reduction of graphical content or delays between both user and avatar movement. Consequently, the SP attacks lead to disruption of the User Immersive

- *Samaikya Valluripally, Benjamin Frailey, Roland Oruche, Aniket Gulhane, Khaza Anuarul Hoque, and Prasad Calyam are with the Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211 USA. E-mail: samaikyavalluripally@gmail.com, {bfrailey, rro2q2, arggm8}@mail.missouri.edu, {hoquek, calyamp}@missouri.edu.*
- *Brady Kruse is with the >Department of Computer Science, Mississippi State University, Starkville, MS 39762 USA. E-mail: bak225@msstate.edu.*
- *Boonakij Palipatana is with the Department of Computer Science, Cornell University, Ithaca, NY 14850 USA. E-mail: bap93@cornell.edu.*

Manuscript received 4 November 2021; revised 15 August 2022; accepted 14 October 2022. Date of publication 24 November 2022; date of current version 8 August 2023.
This work was supported by the National Science Foundation under Awards CNS-2114035, CNS-1647213 and CNS-1659134.
This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by Institutional Review Board University of Missouri-Columbia under Application No. 2006157, and performed in line with the Expedited Categories: 45 CFR 46.110.a(f)(6) and 45 CFR 46.110.a(f)(7).
(Corresponding authors: Khaza Anuarul Hoque and Prasad Calyam.)
Digital Object Identifier no. 10.1109/TSC.2022.3216539

Experience (UIX), which we define as a combination of *usability of application* and *immersive experience* factors. This UIX factor can serve as the overall user experience metric in VRLE sessions and also aid towards developing an effective VRLE application.

In this paper, we propose a novel continuous monitoring methodology using machine learning (ML) [9] and statistical analysis techniques for the detection of SP attacks disrupting UIX in VRLE applications. We further calculate a numerical value termed as "UIX score" which is an aggregated mean of the subjective data collected based on the considered usability and immersion factors during a VRLE session. With the obtained UIX score, we perform a graphical analysis that shows the impact of all the detected SP attacks on VRLE users' experience. Further, we develop our methodology using a realistic VRLE application case study viz., vSocial [4]. The vSocial [10] serves as an immersive learning platform using an integrated cloud-based infrastructure which raises the need to ensure security and safety of the VRLE users when subjected to cyberattacks. Failure to address the SP attack issues in a VRLE can result in undesirable user experience (e.g., poor student engagement, disruption of learning/collaboration, reputation loss for the institution hosting the VRLE, and disruption of user safety (e.g., physical harm, eye strain, cybersickness).

We jointly analyze the combination of SP issues that can impact UIX in the vSocial VRLE as follows. First, based on our prior works [11], we characterize potential SP attacks that can disrupt UIX factors and model them into a novel security and privacy - user immersive experience (SP-UIX) attack tree [11]. We then perform impact analysis of SP issues on UIX factors on a vSocial VRLE instance, via a subjective survey deployed in the form of a virtual questionnaire (VQ) in the VR environment. Second, we create a framework to continuously learn the dynamic resource profiles at the network and application levels in an operational VRLE in order to apply relevant anomaly detection methods. To the best of our knowledge, our work is the first to investigate anomaly detection approaches that focus on analyzing the impact of potential SP attacks on users' UIX in VRLE sessions. We highlight our attack detection shown in Fig. 5 as part of a monitoring system that analyzes relevant data (user, system information) to detect any cyber-attack events. Specifically, we perform pertinent anomaly detection method for *network-based attacks* (i.e., related to security issues arising from e.g., Denial-of-Service (DoS) attacks) and *application-based attacks* (i.e., related to both security and privacy issues arising from e.g., unauthorized access).

In our work, we adopt a combination of ML classifier algorithms (e.g., KNN, multi-label KNN) for the attack datasets with common features (i.e., network-based DoS attacks involving *packet duplication, packet drop and packet tampering*). In addition, we use a statistical analysis technique involving a combination of threshold functions (Z-scores) and boolean conditions for application-based attack datasets (i.e., different forms of unauthorized access that can cause disclosure of user information) with unique/sparse features. In both cases, we collect the different attack datasets from preliminary SP analysis as well as from attack simulations and store them in a Knowledge Base. The Knowledge Base capability allows for training new classification algorithms and for developing new statistical techniques beyond those considered in this paper scope.

We validate the SP attacks outlined in our generated SP-UIX tree of vSocial application using simulation tools (such as Clumsy 0.2 [12] and Wireshark [13]). Based on the collection and analysis of network and application based SP attack traces, we create a new metric viz., *suspiciousness score* (SS_{attack}). The SS_{attack} is calculated as a numerical score based on the attack type and UIX parameters for each of the detected SP attacks. Lastly, we evaluate our proposed anomaly detection methods in a vSocial testbed hosted on an Open Cloud infrastructure [14]. Based on our detection results, we validate the impact of salient quantitative parameters (attack occurrence events, SS_{attack}) related to network and application based SP attacks on the VRLE users' UIX scores.

The remainder of this paper is organized as follows: Section 2 presents related works and issues in similar applications. Section 3 presents our problem formulation and scope. Section 4 outlines our anomaly detection solution approach. Section 5 presents evaluation results on the anomaly detection method. Section 6 provides discussion of the analytical results. Section 7 concludes the paper.

2 Related Works

The literature on the assessment of user experience and attack detection mechanisms for networked and cloud-hosted applications is vast, which motivates us to focus mainly on the prior works applicable to VRLE applications. These works feature common techniques used to address related issues and we contrast them with our novel contributions as summarized in Table 1.

2.1 Immersion and Usability in Virtual Reality

The works in [16], [28] outline user experience models for an immersive virtual environment to measure user feedback via questionnaires using traditional survey methods. The authors in [28] create a tool that measures user experience based on the questions related to judgment, emotion, and technology adoption. In addition, the authors in [15] showcase how a VRLE user can experience side-effects due to the virtual realism of the VRLE termed as *cybersickness* [29].

To evaluate a virtual simulation in a VRLE, the work in [30] use surveys in the form of virtual questionnaires (VQs) instead of the traditional user experience assessment. In our work, we use a combination of both traditional surveys and VQ as part of our UIX assessment [30]. For our survey regarding the questions related to immersion and usability, we adapt the closely related questions from the works in [28], [31] based on our experiments planned for the vSocial users.

2.2 Security, Privacy Risks in VR Systems

We found limited prior works on the specific SP issues in VR systems. In [32] the authors showed how simple attacks can jam or even manipulate the entire position and pose tracking process in VR with their possible countermeasures. Consequently, the authors in [33] showed that cyber-attacks can potentially cause cybersickness based on frame rate manipulation via exploitation of GPU and network vulnerabilities.

TABLE 1
Literature Review of State-of-the-Art of VRLEs

Areas	Contributions	Our Focus/Novelty
Immersion & Usability in VR	- Measures user experience from participants via traditional survey methods [15], [16] - User preference in virtual questionnaires (VQs) as surveys in VRLEs	- Perform UIX assessment via combination of traditional surveys and VQs - Adapted questionnaires related to cyber-sickness and virtual reality
Security, Privacy in VR	- Cybersickness caused by immersive cyberattacks [17], [18] and exposure in VR [19] - SP challenges but lack in vulnerability analysis that impact users [20], [21]	Generate a SP-UIX attack tree to perform threat modeling related to novel SP attack surfaces impacting VRLE user's experience (UIX) based on our prior work [18], [22]
Detection methods for SP issues	- Using network parameters [18] to detect anomaly event over a variety of methods such as attacker POV [23], anomaly-based intrusion detection system [24], and attack fault trees [22] for attack profiles - Development of machine learning and deep learning models [25], [26] for anomaly detection in network scenarios and network intrusion detection systems (NDIS) [27]	- Employ a ML-based approach for network-based attacks \newline - Employ a statistical technique using a boolean z-score calculation or application-based attacks

Moreover, the authors in [8] discuss immersion attacks that can cause physical harm and disrupt user immersion by simulating attacks such as creation of *physical collision attack* with the real world objects by modifying the SteamVR [17] chaperone file, a *disorientation attack* implemented by changing translation and yaw, a *human joystick attack* meant to make a user unintentionally and incrementally move and an *overlay attack* meant to display unintended images. The work in this study [8] serves as one of the main sources for our attack data i.e., to study attack patterns that can potentially affect UIX in a VRLE.

In addition, existing works such as [19], [20] present several security and privacy challenges and their associated threat surface areas on Augmented Reality (AR) and Virtual Reality (VR) systems. However, these works fail to explore the specific vulnerabilities that impact the user in virtual environments at room-scale. The authors in the work [21] discuss how the participants get disoriented due to the exposure in a virtual environment even for a 20 minutes session. Our preliminary work in [11], [18] proposes a novel threat model related to security, privacy and safety issues in VRLE systems. Therein, we perform SQL injections, packet analysis under different attack scenarios to capture the adverse effects on the functionality of the VRLE [22]. We use these results as a baseline for the attack simulations we generate for our SP-UIX attack tree model. Thus, our work uniquely focuses on exploring novel attack surfaces related to SP in VRLE applications and their impact on UIX.

2.3 Detection Methods and Parameters for SP Issues

As part of our proposed anomaly detection method development, we focused on the works relevant to novel SP attacks in human-computer interaction (HCI) and cloud applications. A subset of works address possible defense mechanisms to mitigate the SP issues. The authors in the work [34] discuss the network bandwidth specifications, network delay and computational requirements in VR systems. Their study provides quantifying parameters that can help us in detecting anomaly events in any of the networked devices in VRLEs. In addition, the work in [35] investigates DoS attacks from an attacker point of view, in the hope of determining an optimal attack strategy that can aid in the development of more efficient defense strategies. Moreover, the work in [24] embeds an anomaly based Intrusion Detection system to defend ethereum smart contracts. The event of a network attack in an immersive VRLE can trigger cybersickness. For instance, the work in [22] details a novel framework that uses attack-fault trees for different attacker profiles, and shows how statistical techniques can determine the most vulnerable threat scenarios that induce cybersickness in users.

There are several prior works [25], [26], [36] that adapt different ML methods as part of attack detection. The work in [25] uses a Naive Bayes model to detect SQL injection related anomalies with 93.3% accuracy. In order to determine the network anomalies that can cause DoS attacks, the work in [26] develops an intrusion detection mechanism using a decision tree ML classifier with twelve features and gives the resultant detection rate of 98%. Similarly, a study in [36] compared two ML models (support vector machines, neural networks) in terms of detection rate to determine the more successful one. However, this study [36] uses only one dataset and the hyperparameters for the neural network were not optimized.

The authors in [23] develop a hierarchical hybrid intrusion detection approach for an open-set attack classification on emerging smart network devices. Similarly, the work in [37] investigates the reduction of computational resources in network intrusion detection systems using an ensemble of autoencoders to track network traffic patterns for detecting various network attacks in a computationally-efficient and online manner. Based on our survey of all these existing works for detection methods, we propose a novel anomaly detection method that employs ML models and statistical analysis techniques based on the attack type (i.e., network-based or application-based).

3 Preliminaries of SP Attacks in a VRLE

In this section, we formulate our problem by stating the relationship between potential SP (Security and privacy) attacks and UIX factors in two phases: (i) threat modeling of

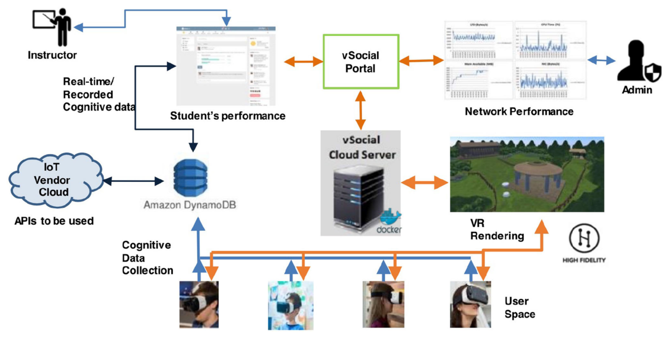


Fig. 1. Overview of vSocial system architecture.

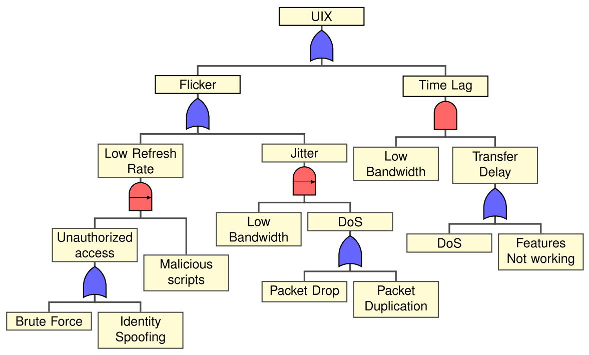


Fig. 2. Attack tree modeling of SP attacks that disrupt UIX.

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.

3.1 Threat Modeling of SP Issues Using Attack Trees

3.1.1 VRLE System Overview

To identify the potential threat scenarios that can disrupt UIX, we use vSocial [4] as our VRLE application case study. In the vSocial application, users interact with each other, move to different spaces within the VRLE, and use their virtual hands (controllers) to complete learning activities as shown in Fig. 1. vSocial features content rendering by High Fidelity [38], an open source platform to run VRLEs featuring virtual worlds. The wearables in vSocial include: VR headsets (e.g., HTC Vive [39]) equipped with VR controllers, EEG headsets (e.g., Muse) on the client-side. A cloud server which runs on an Open Cloud infrastructure slice [14] where the VRLE content is delivered to the users in these virtual classrooms in a real-time manner.

3.1.2 SP Attack Scenarios in vSocial Environment

To study SP attacks on vSocial, there are existing works [27] that can aid in enlisting the potential SP attacks in a VRLE. Table 2 lists the salient acronyms used in remainder of this paper for threat modeling of SP issues. However they do not account for the SP attacks that can occur concurrently or serially in real-world systems. In order to study different SP attacks on vSocial, we use an attack tree (AT) method which has been used extensively in threat modeling [40], [41]. Our proposed work applies the concept of ATs to explore the security, privacy (SP) threat scenarios that can impact UIX factors via a hierarchical visual representation of potential vulnerabilities and threats as shown in the SP-UIX AT in Fig. 2.

TABLE 2
Salient Acronyms Related to Our Proposed SP Anomaly Detection Approach

Acronym	Definition
VRLE	Virtual Reality Learning Environment
UIX	User Immersive Experience
SP	Security and Privacy
SP-UIX	Security and Privacy for User Immersive Experience
VQ	Virtual Questionnaire
SS_{attack}	Suspicousness score of a given attack

Our SP-UIX AT has a goal node (i.e., disruption in UIX) which can be compromised by an attacker via potential attack steps shown as child nodes (i.e., intermediate and leaf nodes). With this AT threat model, we also explore the temporal dependencies in terms of sharing subtrees where the cause and effect relationship of SP attacks on UIX is outlined [22]. In our study context, we generate this SP-UIX attack tree by validating the potential threat scenarios from our prior work in [11], [18] and penetration testing on the vSocial as explained in Section 5.4. Due to the need for ensuring a reliable and safe VRLE application while the user is in session, we focus on the specific VRLE attack "cause and effect relationship" scenarios (e.g., DDoS, Unauthorized access) that disrupt the UIX as shown in Fig. 2. For instance, the leaf node packet drop can be maliciously used for triggering an attack scenario (i.e., intermediate nodes such as e.g., Denial of Service) as shown in Fig. 2. Our SP-UIX attack tree approach is also applicable to other SP attack surfaces that are novel and specific to VRLE applications (e.g., tampering of VRLE learning content, modification of chaperone file as detailed in our prior works [11], [18]).

3.2 Impact Analysis of SP Factors on UIX

3.2.1 Experimental Setup for SP Attacks in vSocial

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 Fig. 3. These activities are part of the learning curriculum of the vSocial environment [18] and rely on fine movement, visual clarity, and clear audio, all of which are disrupted by simulated attacks. We perform an experimental analysis based on one of our earlier works [18] 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 Fig. 3.

After the orientation of the vSocial, a test subject participates and his/her feedback is collected at the end of activity 1 which is the baseline data for a normal 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 [18], 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

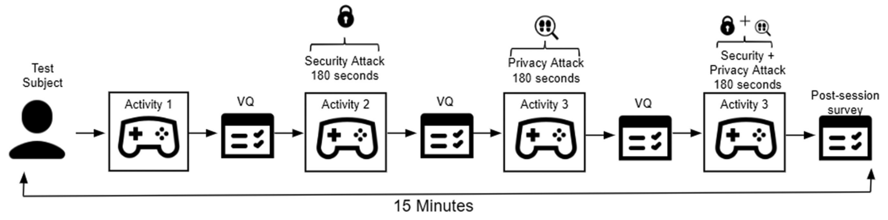

Test Subject
Activity 1
VQ
Security Attack 180 seconds
Activity 2
VQ
Privacy Attack 180 seconds
Activity 3
VQ
Security + Privacy Attack 180 seconds
Activity 3
Post-session survey
15 Minutes

Fig. 3. Activities and their associated virtual questionnaires used for the immersion survey experiment.

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.

These SP attacks are simulated for a duration of 180 seconds (i.e., until the next checkpoint) and then are stopped to avoid further discomfort to the user in relation to their time spent in the VRLE. The surveys conducted for this attack analysis using VQs and post-session paper format are used to measure each of the UIX factors i.e., usability of the application and immersiveness which the VRLE users feel. These VQs' are integrated into the VRLE such that they do not cause any disruption to the users [30].

We recruited 15 participants who are college students for this survey on UIX due to SP attacks. These participants had prior experience in using VR applications and were aware of various VR-related functionalities. We collected their feedback for each of the questions present in our UIX survey on a scale of 1 (very poor) to 5 (excellent). This collected immersion and usability factors subjects data can be used to quantify the aggregated mean value of UIX termed as "UIX scores" in a VRLE setting. Each of the UIX score for every considered SP attack scenario is used for performing the validation analysis of our proposed anomaly detection approach as detailed in Section 5.

3.2.2 Results of the Analysis About the Impact of SP Attacks on UIX Factors

Based on our data collected for the SP versus UIX factors analysis, we outline the data points for each question versus the average rating given by the users as shown in Fig. 4. We observe that the security, privacy, and combination of SP attacks have significantly impacted both the immersion and usability factors. In addition, using the results shown in Fig. 4, we further analyze whether the UIX factors i.e., immersion or usability are negatively impacted by the SP attack scenarios. Based on these results, we observed that such SP attack scenarios triggered an impact of 56% decrease in immersion and 43% decrease in usability. In addition, we also observe that the mental engagement (ME) factor gets more impacted in the case of a privacy breach and in combination scenarios, whereas smoothness of movement (SoM) is affected during a security breach. Similarly, visual clarity (VC) is the usability factor that gets negatively impacted during an SP attack event in a VRLE.

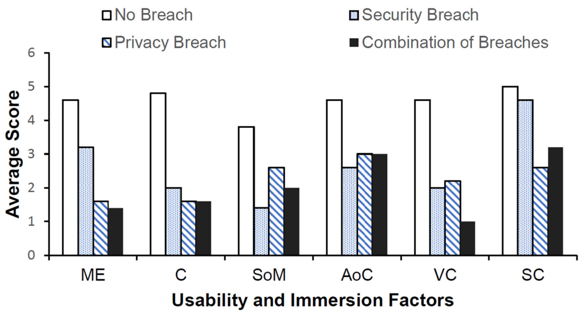


Fig. 4. Results of the UIX survey experiments.

From these results, we understand that SP attacks do certainly affect UIX factors and cause disruption in the functionality of the VRLE. With this preliminary assessment and our modeled SP-UIX attack tree, we further explore the threat signatures to use for detection of such attack scenarios before they disrupt UIX in a VRLE.

4 Anomaly Detection Method for VRLEs

In real-world VRLE applications [3], [4], [5] of dynamic user-system interactions, there exists SP attacks (individual and combination of attacks) with diverse attack patterns. 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 VRLE user data (*security attack*). We propose a novel anomaly detection method to continuously learn from such anomalous VRLE system behaviors and their impact on UIX factors detailed in Section 3.2.2. Our anomaly detection approach is based on DevSecOps principles [42] that suggest the use of diverse solutions for detection of anomalous events (pertaining to both attacks and faults detection) in resilient systems.

The primary goal of our proposed anomaly detection method is to enable the development of a reliable and safe VRLE with capabilities to: (a) identify vulnerabilities against cyber-attacks, and (b) avoid the disruption of UIX caused due to anomaly events during operational use. In this section, we present an overview of our novel anomaly detection approach that is categorized 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 (i.e., network-based attacks and application attacks) in the VRLE data as shown in Fig. 5.

4.1 Attack Detection Module

Our anomaly detection module 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

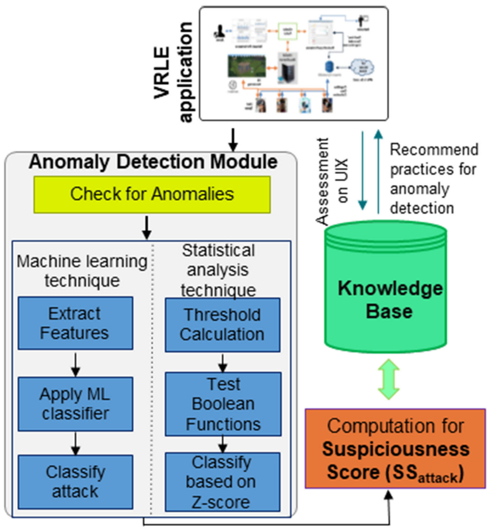


Fig. 5. Anomaly detection of SP attacks on VRLE(s).

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

4.1.1 *Machine Learning (ML) Techniques*

Once an anomaly event is identified, our attack detection module uses ML techniques to classify the type of network-based attacks. Incorporating ML techniques to deal with the continuous input data (session and network information) can aid in automating the detection process and security analysis (e.g., malware, network log classification) [43]. However, such existing works [43] lack in correlation analysis of anomaly events related to user and VRLE system behaviors. For example, large sets of data logs generated from multiple wearable devices (headsets, controllers, emotion headbands) corresponding to geographically distributed users in a VRLE setting can be parsed into an ML model. Moreover, ML can be integrated into functionalities such as network performance and user emotion monitoring, and also 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 significant amounts of data, these attacks share common traits for feature extraction [44], which in turn suggests a good fit for applying ML techniques.

Our ML model used in the anomaly detection method mainly consists of two steps: (i) pre-processing of the anomaly event data, and (ii) classification of the attack type as shown in Fig. 5. In case of network-based attacks, the pre-processing step considers the packet data and the user data for training an ML classifier. Based on pattern analysis, our ML technique can categorize the baseline data (VRLE components, user interaction data) of benign behavior versus attack classification data. However, in the case of SP issues related to application-based attacks, attacks evolve with varied 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.

4.1.2 *Statistical Analysis Technique*

Application-based attacks e.g., unauthorized access that is recorded in a database log as a privacy breach cannot provide enough quantifiable data to use for feature extraction and training ML classifiers. To address this issue, our proposed anomaly event detection module includes performing statistical analysis to detect such application-based attacks.

Due to the single-dimensional nature of the input data used for statistical analysis, we adopt the Z-scores calculation method [45] instead of principal component analysis [46]. Algorithm 1 shows the steps involved in our method to perform Z-score analysis. Our statistical Z-score analysis technique utilizes flag conditions that are further separated into threshold functions [46] and boolean conditions. To flag the benign data (no anomaly) with respect to application-based attack data in VRLEs, we need to determine the thresholds for each triggered attack. For calculating the threshold value, our statistic analysis approach first calculates the standard deviation for each sample of *baseline data*. The Z-score is computed using the standard deviations from a mean, where standard deviation alone only indicates variance.

We adapt the formulations [47], [48] for standard deviations (Equation (1)) and Z-score (Equation (2)) for sample size n, as follows:

$$SD = \sqrt{\frac{\sum (X - \overline{X})^2}{n-1}} \tag{1}$$

$$z = \frac{(X - \mu)}{\sigma}, \tag{2}$$

where X is the individual value, μ is mean, SD is standard deviation, $\overline{X}$ is sample mean and n is the sample size. Based on the deviation from the mean using the formulations in Equations (1) and (2), the threshold values are calculated. 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 5.3.

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 σ and μ, respectively.

$$f(z) = \begin{cases} \textit{Baseline data}, & \text{if } z \in [x, y] \\ \textit{Anomaly}, & \text{otherwise} \end{cases}. \tag{3}$$

More specifically, we use the Equation (3) to compare Z-scores of the incoming data versus the baseline data in the threshold function listed in line 3 of Algorithm 1. In addition to the Z-score, the threshold function in Algorithm 1 also calculates suspiciousness score (SS_{attack}) of the identified anomaly event.

The second set of flag conditions is the Boolean function (see line 15 of Algorithm 1) which is used to validate the boolean conditions especially for application-based attacks such as 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 in order to determine the threshold and boolean conditions to flag malicious events as indicated in the Algorithm 1.

4.2 Calculation of Suspiciousness Score

We define the suspiciousness score (SS_{attack}) as a numerical score for a specific attack pattern detected and classified via our anomaly detection module. The (SS_{attack}) listed in Algorithm 1 is calculated based on the estimated level of impact on the UIX and functionality of the VRLE. This SS_{attack} defined in Equation (4) serves as a quantitative analysis metric of the risk associated with an attack or a specific user in a VRLE. The suspiciousness score calculated for each detected attack in VRLE is defined as follows:

$$SS_{attack} = \sum_{i=1}^{n} (AttackScenario_i * EstLevel\text{Of}Impact_i). \tag{4}$$

Our anomaly detection method stores baseline data, VRLE session information, detected attack patterns (qualitative descriptions of each attack) along with the associated SS_{attack}, and user data into a database, created to serve as a *knowledge base* for training the models employed in our detection approach.

Algorithm 1. Statistical Analysis Pseudocode

```
Input: $D_i$: Input application data (user, session info) from logs
Output: Return suspiciousness score value
1  begin
2    Function $SA$ ():
3      Function $Threshold$ ():
4        Function Z-scores ():
5          Calculate Z-scores: $Z_b$, SD : $\sigma_b$ for baseline data using Eqn. (2);
6          for each $y \in D_i$ do
7            $SD_i = \sqrt{\frac{(x-\mu_b)^2}{n-1}}$
8            $Z_i = \frac{x-\mu_b}{SD_b}$
9          end
10       end Function
11       if $Z_i \notin range(Z_b)$ then
12         Suspiciousness score();
13       end
14       else
15         return as benign data;
16       end
17     end Function
18     Function $Boolean$ ():
19       if $Bcon\_attackisTrue$ then
20         Suspiciousness score();
21       end
22     end Function
23     Function $Suspiciousness\ score$ ():
24       $SS = Est.Impact * Attack$
25       return as SS;
26     end Function
27   end Function SA
28 end
```

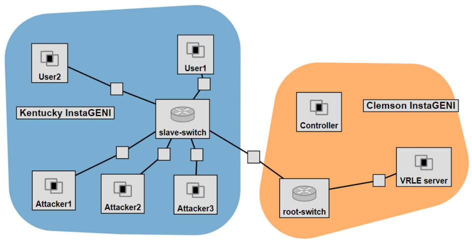


Fig. 6. VRLE architecture implemented in an OpenCloud testbed setup.

5 Evaluation of Anomaly Detection Method

In this section, 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, and (iii) calculation of suspiciousness score for each of the detected network and application-based attacks to show their impact on UIX scores 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.

5.1 Attack Data Generation in the vSocial Application

5.1.1 Testbed Setup

The realistic, Open Cloud [14] testbed that we used in our anomaly detection experiments is shown in Fig. 6. 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 the policy updates. 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 [49] to link the nodes together and subsequently implement a `slow-httptest` script in order to simulate a DoS attack in the Open Cloud testbed.

5.1.2 Tools Used

In order to simulate a DoS attack on vSocial, we used Clumsy0.2 [12] and Wireshark [13] 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).

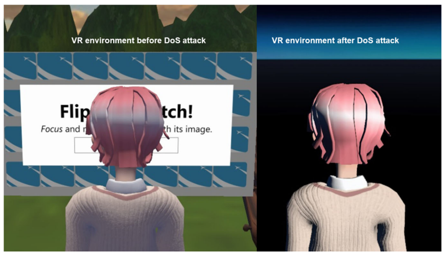

VR environment before DoS attack
VR environment after DoS attack
Flip ... tch!

Fig. 7. A before and after scenario showcasing the effect of DoS attack on vSocial causing a server crash.

5.1.3 *Attack Simulations*

In order to generate the attacks listed in the SP-UIX tree shown in Fig. 2 related to the 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. Moreover, an attacker requires a considerable amount of resources to gain unauthorized access via brute-force as detailed in Section 5.3.

b) Denial-of-Service (DoS) Attack. To launch a DoS attack on the vSocial environment, we perform *packet tampering*, *packet duplication*, and *packet drop* with malicious intent. These attacks impact the VRLE server such that when the DoS attack occurs, the server crashes as shown in Fig. 7. To elucidate, the packet dropping attack targets the communication between the user and VRLE server to disrupt the learning experience [18]. Based on our experimentation, we identify that a packet drop at 80% disrupts the connection to the server very quickly [18], in a worst-case scenario as shown in Fig. 8. Using packet tampering in a man-in-the-middle attack scenario can reveal confidential information as discussed in [18]. From our experiments, we observe that a tamper rate of 20% is sufficient to crash the VRLE server as shown in Fig. 8.

In our proposed work, we use the Attacker Profile (AP) characteristics [50], [51], [52] such as – (i) skill level of the attacker, (ii) resources required to perform a SP attack [11], [22] in VRLEs. Through the SP attack simulation experiments detailed in Section 5.1 we develop the APs as shown in Table 3 as part of validating the SP-UIX tree shown in Fig. 2. Each of the enlisted APs in Table 3 detail the characteristics to perform SP attacks and can be used to estimate the impact value using a uniform scale of 1 to 5.

We categorize the level of impact using the numeric impact value based on the number of VRLE components that get disrupted during an SP attack scenario as shown in Table 3. The APs shown in Table 3 contribute primarily to the calculation of the suspiciousness score discussed in Section 5.4. However, these APs can be further extended for risk management and deploying defense mechanisms on VRLEs in the future. Based on the simulated SP attack data [53], and the modeled SP-UIX attack tree, we illustrate the evaluation of our detection approach employed in the vSocial application.

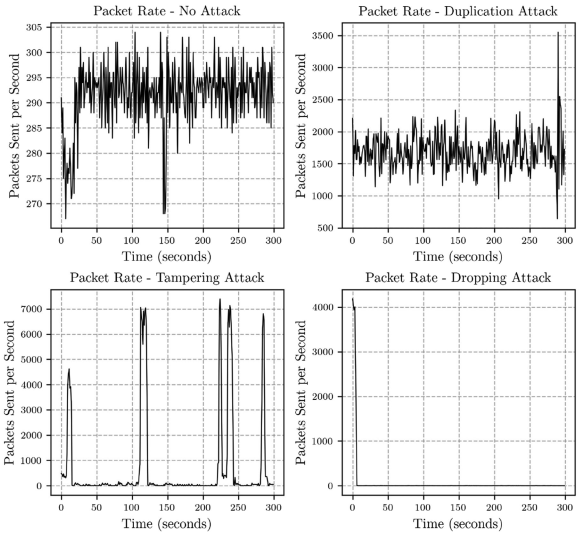


Fig. 8. Packet rate time series for single attacks.

TABLE 3
Attacker Profiling With Estimated Level of Impact

Attack	Resources/ Skill	Level of Impact$^{\phi}$
Unauthorized Access	High	5
Obtaining Local Files	Medium	4
Dropping Packets[1]	Medium	3
Duplicating Packets[2]	Medium	3
Tampering Packets[3]	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

[1]80% Drop Rate
[2]20x Rate, 40% Likelihood
[3]20% Tamper Rate

Impact Scale$^{\phi}$
1. Normal System Functionality
2. Minor Decrease in System Functionality
3. Unusable System Functionality
4. Partially Compromised System
5. Fully Compromised System

5.2 Proposed Anomaly Detection in vSocial Testbed

With the collected data from different SP attacks generated, our proposed anomaly detection uses a two-stage ensemble learning scheme from [18]. 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 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 attacks (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 and then present how our anomaly detection approach can be used in real-time.

5.2.1 *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.

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). Moreover, we use the attack simulation data obtained using our SP-UIX tree analysis method detailed in Section 3.1.2, and also the data obtained through experiments in Section 5.1 for training our ML models to distinguish between benign and anomaly event datasets.

In our preprocessing 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. We collect data at equally spaced time intervals (1 s) i.e., packets rate per second (*PRS*). We can model the packet 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 Fig. 8. This motivates our work to use this time series data to train an effective ML 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.

Feature Extraction. As part of feature extraction, we use `tsfresh` [54], 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 ML models. To select the features (i.e., predictors) that contribute most to the target variable, we used the *Select K Best (SKB) method*. Moreover, we used filter-based feature extraction methods in our proposed anomaly detection approach where the five most distinguishing features among the 212 relevant returned features are selected as shown in Fig. 9. Using our feature extraction, we determine 5 distinguishing features – *Sum of Values*, *Mean of Values*, *Ricker Wavelet*, *Fourier Coefficient (real)*, and *Fourier Coefficient (absolute value)* that clearly differentiate the various classes from one another, and are used for training an ML classifier.

Before we implement an ML classifier, we perform several tests based on quantifiable metrics that suit our data. We repeat the feature extraction step on multiple ML classifiers to aid in developing an algorithm for an accurate

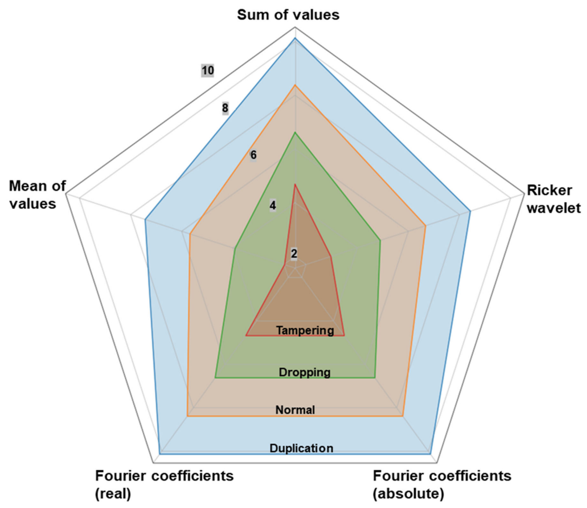


Fig. 9. Five of the most distinguishing extracted features.

detection approach. We implement the following classifiers i.e., {*Decision Tree, Random Forests, K Nearest Neighbors, Ensemble Voting Classifier*} [55] by adapting from the `sklearn` python machine learning library for the best-fit analysis. For each of the classifiers, we run 1,000 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 Fig. 10. Based on our analysis shown in Fig. 10, 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 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.

In addition, we compared the performance of our proposed system with baseline models (i.e., Logistic regression (Baseline 1) and Support Vector Machine (Baseline 2)) [56]

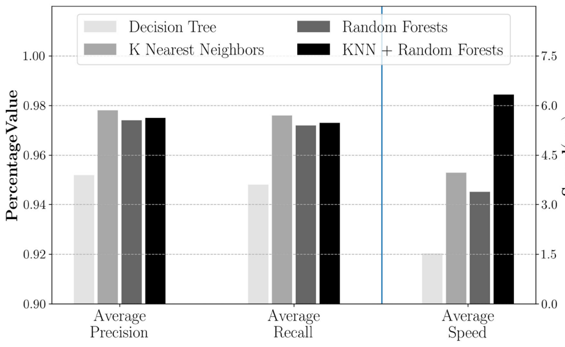


Fig. 10. Single-label classifier accuracy comparison.

TABLE 4
Performance Comparison of Our Approach versus Baseline Models

Name of the model	Classification Accuracy
Baseline 1	52%
Baseline 2	70%
Two-stage ensemble	95%

in terms of accuracy for prediction of anomaly event categories. These baseline models can identify an anomaly event but cannot accurately identify the type of anomaly that is triggered. In contrast, our proposed approach can classify a detected anomaly event into a specific anomaly event category. Table 4 presents the classification accuracy values for these baseline models in comparison with our proposed anomaly detection approach. Based on our analysis shown in Table 4, we determine that the KNN with the K-parameter set to 2 in our anomaly detection approach is most suitable with 95% accuracy when compared to both the baseline models.

Further, 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 Fig. 10, 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 5.

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 [18], 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 5 are the set of: *{A_1– Dropping, A_2– Duplication, A_3– Tampering}* simulated single attack scenarios and *N– represents a 'no breach scenario'.*

5.2.2 Multi-Label Network Based Attack Detection and Classification Using Multi-Label KNN

Based on the results shown in Fig. 10 and in Table 5, 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-labeled K-nearest neighbors (KNN) classification based on the existing works [57]. 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. 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 [58], [59], [60].

In addition, the multi-label KNN initially identifies the k nearest neighbors of the test instance where the label sets of its neighboring instances are obtained. Next, the multi-label KNN uses a *maximum a posteriori* (MAP) principle to predict the set of labels of the test instance using the information of the labels obtained in the neighborhood of an instance [57], [60]. Using a Laplace smoothing method on the Bayes theorem, it determines the posterior probability of an instance labeled for a particular class given the number of neighbors that are in that same class. We employ this multi-label KNN as part of our anomaly detection approach for different

TABLE 5
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	87.5%
Single labeled KNN (single attack scenario)	4.4	2.4	2.8	-	-	-	-	3.2	98.8%	7	97.5%

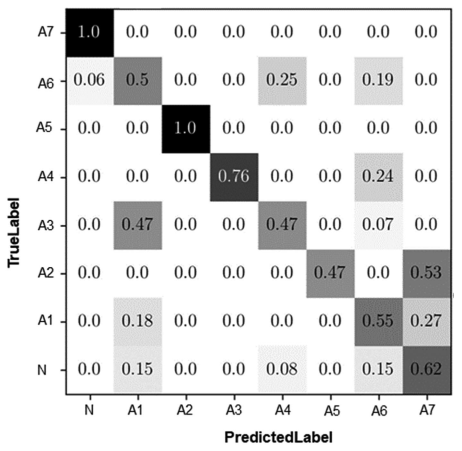


Fig. 11. Multi-label K-NN classifier confusion matrix.

network scenarios (e.g., benign or DoS) in a network-based VRLE. We determine that the time complexity of our anomaly detection method for: (i) training phase is $O(n^2\ d + nqk)$ and for (ii) testing phase is $O(nd + qk)$, where n is the length of unseen instances (n), d is the distance computation between an instance and its neighbors, q is the number of classes, and k is the hyperparameter for selecting the closest neighbors [60].

We run our developed multi-label KNN classifier model for 1,000 tests where a combination of network attack events from the normal behavior (benign) class of data with an average accuracy of 99.5% are detected. We compare these results to check the capability of our model with the existing work [26], 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 1,000 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 powerset models. 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 consider 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 5. We generate a confusion matrix for the multi-labeled classifier as shown in Fig. 11 with the correct classifications and the incorrectly labeled predictions. The listed anomaly event types shown in the Table 5 includes the combination of *{A_4– Duplication + Dropping, A_5– Dropping + Tampering, A_6– Duplication + Tampering, A_7– Tampering + Duplication + Dropping}* multi-attack scenarios simulated and *N– represents a no breach scenario* as shown in Fig. 11. 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 5.

5.3 Application-Based Attack Detection Using Statistical Analysis

Next, our proposed anomaly detection method employs statistical techniques for identifying application-based attacks such as unauthorized access, and disclosure of confidential information which are not feasibly detected by an ML classifier. 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, we implemented the boolean conditions to flag malicious events and thresholding equations to perform Z-score analysis as shown in Section 4.1. We illustrate our experimental evaluation of our statistical technique to calculate the thresholds for the application-based attacks using Z-scores. We consider this baseline data and further compare it with the calculated z-scores of the input data as mentioned in Equation (3). This calculation of SS_{attack} can aid in alerting the VRLE server administrator about the attack.

5.3.1 Data Collection for Z-Score Based Statistical Analysis

In order to test the effectiveness of our statistical analysis technique, our anomaly detection method collects the data by simulating 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 [61], 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 4,958 passwords. We use 123,456 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 1,000 times to record the data that determines the number of attempts taken to identify the correct password. To collect

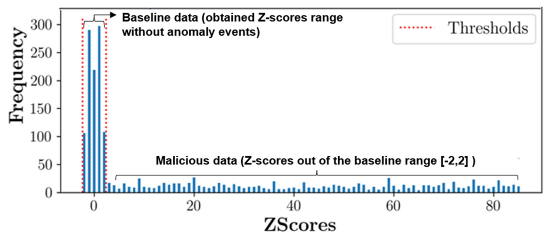


Fig. 12. Bruteforce password attempt Z-score distribution.

the data related to unauthorized access (malicious user trying to login), we run the same test 1,000 times using the attacker database of passwords to simulate a Brute-force attack scenario onto a VRLE system. The data is recorded to identify the number of attempts taken by a malicious user to guess the correct password.

5.3.2 *Application-Based Attack Detection Using Z-Score Analysis*

The data collected in terms of normal (benign) user attempts and malicious (attacker) attempts are analyzed using the deviations calculated 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 1,000 tests due to system limitations and computation time for the statistical analysis. Using the Boolean and threshold functions, we generate a 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 Fig. 12.

The Z-score data related to benign users shown in Fig. 12, 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 Z-score analysis 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 6 where we identify that these both statistics are with higher levels of 99% in terms of *precision* and *recall* factors.

TABLE 6
Bruteforce Attack Detection Scenarios in Terms of Precision and Recall Metrics

Statistic	Good password dataset	Attacker database
Precision	99.95%	99.7%
Recall	99.92%	99.93%

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 passwords. This, considered 100 sets of passwords, represents the scenario where an attacker tries to gain access using other forms of sniffing methods to narrow down 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 statistical parameters *precision* and *recall* for other forms of attacks as shown in Table 6 to understand the efficiency of our Z-score based analysis.

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 solution in Section 4.1 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 Fig. 12 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 Fig. 12. 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 Fig. 12. Thus, using statistical analysis, we identify the application-based attacks (i.e., unauthorized access) with the data threshold shown in Fig. 12.

5.4 Calculation of Suspiciousness Scores (SS_{attack})

Once an attack is detected as shown in Fig. 5, next we calculate the respective suspiciousness score SS_{attack} as described in Section 4.1. Table 7 shows the calculated suspiciousness

TABLE 7
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

scores based on multiple combination of attacks. Using a sandbox technique the SS_{attack} scores are calculated based on the parameter 'Estimated level of impact of an attack on the UIX' as defined in the following equation:

$$est.level_I = \frac{(x - min_{impact})}{max_{impact} - min_{impact}}. \tag{5}$$

Each of 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 the 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 [8]. On the other hand, unauthorized access encompasses any kind of brute-force attack or administrator login from a non-administrator user role. 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 [50], [51]. We remark that these can be extended for future works to determine quantifiable parameters for the application-based attacks.

These SS_{attack} scores can also be used for defense mechanisms when an attack is detected. In this study, we remark that we focus only on developing an anomaly detection method through SS_{attack} score calculation. 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.

6 Discussion

Based on the analytical results discussed in Sections 3.2 and 5, we highlight salient takeaways that can provide insights for employing effective and efficient anomaly detection mechanisms in future VRLE designs.

6.1 Faster Attack Detection

We created a Knowledge Base (KB) to map different attack patterns which have been detected and can be used as a reformulated knowledge in the detection of future anomaly events. Using the KB, we can also train ML models for identifying zero-day attack anomaly events and achieve faster attack detection by continuously learning the dynamic interactions in a VRLE. The KB has information based on our experimental validation in our prior works [18]. For instance, it has information on how we determined the average time taken to crash a VRLE server e.g., complete disruption of a VRLE session due to a DoS attack is 85.5 seconds. In Section 5, we showed that the average detection time of single and multi-attack scenarios by our anomaly detection approach is within 3 seconds. Such an amount of time is less than the determined duration (i.e., 85.5 seconds) that is required to trigger a complete disruption of user experience in a VRLE.

Thereby, we showcase how our anomaly detection method can alert the system administrator in a timely manner to notify that an anomaly event is occurring during a VRLE session. Based on this notification, the VRLE administrator can look into employing counter measures to mitigate the impact of the detected anomaly events that cause bottlenecks. Using our SS_{attack} (that indicates the attack occurrence) history parameter, we can perform even faster detection such that the rate of user experience disruption can be reduced. Thereby, with the KB approach, our anomaly detection can aid in collecting sufficient labeled data for continuously rising attack variants in the form of threat intelligence and can aid in notifying alerts that prompt countermeasure responses to address the anomaly events that cause bottlenecks.

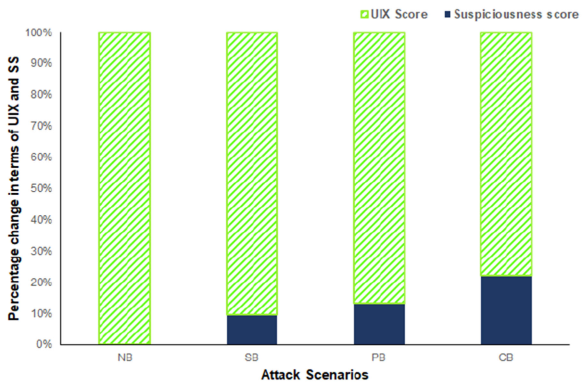


Fig. 13. Relative change in UIX as the Suspiciousness scores increase for different type of SP breaches.

6.2 Validation of SP Attacks Impact on UIX

After the detection of SP attacks and calculation of SS_{attack} using our anomaly detection method, the next step is to validate the impact of the detected attacks on the UIX score. In this section, we provide a validation of the impact of detected SP attacks that aids with the SS_{attack} calculation, which in turn impacts the overall UIX score. We calculate the UIX score for different types of scenarios (attack and benign user), as shown in the Fig. 13.

The analysis between the calculated SS_{attack} and UIX score for different types of scenarios (attack and benign user) is shown in Fig. 13. The x-axis in Fig. 13 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 Fig. 13, 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} and UIX score can be used as salient quantitative parameters in our proposed anomaly detection method. Further, it can be leveraged 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 for handling recurring anomaly events impacting UIX.

Due to this, we determine that a particular anomaly event is more prevalent in VRLEs and can be classified as a critical threat for future recurring anomaly detection. In addition, the SS_{attack} that is associated with estimation of level of impact of several attacks for different attacker profiles can aid in choosing suitable mitigation strategies. For example, a potential defense mechanism, based on SS_{attack}, would include blacklisting the IP of the malicious user, encrypting log files or routine scanning for open ports.

7 Conclusion and Future Work

VRLEs deliver learning content in a collaborative manner to provide an immersive experience for geographically distributed users (i.e., students and instructor(s)). Unauthorized access and DoS attacks affecting VRLE components or data can disrupt the functionality of VRLEs and impact UIX. With our experimental survey, 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.

As part of future work, studies can address security and privacy attacks that target the creation of cybersickness issues amongst users, thus not only impacting the UIX but also the user safety in VRLEs. Further, our findings can be used to inform the best practices to incorporate suitable mitigation strategies for bottleneck anomaly events in future VRLE designs. In addition, we can further improve our ML algorithm by using explainable artificial intelligence techniques (XAI) that enhances the interpretability and trust in our anomaly detection approach [62], [63].

Acknowledgments

Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Impact of IoT on augmented and virtual reality, 2017. Accessed: Oct. 13, 2021. [Online]. Available: https://www.allerin.com/blog/impact-of-iot-on-augmented-and-virtual-reality

[2] M.-C. Tsai et al., "An intelligent virtual-reality system with multi-model sensing for cue-elicited craving in patients with methamphetamine use disorder," *IEEE Trans. Biomed. Eng.*, vol. 68, no. 7, pp. 2270–2280, Jul. 2021.

[3] J.-W. Wu, D.-W. Chou, and J.-R. Jiang, "The virtual environment of things (VEoT): A framework for integrating smart things into networked virtual environments," in *Proc. IEEE Int. Conf. Internet Things*, 2014, pp. 456–459.

[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 *Proc. IEEE 15th Annu. Consum. Commun. Netw. Conf.*, 2018, pp. 1–4.

[5] B. Peixoto, R. Pinto, M. Melo, L. Cabral, and M. Bessa, "Immersive virtual reality for foreign language education: A PRISMA systematic review," *IEEE Access*, vol. 9, pp. 48952–48962, 2021.

[6] D. M. Mendez, I. Papapanagiotou, and B. Yang, "Internet of Things: Survey on security and privacy," 2017, *arXiv:1707.01879*.

[7] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, pp. 2787–2805, 2010.

[8] P. Casey, I. Baggili, and A. Yarramreddy, "Immersive virtual reality attacks and the human joystick," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 2, pp. 550–562, Mar./Apr. 2021.

[9] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, Fourth Quarter 2019.

[10] S. S. Nuguri et al., "vSocial: A cloud-based system for social virtual reality learning environment applications in special education," *Multimedia Tools Appl.*, vol. 80, pp. 16827–16856, 2021.

[11] S. Valluripally, A. Gulhane, R. Mitra, K. A. Hoque, and P. Calyam, "Security, privacy and safety risk assessment for virtual reality learning environment applications," in *Proc. IEEE 17th Annu. Consum. Commun. Netw. Conf.*, 2020, pp. 1–9.

[12] clumsy0.2, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://jagt.github.io/clumsy/download.html

[13] Wireshark, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://www.wireshark.org/

[14] M. Berman et al., "GENI: A federated testbed for innovative network experiments," *Elsevier Comput. Netw.*, vol. 61, no. 14, pp. 5–23, 2014. [Online]. Available: https://ieeexplore.ieee.org/document/8392768

[15] A. Tiiro, "Effect of visual realism on cybersickness in virtual reality," 2018. [Online]. Available: https://members.aixr.org/storage/nbnfioulu-201802091218.pdf

[16] A. Sutcliffe and B. Gault, "Heuristic evaluation of virtual reality applications," *Interacting Comput.*, vol. 16, no. 4, pp. 831–849, 2004.

[17] Steamvr, 2019. Accessed: Oct. 13, 2021. [Online]. Available: http://store.steampowered.com/steamvr

[18] A. Gulhane et al., "Security, privacy and safety risk assessment for virtual reality learning environment applications," in *IEEE Proc. 16th Annu. Consum. Commun. Netw. Conf.*, 2019, pp. 1–9.

[19] Securing your reality: Addressing security and privacy in virtual and augmented reality applications, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://tinyurl.com/yxlplqqe

[20] J. A. De Guzman, K. Thilakarathna, and A. Seneviratne, "Security and privacy approaches in mixed reality: A literature survey," *ACM Comput. Surv. (CSUR)*, vol. 52, no. 6, pp. 1–37, 2019.

[21] Virtual reality headsets could put childrens health at risk, 2017. Accessed: Oct. 13, 2021. [Online]. Available: https://www.theguardian.com/technology/2017/oct/28/virtual-reality-headset-children-cognitive-problems

[22] S. Valluripally, A. Gulhane, K. A. Hoque, and P. Calyam, "Modeling and defense of social virtual reality attacks inducing cybersickness," *IEEE Trans. Dependable Secure Comput.*, pp. 1–18, 2021.

[23] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapé, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *Proc. IEEE Glob. Commun. Conf.*, 2020, pp. 1–7.

[24] X. Wang, J. He, Z. Xie, G. Zhao, and S.-C. Cheung, "Contractguard: Defend ethereum smart contracts with embedded intrusion detection," *IEEE Trans. Serv. Comput.*, vol. 13, no. 2, pp. 314–328, Mar./Apr. 2020.

[25] A. Joshi and V. Geetha, "SQL injection detection using machine learning," in *Proc. Int. Conf. Control, Instrumentation, Commun. Comput. Technol.*, 2014, pp. 1111–1115.

[26] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Comput. Commun.*, vol. 34, no. 18, pp. 2227–2235, 2011.

[27] R. Akella, S. Debroy, P. Calyam, A. Berryman, K. Zhu, and M. Sridharan, "Security middleground for resource protection in measurement infrastructure-as-a-service," *IEEE Trans. Serv. Comput.*, vol. 12, no. 4, pp. 621–638, Jul./Aug. 2019.

[28] K. Tcha-Tokey, E. Loup-Escande, O. Christmann, and S. Richir, "A questionnaire to measure the user experience in immersive virtual environments," in *Proc. Virtual Reality Int. Conf.*, 2016, Art. no. 19.

[29] 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?," *J. Appl. Physiol.*, vol. 125, no. 6, pp. 1670–1680, 2018.

[30] G. Regal, R. Schatz, J. Schrammel, and S. Suette, "VRate: A unity3D asset for integrating subjective assessment questionnaires in virtual environments," in *Proc. 10th Int. Conf. Qual. Multimedia Exp.*, 2018, pp. 1–3.

[31] 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, Art. no. e111511.

[32] M. U. Rafique and S. C. Sen-Ching, "Tracking attacks on virtual reality systems," *IEEE Consum. Electron. Mag.*, vol. 9, no. 2, pp. 41–46, Mar. 2020.

[33] B. Odeleye, G. Loukas, R. Heartfield, and F. Spyridonis, "Detecting framerate-oriented cyber attacks on user experience in virtual reality," in *Proc. 1st Int. Workshop Secur. XR XR Secur.*, 2021, pp. 1–5.

[34] C. Westphal, "Challenges in Networking to Support Augmented Reality and Virtual Reality," in *Proc. Int. Conf. Comput. Netw. Commun.*, 2017, pp. 1–5.

[35] H. Zhang, P. Cheng, L. Shi, and J. Chen, "Optimal denial-of-service attack scheduling with energy constraint," *IEEE Trans. Autom. Control*, vol. 60, no. 11, pp. 3023–3028, Nov. 2015.

[36] A. Osareh and B. Shadgar, "Intrusion detection in computer networks based on machine learning algorithms," *Int. J. Comput. Sci. Netw. Secur.*, vol. 8, no. 11, pp. 15–23, 2008.

[37] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," 2018, *arXiv:1802.09089*.

[38] High fidelity: A social VR platform, 2019. Accessed: Oct. 13, 2021. [Online]. Available: www.highfidelity.com

[39] Vive: Discover virtual reality beyond imagination, 2019. Accessed: Oct. 13, 2021. [Online]. Available: www.vive.com

[40] Attack trees, 1999. Accessed: Oct. 13, 2021. [Online]. Available: http://www.drdobbs.com/attack-trees/184411129

[41] E. J. Byres, M. Franz, and D. Miller, "The use of attack trees in assessing vulnerabilities in SCADA systems," in *Proc. Int. Infrastructure Survivability Workshop*, 2004, pp. 3–10.

[42] H. Myrbakken and R. C. Palacios, "DevSecOps: A multivocal literature review," in *Proc. Int. Conf. Softw. Process Improvement Capability Determination*, 2017, pp. 17–29.

[43] M. Rege and R. Mbah, "Machine learning for cyber defense and attack," in *Proc. 7th Int. Conf. Data Analytics*, 2018, pp. 73–77.

[44] J. Zhao, S. Shetty, and J. W. Pan, "Feature-based transfer learning for network security," in *Proc. IEEE Mil. Commun. Conf.*, 2017, pp. 17–22.

[45] Z-score: Definition, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://stattrek.com/statistics/dictionary.aspx?definition=z_score

[46] Y. Zhang, S. Debroy, and P. Calyam, "Network-wide anomaly event detection and diagnosis with perfSONAR," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 666–680, Sep. 2016.

[47] M. Barde and P. Barde, "What to use to express the variability of data: Standard deviation or standard error of mean?," *Perspectives Clin. Res.*, vol. 3, pp. 113–116, 2012.

[48] P. Driscoll, F. Lecky, and M. Crosby, "An introduction to estimation–1. Starting from Z," *Emerg. Med. J.*, vol. 17, no. 6, pp. 409–415, 2000. [Online]. Available: https://emj.bmj.com/content/17/6/409

[49] N. Foster et al., "Frenetic: A network programming language," in *Proc. 16th ACM SIGPLAN Int. Conf. Funct. Program.*, 2011, pp. 279–291.

[50] M. Rocchetto and N. O. Tippenhauer, "On attacker models and profiles for cyber-physical systems," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2016, pp. 427–449.

[51] Attacker classification to aid targeting critical systems for threat modelling and security review, 2005. Accessed: Oct. 13, 2021. [Online]. Available: http://www.rockyh.net/papers/AttackerClassification.pdf

[52] G. Norman, D. Parker, and J. Sproston, "Model checking for probabilistic timed automata," *Formal Methods Syst. Des.*, vol. 43, no. 2, pp. 164–190, 2013.

[53] Code for attack simulation, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://github.com/boonakij/Packet-Rate-Monitoring

[54] Ts-fresh, 2019. Accessed: Oct. 13, 2021. [Online]. Available: https://tsfresh.readthedocs.io/en/latest/

[55] Scikit-learn, 2021. Accessed: Oct. 13, 2021. [Online]. Available: https://scikit-learn.org/stable/

[56] Multilabel k nearest neighbours, 2017. Accessed: Oct. 13, 2021. [Online]. Available: http://scikit.ml/api/skmultilearn.adapt.mlknn.html

[57] M.-L. Zhang and Z. W. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," in *Proc. IEEE Int. Conf. Granular Comput.*, 2005, pp. 718–721.

[58] Scikit-multilearn, 2017. Accessed: Oct. 13, 2021. [Online]. Available: http://scikit.ml/

[59] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, 2007.

[60] M. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.

[61] Bruteforce-database, 2015. Accessed: Oct. 13, 2021. [Online]. Available: https://github.com/duyetdev/bruteforce-database/

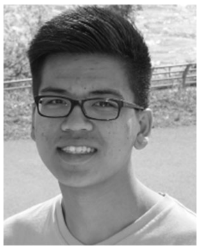
[62] A. Nascita, A. Montieri, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapé, "XAI meets mobile traffic classification: Understanding and improving multimodal deep learning architectures," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 4, pp. 4225–4246, Dec. 2021.

[63] M. Zolanvari, Z. Yang, K. Khan, R. Jain, and N. Meskin, "TRUST XAI: Model-agnostic explanations for AI with a case study on IIoT security," *IEEE Internet Things J.*, pp. 1–13, 2021.

Samaikya Valluripally received the bachelor of technology degree in computer science from Jawaharlal Nehru Technological University, India, in 2014, and the MS degree in computer science from the University of Missouri-Columbia, in 2016. She is currently working toward the PhD degree in computer science with the University of Missouri-Columbia. Her current research interests include cloud security for AR/VR and IoT applications, Big Data analytics.

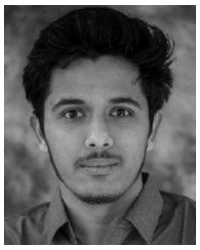
Benjamin Frailey is currently working toward the BS degree in computer science and a minor in mathematics with the University of Missouri-Columbia. His research interests include cyber security, cloud computing, machine learning, Internet of Things, and computer networking.

Brady Kruse is currently working toward the BS degree in computer science with emphasis in cyber security and a minor in mathematics and english with Mississippi State University. His research interests include quantum computing, Internet of Things, cyber security, and science fiction.



Boonakij Palipatana is currently working toward the BA degree in information science with a concentration in data science with Cornell University. His research interests include machine learning, natural language processing, and computational social science.

Roland Oruche received the BS degree in information technology from the University of Missouri-Columbia. He is currently working toward the PhD degree in computer science with the University of Missouri-Columbia. His research interests include machine learning, recommender systems and human-computer interaction.



Aniket Gulhane received the BE degree in computer engineering from Savitribai Phule Pune University, India, in 2016. He is currently working toward the MS degree in computer science with the University of Missouri-Columbia. His research interests include cloud security, human computer interaction, cloud computing.

Khaza Anuarul Hoque (Senior Member, IEEE) received the MSc and PhD degrees from the Department of Electrical and Computer Engineering, Concordia University, Montreal, Canada, in 2011 and 2016, respectively. He is currently an assistant professor with the Department of Electrical Engineering and Computer Science, University of Missouri-Columbia (MU) where he directs the Dependable Cyber-Physical Systems (DCPS) Laboratory. Before joining MU, he was a postdoctoral research fellow with the University of Oxford, U.K. His research interests include formal methods, cyber-physical systems, cybersecurity, and safe AI/ML. He is a member of ACM, and AAAS.

Prasad Calyam (Senior Member, IEEE) received the MS and PhD degrees from the Department of Electrical and Computer Engineering, The Ohio State University, in 2002 and 2007, respectively. He is currently an associate professor with the Department of Computer Science, University of Missouri-Columbia. Previously, he was a research director with the Ohio Supercomputer Center. His research interests include distributed and cloud computing, computer networking, and cyber security.