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, and Prasad Calyam

Abstract—Virtual Reality Learning Environments (VRLE) are a new form of immersive environments which are integrated with IoT 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 users' immersive experience (UIX). In this paper, we identify potential SP attack surfaces that impact the application usability and presence of immersion factors, and propose a novel anomaly detection system 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 includes 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 approach 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 users' UIX.

Index Terms—Social Virtual Reality, Security and Privacy, User Immersive Experience, Anomaly Detection, Machine Learning

1 Introduction

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 [1]. 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), disaster management (virtual smart cities), and education (virtual classrooms). Existing works [2]–[4] 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. To increase the user engagement effectively, real-time IoT applications (e.g., public safety in smart cities) integrate VRLEs for disaster management and damage containment [5]–[7]. The VRLE systems render immersive content from the network-connected IoT devices to the users [8] and enhance human cognition in several subjects such as psychology and physiology [9]. 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 application [10].

To understand the SP issues, let's consider an exemplar VRLE architecture shown in Figure 1. Networked devices

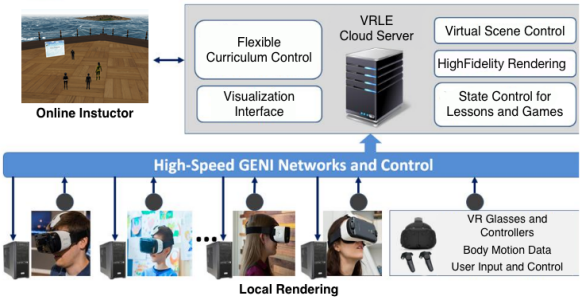


Figure 1: Architecture of a VRLE application across high-speed networks for an online learning environment

in the VRLE collect and aggregate data from distributed user/instructor locations to a central cloud storage instance. This interconnectivity 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 [11]. For instance, malicious users can gain access to sensitive information related to entities in the VRLE without any consent or authorization [12], which can create a loss of confidentiality (LoC) issue. An example of a security attack can be seen in the case where an attacker can capture confidential data to tamper the VRLE content that can disorient an avatar (user in the virtual world), which creates a Loss of Integrity (LoI) issue. These SP attacks can also cause reduction of graphical content or delays between both user and avatar movement. Consequently, the SP attacks lead to disruption of the User Immersive Experience (UIX), which we define as a combination of *usability of the application* and *sense of immersiveness* factors.

- *S. Valluripally, B.Frailey, R. Oruche, A. Gulhane, K. A. Hoque, P. Calyam* are with the Department of Electrical Engineering and Computer Science, University of Missouri-Columbia, Columbia, MO, 65211. *Corresponding Author; E-mail: calyamp@missouri.edu*
- *B. Kruse is with the Department of Computer Science at Mississippi State University; B. Palipatana is with the Department of Computer Science at Cornell University.*

In this paper, we propose a novel continuous monitoring methodology using machine learning (ML) and statistical analysis techniques for the detection of SP attacks disrupting UIX score in VRLE applications. We define UIX score as the aggregated value of the data collected based on the considered presence and immersion factors during a VRLE session. Our approach involves using a realistic VRLE application (viz., vSocial) case study [8]. vSocial is a cloud-based virtual reality platform developed for immersive online social learning based on the Social Competence Intervention (SCI) curriculum for youth with Autism Spectrum Disorder (ASD). We jointly analyze the combination of SP issues that can impact UIX in the vSocial VRLE as follows. Firstly, based on our experimental behavior analysis on a vSocial VRLE instance, we characterize potential SP attacks that affect the UIX factors. We capture different attack signatures, corresponding to specific layers of the VRLE platform and study their impact on the UIX factors. We then show the dependency relationship between the potential SP issues and UIX factors via a subjective survey that is based on UIX factors, and deployed in the form of virtual questionnaires (VQ) in the vSocial VRLE. Based on the preliminary results, we formalize the potential SP issues of the VRLE into a novel security and privacy - user immersive experience (SP-UIX) attack tree. An attack tree [13] is defined as a graphical structure that represents attacks and their respective consequences towards a common system disruption goal.

Secondly, 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 IoT-based VRLE applications. Specifically, we develop pertinent anomaly detection approaches 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). Our attack detection strategy builds on the principles of DevSecOps [14], [15] which suggest that attacks have unique sets of features and varying data types, and thus a 'one size fits all' technique is not suitable. 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 [16], BeEf -xss [17] and Wireshark [18]). 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 using human subject based testing in a vSocial testbed hosted on the GENI Cloud infrastructure [19]. Based on our detection results, we validate the impact of salient quantitative paramenters (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 summarizes related works and the current findings of issues in similar applications. Section 3 determines our problem formulation and issues that we address. Section 4 outlines our solution approach. Section 5 presents experimental results on the anomaly detection approach. Section 6 discusses the validation of SP attacks on the VRLE users' UIX. Section 7 concludes the paper.

2 Related Works

The literature on the assessment of user experience and attack detection mechanisms for IoT based applications is vast, so we focus on the prior works that are applicable to VRLE application scenarios. These findings define common techniques used to address related issues and contrast with novel contributions of our work.

2.1 Immersion and Usability in Virtual Reality

The literature in [20] discusses a user experience model for an immersive virtual environment to measure user feedback via questionnaires over traditional survey methods. The authors in this work create a tool that measures user experience based on the questions related to judgement, emotion, and technology adoption. In addition, the authors in [21] showcase how a VRLE user can experience side-effects due to the virtual realism of the VRLE termed as *cybersickness* [22]. To evaluate the sense of presence in a virtual environment, the work in [23] discusses the practical methods using a rating scale of net positive values to reflect user responses. The analytical results in this work explain the potentiality of using such heuristic-based methods to evaluate a virtual simulation in VRLE.

Instead of using traditional methods of assessing the user experience, there are works on implementing the survey questions in the form of virtual questionnaires (VQ) in the VRLE. The work in [24] implements VRate, an asset that allows integration of questionnaires in a three-dimensional engine. In order to assess the user experience (UX) and quality of experience (QoE), a two-dimensional interface is used in fully immersive and interactive virtual environments (VE). Based on their survey results, the authors in [24] observe that the users prefer virtual questionnaires over paper-based survey forms.

In our work, we use a combination of both traditional surveys and VQ as part of our UIX assessment. We adapt the VQ format from the work discussed in [24] and integrate it into our VRLE and the traditional survey for post-session assessment. For our survey regarding the questions related to immersion and usability, we adapt the closely related questions from the works in [20], [25] based on our activities planned for the vSocial users.

2.2 Security, Privacy Risks in VR Applications

We found limited prior works on the specific SP issues in VR systems. There have been methods proposed to simulate several SP attacks in other IoT-based applications that can be adapted to a VRLE. The authors in [26], 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 [27] 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 [26] serves as a fundamental source for our attack data i.e., to study attack patterns that can potentially affect UIX in VRLE.

In addition, existing works such as [28], [29] presents several security and privacy challenges and their associated threat surface areas on Augmented Reality (AR) and Virtual Reality (VR). However, these works fail to explore the specific vulnerabilities that impact the user for virtual environments at room-scale. Similarly, the authors in [30] discuss safety issues in virtual environments and their impact on the human performance due to the tasks included. Furthermore, the authors in the work [31] discuss about how the participants get disoriented due to the exposure in a virtual environment even for a 20 minutes session. One of our preliminary works [32], develops a novel threat model related to security, privacy and safety issues in VRLE applications. Therein, we perform SQL injections and packet dropping to capture the adverse effects on the functionality of the VRLE. We use this study [32] as a baseline for the attack simulations we intend to generate for our formalized SP-UIX attack tree model. Thus, our work focuses on exploring potential novel attack surfaces related to security, privacy in VRLE applications and their associated impact on UIX.

2.3 Existing Detection Methods and Parameters for SP Issues in Other Applications

As part of our proposed anomaly detection approach, we surveyed existing works that employ different solutions for detecting various cyberattacks in IoT based applications. A subset of our findings discusses possible defense mechanisms to mitigate the SP issues. However, for our proposed approach, we limited our focus on detecting the attacks for human-computer interaction (HCI) and IoT related applications. The authors in the work [33] discuss the network parameters such as specifications of bandwidth, network delay and computational requirements in VR applications. Moreover, the authors explore different network architectures and their adaptability in terms of network configuration in virtual environments. This study provides quantifying parameters that can help us in detecting an anomaly in any of the networked devices in VRLEs. In addition, the work in [34] 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, this work [34] determines the packet loss rate along with the energy and sensor strength parameters to distinguish between an attack and a fault.

There are several prior works [35]–[37] that adapt different ML methods as part of attack detection. The work in [35] 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 [36] 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 [37] compared two ML models (support vector machines, neural networks) in terms of detection rate to determine the more successful one. However, this study [37] uses only one dataset and the hyperparameters for the neural network were not optimized. Based on our survey of existing works for detection methods, we propose a novel detection method that employs ML techniques and statistical analysis based on the attack type (network or application-based attacks).

3 Preliminaries of SP Attacks in VRLE

We collect the data related to potential vulnerabilities via different subjective assessments as shown in the overview Figure 2. In this section, we formulate our problem by stating the relationship between potential SP attacks and UIX factors in two phases: (i) Analyzing the relationship of the impact of SP attacks on UIX due to the vulnerabilities in a VRLE, and (ii) Formalization of the vulnerable components that can cause potential SP issues that disrupt users' UIX.

3.1 Impact Analysis of SP Factors on UIX

3.1.1 VRLE system overview

To analyze the relationship of potential SP attacks on UIX factors, we use vSocial as our VRLE application case study. The users interact with each other, move to different spaces within the VRLE, and use their virtual hands (controllers) to complete learning activities. The cloud server in our vSocial architecture shown in Figure 3 delivers the VRLE content to the users in these virtual classrooms and stores the user engagement information and activities in real-time.

3.1.2 SP attack scenarios in vSocial Environment

In order to study different SP attacks on vSocial, We utilize a preliminary threat model that includes different layers of VRLE application such as network, physical, and application platforms as shown in Figure 4. For each of these layered platforms, we classify the potential SP attacks via different vulnerabilities, exploitable VRLE components as shown in Figure 4.

For instance, an attacker at the application level can perform *unauthorized access* by spoofing a valid user identity, assuming credentials are obtained via a malicious action. Similarly, an intruder can *insert malicious scripts* to run as part of the VRLE application thereby causing a *Data tampering* attack. Moreover, at the network level, an attacker can send continuous requests to a server and cause a *Denial-of-service* attack. An example of a Physical layer attack can be seen when an attacker can intercept and sniff the data related to the devices by *eavesdropping or gaining access* to the IoT devices. Although, such preliminary threat models can aid in enlisting the potential SP attacks due to vulnerabilities in a VRLE, they however lack flexibility in addressing the SP attacks that can occur concurrently in real-world systems.

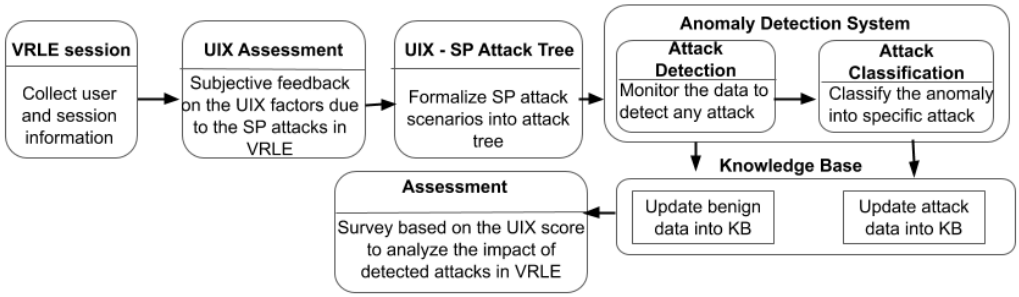


Figure 2: Overview of our anomaly detection approach for addressing SP attacks on a VRLE

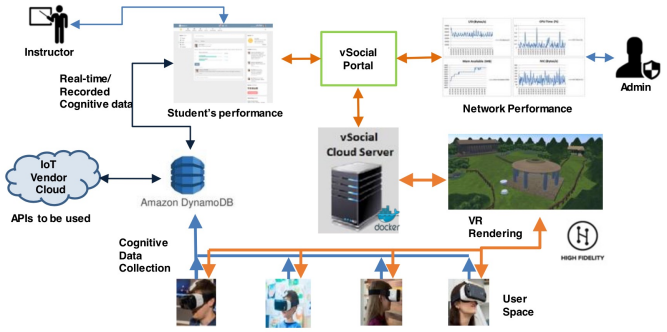


Figure 3: Overview of vSocial system architecture.

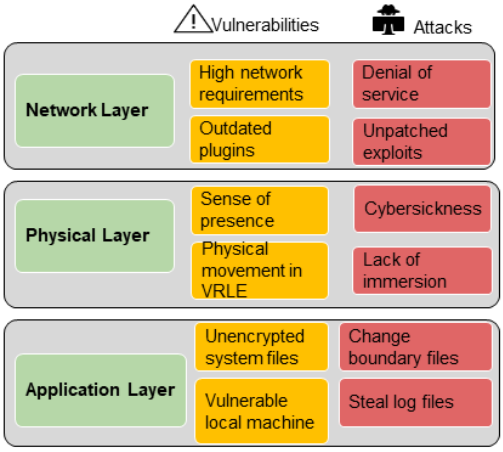


Figure 4: A layered diagram of vulnerabilities and their associated attacks in VRLE

For example, sensitive user data can be disclosed by gaining *unauthorized access* to the data termed as a combination of security and privacy attacks that occurs concurrently in a VRLE. Thus, we outline the preliminary version of potential SP attacks using a layered approach.

3.1.3 Experimental Setup for SP attacks in vSocial Environment

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 learning environment with three different activities for users to perform as shown in Figure 5. The subjects enter the VRLE and are given orientation instructions about how to interact with the learning environment. We perform this experimental behavior analysis based on one of our earlier works [32] to assess significant factors that disrupt UIX in VRLE. We simulate

Table 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	[38] [20]
Comfort	Do you feel comfortable in the virtual environment?		[25]
Smoothness of Movement	How Smooth is movement in the virtual environment?	PQ	[38] [20]
	Usability Question		
Accuracy of Controls	Are the controls sensitive to your movement and capable of performing tasks?	PQ	[38] [20]
Visual Clarity	Are you able to see the games clearly?	PQ	[38] [20]
Sound Clarity	Are you able to hear the sounds in the virtual environment?	PQ	[38] [20]

three attacks across the activities and quantify UIX using a set of virtual questionnaires (VQ) as shown in Figure 5, which gives an overview of our analysis.

After the orientation, the test subject participates and the user feedback is collected at the end of activity 1 using a VQ as shown in Figure 6. This feedback data related to activity 1 serves as a baseline data for a 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 [32], 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 play a distracting noise 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 simulated 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.

The activities performed by the subjects are part of the learning curriculum of the vSocial environment [32] and rely on fine movement, visual clarity, and clear audio, all of which are disrupted by our attacks. We run the attacks until the next checkpoint is reached–if the attack duration reaches 180 seconds, we stop the attack simulation to avoid further discomfort to the user with respect 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 factor i.e., usability of the application and sense of immersiveness which the VRLE users feel.

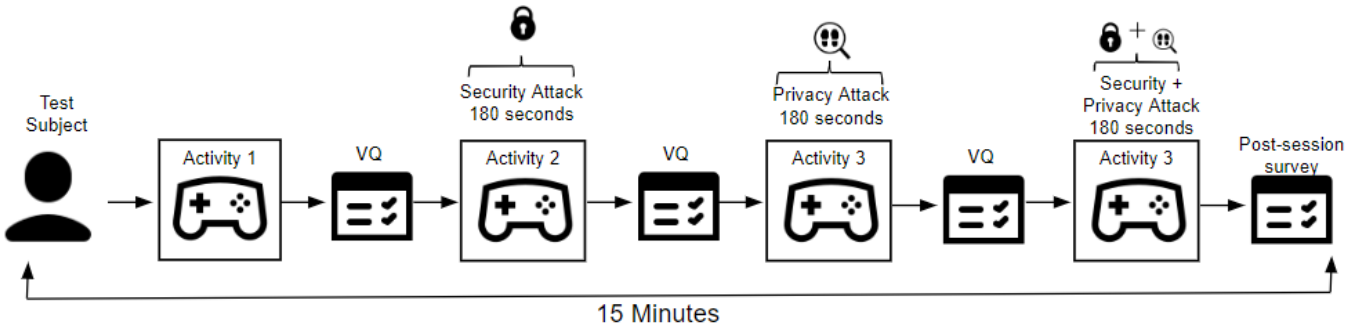


Figure 5: Activities and their associated virtual questionnaires used for the immersion survey experiment

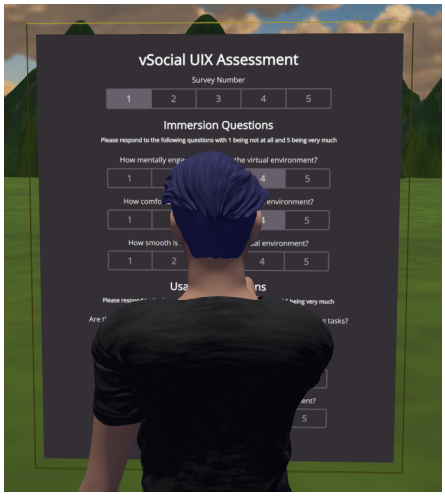


Figure 6: User giving feedback via virtual questionnaire after 1st activity in vSocial Learning Environment

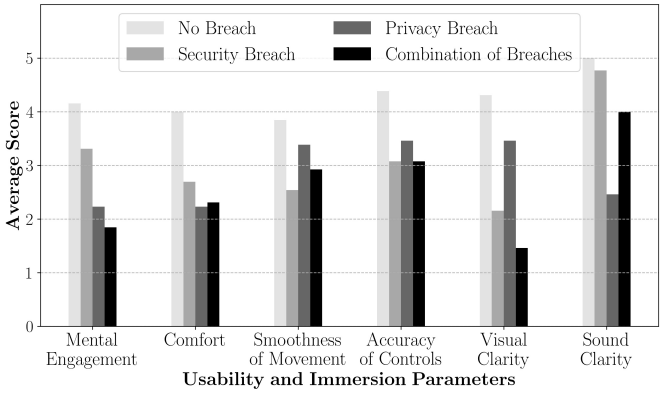


Figure 7: Results of survey questions from UIX survey experiment

These VQs' are integrated into the VRLE such that they do not cause any disruption to the users [24]. We model the questions in these VQs' based on the existing works and surveys [38] [20], such as the Presence Questionnaire (PQ) and Immersive Tendencies Questionnaire (ITQ). For the purpose of our preliminary data analysis, we use only 6 questions as to measure usability and immersiveness, though the questions can be extended for future research. To maintain uniformity across the collected feedback data, we used the same PQ and ITQ questions in every VQ as shown in Figure 7. We categorize each of these questions into Immersion and Usability categories listed in Table 1.

Table 2: Correlation between usability and immersion

Correlation Test	*Correlation Coefficient*	*p-value*
Pearson	0.708	$4.27e^{-9}$
Kendall	0.570	$3.91e^{-8}$
Spearman	0.727	$1.03e^{-9}$

We recruited 15 human subjects to participate in this UIX survey and to obtain their feedback for each of the questions present in our UIX 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 UIX scores in the VRLE.

3.1.4 Results of the analysis about the impact of SP attacks on UIX factors

To investigate the impact of SP attacks on UIX factors, we first study whether our chosen usability and immersion related questions are dependent on each other. Based on the definition of UIX factors (usability of the application and sense of immersiveness), we perform several statistical correlation tests using `scipy` [39] between the data collected for these set of questions. From our statistical correlation analysis, we observe that the correlation coefficient for different tests between usability and immersion is high ≥ 0.50 as shown in Table 2. For this statistical analysis, we consider our null hypothesis as *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 UIX factors analysis, we outline the data points for each question versus the average rating given by the users as shown in Figure 7. We observe that the security, privacy, and combination of SP attacks has significantly impacted both the immersion and usability factors. 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 analysis, we continue to explore the potential security and privacy threats in detail (single and combination of threats) that could disrupt a user's UIX in a VRLE.

3.2 Formalization of SP Issues into an Attack Tree Model

As mentioned in Section 3.1.2, it would be complex to show the interplay for different combination of SP attacks that can

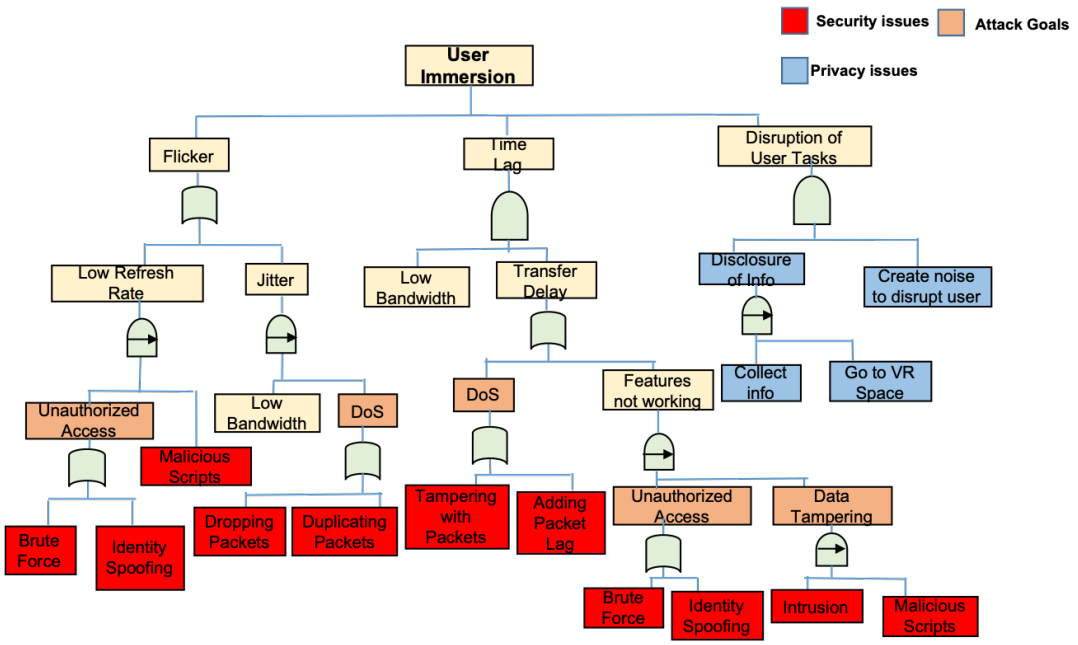


Figure 8: Formalized immersion attack tree with logical gates

act concurrently or even serially. Using our experimental results on SP issues' impact on UIX factors, we further outline more potential combinations of SP attacks that occur as part of the dynamic VRLE interactions. Using our preliminary threat model shown in Figure 4, we create a formalized graphical representation of the VRLE attack surface known as an "attack tree".

Definition: ***Attack trees:*** They are hierarchy based graphical models that show how the disruption goal of an attacker (root node) can be refined into smaller goals (intermediate nodes) until it reaches to the leaf nodes (basic threats) where no further refinement is possible.

Existing works discuss the exemplar attack trees that simplify the mapping of different vulnerabilities across different layers of SCADA applications [40], [41]. We generate a formalized attack tree whose root node (disruption in user immersion) is the goal of an attacker, and intermediate nodes are the potential attacks. The leaf nodes of this generated SP-UIX attack tree consists of security and privacy threats as shown in Figure 8, which describe the potential causes of their respective 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 attack tree by validating the potential threat scenarios from our prior work in [32] and penetration testing on the vSocial application study which is explained in detail in Section 6. With the results from our preliminary SP versus UIX factors analysis and the formalized attack tree, we next explore the threat signatures pertaining to each of the attack scenarios described in the SP-UIX attack tree. This preliminary data informs detailed characterization of SP attacks that impact users' UIX.

4 Proposed Anomaly Detection Approach

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

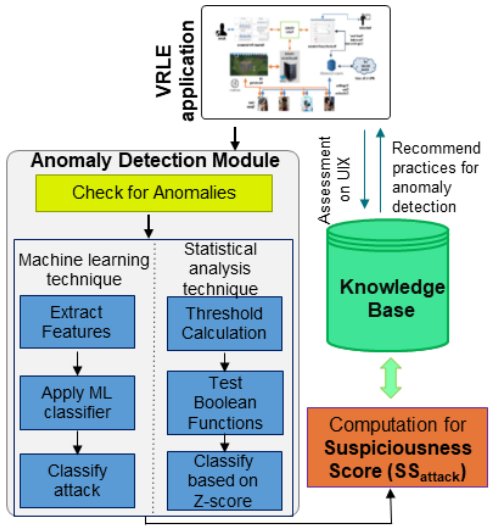


Figure 9: Anomaly detection approach to address SP attacks on VRLE(S)

and mitigate the impact of SP attacks on UIX factors, we propose a novel anomaly detection framework approach that is based on DevSecOps [14], [15] principles of diverse solutions for change (i.e., attack, fault) detection in resilient systems. In this section, we present an overview of our anomaly detection framework approach as shown in Figure 9 for our VRLE application case study. We categorize our anomaly detection approach into three main modules: i) *Anomaly Detection module* to check and classify for anomalies in the incoming data, ii) *Compute Suspiciousness Score* SS_{attack} for the detected attacks in the VRLE data, iii) *Knowledge Base* created for storing the attack patterns and VRLE session information. Thus, through our anomaly detection framework approach, multiple data streams can be checked for anomalies using different methods (i.e., the DevSecOps solution diversity principle) as a continuous (iterative) monitoring process.

4.1 Attack Detection Module

The primary goal of our anomaly detection system is to develop a secure and reliable IoT based VRLE that can detect vulnerabilities against cyber-attacks and avoid the disruption of UIX via anomaly events during operational use. We highlight our attack detection module shown in Figure 9 as a monitoring system that acts upon the data (user, system information) to detect any attack events. For this, the data is checked for an anomaly event using the mechanisms employed in our anomaly detection module as shown in Figure 9. The data collected from the VRLE components and through user interactions in real-world VRLE applications without any attack scenarios (benign behavior) is termed as *baseline data* in our study. To check for anomalies, these mechanisms utilize the baseline data stored in the Knowledge Base created for the vSocial application. Our detection module based on DevSecOps [14], [15] principles of diverse solutions 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.

Machine Learning (ML) Techinques: 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 [42]. In our proposed anomaly detection module, we implement a ML classifier that can accurately differentiate normal (benign) behavior from a number of network-based attacks (DoS). Even though these network attacks generate a lot of data, these attacks share common traits in features [43], [44] as shown in Table 4 which in turn suggests a good fit for applying ML techniques.

Our attack detection ML technique shown in Figure 9 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 normal baseline data to attack type. 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.

Statistical Analysis: Our proposed anomaly detection approach performs statistical Z-score analysis [45] 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. 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 [46]. However, due to the single-dimensional nature of the data considered for the statistical analysis, we calculate Z-scores instead of principal component analysis as discussed in [46]. To determine the deviation of attack data from normal (benign) VRLE application behavior, Z-score has the potential in determining the thresholds for each attack encountered. The Z-score is a calculation of standard deviations from a mean, where standard deviation alone only indicates variance. Based on the existing literature [47], [48], we adapt the formulations of standard deviations (Equation 1) and Z-score (Equation 2) for sample size n as follows:

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

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

where X is individual value, μ is mean, SD is standard deviation, $\overline{X}$ 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 1, 2 threshold values are calculated.

Once the Z-scores for *baseline data* are calculated, our anomaly detection approach proceeds to calculate the Z-scores for the incoming data and determines any suspicious data (anomaly) 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. Our anomaly detection approach determines the suspicious patterns anomaly category in VRLE data based on the outlined categories in Equation 3.

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

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 and determines the Z-scores for baseline data and incoming application data. 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 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 anomalies, we compare Z-scores of the incoming data to the baseline data in the Threshold () function listed in 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, our Algorithm 1 utilizes Boolean () function 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 user event. Thus, the application-based attacks are detected based on the calculation of Z-scores using the equations 1, 2 to determine the threshold and boolean conditions. Such a determination helps to flag malicious events as detailed in Table 9 and Algorithm 1.

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 : σ_b for baseline data using eqn 2;
                for each y ∈ D_i do
                    SD_i = sqrt( (x−μ_b)^2 / (n−1) )
                    Z_i = (x−μ_b) / SD_b
                end
            end Function
            if Z_i ∉ 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
```

4.2 Calculation of Suspiciousness Score

Based on the attack classification results, our proposed anomaly detection approach proceeds to calculate 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 VRLE as mentioned in the Equation 4 and Algorithm 1. The suspiciousness score calculated for each detected attack in VRLE is as follows:

$$\text{SS}_{attack} = \sum_{i=1}^{n} (AttackScenario_i * EstLevelOfImpact_i) \quad (4)$$

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} acts as quantitative analysis of 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 based on the ML classifier and Z-score statistical analysis to add to the overall suspiciousness score iteratively.

4.3 Knowledge Base Module

Our anomaly detection approach 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 contains and actively stores VRLE session information, detected attack patterns (qualitative descriptions of each attack) along with its associated SS_{attack}, and user data. The data can be stored to create a better model (those not considered in our current study) and use as knowledge for detection of zero-day attack anomaly events. With data being iteratively appended into the knowledge base, it can be used for training any anomaly detection approach to understand the dynamic interaction in a VRLE and accurately identify attack patterns. The Knowledge base in this paper can be enhanced and adapted for future VRLE system enhancement activities.

5 Evaluation of Anomaly Detection Method

In this section, we establish the effectiveness of our proposed anomaly detection approach. We evaluate the performance of our two mechanisms (ML classifier, Statistical 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 (iv) Analysis of the impact on UIX scores due to the SP attacks detected using our proposed anomaly detection framework in realistic settings. To demonstrate our anomaly detection module 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 vSocial application

5.1.1 Testbed setup

We evaluate the efficacy of our anomaly detection system using a realistic, GENI Cloud [19] testbed as shown in Figure 10. 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. 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 [49] to link the nodes together and subsequently implement a `slowhttptest` script in order to simulate a DoS attack in the GENI Cloud testbed setup that hosts the vSocial application as shown in Figure 10. Before illustrating all the simulated SP attacks, we first discuss the tools used as part of performing attack scenarios on the vSocial application client.

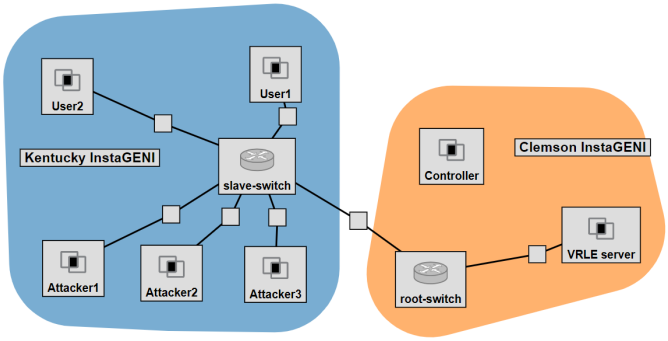


Figure 10: VRLE architecture replicated in GENI

5.1.2 Tools Used

As part of simulating several SP attacks, we used a Kali Linux [50] VM specifically with a few built-in tools such as: (i) BeEf -xss [17] to perform web hooking attacks, (ii) Metasploit for port scanning and running attacks using the CVE database [51], and vulnerabilities related to unauthorized access found in Trivial File Transfer Protocol (TFTP) [52]. In order to simulate a DoS attack on vSocial, we used Clumsy0.2 [16] and Wireshark [18] 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).

5.1.3 Attack simulations

In order to generate the attacks listed in the SP-UIX tree shown in Figure 8 related to VRLE application, we perform SP attacks. Using the leaf nodes 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), XSS Browser attack, Overlay attack and modification of Chaperone file that causes an attack.

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 5.3.2. Moreover, an attacker requires considerable amount of resources to gain unauthorized access via brute-force. In addition, the likelihood of a server being too vulnerable would be low. In order to obtain the local files from such server, we determine based on the existing work [32], the amount of resources required would go from medium to high.

b) Denial-of-service (DoS) attack: To perform a DoS attack on 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 occurs, the server crashes as shown in Figure 11. To elucidate, the packet dropping attack targets the communication between the user and VRLE server to disrupt the learning experience [32], which could be an easy task for the attacker if present in the same network to get the server IP information. Based on our experimentation, we identify that packet drop at 80% disrupts the connection to the server very quickly [32], in a worst-case scenario as shown in Figure 12. Similarly, in a worst-case scenario for packet duplication, 40% of the packets sent over the network are duplicated 20 times each. This causes the user to be disconnected from the VRLE very quickly. On the other hand, packet tampering causes the VRLE crash for all users accessing the server. Using packet tampering in a man-in-the-middle attack scenario can reveal confidential information as discussed in [32]. From our experiments, we observe that a tamper rate of 20% would be good enough to crash the VRLE server as shown in Figure 12.

Figure 11: A before and after scenario showcasing the effect of DoS attack on vSocial causing a server crash

c) Miscellaneous SP attacks: Based on the work discussed in [26], 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 is concerned 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 [26]. 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. 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.

Using the observations (level of impact, resources required) through our SP attack simulations, we develop the attacker profiles shown in Table 3 for every SP attack simulated as part of SP-UIX tree validation. In this study, we adapt the definition of attacker profile from several existing works [53], [54] 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 attack profiles contribute primarily to the calculation of the suspiciousness score discussed in Section 5.4. 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 and using the attacker profiles presented in Table 3 and in SP-UIX attack tree, we illustrate the evaluation of our detection approach employed in the vSocial application.

Table 3: Attacker profiling table with accompanying levels 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

Table 4: Example of machine learning data structure

SampleID	Class	Time	Value
1	Normal	1	290
1	Normal	2	295
41	Dropping	1	4
41	Dropping	2	8
81	Duplication	1	1820
81	Duplication	2	1940
121	Tampering	1	410
121	Tampering	2	1260

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 [32]. 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 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. Next, 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.

5.2.1 ML on a single label network-based attack dataset

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 approach aims to develop a pertinent machine learning classifier. We generate training and testing datasets by sampling different types of network-based attacks such as DoS and also the causes of DoS attack (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` [55] and `scikit-multilearn` [56] are used for implementing the classifier models.

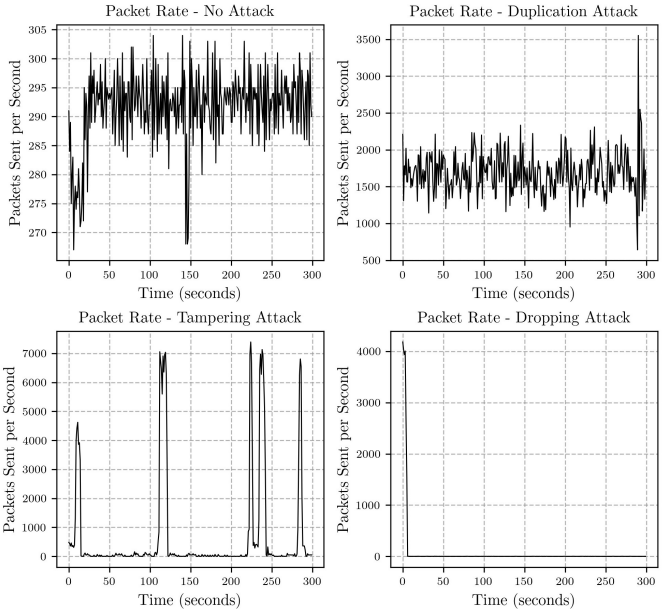


Figure 12: Packet rate time series comparison for single attacks

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. A sample of the data set structure we used is shown in Table 4.

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) as shown in Table 4. For each class, we collect 40 samples of 15-second sequences, where each sequence contains a packet rate at 1-second intervals. 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 12. 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.

Feature Extraction: As part of feature extraction, we use `tsfresh` [57], 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 13. We use these distinguishing five

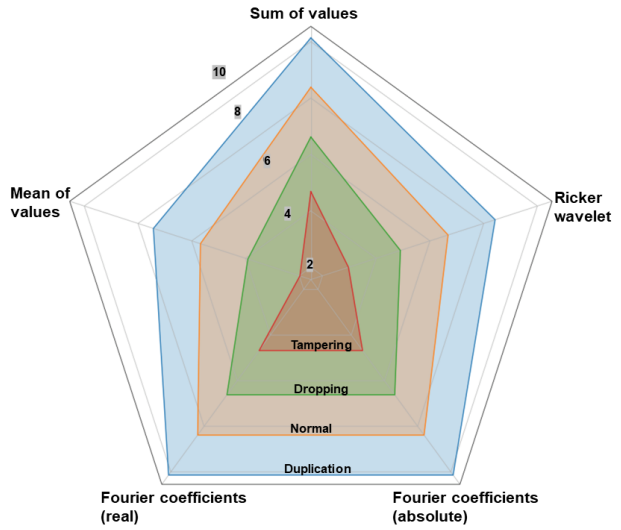


Figure 13: Five of the most distinguishing extracted features

features – *Sum of Values*, *Mean of Values*, *Ricker Wavelet*, *Fourier Coefficient (real)*, and *Fourier Coefficient (absolute value)* for training a ML classifier.

Before we implement a ML classifier, we perform several tests based on a few quantifiable metrics to suit for our data. We repeat the feature extraction step on multiple machine learning classifiers to aid in developing an algorithm for an accurate detection approach. We implement these classifiers by adapting from the `sklearn` python machine learning library [58].

Based on the common approach in prior works, we define different machine learning classifiers considered in our case for the best-fit analysis as follows:

1) **Decision Tree**: A Decision Tree consists of internal nodes and terminal nodes, or leaves. Starting with the root node, each node represents a test condition that, given a sample, determines what next node or leaf to move to next [55].
2) **Random Forests**: Random Forests is an ensemble learning method for classification that involves a collection of Decision Trees. A multitude of Decision Trees are randomly constructed with the use of bootstrap aggregating, also known as bagging [55].
3) **K Nearest Neighbors**: K Nearest Neighbors (KNN) takes a group of X-dimensional labeled training data and lays it in X-dimensional space. Then, when given a new unlabeled sample, the algorithm lays the sample in the space and uses the K-nearest points to determine what to classify the sample as [55].
4) **Ensemble Voting Classifier**: The Voting Classifier is an ensemble model that simply enlists the results of two or more machine learning models. In our case, we use both the KNN and the Random Forests to construct our Voting Classifier [55].

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 evaluation performance metrics that can be used for a classification exercise. However, in this paper, the performance

Table 5: Single-label classifier accuracy statistics

ML Model	Average Speed (ms)	Average Precision	Average Recall
Decision Tree	1.52	95.2%	94.8%
K Nearest Neighbors	3.97	97.8%	97.6%
Random Forests	3.38	97.4%	97.2%
KNN + Random Forests	6.33	97.5%	97.3%

Table 6: Time between start of attack and detection of attack

Type of Attack	Time to Detect
Dropping	4.4 seconds
Duplication	2.4 seconds
Tampering	2.8 seconds

metrics such as *average time*, *precision* and *recall* are computed to decide the suitable classifier for our data collected that can classify a sample as an attack or benign data class. We calculate the considered performance metrics as follows:

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

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

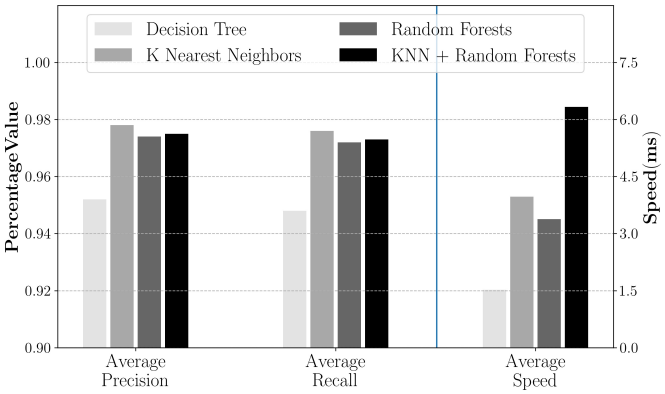


Figure 14: Single-label classifier accuracy comparison

Table 5 presents the *Precision* and *Recall* values of the considered ML classifiers. Based on our analysis shown in Table 5, 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 (ms) 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. 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 are outlined in a graphical manner as shown in Figure 14.

Based on the graphical analysis shown in Figure 14, we use the most suited classifier *KNN* and use it for attack detection and classification for single labeled data. We want to determine the most suitable classifier, that detects attack faster along with maintaining high accuracy in data classification considering the real-time impact on the user experience in a VRLE. For this, we use average speed defined as the amount of time needed for the classifier on a single sample as shown in Table 5. We determine how long our model takes to recognize that an attack has started, given that it would be used to monitor the VRLE as a real-time intrusion detection system. 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 6. 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. With one of our preliminary works [32], we also 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 results, it is evident that our anomaly detection approach works effectively and can avoid any VRLE server crash before the user gets interrupted or causes a significant impact on VRLE sessions.

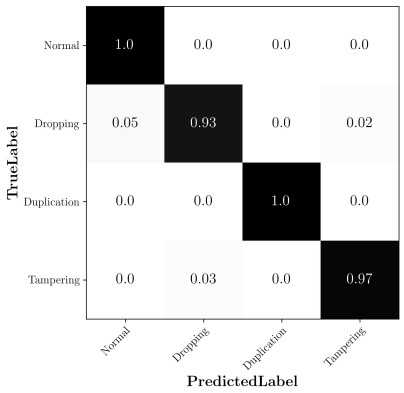


Figure 15: Single-label KNN classifier confusion matrix

Figure 15 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.

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

Based on the results shown in Figures 14 and 15, we justify the potential of employing a suitable classifier in our proposed anomaly detection approach to identify any network-based anomaly events from a sample of time series based packet data. However, considering the dynamic interactions in 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 concurrently.

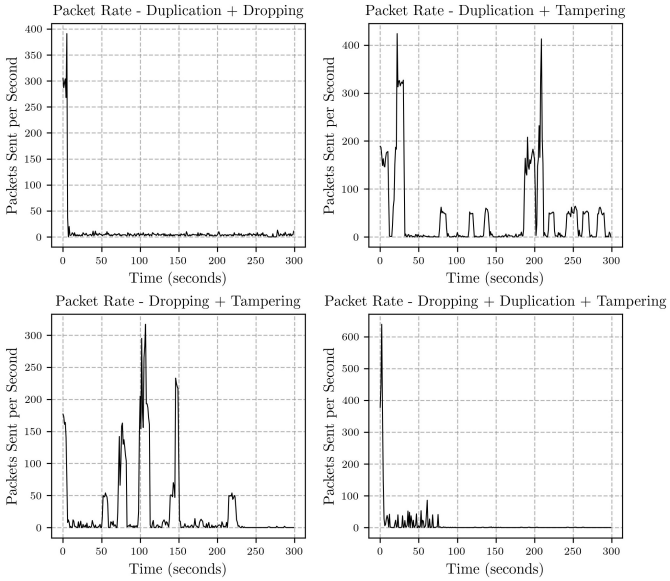


Figure 16: Combinational attack time series comparison

In order to address such multi-attack scenario, our anomaly detection approach adapts multi-label KNN classification based on the existing works [59], [60], [61] using a multi-label KNN classifier. The single-label KNN classifier in our anomaly detection approach uses adaptation methods (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 data collection process in single-label classifier. 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 as shown in Figure 16. By observing the time series data, we proceed to develop our multi-label KNN classifier by adapting from the works in [56], [59]–[61]. Our developed multi-label KNN classifier is able to detect the combination of network 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 [36], 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

Table 7: Time between start of attack and detection of attack

Type of Attack	Time to Detect
Dropping	4.0 seconds
Duplication	2.0 seconds
Tampering	2.8 seconds
Dropping + Duplication	2.6 seconds
Dropping + Tampering	3.6 seconds
Duplication + Tampering	1.2 seconds
Dropping + Duplication + Tampering	3.6 seconds

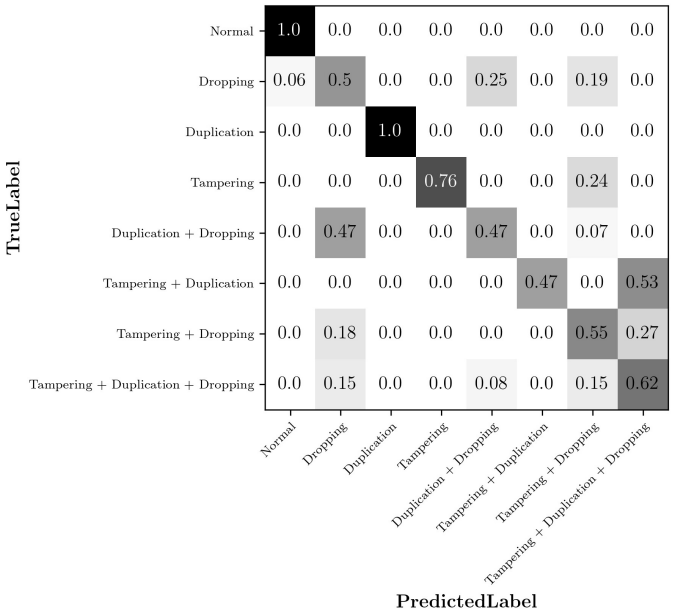


Figure 17: Multi-label K-NN classifier confusion matrix

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 powerset 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 7. We generate a confusion matrix for the multi-labeled classifier as shown in Figure 17, where it showcases the correct classifications and the incorrectly labeled predictions. 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 approach 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 (precision, recall, time taken to detect and classify) as shown in Table 8.

5.3 Application-based attack detection using statistical analysis

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 approach 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 4.1 implements the boolean conditions to flag malicious events and thresholding equations to perform Z-score analysis.

We illustrate our experimental evaluation of our statistical technique for the attacks listed in Table 9. To elucidate, the thresholds are calculated for each application-based attack considered in this study as shown in Table 9. 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. If the input log data is determined as anomaly, 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.

5.3.1 Data collection for Z-score based statistical analysis

In order to test the effectiveness of our Algorithm 1, our anomaly detection approach 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 [62], and we 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

Table 8: Time taken and accuracy statistics for our employed ML model to detect and classify an attack

Type of model	*Stage1: Detection*		*Stage2: Classification*	
	Average time taken to detect network attacks by the ML model (seconds)	**Average accuracy to detect network attacks by the ML model**	**Average time taken to classify into network attacks by the ML model (milli seconds)**	**Average accuracy taken to classify into network attacks by the ML model**
Single attack scenario (single labeled ML model)	3.2 seconds	98.8 %	7 milli seconds	97.5%
Multi attack scenario (Multi labeled ML model)	2.82 seconds	99.5%	1.3 milli seconds	87.5%

Table 9: List of statistical analysis system functions

Function	Description
Booleans	
SQLInjection (input)	Scans inputs for illegal characters that could cause an SQL Injection
TFTP()	Checks for open ports and new files sent from IPs over TFTP, a sample vulnerability for unauthorized access
nonDefaultFile()	Parses system files for a non-default file
recentlyModified()	Checks if files have been modified since the start of the virtual reality session
checkUsernameIP()	Checks for a username being logged in from an unfamiliar IP
Thresholds	
bruteforce()	Uses Z-scores to calculate when a number of login attempts exceeds the normal threshold
bruteforceUsername()	Uses Z-scores to calculate when a number of login attempts for a specific username exceeds the normal threshold

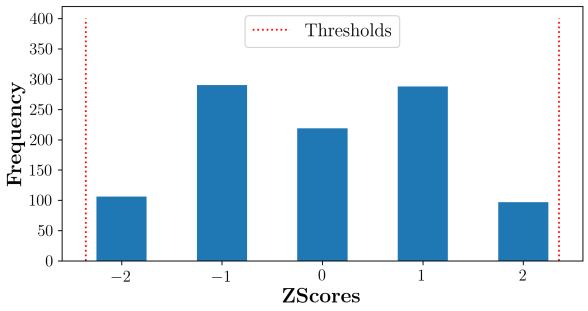


Figure 18: Normal password attempt Z-score distribution that shows benign behavior range in vSocial data.

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 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 based statistical analysis

The data collected in terms of normal (benign) user attempts and malicious (attacker) attempts are sent to Algorithm 1 listed in Section 4.1 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

Table 10: 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%

Figure 19: Brute-force password attempt Z-score distribution

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 9, we generate histograms to show the distribution of Z-score frequency for normal (benign) user data on login attempts as shown in Figure 18 and for malicious user (unauthorized access) login attempt data as shown in Figure 19.

The Z-score data in benign user histogram shown in Figure 18, 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 10 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 algorithm. For simplicity, we run tests related to an attacker database of 100 passwords instead of the 4958 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 10 to understand the efficiency of our Z-score based algorithms.

Table 11: 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

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 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 Figures 18 and 19 represents the frequency of the data and the x-axis represents the Z-score value ranges. The thresholds for the normal data (benign user) shown in Figure 19 are in the form of the dotted line on the extreme left which is adapted from Figure 18. The malicious attack data is detected and flagged as the Z-score distribution falls out of the threshold range which is annotated as malicious data in the bar graph shown in the Figure 19. Thus, using statistical analysis, we identify the application-based attacks (i.e., unauthorized access) as shown in Figure 19 based on comparison of the data threshold shown in Figure 18.

5.4 Calculation of Suspiciousness Scores (SS_{attack})

Based on the approach shown in Figure 9 that is described in Section 4.1, our proposed anomaly detection module calculates the suspiciousness score SS_{attack} for a specific detected attack via our proposed anomaly detection methods. 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' that is detailed in Table 3. 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 attacker profile as shown in Table 3 and these can be extended for future works to determine parameters that can be quantified for the application-based attacks.

Our anomaly detection module calculates these SS_{attack} based on Equation 4 which is mentioned in the Algorithm 1. Our anomaly detection approach 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. 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 4, where we calculate the estimated level of impact of each attack on the UIX as follows:

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

The SS_{attack} gets updated by the above formulated estimated level of impact as shown in Table 11 using the Equations 4 and 5. Our proposed anomaly detection will generate the normalized SS_{attack} scores so that they can be used for triggering an alert. These SS_{attack} scores can be used for defense mechanisms when an attack is detected.

In this study, we remark that we focus only on developing an anomaly detection approach through SS_{attack} score calculation, which is used to analyze the relationship between the UIX score (aggregated numerical value of UIX factors considered) and SP attacks.

5.5 Creation of a Knowledge Base

We collect all the different attack signatures and benign data (VRLE session information) along with the calculated SS_{attack} into a database that serves as a Knowledge Base. We generated this Knowledge Base by collecting the packet data during the simulation of different DoS attack scenarios (packet tampering, packet duplication, and packet dropping) and different log information related to access controls during application-based attacks (different forms of unauthorized access). This Knowledge Base can be adapted to other models and it is effective especially when a user who was active in previous sessions tries to perform an attack, our Knowledge Base can help in detecting whether it was from the user who tried to login previously. However, the full capability of our Knowledge Base can be extended from other applications and utilize for employing defense mechanisms that are part of our future work for VRLE systems.

6 Validation of SP attacks Impact on UIX

With the results obtained from the techniques in our proposed anomaly detection approach, we identify that the machine learning method is able to accurately detect when a network-based attack occurs, and shows the potential in determining which attack(s) are occurring concurrently. On the other hand, our Z-score analysis technique is able to detect application-based attacks with high accuracy. After the detection of SP attacks and calculation of SS_{attack} using our proposed anomaly detection approach, the next step is to validate the impact of the detected attacks on the UIX score. The subjective assessment in Section 3 details about the relationship between different SP attacks and each of the UIX factors. In this section, we provide a validation of the impact of detected SP attacks that triggers the SS_{attack} and their impact on the overall UIX score. Herein, the SS_{attack} and overall UIX score are considered as the metrics which can be used for the impact analysis.

The analysis between the calculated SS_{attack} and UIX score for different types of scenarios (attack and benign user) is shown in Figure 20. The x-axis in Figure 20 denotes the breaches occurred where, NB – *No breach*, SB – *Security breach*, PB – *Privacy breach* and CB – *Combination of breaches* in VRLE application case study. The y-axis in Figure 20,

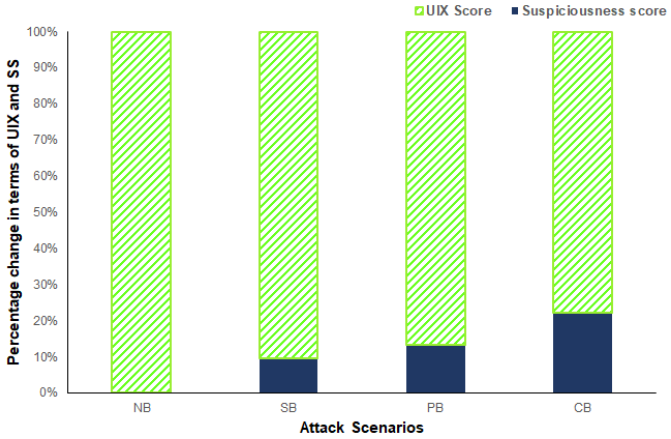


Figure 20: Relative change in UIX as the Suspiciousness scores increase for different type of SP breaches

represents the percentage change value that occurs in both UIX and SS_{attack} parameters. We observe that for every security, privacy breach scenario in 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 approach for detecting attacks before they occur in real-world VRLE systems.

7 Conclusion

VRLEs are IoT-based environments that deliver important distance learning content in a collaborative manner with immersive experience to geographically distributed users (i.e., students and instructor(s)). Unauthorized access and Denial of Service attacks to the VRLE components or data can cause potential security and privacy attacks which can affect the functionality of VRLEs and impacts user immersive experience (UIX). 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 this established relationship, we proposed a novel anomaly detection framework approach that effectively identifies an attack event or a combination of attack events, and correspondingly classifies the events into a specific attack category recorded in a knowledge base.

We formalized all the SP attacks into a new attack tree termed as *SP-UIX*. Using this attack tree formalism, we detailed the vulnerabilities that VRLEs possess and, more importantly, the consequences of each SP attack type on the UIX factors. Our proposed novel anomaly detection framework approach involved two major techniques: (i) machine learning KNN classifiers for identifying e.g., DoS attacks, and (ii) Z-score analysis for detecting e.g., unauthorized access. Using a real-world VRLE application case study viz., vSocial for special education, we successfully detected single network attack scenarios (a form of DoS 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 such as more than one form of DoS attack within 2.82 seconds, and classified the attack categories with an accuracy of 87.5%. Additionally, our anomaly detection module detected unauthorized access via a Z-score based statistical analysis with an average *precision* of 99.7% and generated an attack-specific 'Suspiciousness Score' metric normalized on a scale of 0-1 for both network-based and application-based attacks. We utilized different attacker profiles generated from our experiments in our extensive attack simulations on a vSocial configuration to demonstrate our proposed anomaly detection approach efficacy in terms of accuracy, precision and recall. In the same context, we created a knowledge base that consists of all the detected attack patterns along with the calculated Suspiciousness Scores and UIX scores. Lastly, we explained the ability of extending our anomaly detection techniques using the created knowledge base and the identified quantitative parameters (from impact analysis) in order to detect any zero-day attack events on VRLE systems.

As part of future work, our two-step ensemble machine learning technique can be replaced with pertinent deep learning methods, including ones that employ e.g., 'adaptive boosting' to more accurately detect sophisticated network-based attacks. Additional efforts can be put in addressing the cybersickness [63]–[65] issues that can eventually impact the UIX in application scenarios that are impacted by available bandwidth bottlenecks. Further, our findings can be used to inform the best practices for incorporating suitable anomaly detection techniques in future VRLE designs.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Award Numbers: CNS-1647213 and CNS-1659134. 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] N. Joshi. (2017) Impact of iot on augmented and virtual reality. Last Accessed 2019-09-30. [Online]. Available: https://www.allerin.com/blog/impact-of-iot-on-augmented-and-virtual-reality

[2] 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," *2014 IEEE International Conference on Internet of Things(iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom)*, pp. 456–459, 2014.

[3] A. V. Aalto, "How virtual reality meets the industrial iot," 2015.

[4] D. You, B.-S. Seo, E. Jeong, and D. H. Kim, "Internet of things (iot) for seamless virtual reality space: Challenges and perspectives," *IEEE Access*, vol. 6, pp. 40 439–40 449, 2018.

[5] J. Hu, W. Wan, R. xiang Wang, and X. Yu, "Virtual reality platform for smart city based on sensor network and osg engine," *2012 International Conference on Audio, Language and Image Processing*, pp. 1167–1171, 2012.

[6] H. Prendinger, K. Gajananan, A. B. Zaki, A. Fares, R. Molenaar, D. Urbano, J. W. C. van Lint, and W. Gomaa, "Tokyo virtual living lab: Designing smart cities based on the 3d internet," *IEEE Internet Computing*, vol. 17, pp. 30–38, 2013.

[7] F. Rabbi, T. Park, B. Fang, M. Zhang, and Y. Lee, "When virtual reality meets internet of things in the gym: Enabling immersive interactive machine exercises," *IMWUT*, vol. 2, pp. 78:1–78:21, 2018.

[8] 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.

[9] 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.

[10] 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.

[11] D. M. Mendez, I. Papapanagiotou, and B. Yang, "Internet of things: Survey on security and privacy," *ArXiv*, vol. abs/1707.01879, 2017.

[12] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, pp. 2787–2805, 2010.

[13] B. Schneier. (1999) Attack trees. [Online]. Available: http://www.drdobbs.com/attack-trees/184411129

[14] H. Myrbakken and R. C. Palacios, "Devsecops: A multivocal literature review," in *SPICE*, 2017.

[15] K. Carter, "Francois raynaud on devsecops," *IEEE Software*, vol. 34, pp. 93–96, 2017.

[16] (2019) clumsy0.2. Last accessed 2019-09-30. [Online]. Available: https://jagt.github.io/clumsy/download.html

[17] (2019) Beef. Last accessed 2019-09-30. [Online]. Available: https://beefproject.com/

[18] (2019) Wireshark. Last accessed 2019-09-30. [Online]. Available: https://www.wireshark.org/

[19] 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

[20] 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.

[21] A. Tiiro, "Effect of visual realism on cybersickness in virtual reality," *University of Oulu*, 2018.

[22] 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.

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

[24] G. Regal, R. Schatz, J. Schrammel, and S. Suette, "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.

[25] 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.

[26] P. Casey, I. Baggili, and A. Yarramreddy, "Immersive virtual reality attacks and the human joystick," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[27] (2019) Steamvr. Last accessed 2019-09-30. [Online]. Available: http://store.steampowered.com/steamvr

[28] "Securing your reality: Addressing security and privacy in virtual and augmented reality applications." [Online]. Available: https://tinyurl.com/yxlplqqe

[29] 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.

[30] 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.

[31] (2017) Virtual reality headsets could put childrens health at risk. Last accessed 2019-09-30. [Online]. Available: https://www.theguardian.com/technology/2017/oct/28/virtual-reality-headset-children-cognitive-problems

[32] A. Gulhane, A. Vyas, R. Mitra, R. Oruche, G. Hoefer, 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.

[33] C. Westphal, "Challenges in networking to support augmented reality and virtual reality," *IEEE ICNC*, 2017.

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

[35] A. Joshi and V. Geetha, "Sql injection detection using machine learning," in *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*. IEEE, 2014, pp. 1111–1115.

[36] 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.

[37] A. Osareh and B. Shadgar, "Intrusion detection in computer networks based on machine learning algorithms," *International Journal of Computer Science and Network Security*, vol. 8, no. 11, pp. 15–23, 2008.

[38] B. G. Witmer and M. J. Singer, "Measuring presence in virtual environments: A presence questionnaire," *Presence*, vol. 7, no. 3, pp. 225–240, 1998.

[39] (2019) Scipy. Last accessed 2019-09-30. [Online]. Available: https://www.scipy.org/

[40] E. J. Byres, M. Franz, and D. Miller, "The use of attack trees in assessing vulnerabilities in scada systems," in *Proceedings of the international infrastructure survivability workshop*. Citeseer, 2004, pp. 3–10.

[41] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer, "Foundations of attack–defense trees," in *International Workshop on Formal Aspects in Security and Trust*. Springer, 2010, pp. 80–95.

[42] M. Rege and R. Mbah, "Machine learning for cyber defense and attack," in *DATA ANALYTICS 2018 : The Seventh International Conference on Data Analytics*, 2018.

[43] 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.

[44] J. Zhao, S. Shetty, J. W. Pan, C. Kamhoua, and K. Kwiat, "Transfer learning for detecting unknown network attacks," *EURASIP Journal on Information Security*, vol. 2019, no. 1, p. 1, Feb 2019. [Online]. Available: https://doi.org/10.1186/s13635-019-0084-4

[45] (2019) Z-score: Definition. Last accessed 2019-09-30. [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 Transactions on Network and Service Management*, vol. 13, no. 3, pp. 666–680, 2016.

[47] 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.

[48] 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

[49] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: a network programming language," in *ICFP*, 2011.

[50] (2019) Kali linux. Last accessed 2019-09-30. [Online]. Available: https://www.kali.org/

[51] (2019) Metasploit. Last accessed 2019-09-30. [Online]. Available: https://www.metasploit.com/

[52] (2019) Solarwinds tftp server. Last accessed 2019-09-30. [Online]. Available: https://www.solarwinds.com/free-tools/free-tftp-server/

[53] M. Rocchetto and N. O. Tippenhauer, "On attacker models and profiles for cyber-physical systems," in *ESORICS*, 2016.

[54] Attacker classification to aid targeting critical systems for threat modelling and security review. Last accessed 2019-09-30. [Online]. Available: http://www.rockyh.net/papers/AttackerClassification.pdf

[55] (2019) Scikit-learn. Last accessed 2019-09-30. [Online]. Available: https://scikit-learn.org/stable/

[56] (2019) Scikit-multilearn. Last accessed 2019-09-30. [Online]. Available: http://scikit.ml/

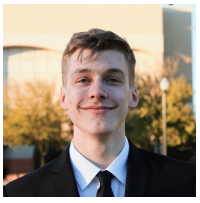
[57] (2019) Ts-fresh. Last accessed 2019-09-30. [Online]. Available: https://tsfresh.readthedocs.io/en/latest/

[58] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

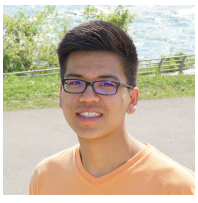
[59] M.-L. Zhang and Z. W. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," *2005 IEEE International Conference on Granular Computing*, vol. 2, pp. 718–721 Vol. 2, 2005.
[60] (2019) Multilabel k nearest neighbours. Last accessed 2019-09-30. [Online]. Available: http://scikit.ml/api/skmultilearn.adapt.mlknn.html
[61] (2018) Deep dive into multi-label classification. Last accessed 2019-09-30. [Online]. Available: https://towardsdatascience.com/journey-to-the-center-of-multi-label-classification-384c40229bff
[62] (2019) Bruteforce-database. Last accessed 2019-09-30. [Online]. Available: https://github.com/duyetdev/bruteforce-database/
[63] J. J. LaViola Jr, "A discussion of cybersickness in virtual environments," *ACM Sigchi Bulletin*, vol. 32, no. 1, pp. 47–56, 2000.
[64] S. Davis, K. Nesbitt, and E. Nalivaiko, "A systematic review of cybersickness," in *Proceedings of the 2014 Conference on Interactive Entertainment*. ACM, 2014, pp. 1–9.
[65] L. Rebenitsch and C. Owen, "Review on cybersickness in applications and visual displays," *Virtual Reality*, vol. 20, no. 2, pp. 101–125, 2016.

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 pursuing her Ph.D. degree in Computer Science at University of Missouri-Columbia. Her current research interests include Cloud Computing, Cloud Security for AR/VR and IoT applications, Big Data Analytics.

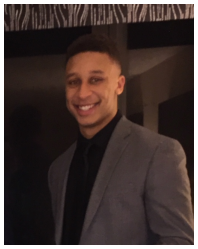
Benjamin Frailey is currently pursuing his BS in Computer Science and a minor in Mathematics at University of Missouri-Columbia. His research interests include Cyber Security, Cloud Computing, Machine Learning, Internet of Things, and Computer Networking.



Brady Kruse is currently pursuing his BS in Computer Science with emphasis in cyber security and a minor in Mathematics and English at Mississippi State University. His research interests include Quantum Computing, Internet of Things, Cyber Security, and Science Fiction.



Boonakij Palipatana is currently pursuing his BA in Information Science with a concentration in Data Science at Cornell University. His research interests include Machine Learning, Natural Language Processing, and Computational Social Science.



Roland Oruche received his BS in Information Technology from the University of Missouri-Columbia. He is currently pursuing his Ph.D. in Computer Science at the University of Missouri-Columbia. His research interests include Machine Learning, Cloud Computing, Virtual Reality, and Computer Vision.

Aniket Gulhane received his BE in Computer Engineering from Savitribai Phule Pune University, India in 2016. He is currently pursing his MS degree in Computer Science at the University of Missouri-Columbia. His research interests include Cloud Security, Human Computer Interaction, Cloud Computing.

Khaza Anuarul Hoque received his MS and PhD degrees from the Department of Electrical and Computer Engineering at Concordia University, Montreal, Canada in 2011 and 2016, respectively. He is currently an Assistant Professor in the Department of Electrical Engineering and Computer Science at University of Missouri-Columbia where he directs the Dependable Cyber-Physical Systems (DCPS) Laboratory. His research interests include: Formal Methods, Cyber-physical Systems, and Cyber Security. He is a Member of IEEE and ACM.

Prasad Calyam received his MS and PhD degrees from the Department of Electrical and Computer Engineering at The Ohio State University in 2002 and 2007, respectively. He is currently an Assistant Professor in the Department of Computer Science at University of Missouri-Columbia. Previously, he was a Research Director at the Ohio Supercomputer Center. His research interests include: Distributed and Cloud computing, Computer Networking, and Cyber Security. He is a Senior Member of IEEE.