Attack Trees for Security and Privacy in Social Virtual Reality Learning Environments

Samaikya Valluripally[a] Aniket Gulhane[a], Reshmi Mitra[b], Khaza Anuarul Hoque[a], Prasad Calyam[a]
University of Missouri-Columbia[a], Webster University [b],
{svbqb, argqm8}@mail.missouri.edu, {calyamp, hoquek}@missouri.edu, reshmimitra25@webster.edu

***Abstract*—Social Virtual Reality Learning Environment (VRLE) is a novel edge computing platform for collaboration amongst distributed users. Given that VRLEs are used for critical applications (e.g., special education, public safety training), it is important to ensure security and privacy issues. In this paper, we present a novel attack tree formalism and model checking techniques in order to obtain quantitative assessments of threats and vulnerabilities. Based on the use cases from an actual social VRLE viz., vSocial, we describe security and privacy attack trees that when converted into a stochastic timed automata allows for rigorous statistical model checking. Such an analysis helps us adopt pertinent design principles such as *hardening*, *diversity* and *principle of least privilege* to enhance the resilience of social VRLEs. Through experiments in a vSocial case study, we demonstrate the effectiveness of our attack tree modeling with a reduction of 26% in probability of loss of integrity (security) and 80% in privacy leakage (privacy) in before and after scenarios pertaining to the adoption of the design principles.**

***Index Terms*—Virtual reality, Special education, Security, Privacy, Attack trees, Formal verification**

I. Introduction

Social Virtual Reality (VR) is a new paradigm of collaboration systems that uses edge computing for novel application areas involving virtual reality learning environments (VRLE) for special education, surgical training, and flight simulators. Typical VR system applications comprise of interactions that require coordination of diverse user actions from multiple Internet-of-Things (IoT) devices, processing activity data and projecting visualization to achieve cooperative tasks. However, this flexibility necessitates seamless interactions with IoT devices, geographically distributed users outside the system's safe boundary, which poses serious threats to security and privacy [1].

Although existing works [2]–[4] highlight the importance of security and privacy issues in VR applications, there are a limited systematic efforts in evaluating the effect of various threat scenarios on such edge computing based collaborative systems with IoT devices. Specifically, VRLE applications are highly susceptible to Distributed Denial of Service (DDoS) attacks, due to the distributed IoT devices (i.e.,VR headsets) connecting to virtual classrooms through custom controlled collaboration settings. Moreover, loss of confidential information is possible as VRLEs track student engagement and other realtime session metadata.

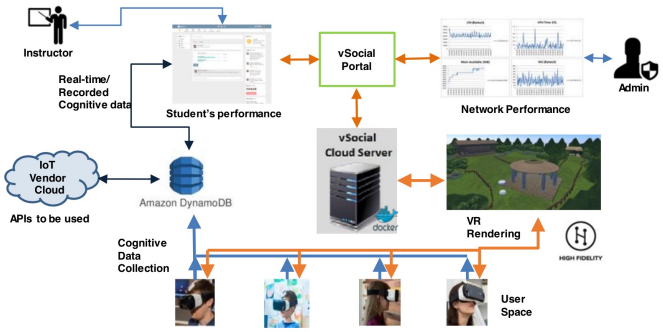

Fig. 1: vSocial system components used for real-time student learning session management.

In this paper, we consider a VRLE application designed for youth with autism spectrum disorder (ASD) as case study viz., vSocial [5][1]. Our multi-modal VRLE system as shown in Figure 1 uses the High Fidelity platform [6], and renders 3D visualizations based on the dynamic human computer interactions with an edge cloud i.e., vSocial Cloud Server. VRLE setup includes: VR headset devices (HTC Vive), handheld controllers, and base stations for accurate localization and tracking of controllers [5]. Any disruption caused by an attacker with malicious intent on the instructor's VR content or administrator privileges will impact user activities and even cause cybersickness. Failure to address these security and privacy issues may result in alteration of instructional content, compromise of learning outcomes, access privileges leading to confidential student information disclosure and/or poor student engagement in ongoing classroom sessions.

Motivated by the importance of ensuring security and privacy in the VRLE application, we propose a novel framework for quantitative evaluation of security and privacy metrics using attack trees to perform Statistical Model Checking (SMC) [7]. In order to check whether the design has met the requirements and compute the security goals, we employ the state of the art SMC techniques. These SMC methods can be used to formally verify the VRLE system and user

This material is based upon work supported by the National Science Foundation under Award Number CNS-1647213. 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.

[1]Moving forward we use the acronym VRLE to sometimes interchangeably refer to our case study application viz., vSocial.

behaviors [8]. Thus, we model our VRLE application case study as stochastic system behavior patterns such that the dynamic interactions in real time VRLE systems can be analyzed.

We use the attack trees concept from [9] and derive graphical models that provide a systematic representation of various attack scenarios. Although attack trees are popular, they lack support for modeling the temporal dependencies between the attack tree components. To overcome this limitation, we utilize an automated state-of-the-art SMC tool UPPAAL [7]. Our approach overall involves translating the constructed security and privacy attack tree of the VRLE application into the Stochastic Timed Automata (STA) in a compositional manner. For the purposes of this paper, we define: (a) *security* – as a condition that ensures a VR system to perform critical functions with the establishment of confidentiality, integrity, and availability [10], and (b) *privacy* – as a property that regulates the IoT data collection, protection, and secrecy in interactive systems [10].

The main paper contributions summary is as follows:
– We propose a framework to evaluate security and privacy of VR applications using the attack tree formalism and statistical model checking. To show the effectiveness of our solution approach, we utilize a VRLE application case study viz., vSocial that uses edge computing assisted IoT devices for students and instructor(s) collaboration.
– We perform a trade-off analysis by evaluating the severity of different types of attacks on the considered VRLE application. From our results, we observe that: i) unauthorized access and causes of DoS attack (in security attack tree), ii) track user movement and user physical location (in privacy attack tree) are the most vulnerable candidates in a VRLE.
– We demonstrate the effectiveness of using design principles (also known as security principles) i.e., *hardening, diversity, principle of least privilege* on the privacy and security of VRLE applications in the event of most severe threats. We show that in terms of security – a combination of {*hardening, principle of least privilege*} is most influential in reducing the probability of Loss of Integrity (LoI). Similarly for privacy – a combination of {*diversity, principle of least privilege*} is most influential in reducing privacy leakage.

The remainder of the paper is organized as follows: Section II discusses related works. Section III introduces the necessary background and terminology. Section IV discusses the proposed security and privacy framework in detail. Section V presents the numerical results using our proposed framework on the VRLE case study. Section VI discusses the effectiveness of design principles on the security and privacy threat scenarios. Section VII concludes the paper.

II. Related Works

There have been several comprehensive studies that highlight the importance of security and privacy threats on IoT and related paradigms such as Augmented Reality (AR) with IoT devices, and edge computing. A recent study [1] on challenges in AR and VR discusses the threat vectors for educational initiatives without characterizing the attack impact. Survey articles [2]–[4], [11]–[13] are significant for understanding the concepts of threat taxonomy and attack surface area of IoT and fog computing. They highlight the need to go beyond specific components such as network, hardware or application, and propose end-to-end solutions that consider system and data vulnerabilities. An observation given the above state-of-art is that there are very few scholarly works on the quantitative evaluation for these security and privacy threats in the context of VR applications.

We borrow the attack trees concept that is used commonly in cyber-physical systems involving SCADA system [14], and adapt it for threat modeling to determine the probability of detection and severity of threats. One of our preliminary works [15] showed risk assessment of security, privacy and safety metrics of the VRLE applications utilizing an attack tree simulation tool. In contrast, this work focus is on formal modeling of attack trees using STAs and utilizes a state-of-the-art formal verification tool to evaluate the developed security and privacy attack trees. In addition, our proposed framework incorporates the concept of design principles to enhance the security and privacy of VRLE applications.

Amongst the numerous prior works on attack trees, the work in [16] presents a novel concept of Attack Fault Tree (AFT), a combination of both attack and fault trees. A model of STA [17] alleviates some assumptions made in timed automata and provides advantages such as choice of transitions requiring satisfaction of very precise clock constraints. Timed automata [18] provides an abstract model of the real system by using clocks as well as timing constraints on the transition edges. As compared to Continuous-Time Markov Chains (CTMC) [19], STA models allow us to express hard time constraints such as $x \leq t$. We studied the above existing modeling techniques to formalize our security and privacy attack trees into STA, which we evaluate using a model checking tool viz., the UPPAAL SMC [20].

III. Background and Terminology

A. Attacks in VRLE application use case:

The users in a social VRLE are networked and geographically distributed, which creates a series of potential attack scenarios. Using vSocial shown in Figure 1 as a social VRLE case study, herein we demonstrate exemplar security and privacy attacks that can affect the VRLE application sessions. **Security Attacks:** An attacker can gain unauthorized access to either tamper any confidential information (user, network, VRLE components) by impersonating as a valid user, or disclose compromised confidential information. To elucidate, the instructional content in a vSocial application is in a web-enabled presentation format using the features present in High Fidelity. To guide the students through activities in the vSocial learning environment, the instructor will have privileged access to control the learning content settings such as e.g., editing the slides, and rewarding the students based on their performance. Gaining *Unauthorized access* to the instructor account as shown in Figure 2 can lead to *disclosure of user information*,

Fig. 2: Unauthorized access to VRLE learning content.

and tampering of the learning content in vSocial to negatively impact the users' (students') learning experience.

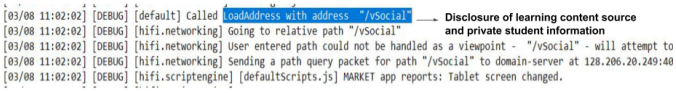

Fig. 3: Privacy attack on vSocial application.

Privacy Attacks: A user privacy breach can involve an intruder entering a VRLE world with fake credentials to snoop into the virtual classroom conversations. The attacker can then disrupt an ongoing VRLE session by obstructing the view of the users in their learning sessions and can even disorient the content. Disorientation can possibly lead to a user running into a wall and getting physically hurt. Privacy attacks can also involve *packet tampering* that was demonstrated in [15], where an attacker performs illegal packet capture in order to extract sensitive information (*packet sniffing attack*). A potential privacy breach can occur when the attacker *discloses the confidential information* obtained from the captured packets as shown in Figure 3. Using this packet sniffing attack, the avatar (user virtual information) can also be disclosed with private location information and student credentials.

B. Statistical model checking

Statistical model checking (SMC) is a variation of the well-known classical model checking [21] approach for a system that exhibits stochastic behavior. The SMC approach to solve the model checking problem involves simulating (Monte Carlo simulation) the system for finitely many runs, and using hypothesis testing to infer whether the samples provide a statistical evidence for the satisfaction or violation of the specification [22].

Stochastic timed automata: Stochastic timed automata (STA) is an extended version of timed automata (TA) with stochastic semantics. A STA associates logical locations with continuous, generally distributed sojourn times [18]. In STA, constraints on edges and invariants on locations, such as clocks are used to enable transition from one state to another [16].

Definition 1 (Stochastic timed automata)**.** *Given a timed automata which is equipped with assignment of invariants $\mathcal{I}$ to locations $\mathcal{L}$, we formulate an STA as a tuple $T = \langle \mathcal{L}, l_{init}, \Sigma, \mathcal{X}, \mathcal{E}, \mathcal{I}, \mu \rangle$, where $\mathcal{L}$ is a finite set of locations, $l_{init} \in \mathcal{L}$ is the initial location, Σ is a finite set of actions, $\mathcal{X}$ is the finite set of clocks, $\mathcal{E} \subseteq \mathcal{L} \times \mathcal{L}_{clk} \times \Sigma \times 2^{\mathcal{X}}$ is a finite set of edges, with $\mathcal{L}_{clk}$ representing the set of clock constraints, $\mathcal{I}$: $\mathcal{L} \longrightarrow \lambda$ is the invariant where λ is the rate of exponential assigned to the locations $\mathcal{L}$, μ is the probability density function (μ_l) at a location $l \in \mathcal{L}$.*

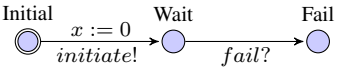

Fig. 4: An exemplar STA.

An exemplar STA is shown in Figure 4 that consists of the locations {Initial, Wait, Fail}. Herein, the $Initial$ location represents the start of execution of an STA and a *clock* x is used to keep track of the global time. The communication in an STA exists between its components using message broadcast signals in a bottom-up approach. The STA is activated by broadcasting *initiate!* signal, which transitions to wait location and waits for the *fail* signal. In an STA, time delays are governed as probability distributions (used as invariants) over the locations. The Network of Stochastic Timed Automata (NSTA) is defined by composing all component automaton to obtain a complete stochastic system satisfying the general compositionality criterion of TA transition rules [7], [18].

UPPAAL SMC: UPPAAL SMC is an integrated tool for modeling, validation, and verification of real-time systems modeled as a network of stochastic timed automata (NSTA) extended with integer variable, invariant, and channel synchronizations [20]. In SMC, the probability estimate is derived using an estimation algorithm and statistical parameters, such as $1-\alpha$ (required confidence interval) and ϵ (error bound) [23]. For instance, if we indicate goal state in the STA of Top_event as $Fail$, then the probability of a successful occurrence within time t can be written as: $Pr[x <= t](<> Top\_event.Fail)$ where, $<>$ represents the existential operator ($\Diamond$) and x is a clock in the STA to track the global time.

C. Design principles

To build a trustworthy VRLE system architecture which ensures security and privacy, integration of design principles in the life cycle of edge computing interconnected and distributed IoT device based systems is essential [24]. We adapt the following three design principles from NIST SP800-160 [10], [24] such as: (i) *Hardening* – defined as reinforcement of individual or types of components to ensure that they are harder to compromise or impair, (ii) *Diversity* – defined as the implementation of a feature with diverse types of components to restrict the threat impact from proliferating further into the system, and (iii) *Principle of least privilege* – defined as limiting the privileges of any entity, that is just enough to perform its functions and prevents the effect of threat from propagating beyond the affected component.

IV. Proposed Framework

In this section, we present details of our proposed framework that uses preliminary results from the previous section (Section III) for security and privacy analysis (i.e., to identify the most vulnerable attacks) of social VRLE applications. An overview of the steps followed in our framework is shown in Figure 5. Firstly, we outline the threat scenarios in a social VRLE application [5] using traditional approaches.

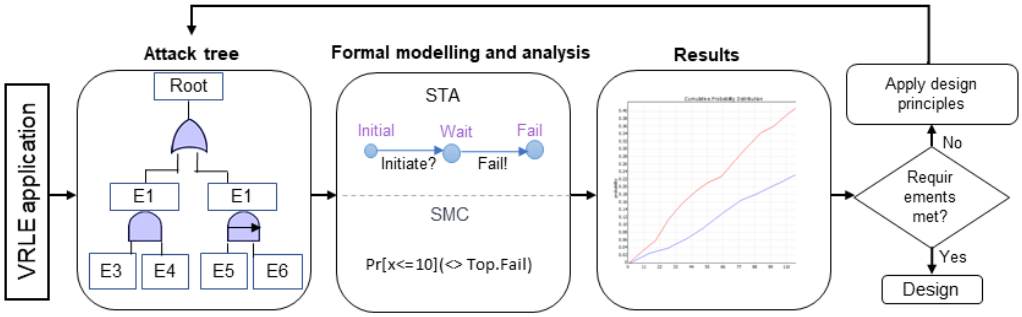

Fig. 5: Proposed framework for security and privacy analysis of a social VRLE.

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

A. Formalization of security and privacy attack trees

Attack trees are hierarchical models that show how an attacker goal (root node) can be refined into smaller sub-goals (child/intermediate nodes) via gates until no further refinement is possible such that the basic attack steps (BAS) are reached. BAS represents the *leaf nodes* of an attack tree [25]. The *leaf nodes* and the *gates* connected in the attack trees are termed as *attack tree elements*. To explore dependencies in attack surfaces, attack trees enable sharing of subtrees. Hence, attack trees are often considered as directed acyclic graphs, rather than trees [16].

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

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

Attack tree gates: Given an attack tree A, we formally define the attack tree gates $G = \{OR, AND, SAND\}$.[2] An AND gate is disrupted when all its child nodes are disrupted, whereas an OR gate is disrupted if either of its child nodes are disrupted. Similarly, SAND gate is disrupted when all its child nodes are disrupted from left to right using the condition that the success of a previous step determines the success of the upcoming child node. The output nodes of the gates using these gates G in an attack tree A are defined as *Intermediate nodes* (I), which will be located at a level that is greater than the leaf nodes.

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

Security and privacy attack trees: Based on the results discussed in Section III and experimental evidence from our prior work [15], we model threat scenarios in the form of a security attack tree (that lists potential VRLE security threats) and privacy attack tree (that lists potential VRLE privacy threats) as shown in Figures 6 and 7, respectively. The descriptions of the leaf nodes are listed in Table I. Exploring the security aspect in CIA triad of {Confidentiality, Integrity, Availability} may result into an enormous number of leaf nodes in the attack tree. Consequently, in this work we only focus on the Loss of Integrity (LoI) and privacy leakage to address the respective security and privacy threat scenarios that can disrupt the user experience in a social VRLE. Creation of new security and privacy trees for issues related to LoC and LoA can be performed similar to the approach presented for LoI; such details are beyond the scope of this paper. Moreover, the listed security and privacy implications in the attack trees are critical especially when it is concerned with user in VRLE. To elucidate, VRLE applications like flight simulations, military training exercises and vSocial [developed for children with ASD] such attacks and their consequences are important because of these sensitive applications target audience information cannot be exposed.

[2]We limit our modeling to these three gates, however attack trees can adopt any other gates from the static/dynamic fault trees.

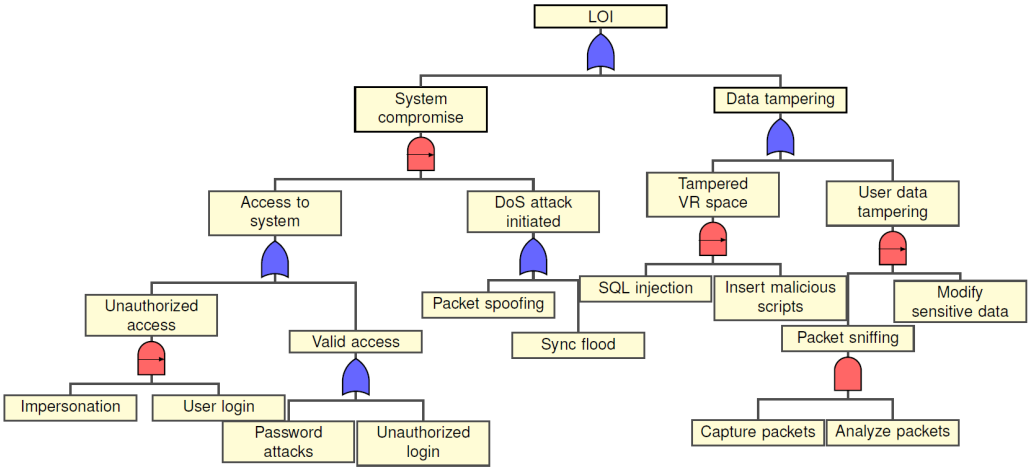

Fig. 6: Formalized security attack tree with threat scenarios disrupting LoI.

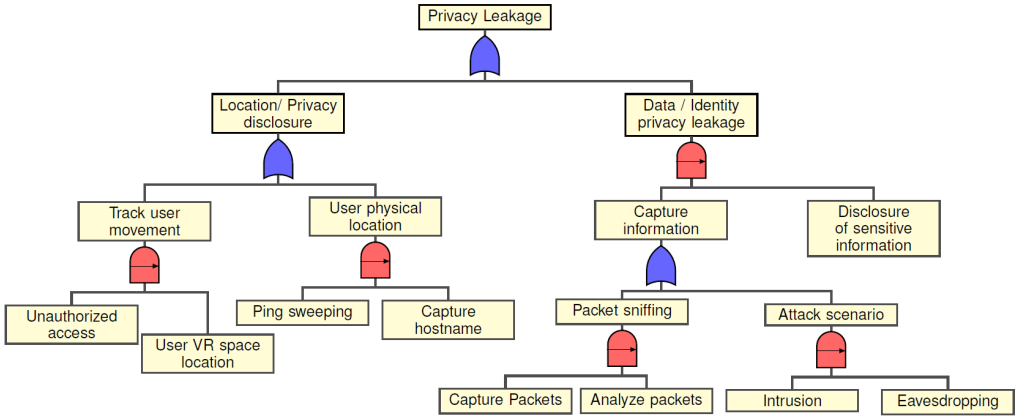

Fig. 7: Formalized privacy attack tree with threat scenarios disrupting privacy leakage.

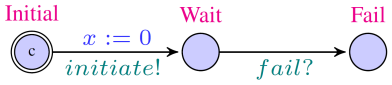

Fig. 8: STA for root node.

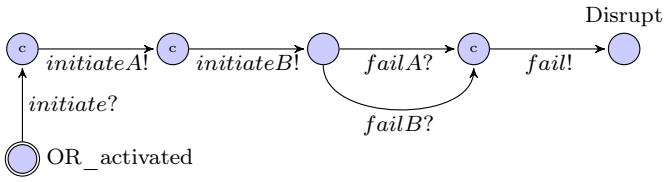

Fig. 9: STA for OR gate and root node of security attack tree.

B. Translation of attack trees into stochastic timed automata

In this section, we generate STA from the corresponding security and privacy attack trees shown in Figures 6 and 7. In our translational approach: (i) each of the leaf nodes in these attack trees is converted into an individual STA. The intermediate events, which are basically the output of the logic gates used at different levels are converted imperatively into STA; (ii) the generated STAs are composed in parallel by including the root node; (iii) the obtained NSTA is then used for statistical model checking in order to verify the security and privacy properties formalized as SMC queries.

To demonstrate the translation of an attack tree into an STA, we consider the security attack tree as shown in Figure 6. As part of the translation, each of the security AT element (leaf and gates) input signals are connected to the output signal of child nodes. The generated network of STA communicates using signals. *initiate* - indicates activation signal of attack tree element. This signal is sent initially from the root node to its children. *fail* - indicates disruption of that attack tree element. This signal is sent to the parent node from its child node to indicate an STA disruption. The scope of the above signals can also be extended by special symbols such as: i)'?' (e.g., initiate?) means that the event will wait for the reception of the intended signal, ii) '!'(e.g., initiate!) implies output signal broadcasts to other STA in the attack tree.

Illustrative example: For instance, we show the conversion of LoI i.e., root node (Top_event) into STA as shown in Figure 8. The converted STA of the LoI is equipped with $initiate!$ and $fail?$ signals. The root node is the OR gate output for the two child nodes: (A) "System Compromise" and (B) "Data Tampering". Top OR gate sends an initiate signal and activates its child nodes "System compromise" and "Data Tampering" as shown in Figure 9 by broadcasting $initiateA!$

TABLE I: Descriptions of leaf nodes in security and privacy attack trees.

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

TABLE II: λ values for leaf nodes of security & privacy ATs.

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

and $initiateB$! signals. After initialization, if either of the nodes (A) OR (B) are disrupted, then a $fail$! signal is sent to the Top_event, which forces a transition to $Disrupt$ state, representing LoI in the system. The clock x is a UPPAAL global variable to keep track of the time progression as mentioned in Section III. Similarly, STAs for the AND gate, SAND gates and the leaf nodes are also developed. Moreover, STAs for leaf nodes such as $impersonation$ in the security attack tree are instantiated with λ (rate of exponential) values. For the given λ values to the leaf nodes, the probability of occurrence is calculated. This value then propagates upward in the tree to calculate the probability of LoI. As mentioned earlier, the developed STAs are composed using the parallel composition [18] technique to form an NSTA, which is then used for SMC by the UPPAAL tool [7].

V. Quantitative Results

In this section, we present the results obtained using our proposed framework. As mentioned in Section IV, the threat scenarios we consider are: *LoI* and *privacy leakage* for security and privacy attack trees (AT), respectively. In the following analysis, we assume that our design requirement is to keep the probability of LoI and privacy leakage below the threshold of 0.25. For evaluation purposes, we use arbitrary values of λ as parametric input to the leaf nodes as shown in Table II obtained from [26], [27]. Note, after providing λ values as parameters to the leaf nodes, we utilize the SMC queries as explained in Section III-B to find the respective probabilities

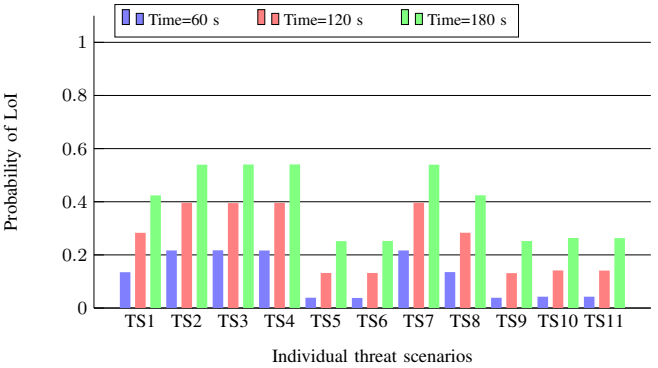

Fig. 10: TS of security AT where - $TS2$, $TS3$, $TS4$, $TS7$ are the most vulnerable nodes.

of LoI and privacy leakage. Any other user specified threshold values can also be used in our framework. This is due to the fact that the model checking approach takes the user specified values at the beginning of an experiment. For our experiment purposes, we consider LoI (security attack tree) and privacy leakage (privacy attack tree) as goal nodes. In the following set of experiments, we present the obtained probability of the goal nodes with respect to the time window used by the attacker.

A. Vulnerability analysis in the security AT

We assign the values of λ shown in Table II. However, when assigning a λ value to a leaf node in the attack tree, we consider a very small positive constant $(K) \approx 0.002$ for the remaining leaf nodes. This is because, in real time systems, multiple attack scenarios can happen. To identify a vulnerability in a security attack tree, we analyze: (i) individual leaf nodes, and (ii) combinations of leaf nodes, to determine their effect on the probability of LoI occurence.

i) Individual leaf node analysis: In Figure 10, we show the probability of LoI over multiple time windows for each leaf node in the security attack tree. We perform a thorough analysis of leaf nodes in the security attack tree for threat scenarios across different time intervals i.e., *t = {0, 60, 120, 180}*. For the individual leaf node analysis, the considered threat scenarios (TS) shown in Figure 10 are termed as: $TS1$

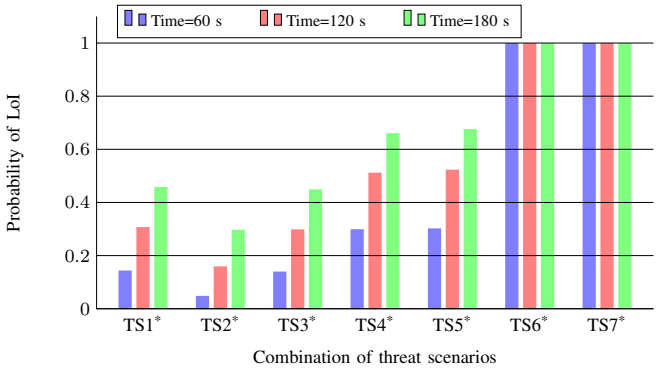

Fig. 11: TS of security AT where – $TS6^*$, $TS7^*$ are the most vulnerable combination.

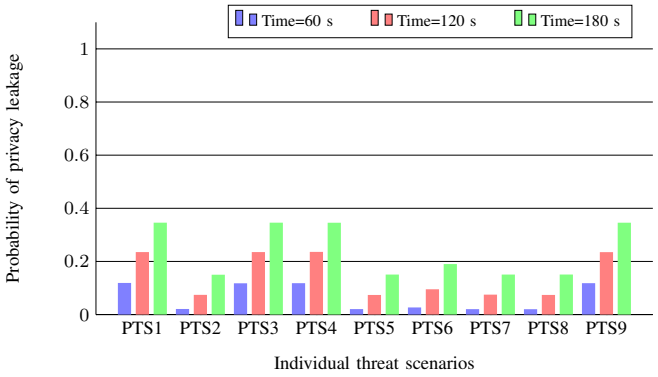

Fig. 12: TS of privacy AT where - $PTS1$, $PTS3$, $PTS4$, $PTS9$ are the most vulnerable nodes.

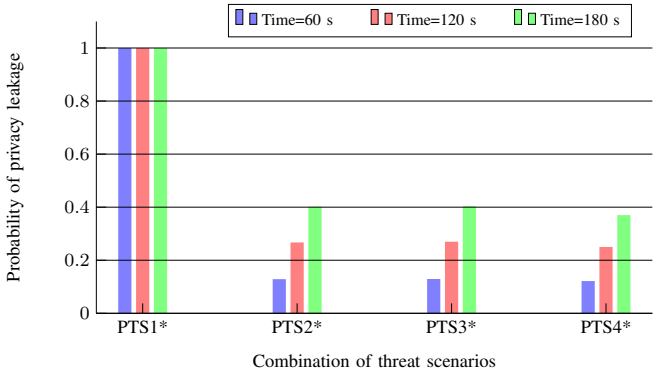

Fig. 13: TS of privacy AT where – $PTS1*$ is the most vulnerable combination.

– insert malicious scripts, $TS2$ – packet spoofing, $TS3$ – unauthorized login, $TS4$ – password attacks, $TS5$ – modify data, $TS6$ – analyze packets, $TS7$ – Sync flood, $TS8$ – SQL injection, $TS9$ – capture packets, $TS10$ – impersonation, $TS11$ – user login. As shown in Figure 10, the leaf nodes $TS3$ and $TS4$ (for unauthorized access) as well as $TS2$ and $TS7$ (for DoS attack) are the most vulnerable in the security attack tree with the probability of 0.53.

ii) Analysis using combination of leaf nodes: Herein, we consider combinations of leaf nodes to identify their impact on LoI. For these experiments, we explore two scenarios: In the first scenario, we consider combinations of leaf nodes that belong to the same sub-tree, and in the second scenario, we consider leaf nodes from different sub-trees. The considered combination of threat scenarios are enlisted as: $TS1^*$ – {impersonation, SQL injection}, $TS2^*$ – {impersonation, modify data}, $TS3^*$ – {SQL injection, capture packets}, $TS4^*$ – {pwd attacks, SQL injection}, $TS5^*$ – {impersonation, packet spoofing}, $TS6^*$ – {packet spoofing, unauthorized login}, $TS7^*$ – {unauthorized login, Sync flood}. As shown in Figure 11, $TS6^*$ and $TS7^*$ are the most vulnerable combination of threat scenarios with a probability of 1 for an LoI event. As part of further analysis in Section VI, we discuss about the potential candidates for design principles to apply on these leaf nodes such that the VRLE application resilience against security threats is enhanced.

B. Vulnerability analysis in the privacy AT

We analyze the privacy attack tree similarly for: (i) individual leaf nodes, and (ii) combinations of leaf nodes. For the considered individual leaf node analysis in the privacy attack tree, the threat scenarios are termed as: $PTS1$ – unauthorized access, $PTS2$ – capture packets, $PTS3$ – user VR space location, $PTS4$ – ping sweeping, $PTS5$ – analyze packets, $PTS6$ – disclosure of sensitive information, $PTS7$ – intrusion, $PTS8$ – eavesdropping, $PTS9$ – capture hostname. As shown in Figure 12, the most vulnerable leaf nodes are: $PTS1$, $PTS3$, $PTS4$, $PTS9$ with the highest probability of privacy leakage of 0.34.

For the analysis of combination of leaf nodes, we refer to the combination of threat scenarios as: $PTS1*$ – {unauthorized access, user VR space location}, $PTS2*$– {capture packets, disclosure of sensitive information}, $PTS3*$ – {unauthorized access, disclosure of sensitive information}, $PTS4*$ – {capture packets, analyze packets}. As shown in Figure 13, $PTS1*$ is the most vulnerable combination of threats for privacy leakage with a probability of 1. In summary, we can conclude that the above numerical analysis shown in Table III on both security and privacy attack trees can help in identifying the LoI and privacy leakage concerns that need to be addressed in the social VRLE design.

VI. Recommended Design Principles

In this section, we are examining the effect of applying various design principles to the most vulnerable components identified in the Sections V-A, and V-B. Existing works such as NIST SP800-160 document [10], [24] suggest that the services for safeguarding security and privacy are critical for successful operation of current devices and sensors connected to physical networks as part of IoT systems. As mentioned in Section III-C, these design principles are essential to construct a trustworthy edge computing based system architecture. The goal is to apply a combination of design principles at different levels of abstraction to help in developing effective mitigation strategies. We adopt a selection of design principles such as *hardening*, *diversity* and *principle of least privilege* among the list of principles available in NIST document [10], and [24]. In

TABLE III: Most vulnerable components considering the individual & combination of leaf nodes.

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

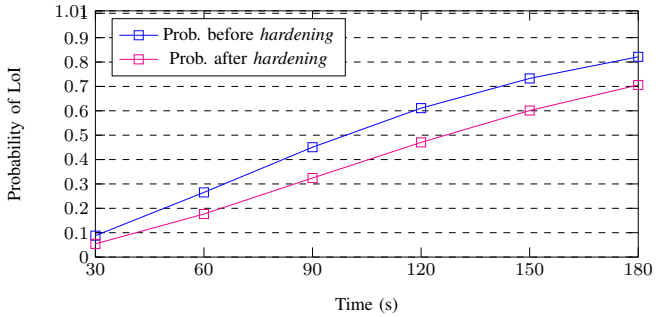

Fig. 14: Prob. in LoI reduced by 15.85% in security AT due to application of design principles.

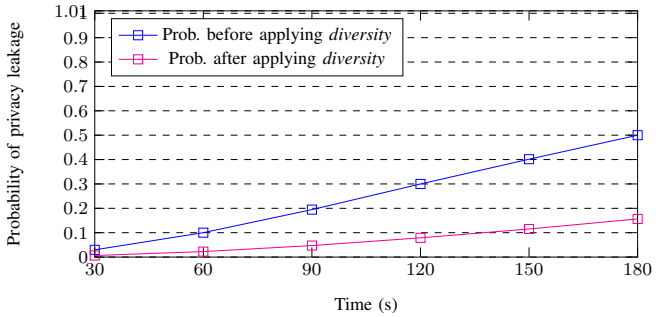

Fig. 15: Prob. of privacy leakage reduced by 68% in privacy AT due to application of design principles.

the following, we demonstrate their effectiveness by showing that there is a reduction in the probability of disruption terms after adopting them in our VRLE system design.

Implementation of design principles on security attack tree: In this section, we apply design principles on one of the identified vulnerable nodes of the security attack tree as shown in Section V-A. For instance, we incorporate *hardening* design principle on the password attacks, to study its effects on the security metric LoI as shown in Figure 14. As part of the *hardening* principle, we added new nodes such as a firewall and a security protocol in the security attack tree. Our results show that the probability of disruption of LoI is reduced from 0.82 to 0.69 (15.85%), with the given attacker profile. The decrease in the disruption of LoI is due to the rise in additional resources that are required by the attacker to compromise such a VRLE application system which is incorporating the *hardening* principle. Similarly, we apply the *principle of least privilege* on the security attack tree, which under-provisions privileges intentionally. This in turn reduced the probability of disruption of LoI from 0.82 to 0.79 (3.66%).

Implementation of design principles on privacy attack tree: Using the similar approach mentioned in design principles on the security attack tree, we apply *diversity* design principle on one of the identified vulnerable nodes (unauthorized access) in the privacy attack tree. After adding multi-factor authentication procedures as part of the *diversity* principle, the probability of disruption on privacy leakage is reduced significantly from 0.5 to 0.16 (68%) as shown in Figure 15. Similarly, we apply the *principle of least privilege* by under-provisioning

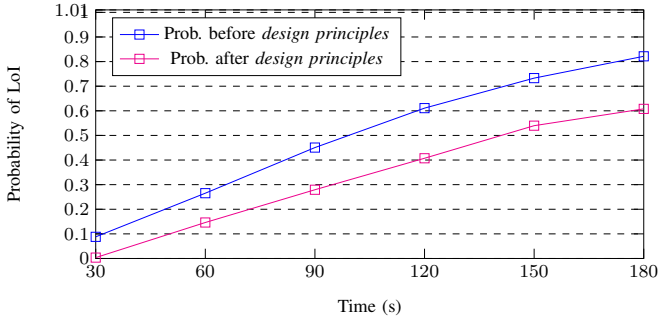

Fig. 16: Prob. of LoI is reduced by 26% in security AT due to application of design principles for a combination of security attack tree nodes.

privileges on the privacy attack tree where the probability of disruption of privacy leakage is slightly reduced from 0.5 to 0.48 (4%). Thus, from the above implementation of individual design principles, we conclude that *hardening* and *diversity* are more effective in reducing the disruption of LoI and privacy leakage, respectively. Thus, our findings shows that some security principles are more effective than others. In addition, our results emphasize the benefits in implementing a combination of design principles in both security and privacy attack trees to overall improve the attack mitigation efforts.

To study the effect on disruption of the LoI and privacy leakage, we adopt a combination of design principles such as: (i) for the security attack tree: {*hardening*, *principle of least privilege*}, and (ii) for the privacy attack tree: {*diversity*, *principle of least privilege*}. We observe that there is a

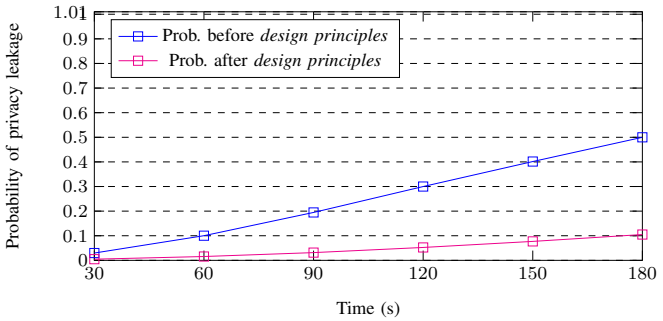

Fig. 17: Prob. of privacy leakage reduced by 80% in privacy AT due to application of design principles for a combination of privacy attack tree nodes.

significant drop in the probability of disruption of LoI from 0.81 to 0.6 (26%), and 0.5 to 0.1 (80%) for privacy leakage as shown in Figures 16 and 17, respectively.

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

VII. Conclusion and Future Works

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

In future, we plan to explore the effect of fault and attacks as a combination using the attack-fault tree formalism [16] for VRLE applications. This will allow us to reason about the safety metrics and study the safety, security and privacy trade-offs. Since, different components in a typical social VRLE application go through different maintenance actions, we also plan to explore the impact of various maintenance strategies on the reliability metric of social VRLE applications using the fault maintenance tree formalism [28].

References

[1] B. Fineman, N. Lewis, "Securing Your Reality: Addressing Security and Privacy in Virtual and Augmented Reality Applications", *EDUCAUSE Review*, 2019.

[2] W. Zhou, Y. Jia, A. Peng, Y. Zhang, P. Liu, "The Effect of IoT New Features on Security and Privacy: New Threats, Existing Solutions, and Challenges Yet to Be Solved", *IEEE Internet of Things Journal*, 2018.

[3] K. Fu, T. Kohno, D. Lopresti, E. Mynatt, K. Nahrstedt, S. Patel, D. Richardson, B. Zorn, "Safety, Security, and Privacy Threats Posed by Accelerating Trends in the Internet of Things", *Computing Community Technical Report*, 2017.

[4] P. Casey, I. Baggili, A. Yarramreddy, "Immersive Virtual Reality Attacks and the Human Joystick", *IEEE Transactions on Dependable and Secure Computing*, 2019.

[5] C. Zizza, A. Starr, D. Hudson, S. S. Nuguri, P. Calyam and Z. He, "Towards a Social Virtual Reality Learning Environment in High Fidelity", *IEEE Consumer Communications & Networking Conf. (CCNC)*, 2018.

[6] High Fidelity, 2019. [Online]. Available: https://highfidelity.com.

[7] A. David, K. G. Larsen, A. Legay, M. MikuËĞcionis, and D. B. Poulsen, "Uppaal SMC Tutorial", *Int. J. on Software Tools for Tech. Transfer*,2015

[8] S. Alireza, R. A. Masoud, N.N. Jafari, R. Reza, "A symbolic model checking approach in formal verification of distributed systems", *Human-centric Computing and Information Sciences*, 2019.

[9] S. Bruce, "Attack trees", *Dr.Dobb's journal*, 24.12, 21-29, 2019.[Online]. Available: http://www.drdobbs.com/attack-trees/184411129

[10] "Systems Security Engineering:Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems", 2019. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-160.pdf

[11] R. Roman, J. Lopez, M. Mambo, "Mobile Edge Computing: A Survey and Analysis of Security Threats and Challenges", *Elsevier Future Gen. Computer Systems*, 2016.

[12] S.Yi, Z.Qin, Q.Li, "Security and Privacy Issues of Fog Computing: A Survey", *Intl. Conf. Wireless Algorithms, Systems, Applications*, 2015.

[13] M. A. Khan, K. Salah, "IoT security: Review, Blockchain Solutions, and Open Challenges", *Elsevier Future Gen. Computer Systems*, 2018.

[14] E. Byres, M. Franz, D. Miller, "The Use of Attack Trees in Assessing Vulnerabilities in SCADA Systems", *IEEE Conf. Int.Infrastructure Survivability Workshop*, 2004.

[15] A. Gulhane, A. Vyas, R. Mitra, R. Oruche, G. Hoefer, S. Valluripally, P. Calyam, K. A. Hoque, "Security, Privacy and Safety Risk Assessment for Virtual Reality Learning Environment Applications", *IEEE Consumer Communications & Networking Conference (CCNC)*, 2019.

[16] R. Kumar, M. Stoelinga, "Quantitative Security and Safety Analysis with Attack-Fault Trees", *IEEE 18th Int. Symposium on HASE*, 2017.

[17] N. Bertrand, P. Bouyer, T. Brihaye, Q. Menet, C. Baier, M. Grosser, M. Jurdzinski, "Stochastic Timed Automata", *Logical Methods in Comp. Sci.*, 2014.

[18] P. Ballarini, N. Bertrand, A. Horvath, "Transient Analysis of Networks of Stochastic Timed Automata Using Stochastic state classes", *Int. Conference on Quantitative Evaluation of Systems*, 2013.

[19] A. Aziz, K. Sanwal, V. Singhal, R. Brayton, "Model-checking Continuous-time Markov chains", *ACM Trans. on Comptl. Logic*, 2000.

[20] D. Alexandre, K. Larsen, A. Legay, M.Mikučionis ,D. Poulsen, "Uppaal SMC Tutorial", *Int. J. on Software Tools for Technology Transfer*, 2015.

[21] E. M. Calrk jr, O. Grumberg, D. Peleg, "Model checking", *MIT Press*, 2000.

[22] H.Younes, M. Kwiatkowska, G. Norman, D. Parker, "Numerical vs Statistical Probabilistic Model Checking", *Int. J. on Software Tools for Technology Transfer*, 2006.

[23] P. Bulychev, A. David, K.G. Larsen, A. Legay, G. Li, D. B. Poulsen, “Rewrite-based Statistical Model Checking of wmtl”, *Int. Conference on Runtime Verification*, 2012.
[24] A. Laszka, W. Abbas, Y. Vorobeychik, X. Koutsoukos,“Synergistic Security for the Industrial Internet of Things: Integrating Redundancy, Diversity, Hardening", *IEEE ICII*, 2018.
[25] G. Norman, D. Parker, J. Sproston, “Model checking for probabilistic timed automata”, *Formal Methods in System Design*, 2013.
[26] P. Saripalli, B. Walters, “QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security”, *IEEE Cloud Computing*, 2010.
[27] M. Kiani, A. Clark, and G. Mohay, “Evaluation of anomaly based character distribution models in the detection of SQL injection attacks”, *Int. Conference on Availability, Reliability and Security*, pp. 47-55, 2008.
[28] N. Cauchi, K. A. Hoque, A. Abate, M. Stoelinga, “Efficient Probabilistic Model Checking of Smart Building Maintenance using Fault Maintenance Trees", *Proc. of ACM Int. Conf. on Systems for Energy-Efficient Built Environments*, 2017.