# SPARQ: a QoS-Aware framework for mitigating cyber risk in self-protecting IoT systems

Alessandro Palma, Houssam Hajj, Georgios Bouloukakis

# SPARQ: A QoS-Aware Framework for Mitigating Cyber Risk in Self-Protecting IoT Systems

Alessandro Palma*, Houssam Hajj Hassan†, Georgios Bouloukakis†
*Sapienza University of Rome, Department of Computer, Control, and Management Engineering (DIAG), 00185, Italy.
†Samovar, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France.
*palma@diag.uniroma1.it*
{*houssam.hajj_hassan, georgios.bouloukakis*}*@telecom-sudparis.eu*

***Abstract*—Today's smart spaces deploy various IoT devices to offer services for occupants. Such devices are exposed to security risks that may pose serious threats to network services and users' privacy. To avoid the disruption of normal operations, self-protecting solutions have been developed to allow IoT networks to autonomously respond to cyber threats in real-time. However, existing self-protecting systems focus solely on architectural adaptations to respond to cyber threats, overlooking the mitigation actions described in cybersecurity standards –which represent the correct cybersecurity posture– as well as the impact of the adaptation strategies on the Quality-of-Service (QoS) performance. To overcome these existing limitations, this paper presents SPARQ, a novel framework for designing self-protecting IoT systems that considers both the security exposure to cyber attacks and the QoS performance. We leverage Attack Graph as a threat model for analyzing the cyber exposure of the system and Queuing Network Models to analyze QoS in IoT systems. Based on the analysis outcomes, SPARQ provides mitigation plans to reduce the cyber risk while also minimizing the impact on QoS. We evaluate the proposed approach on two use cases from real-world scenarios including a critical infrastructure and a smart home. The experimental evaluation shows that SPARQ is capable of reducing the cyber risk significantly while also improving the QoS performance by 35% compared to existing approaches.**

***Index Terms*—Self-protection, Attack Graph, Quality of Service, Cyber Risk.**

## I. Introduction

Addressing security threats and safeguarding organizations from cyber attacks is becoming increasingly challenging due to the ever-changing vulnerability landscape and the growing sophistication of attackers. This is further aggravated by the need for modern systems to be continuously active in providing digital services in both critical and non-critical infrastructures. Among these systems, the Internet of Things (IoT) is becoming pervasive in providing services in interconnected ecosystems of devices, such as those in smart homes and cities, healthcare, energy, and utility systems. IoT networks consist of devices that are interconnected with applications to offer services to users. They are normally configured to serve operational functions of specific application domains, bringing researchers and practitioners to focus highly on the design of operational services in these networks [1]. This often implies less attention to the presence of vulnerabilities due to misconfiguration, bugs in the source code, or resource limitations [2]. An attacker can exploit such vulnerabilities to intrude and compromise networks [3]. This is particularly relevant in IoT networks, where devices must generate and exchange data continuously. Many new vulnerabilities are discovered every day and cannot be immediately patched due to a multitude of reasons, such as the lack of knowledge to fix them, cost factors, and organizational preferences placing availability and usability over security [4]. This leads to the exposure of these networks to cyber attacks.

To support continuous monitoring and improvement of the cybersecurity posture of IoT devices, self-protecting systems have garnered significant attention as they enable devices to autonomously adapt to dynamic environments, offering responsiveness, agility, and cost-effectiveness [5], [6], [7], [8]. Self-protection refers to the capability of a system to autonomously detect, assess, and respond to threats or changes in its environment to maintain its security and functionality without external intervention [9].

Designing self-protecting IoT systems is a challenging task, mainly because security mitigation typically requires a discontinuation of the services: this involves a human expert manually patching vulnerabilities (e.g., removing bugs in the source code) [10]. To this aim, there exist approaches in the literature proposing self-protecting systems that autonomously execute modifications of the network infrastructure to avoid human intervention [11], [12], [13], [14]. The majority of these approaches rely on *architectural adaptation*, which consists of taking actions that change the architecture of the IoT network to avoid the exploitation of vulnerabilities. Such actions may be excluding an IoT device from the network (e.g., by turning it OFF), or removing some communication links.

While these approaches are a valuable solution for autonomous cybersecurity, there are cases where relying on architectural adaptations should be integrated with security vulnerability patching. For example, in critical infrastructures such as healthcare networks, shutdown time and service interruption should be avoided as much as possible due to the criticality of the data and operations (e.g., patients' requests) that must always be retrievable [15]. Self-protecting solutions relying on architectural adaptation usually overlook the impacts that adaptations have on Quality-of-Service (QoS). On the other hand, replacing such architectural adaptations with vulnerability patching might result in prolonged waiting times due to the need for human intervention, which could compromise the autonomous capabilities of self-protecting

systems [16]. This is mainly due to the difficulty of defining methodologies to collect and consider heterogeneously QoS metrics (e.g., latency) and security metrics (e.g., cyber risks) and integrating this information for comprehensive mitigation plans.

To address these challenges, this paper presents SPARQ, a Self-Protecting framework for Autonomous Response with QoS-awareness for mitigating cyber risks in IoT networks. To realize SPARQ, we introduce the concept of *security adaptations*, which are actions that can be employed in IoT devices to patch specific vulnerabilities without changing the network configuration (e.g., installing software updates), contrary to architectural adaptations. We propose a taxonomy to consider security and architectural adaptations comprehensively. Consequently, we leverage Attack Graph (AG), a graph-based threat model, to evaluate the security effects of security and architectural adaptations in the IoT network, and Queuing Network Models to simulate such adaptations to assess the QoS effects. This enables the collection of security and QoS metrics that are used as input to Automated Planning (AI Planning) approaches to automatically provide mitigation plans through heuristics. SPARQ has been extensively evaluated on two scenarios from real IoT networks, including smart home and critical healthcare infrastructure, showing its benefits in reducing cyber risks while also considering the QoS. In summary, this paper contributes:

- the integration of a security model, based on Attack Graphs, and QoS model, based on Queuing Networks to analyze the security and QoS effects of adaptations in IoT networks (§IV-B1,§IV-B2);
- Automated Planning domain models including security and QoS metrics that enable an AI planner to automatically provide mitigation plans (§IV-C);
- a working prototype of SPARQ[1] and its experimental evaluation on two real scenarios (§V).

Additionally, we report related work (§II), a motivating example with the overview of the approach (§III) and finally conclude this article (§VI).

## II. Related work

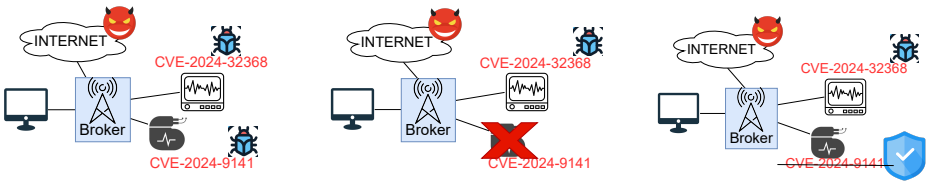
The expanding attack surfaces of IoT networks increased the attention to their self-protection to autonomously mitigate cyber risks [17], [18], [19]. Different works handle self-protection approaches for their capability to react and prevent potential attacks [20], [16], [18], [15]. Many of these works focus on architectural adaptations for resource management, such as Yuan et al. [21] who propose software patterns to activate for security breaches, and Li et al. [8] who design a pipeline with various Machine Learning (ML) models for predicting repair actions based on monitoring logs. Similarly, Gill et al. [22] propose a mechanism to self-adapt resource utilization based on software, hardware, and network faults, and Degeler et al. [23] introduce a system to block network communication or program execution.

[1]https://github.com/satrai-lab/sparq/

In contrast to the above works that focus on architectural adaptations, other approaches put focus on the cybersecurity perspective. For example, Dorsey et al. [24] and Kovalenko et al. [25] detect and adapt network systems for data safety and processing, while Wohlrab et al. [26] focus on the tradeoffs between quality attributes and their weights in ML utility functions for explainability purposes. Although these works handle the adaptation in the presence of failures and faults, they lack a deeper investigation of their effects in cyberspace. To address this point, Li et al. [12] propose a self-adaptation framework using Bayesian games to model attacks and defenders. Girdler et al. [27] propose an Intrusion Detection System (IDS) to dynamically detect malicious traffic and adapt the blacklist of malicious devices, and Kholidy [11] introduces an automated response controller for Cyber-Physical Systems (CPS) based on game theory to mitigate cyber threats based on their criticality. Pasquale et al. [28] automate security requirements analysis in order to determine the maximum achievable satisfaction level in terms of security, while Ramadan et al. [29] introduce a framework to balance security, data-minimization, and fairness requirements for business process modeling.

Given the complexity of modern cyber-attacks, recent works found a promising solution in attack graphs. Zeller et al. [14] employ attack graphs to analyze the risks of logic vulnerabilities and apply reconfigurations at runtime. Besides, Skandylas et al. [13] introduce a formal approach to designing AG-based self-protecting systems, while Khakpour et al. [6] use threat modeling to assess the risks of transient configurations during adaptation steps. Although these works focus on a security perspective, they do not consider the role of QoS. As a result, the existing self-protecting systems imply a degradation of the operational performance due to the higher priority of security over QoS [6]. For example, they negotiate increased computational overhead for enhanced security [11], up to the point of making service operations non-functional [13]. This indicates that QoS cannot be ignored while considering security in self-protecting systems as it may cause huge damage, especially in critical infrastructures.

While numerous approaches have been developed for QoS-aware self-adaptive systems, they typically overlook their security aspects. For instance, Delouee et al. [30] and Eryilmaz et al. [31] focus on dynamically adapting sensor deployment and data source assignment using quality evaluation to meet application requirements. Distream [32] introduces dynamic workload balancing for low-latency, high-throughput live video analytics. Hajj Hassan et al. [33] leverage planning methodologies for adaptive data flow configuration in IoT environments based on QoS requirements. Despite the effectiveness of such approaches to managing QoS requirements, they largely ignore security risks affecting IoT systems. Therefore, a gap remains in designing self-adaptive systems that properly address security requirements and, to the best of the authors' knowledge, it is currently missing a system for self-protection through a QoS-aware approach, that we propose in this paper.




(a) Sample network. (b) Architectural adaptation. (c) Security adaptation.

Fig. 1: Motivating scenario showing the different impact of (a) absence, (b) architectural, and (c) security adaptations.

## III. Proposed Approach

This section presents a motivating scenario that highlights the challenges of QoS-aware self-protection and an overview of how SPARQ addresses these challenges.

### A. Motivating scenario

Consider the simple network in Figure 1a, a healthcare network composed of an electrocardiogram machine (ECG), a wearable medical device (blood pressure sensor), and a laptop to manage the data from the two medical devices. A local message broker is used to exchange data. The ECG machine has a vulnerability (CVE-2024-32368) allowing an attacker to deny its services by compromising the Bluetooth component, used to exchange sensor data. The wearable device has a vulnerability (CVE-2024-9141) allowing a remote attacker to perform a cross-scripting attack and compromise the sensor data sent to the laptop. This scenario represents a common hospital setting (e.g., blood analysis laboratory), where vulnerabilities can be newly discovered and not yet patched [34].

The context of the presented network does not include the presence of security experts, rather it includes doctors and nurses, who most likely have low expertise in technology and security. For this reason, employing *self-protection* in such a network would be a mandatory property to promptly protect the network and sensitive data. Existing self-protecting approaches focus on mitigation strategies based on architectural adaptations [35] such as switching off a device or removing communication links. In the above example, the vulnerability of the wearable device is critical because it represents a high risk of compromising medical data. According to existing architectural adaptations, as soon as the system detects the vulnerability, it removes the communication link to prevent the exchange of compromised data (see Figure 1b). This solution protects the network from compromised communication; nevertheless, it has a high impact on the QoS because it affects the termination of the operation of the whole network (i.e., blood analysis services), with all the side effects this implies (e.g., damages of the reputation and disappointment of the patients).

According to cybersecurity catalogs that describe vulnerability mitigation –i.e., Common Weakness Enumeration (CWE)[2]–, it is sufficient to restrict network access only to authorized users to patch the medical device. Therefore it is not worth interrupting the service, but rather adjusting the access controls, representing an alternative to architectural adaptation (see Figure 1c). At the same time, the vulnerability in the ECG requires a higher complexity to be patched because a network administrator must determine trust boundaries that may be complex and time-consuming in such critical infrastructures [36]. In this case, it is worth temporarily blocking the communication to avoid the risk of a cyber attack and waiting for the intervention from security experts[3].

This indicates that determining autonomic mitigation strategies in the presented scenario is far from trivial because architectural adaptations are not always the best solution and, when necessary, their impact on QoS must be evaluated. Evaluating such a trade-off is challenging because there is no standard collection of metrics that may be exploited from a self-protecting system to determine a mitigation plan. This paper addresses these challenges by proposing SPARQ, a novel framework for QoS-aware self-protecting systems.

### B. SPARQ Overview

Figure 2 depicts the overview of SPARQ. The environment is represented by an IoT network, suitably configured to give as **input** of the proposed framework (i) the security posture with respect to cybersecurity standards and (ii) the network performance. The former includes vulnerability detection and cybersecurity knowledge bases providing information about the vulnerabilities detected in the network, their risks, and potential mitigation. The latter monitors the performance of IoT devices through QoS metrics such as latency, which dynamically adapt when changes occur in the network. Additionally to these inputs, SPARQ contributes a comprehensive taxonomy of mitigation strategies. They represent the actions that are necessary to self-protect the IoT network and they can either be architectural or security adaptations. *Architectural adaptations* include actions that change the physical configuration of the network. *Security adaptations* act only to specific vulnerabilities and do not change the network topology. The pool of strategies is the driving component for the subsequent phases of the framework.

In fact, during the **models and analysis** phase, SPARQ leverages security and QoS metrics from the monitored environment to model and analyze the effects of the different adaptations from the taxonomy of mitigation strategies. To achieve such analysis, SPARQ leverages Attack Graph models [37], [38] to analyze the security effects of the different adaptations and Queuing Network Models [39], [40] to analyze their QoS effects. The former is a graph-based representation of the

[2]https://cwe.mitre.org/

[3]This depends on the duration of interventions, as a healthcare network may tolerate only short interruption of services

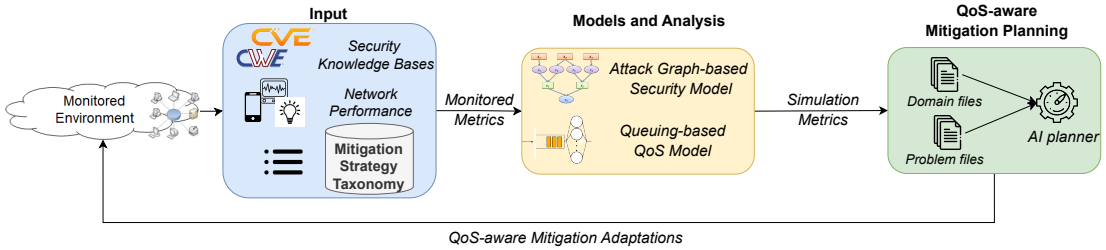



Fig. 2: Design overview of SPARQ considering security and QoS models.

possible attacks in the network that allows for quantitatively estimating the *cyber risk* and *attack surface* as driving metrics to assess the security exposure of a network [41], [42]. The latter enables the evaluation of the *latency* of different adaptations in IoT networks to assess the communication delays. Based on such simulations, SPARQ collects the security and QoS metrics resulting from the application of each mitigation strategy from the taxonomy.

The results from such simulations are given as input to **QoS-aware Mitigation Planning**, which consists of using the security and QoS metrics to create domain and problem files –which contain information about the possible adaptation actions and current state of the network–, given as input to an AI planner that automatically defines the set of adaptations to include in a mitigation plan according to their security and QoS effects. This is achieved by designing the planning and domain files to maximize security while minimizing the effects on the QoS. Consequently, the AI planner can provide the optimal plans automatically [43].

State-of-the-art self-protecting systems typically overlook the combined effects of security and QoS, therefore, a human expert should manually evaluate the trade-off between QoS and security protection, which is a task that is time-consuming, qualitative, and prone to errors. In contrast, SPARQ automatically provides mitigation plans considering both aspects. For example, in the motivating scenario of Figure 1, SPARQ can automatically identify the mitigation strategies to patch the vulnerability in the medical device, saving the continuity of healthcare operations (i.e., without significant reduction of the QoS). This is achieved by simulating the impact of architectural adaptations that would interrupt the data exchange from the medical device (i.e., high degradation of QoS), while the vulnerability does not expose to high-risk situations (i.e., low security effects). Similarly, the AI planner identifies whether the optimal solution for the vulnerability in the ECG machine is to retain the risk or temporarily block the communication.

## IV. SPARQ System Design

Consider an IoT network composed of devices that exchange data to provide specific services (e.g., healthcare analysis). The most used paradigm to exchange data in IoT networks is publish/subscribe (pub/sub), where publishers send messages to a topic and subscribers receive messages from topics they are subscribed to [44]. Additionally, these networks can monitor both QoS and security. The former is measured through sensors that monitor the performance of the services in the network. The latter is achieved through vulnerability scanners that discover vulnerabilities affecting each device. They enable the definition of security and architectural adaptations, which we collect in a comprehensive taxonomy that drives the different phases of the SPARQ.

This section introduces the mitigation strategies employed in SPARQ first. Then, it describes the design of the models employed during the analysis phase and how they allow us to measure security and QoS metrics. We finally present the planning models to automatically provide mitigation plans from these metrics through an AI planner.

### A. A taxonomy of mitigation strategies

One of the main goals of SPARQ is identifying the optimal set of adaptations to mitigate the cyber risks in IoT networks while being aware of their effects on the QoS. To comprehensively consider security and QoS in self-protecting systems, we distinguish two categories of adaptations: architectural or security ones. *Architectural adaptations* refer to changes or modifications to the architectural components of the IoT networks, such as their configurations and interactions to mitigate cyber risks. In contrast, *security adaptations* refer to the actions taken in specific IoT devices to patch their vulnerabilities, without changing the architectural components.

For example, switching off a device degrades the performance more than updating the software of its vulnerable service, because it terminates some services and needs to reroute the IoT network traffic. Therefore, it is fundamental to consider architectural adaptations together with security ones to determine their trade-off in terms of both performance and security. On the one hand, security adaptations are more effective in terms of security and QoS; on the other hand, they have the drawback that they cannot always be executed automatically but rather require human intervention. To indicate the effort of the different mitigation strategies, we provide a taxonomy of the adaptations used in SPARQ and highlight for each one the effects on the QoS and whether they imply changes in the network (architectural adaptations) or not (security adaptations). Table I reports the taxonomy and, to the best of the authors' knowledge, represents a baseline for the relation between mitigation strategies and their QoS effects.

Security adaptations are inspired by the CWE knowledge base and the architectural ones are collected from different research papers on self-adaptive systems [45], [46], [47]. We can recognize two main QoS effects of mitigation strategies: either they require human intervention or can be automatically

TABLE I: Taxonomy of SPARQ adaptations and QoS effects.

| Mitigation adaptation | QoS effects | Category |
|---|---|---|
| Update software libraries | Increased delay | Security |
| Avoid dynamic refactoring | Human intervention | Security |
| Double-Check the used programming language | Human intervention | Security |
| Check compilers correctness | Human intervention | Security |
| Environmental variables hardening | Human intervention | Security |
| Isolate code running from other processes | Human intervention | Security |
| Separate code and data and limit their interaction | Human intervention | Security |
| Separation of privilege: only necessary messages to necessary destinations | Reduced message size + Reachability adaptation | Security |
| Specify output encoding in messages | N/A | Security |
| Input validation | Increased delay | Security |
| Quarantining system files of a device | N/A | Security |
| Use an application firewall | Reachability adaptation | Architectural |
| Traffic redirection | Reachability adaptation | Architectural |
| Attack surface reduction: block untrusted sources | Reachability adaptation | Architectural |
| Limit resource utilization | Reduced message rate | Architectural |
| Change IP address | Reachability adaptation | Architectural |
| Restart a device | Increased delay + Shutdown period | Architectural |
| Packet dropping: drop all the packets to vulnerable destination | Reduced message rate + Reachability adaptation | Architectural |
| Disconnect the device from the Internet | Reachability adaptation | Architectural |
| Terminate one or more services in a device | Reduced message rate + Shutdown period | Architectural |
| Block one or more ports of a device | Reduced message rate | Architectural |
| Terminate all services in a device | Shutdown period | Architectural |

applied to the IoT network. In the former case, there is not an immediate adaptation of the network but the execution time depends on many different parameters that, in most of the cases, cannot be controlled prior to the actual execution. For example, "Check compiler correctness" requires a developer to manually evaluate the status of current compilers and the time necessary to complete this check may vary depending on the developer's expertise. However, the QoS is not altered until the execution starts and human intervention does not mean that it takes necessarily long time. In fact, most of the human-dependent adaptations are supported by automated tools (e.g., code isolation tools). Additionally, these kinds of adaptations typically correspond to patching vulnerabilities by checking some security parameters (e.g., vulnerable code) and have the advantage of applying to specific services running on the devices, without altering the network configuration. In this sense, they are less intrusive and more effective than architectural adaptations that modify the network architecture possibly without the need to do it. For example, instead of switching off a server with sensitive data, the risk can be lowered with a less intrusive action such as software updates, that keep the server on, and therefore without compromising the availability of the operational services. To distinguish these cases, a comprehensive taxonomy such as the one we proposed in Table I is necessary. This taxonomy is used in the analysis phase to simulate mitigation actions' effects on the security and QoS of the IoT networks. Similarly, actions are modeled in the AI planning domain files to enable an AI planner to provide the optimal QoS-aware mitigation plan.

### B. SPARQ models and analysis

*1) Security Model:* To model the effects of security adaptations we leverage the AG model [37], [38]. This is a graph-based representation of the possible ways an attacker can intrude and compromise a network. Such a model is based on the *network reachability graph* and the system *vulnerability inventory* [42]. In a network reachability graph, nodes are associated with network devices and edges represent reachability conditions between them (taking into account firewall rules and routing information). A vulnerability inventory lists devices with associated vulnerabilities, typically obtained automatically by vulnerability scanners (e.g., Nessus[4]). Vulnerabilities are elements of the CVE knowledge base, which is a cybersecurity standard that regularizes references to vulnerabilities across different security tools, databases, and vendors. Starting from these inputs, automated tools exist to model the AG, where a node represents the capability that an attacker needs to perform an exploit and its consequent attacker state, while edges represent a successful attacker's exploit [42], [48]. More formally, an Attack Graph $G = (V, E)$ is a directed multi-graph in which $V$ is the set of security condition nodes and $E$ is the set of labeled edges where an edge $e = (v_1, v_2, u) \in E$ indicates that the attacker can move from condition $v_1$ to condition $v_2$ by successfully exploiting vulnerability $u$.

Once the AG model is defined, the existing attack paths are computed to perform security analysis and estimate the level of exposure to cyber risks or to online monitor the evolution of possible attacks. These paths represent sequences of compromised devices and vulnerabilities exploited during the attack, thus determining the possible attacks in the network. An Attack Path $AP = \langle v_1, u_{1,2}, v_2, u_{2,3}, \cdots v_n \rangle$ is the ordered sequence of attacker states $v_1, \cdots, v_n$, interleaved by the sequence of vulnerabilities $u_{1,2}, \cdots, u_{n-1,n}$ which allows an attacker to move between consecutive states.

We calculate the *cyber risk* associated with attack paths through the Common Vulnerability Scoring System (CVSS[5]), which is a standardized framework assessing security metrics (with numerical scores) of CVE vulnerabilities. In particular, we employ existing cyber risk models [41] that estimate the cyber risk based on its standard definition: $R = risk(AP) = likelihood(AP) \cdot impact(AP)$, where the likelihood is calculated using the CVSS-3.1 exploitability metrics (Attack Vector, Attack Complexity, Privilege Required, and User Interaction), and the impact is determined by CVSS-3.1 impact metrics (Confidentiality, Integrity, and Availability). In particular, an aggregation function (e.g., max) is applied to all vulnerabilities participating in the attack path $AP$ to compute the risk [41]. Since CVSS metrics are defined in the range $[0, 1]$, the risk is in the same range too. Beyond cyber risk, AG is a useful analytics tool to measure the *attack surface* $AS = count(AP)$ determined as the number of possible strategies an attacker can perform to attack network devices, where the higher the attack surface is, the larger the pool of attacks. Therefore, cyber risk $R$ and attack surface $AS$ are the driving security metrics to analyze the exposure to cyber attacks [6].

AG provides comprehensive support to investigate the ex-

[4]https://www.tenable.com/products/nessus

[5]https://www.first.org/cvss/v3.1/specification-document

posure of the IoT network to cyber attacks during different instants of time, reflecting the network changes during monitoring. For example, when a vulnerability is patched, the exposure (cyber risk and attack surface) changes as the attacker has one less entry point to successfully perform the attack. Similarly, changes in the network architecture (e.g., switching off a device) reflect changes in security exposure. This means that we can leverage AG to model the security effects that an adaptation has on the network. Note that we are considering both architectural and security adaptations and AG is capable of modeling both. To analyze the security benefits of an adaptation, we simulate it within the AG model and recompute the security metrics. Security adaptations correspond to patching one (or more) vulnerability in some devices; we simulate them by removing the vulnerabilities under investigation from the vulnerability inventory from which the AG is modeled. Let $AP = \langle v_1, u_{1,2}, v_2, u_{2,3}, \cdots v_n \rangle$ be the attack path where the vulnerability $u_{1,2}$ is patched through a security adaptation. Then, we simulate such an adaptation by modeling the attack paths $AP^* = \langle v_2, u_{2,3}, \cdots v_n \rangle$ instead of $AP$, thus calculating the new risk value $R^* = risk(AP^*)$ and attack surface $AS^* = count(AP^*)$. Architectural adaptations correspond to modifying the network configurations and we simulate them by reflecting such modifications in the reachability graph from which the AG is modeled. Let $AP = \langle v_1, u_{1,2}, v_2, u_{2,3}, \cdots v_n \rangle$ be the attack path where the device $v_1$ is reconfigured (e.g., switched off) due to an architectural adaptation. Then, we simulate such an adaptation by modeling the attack paths $\hat{AP} = \langle v_2, u_{2,3}, \cdots v_n \rangle$ instead of $AP$, thus calculating the new risk value $\hat{R} = risk(\hat{AP})$ and attack surface $\hat{AS} = count(\hat{AP})$. Note that while security adaptations are modeled at the level of vulnerabilities, architectural ones are modeled at the level of devices. In this way, simulating adaptations over the AG becomes a very flexible task without modifying the AG algorithms, but only modeling their inputs.

*2) QoS Model:* To evaluate the performance of IoT networks when applying different adaptations, we design a QoS model using *Queueing Networks* [39], [40]. Queueing networks have been widely used for modelling the behavior of complex systems involving multiple processes, resources, and interactions [49]. They also provide a strong formal framework for representing and analyzing computer networks analytically. Our QoS model relies on the publish/subscribe paradigm [44] to abstract data exchange in IoT networks due to its wide usage in IoT environments [50]. For this purpose, each IoT device $d_i \in D$ acts as a publisher and generates messages tagged with a topic $t_j$, characterized by a message size $G_{t_j}$ and a generation rate $\lambda_{t_j}$. $\lambda_{t_j}$ may generate messages based on a probability distribution (e.g., Exponential, Deterministic) or by relying on real-world traces of existing smart spaces. A message broker accepts publications from the devices $D$ via an input queue $Q_{in}$ of type G/G/1[6]. $Q_{in}$'s service rate $\mu_{in}$ represents the message broker's processing time. A network queue $Q_{net}$ represents the networking infrastructure; its service rate $\mu_{net}$ represents the available bandwidth between message broker and applications. Each application $\mathrm{a}_i \in A$ is represented by a queue $Q_{\mathrm{a}_i}$ and indirectly subscribes to topics by specifying subscription filters. We define a subscription as the tuple $r_j = (\mathrm{a}_i, t_j) \in \overline{R}$. For each subscription $r_j$, we define a data flow $f_i \in F$ matching $r_j$ starting from the broker towards an application $\mathrm{a}_i$.

[6]A G/G/1 queue denotes a single-server FIFO queue with general distributions of inter-arrival and service times.

Given an IoT network description, we can instantiate and compose a corresponding QoS model for evaluating the performance of data flows $F$. A QoS model can be simulated using any queueing simulator to generate a dataset containing QoS metrics including the end-to-end latency for each data flow $\Delta_{f_i}$, the throughput $\Theta_{f_i}$, data drop rate $\Xi_{f_i}$ and other relevant metrics. SPARQ's QoS model is primarily used for evaluating the QoS performance of the IoT network. For each adaptation $m \in M$, which does not require human intervention (see Table I), SPARQ instantiates the QoS model and composes an IoT network where $m$ is applied. This may include turning OFF devices/links, redirecting traffic, blocking some ports, or other possible actions that can be performed.

### C. Planning mitigation strategies

An AI planning system ("planner", for short) takes a problem formalization, or model, as input and uses some problem-solving technique, such as heuristic search or propositional satisfiability to work out its solution [43]. The descriptive models used by planning systems are called *planning domains*. These include a description of the planning environment, i.e., *states*, and *actions* that can be taken by the planner in order to reach a certain *goal*. In addition, a *cost* can be associated with one or more actions.

*Definition 1:* **Planning Domain.** A planning domain is a state transition system $\Sigma = (S, \mathcal{A}, \gamma, \mathcal{C})$, where:

- $S$ is a finite set of states of the system. These refer to the states of the IoT network in terms of risk level and QoS performance. For instance, a device in the IoT network is exposed to high risk (e.g., $0.9$) and continues working normally (e.g., latency unchanged).
- $\mathcal{A}$ is a set of actions that an agent may perform. Actions are used to alter systems based on conditions and effects. In SPARQ, we consider mitigation actions to reduce the cyber risk (see Table I).
- $\gamma : S \times \mathcal{A} \rightarrow S$ is the state transition function. If $\gamma(s, \alpha)$ is defined then action $\alpha$ is applicable to state $s$, with $\gamma(s, \alpha)$ being the predicted outcome. For example, updating software libraries of a host for patching a security vulnerability incurs a higher latency for data flows associated with that host.
- $\mathcal{C} : S \times \mathcal{A} \rightarrow [0, \infty)$ is a cost function with the same domain as $\gamma$. This can represent a cost function minimizing cyber risk, increasing latency, or other parameters that have to be optimized.

Given a planning domain, we can then define one or more planning problems $P = (\Sigma, s_0, G)$ where $\Sigma$ is a state-transition domain, $s_0$ is the initial state, and $G$ is a set of

ground literal goals. The goal state is typically the desired final state of the system, for example reducing the overall cyber risk of a network while satisfying QoS requirements of applications deployed in the network. We propose a planning problem to generate mitigation plans for reducing cyber risk in IoT networks, while also considering the effects of such plans on QoS metrics. SPARQ currently considers cyber risk $R$ and attack surface $AS$ as security metrics and latency as QoS metrics. However, note that SPARQ's definition of a planning problem is generic enough to cover additional risk and QoS factors that may be relevant in particular scenarios.

To define the planning problem, we first model a planning domain (defined earlier) and an initial state $s_0$, where no mitigation action is applied to the IoT network. $s_0$ also includes information about the current security metrics and QoS metrics. In addition, constraints such as minimizing or maximizing QoS metrics and security exposure are defined. The goal state $G$ defines that all constraints should be met such as minimizing (e.g., latency) and/or maximizing (e.g., throughput) related QoS metrics, as well as minimizing cyber risk and attack surface.

An AI planner takes as input a planning domain and a planning problem, and finds a solution consisting of a set of actions that transform the system from the initial state to reach the final desired goal state.

*Definition 2:* **Plan.** A plan is a finite set of actions:

$$\pi = \langle \alpha_1, \alpha_2, \ldots, \alpha_n \rangle \tag{1}$$

where the plan's length $|\pi|$ is $n$, and its *cost* is the sum of the action costs: $cost(\pi) = \sum_{i=1}^{n} cost(a_i)$ . A plan $\pi$ is applicable to a state $s_0 \in S$ if there are states $s_1, s_2, \ldots, s_n$ such that $\gamma(s_{i-1}, \alpha_i) = s_i$ for $i = 1, \ldots, n$. In this case, $\gamma(s_0, \alpha_\pi) = s_n$ (with $\alpha_\pi$ being the last action in plan $\pi$). A solution for $P$ is a plan $\pi'$ such that $\gamma(s_0, \alpha_1) \ldots \gamma(s_m, \alpha'_\pi)$ satisfies $G$.
In our case, the planner generates a plan $\pi = \langle \alpha_1, \alpha_2, ..., \alpha_n \rangle$ where the total cyber risk and attack surface of the system is minimized while also minimizing the overall latency of the system $\delta_{system}$, meeting all goal conditions. $\pi$ is a plan consisting of a set of adaptations to be applied to the vulnerable devices in the IoT network, according to the initial situation of the building, defined in $s_0$. In this way, the set of adaptations from the plan provided by the planner is the optimal ones that allow self-protection while being QoS-aware. Techniques used to generate plans from the initial state to the goal state include: (i) graph based search techniques, (ii) state-transition systems, (iii) constraint solvers that make use of symbolic predicates, constraints and effects, (iv) heuristic approximations such as removing negative predicates. Comparison of scale, benchmarking and speed of AI planning solvers has been evaluated within the International Planning Competition[7].

## V. Experimental Results

This section presents the experimental evaluation of SPARQ in two real network settings. We discuss the impact of security and QoS, and compare SPARQ against existing approaches that focus on architectural adaptations or consider only QoS adaptations, as they represent the current baselines of self-protection. In addition, we validate the correctness of the proposed mitigation plans with respect to ground truth from security knowledge bases.

### A. Experimental Setup

To experimentally evaluate the SPARQ approach, we emulate two real IoT networks from different domains and simulate network traffic according to real specifications. One is a healthcare network from a hospital laboratory with 5 medical devices, 3 intermediate servers, and 5 laptops. The vulnerability inventory has been collected with the OpenVAS vulnerability scanner and manually removing false positives. The resulting vulnerability inventory consists of 512 vulnerabilities in the whole network. The second network is a smart home environment composed of printers, energy management platforms, cameras, Amazon Echo, and different sensors such as occupancy and fire detectors. It includes 216 vulnerabilities representing the vulnerability inventory, obtained by running OpenVAS and Nessus vulnerability scanners.

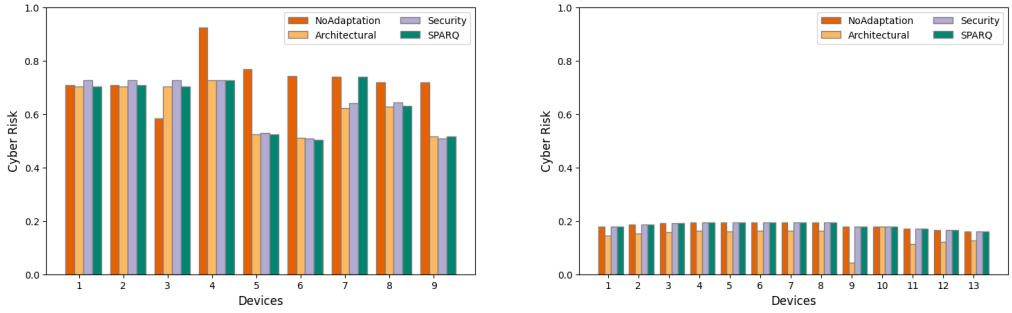
For the security model, we simulate the human intervention of security adaptations according to three different strategies: (i) security operators decide to *retain the risk*, (ii) security operators decide to patch vulnerabilities based on *hosts* (i.e., all vulnerabilities in a host), (iii) security operators decide to patch *single vulnerabilities*. We also consider their combinations to have an exhaustive set of possible scenarios. Architectural adaptations are simulated directly by changing the vulnerability inventory and reachability graphs as reported in Section IV-B1. For simulating queuing models representing IoT networks, we rely on the Java Modelling Tools (JMT) simulator [39]. We run simulations for both QoS models of healthcare and smart home IoT networks based on the applied mitigation plans. We then extract the results of the simulations from the JMT XML files and create a dataset of performance metrics consisting of CSV files. Planning techniques are used to find the optimal mitigation plan. We use PDDL [51] to write the domain and problem files used as an input to the Metric-FF solver [52] that generates the necessary plans.

To conduct the evaluation, we consider and compare our approach with three different scenarios: (i) self-protecting systems that do not implement any adaptation, thus providing a standard baseline (*NoAdaptation*); (ii) state-of-the-art self-protecting systems modeling only architectural adaptations to the network: [23], [6], [13], [14] (*Architectural*); (iii) response systems modeling only security adaptations: they represent the current state-of-the-art on cybersecurity response, typically not employable in self-protecting systems, but rather as response systems [53], [41], [54] (*Security*). The reference adaptation strategies are the ones proposed in Table I.

The evaluation aims to answer the following research questions (RQs):

**RQ1**: To what extent is considering both architectural and security adaptations beneficial for security?

[7]https://www.icaps-conference.org/competitions/




(a) Smart Home network.

(b) Healthcare network.

Fig. 3: Cyber risk for (a) smart home and (b) healthcare networks.

**RQ2**: To what extent is considering both architectural and security adaptations beneficial for QoS?

**RQ3**: What is the security-QoS trade-off?

To answer RQ1, we evaluate the security effects of the proposed approach by considering cyber risk and attack surface assessments. The former shows the ability to reduce the risk among the different devices, and the latter quantitatively measures the attack surface, intended as the number of possible strategies an attacker can employ to compromise a device. Consequently, we evaluate the QoS to show the effects of the latency between the different approaches to answer RQ2. Finally, we evaluate the accuracy of the mitigation plans provided by SPARQ with respect to the ideal security actions expected by the CWE knowledge base to answer RQ3.

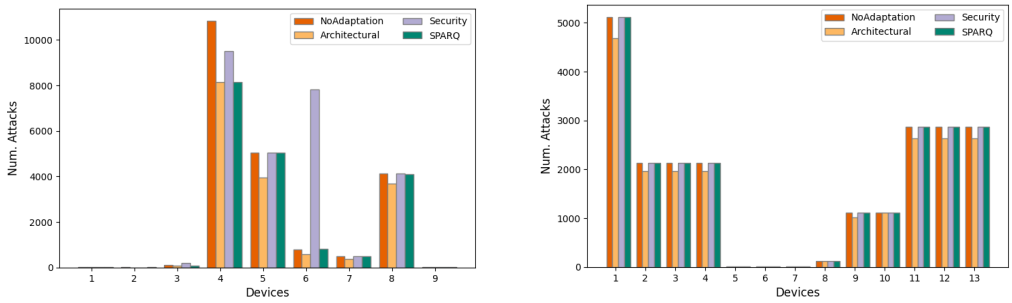
Experiments are executed on a PC with an Intel Core i7-11800H 2.3GHz processor and 16 GB memory. The source code of SPARQ and data for reproducibility are available on: https://github.com/satrai-lab/sparq/.

### *B. Security evaluation*

*1) Cyber risk assessment:* To assess cybersecurity in self-protecting systems, the cyber risk is commonly used to determine the attack exposure [41], [11]. Thus, we measure the average cyber risk in the different devices for the smart home (Figure 3a) and healthcare (Figure 3b) networks, according to the security model described in Section IV-B1. The bar chart shows that the cyber risk is higher when no adaptation is applied, for both networks. This is an expected result as there are no actions to mitigate the risk. Considering the smart home scenario, the average risk with no adaptation is $0.74$, reduced to $0.63$, $0.64$, and $0.64$ in the architectural, security, and SPARQ systems, respectively. Similarly, for the healthcare network, the average risk is $0.18$, $0.14$, $0.18$, and $0.18$, respectively. An interesting outcome of this investigation is that architectural adaptations provide mitigation plans that reduce cyber risk more than others. Apparently, this is a good outcome. However, we need to consider that such a reduction is due to the fact that architectural adaptations represent more drastic solutions. For example, instead of performing an easy vulnerability patching, they modify the network structure (e.g., switching off a device). Hence, this reduces the attack surface, but also affects the QoS, as shown in the next section.

Another interesting observation, looking at the trend of the cyber risk among the different devices, is that the proposed approach effectively trades off cyber risk so that the cyber risk is between the values provided with architectural and security adaptations. This is particularly visible in the smart home network. The healthcare network presents less variability of cyber risk, especially between security and SPARQ. The reason behind this behavior is the presence of fewer vulnerabilities in the devices due to their critical infrastructure. This indicates that, in critical networks (commonly less vulnerable because some critical vulnerabilities are patched), the proposed hybrid approach tends to be similar to security mitigation plans rather than architectural ones. The interesting point is that the risk in such a network is not particularly high, and therefore it is not worth switching off devices or redirecting traffic, because it has an impact on the QoS that cannot be tolerated in critical infrastructures. In contrast, existing architectural approaches overlook this problem as they do not consider cyber risk as metrics for the self-protecting workflow.

*2) Attack surface assessment:* When removing vulnerabilities from the network through architectural or security adaptations, a remote attacker can no longer exploit some steps of the possible attacks. Even though this impacts the system's performance (see next QoS evaluation), it reduces the attack surface in the network, measured as the number of possible attacks that can be employed, which is the number of paths in the AG. Figure 4 reports the number of attack paths targeting each device. The bar chart illustrates that the highest reduction is obtained with the architectural adaptations. This is again an expected result because such adaptations remove more vulnerabilities than needed, thus drastically reducing the paths in the AG. As for the cyber risk assessment, this advantage must be traded off with the impact on QoS as dropping out network traffic or devices increases the latency of network services. In particular, in the smart home network, the architectural adaptations reduce the attack surface by $20\%$ than the baseline with no adaptations. In contrast, security adaptations reduce the attack surface by $10\%$, while SPARQ reduces the attack surface by $15\%$. This is due to the fact that security adaptations imply that only specific vulnerabilities are patched, without interfering with the QoS of entire devices or services, while SPARQ shows traded-off effects.




(a) Smart Home network.

(b) Healthcare network.

Fig. 4: Attack surface for (a) smart home and (b) healthcare networks.

The healthcare network shows a flatter trend, where SPARQ and security adaptations are comparable with the baseline scenario, while architectural adaptations slightly reduce the attack surface. The reasons for this different behavior can be again attributed to the criticality of the network that brings the proposed approach to resemble the security adaptation plans to not interfere with the QoS. This enhances the importance of considering QoS in self-protecting systems.

**Answer RQ1**: *While security adaptations represent the desiderata in terms of security (i.e., ideally patching all vulnerabilities), SPARQ identifies when architectural strategies are necessary in more risky situations and when it is not worth changing the network configuration because of low risk. This advances the literature where existing self-protecting systems are not capable of identifying such context-aware information due to missing consideration of the security-QoS trade-off.*

### C. QoS evaluation

Both architectural and security adaptations have effects on the QoS of self-protecting systems. In particular, we are interested in understanding these effects, especially in relation to the security assessment. We analyze the latency of communication messages as a QoS metric as a pivotal indication of the network performance in terms of its services. The latency measures the time taken for data to be exchanged from a source to a destination in a network, thus we ideally want the lowest possible latency, especially in critical networks such as the healthcare one. Figure 5 reports the analysis for the smart home (Figure 5a) and healthcare (Figure 5b) networks of SPARQ compared with the three scenarios under analysis (no adaptation, architectural, and security). The chart shows that, as expected in both networks, the system with no adaptation has the best performance as there are no adaptations that interfere with the network structure, therefore the network is not changed with respect to its initial design. The average latency is $0.17$ seconds in the smart home network and $0.32$ in the healthcare one. Architectural adaptations provide an almost double latency in the smart home network(equal to $0.45$) and the healthcare one (equal to $0.39$). The reasons behind this performance degradation reside in the fact that architectural adaptations change the network structure and therefore the initial design of the network. As a result, the service provisioning must be changed and the traffic re-routed with the consequent increased latency. A similar behavior, but with less effects on the QoS, is attributed to the security adaptations. In fact, some of them still require shutdown times of the devices (e.g., the time during the software updates) that impact the latency with an average value of $0.39$ in the smart home network and $0.32$ in the healthcare one.

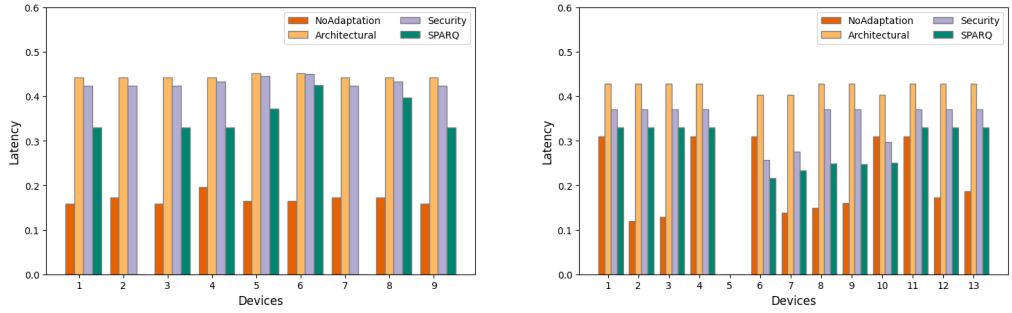
Finally, some devices in both networks have a small attack surface (i.e., a low number of possible attacks), with a high cyber risk. This indicates that such devices have few vulnerabilities (therefore they allow a low number of attacks) but expose the network to critical security risks. Looking at the corresponding latency values of such devices (devices with IDs $1, 2, 3, 9$ for the smart home network and $5, 6, 7, 8$ for the healthcare one) we can notice that the proposed approach has the lowest latency, offering better service performance. This indicates that SPARQ is capable of detecting the most critical vulnerabilities where it is more appropriate to provide security adaptation without interrupting network services. This is because the planner takes into account latency, cyber risk, and attack surface of the different devices.

**Answer RQ2**: *SPARQ outperforms the existing solutions, showing the capability to correctly identify adaptations worth applying or not: if the cyber risk is low it may not be worth changing the network infrastructure. This results in an average latency of* $0.27$ *seconds in the smart home network and* $0.26$ *seconds in the healthcare one. These values are close to the ideal QoS behaviors (i.e., no adaptation applied), showing a clear trade-off of the existing solutions.*

### D. Evaluation of security plans

The last aspect to validate is the correctness of the mitigation plans provided by the SPARQ automated planner. Note that the mitigation plans are a set of actions to be applied to each device, including the different adaptations reported in Table I. In terms of security, the optimal solution is to patch all the vulnerabilities in each device, although impractical and sometimes impossible in IoT networks [2]. For this reason, we consider CWE as the ground truth knowledge base that reports the mitigation actions for each vulnerability. We compare the strategies from the planner with the ones from CWE and considered the following metrics for each device:

- True Positives (TP) are the strategies that are in the security plan and expected by CWE;

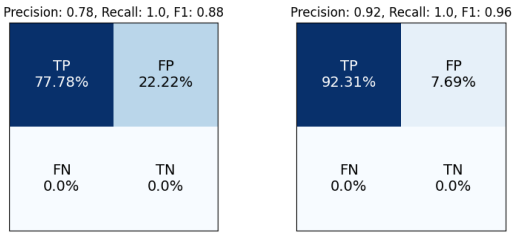



(a) Smart Home network. (b) Healthcare network.

Fig. 5: Network latency for (a) smart home and (b) healthcare networks.

- False Positives (FP) are the strategies that are in the security plan, but not expected by CWE;
- False Negatives (FN) are the strategies that are not in the security plan, but are expected by CWE;
- True Negatives (TN) are the strategies that are neither in the security plan nor are expected by CWE.

Figure 6 reports the confusion matrices of the networks.




(a) Smart Home network. (b) Healthcare network.

Fig. 6: Confusion matrices of mitigation plans.

As expected, there are no true negatives because they correspond to undetectable adaptations both in the ground truth and in the provided plans. It is interesting to note the absence of false negatives as it corresponds to adaptations that should be present but are not. This indicates that the planner is *conservative* in terms of mitigation plans with respect to the security knowledge bases, and this is due to the wide generality of CWE, which includes many strategies for the same vulnerabilities. On the one hand, this explains the absence of false negatives; on the other hand, this enforces the need for more structured plans as we proposed in this paper. The percentage of false positives is $22.22\%$ in the smart home network and $7.69\%$ in the healthcare one. They correspond to security strategies that are not really necessary according to CWE. However, they still help to reduce the cyber risk, thus enforcing the conservatives of the proposed approach, which tends to apply more strategies rather than less. Concerning true positives, Fig. 6 shows that most of the strategies are correctly selected from the planner with accuracy from $0.78$ (smart home) up to $0.92$ (healthcare). This indicates the good performance of the planner in predicting the security adaptations with respect to CWE. The presence of false positives is because SPARQ takes into account architectural adaptations too, which are generally not reported in security knowledge bases. However, they are important because they represent strategies that can immediately be applied in high-risk situations. Thus, we consider the results promising since the overall accuracy is high and the presence of false positives is the price to pay for the security-QoS trade-off.

**Answer RQ3**: *Existing solutions that apply only architectural adaptations degrade the performance of the QoS due to the drastic actions that are put in place even when not necessary. Conversely, applying only security adaptations provides very specific mitigation actions, but disregards the QoS and the possibility of employing them autonomously. SPARQ balances security and QoS and shows promising results in providing QoS-aware mitigation plans: in terms of QoS it outperforms existing solutions; in terms of security it shows an average of* $0.85$ *accuracy in determining the ideal plans.*

## VI. Conclusion

This article introduces SPARQ, a QoS-aware framework for self-protecting systems based on Attack Graphs and Queuing Network models. Collecting security and QoS metrics from the monitored environment and employing a taxonomy of mitigation strategies, SPARQ analyzes their security and QoS effects and defines optimal mitigation plans accordingly. This addresses the problem of taking too drastic architectural adaptations when not necessary, without sacrificing security. We validated SPARQ on two networks from real scenarios. Results show its capability to reduce the risk without sacrificing the QoS with network performance improved by 35% compared to existing self-protecting systems. In future work, we plan to improve the execution component of the proposed framework by designing methodologies to automate security adaptations. In addition, we will explore alternative AI planners that can learn from previous plans beyond security and QoS metrics. Finally, we will explore how to enhance the context-awareness of SPARQ by providing additional parameters to the planner, such as the cost of each adaptation.

## Acknowledgment

This work is supported by the Horizon Europe projects DI-Hydro (grant agreement No. 101122311) and MEDIATE (grant agreement No. 101168465). It is also supported by the *Avvio alla Ricerca* Sapienza grant No. AR2241902B426269.

## References


[1] W. H. Hassan *et al.*, "Current research on internet of things (iot) security: A survey," *Computer networks*, vol. 148, pp. 283–294, 2019.

[2] T. Sasi, A. H. Lashkari, R. Lu, P. Xiong, and S. Iqbal, "A comprehensive survey on iot attacks: Taxonomy, detection mechanisms and challenges," *Journal of Information and Intelligence*, 2023.

[3] J. Navarro, A. Deruyver, and P. Parrend, "A systematic survey on multi-step attack detection," *Computers & Security*, vol. 76, pp. 214–249, 2018.

[4] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying iot security: An exhaustive survey on iot vulnerabilities and a first empirical look on internet-scale iot exploitations," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019.

[5] P. Arcaini, E. Riccobene, and P. Scandurra, "Modeling and analyzing mape-k feedback loops for self-adaptation," in *SEAMS, Firenze, Italy*. IEEE/ACM, 2015, pp. 13–23.

[6] N. Khakpour, C. Skandylas, G. S. Nariman, and D. Weyns, "Towards Secure Architecture-Based Adaptations," in *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, May 2019, pp. 114–125, iSSN: 2157-2321.

[7] H. Lee, A. Mudgerikar, A. Kundu, N. Li, and E. Bertino, "An infection-identifying and self-evolving system for iot early defense from multi-step attacks," in *Computer Security – ESORICS 2022*, V. Atluri, R. Di Pietro, C. D. Jensen, and W. Meng, Eds. Cham: Springer Nature Switzerland, 2022, pp. 549–568.

[8] R. Li, Z. Cheng, P. P. C. Lee, P. Wang, Y. Qiang, L. Lan, C. He, J. Lu, M. Wang, and X. Ding, "Automated Intelligent Healing in Cloud-Scale Data Centers," in *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*, Sep. 2021, pp. 244–253, iSSN: 2575-8462.

[9] B. Yuan, Z. Pan, F. Shi, and Z. Li, "An attack path generation methods based on graph database," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1, 2020, pp. 1905–1910.

[10] L. Gressl, A. Rech, C. Steger, A. Sinnhofer, and R. Weissnegger, "Security based design space exploration for cps," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, ser. SAC '20, 2020.

[11] H. A. Kholidy, "Autonomous mitigation of cyber risks in the Cyber–Physical Systems," *Future Generation Computer Systems*, vol. 115, pp. 171–187, Feb. 2021.

[12] N. Li, M. Zhang, J. Li, S. Adepu, E. Kang, and Z. Jin, "A Game-Theoretical Self-Adaptation Framework for Securing Software-Intensive Systems," *ACM Transactions on Autonomous and Adaptive Systems*, p. 3652949, Mar. 2024.

[13] C. Skandylas and N. Khakpour, "Design and implementation of self-protecting systems: A formal approach," *Future Generation Computer Systems*, vol. 115, pp. 421–437, 2021.

[14] S. Zeller, N. Khakpour, D. Weyns, and D. Deogun, "Self-protection against business logic vulnerabilities," in *Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. Seoul Republic of Korea: ACM, Jun. 2020, pp. 174–180.

[15] S. Walker-Roberts, M. Hammoudeh, and A. Dehghantanha, "A systematic review of the availability and efficacy of countermeasures to internal threats in healthcare critical infrastructure," *IEEE Access*, vol. 6, pp. 25 167–25 177, 2018.

[16] A. Dehghantanha, A. Yazdinejad, and R. M. Parizi, "Autonomous cybersecurity: Evolving challenges, emerging opportunities, and future research trajectories," in *Proceedings of the Workshop on Autonomous Cybersecurity*, ser. AutonomousCyber '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 1–10.

[17] I. Pekaric, R. Groner, T. Witte, J. G. Adigun, A. Raschke, M. Felderer, and M. Tichy, "A systematic review on security and safety of self-adaptive systems," *Journal of Systems and Software*, vol. 203, p. 111716, 2023.

[18] C. Rouff, L. Watkins, R. Sterritt, and S. Hariri, "Sok: Autonomic cybersecurity - securing future disruptive technologies," in *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, 2021, pp. 66–72.

[19] E. Yuan, N. Esfahani, and S. Malek, "A systematic survey of self-protecting software systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 8, no. 4, jan 2014.

[20] M. Bashendy, A. Tantawy, and A. Erradi, "Intrusion response systems for cyber-physical systems: A comprehensive survey," *Computers & Security*, vol. 124, p. 102984, Jan. 2023.

[21] E. Yuan, S. Malek, B. Schmerl, D. Garlan, and J. Gennari, "Architecture-based self-protecting software systems," ser. QoSA '13. ACM, 2013, p. 33–42.

[22] S. S. Gill, I. Chana, M. Singh, and R. Buyya, "Radar: Self-configuring and self-healing in resource management for enhancing quality of cloud services," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 1, p. e4834, 2019.

[23] V. Degeler, R. French, and K. Jones, "Self-healing intrusion detection system concept," in *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS)*, 2016, pp. 351–356.

[24] L. C. Dorsey, B. Wang, M. Grabowski, J. Merrick, and J. R. Harrald, "Self healing databases for predictive risk analytics in safety-critical systems," *Journal of Loss Prevention in the Process Industries*, vol. 63, p. 104014, 2020.

[25] A. Kovalenko and H. Kuchuk, *Methods to Manage Data in Self-healing Systems*. Cham: Springer International Publishing, 2022, pp. 113–171.

[26] R. Wohlrab, J. Cámara, D. Garlan, and B. Schmerl, "Explaining quality attribute tradeoffs in automated planning for self-adaptive systems," *Journal of Systems and Software*, vol. 198, p. 111538, 2023.

[27] T. Girdler and V. G. Vassilakis, "Implementing an intrusion detection and prevention system using Software-Defined Networking: Defending against ARP spoofing attacks and Blacklisted MAC Addresses," *Computers & Electrical Engineering*, vol. 90, p. 106990, Mar. 2021.

[28] L. Pasquale, P. Spoletini, M. Salehie, L. Cavallaro, and B. Nuseibeh, "Automating trade-off analysis of security requirements," *Requirements Engineering*, vol. 21, pp. 481–504, 2016.

[29] Q. Ramadan, D. Strüber, M. Salnitri, J. Jürjens, V. Riediger, and S. Staab, "A semi-automated bpmn-based framework for detecting conflicts between security, data-minimization, and fairness requirements," *Software and Systems Modeling*, vol. 19, pp. 1191–1227, 2020.

[30] L. Delouee *et al.*, "AQuA-CEP: Adaptive Quality-Aware Complex Event Processing in the Internet of Things," in *ACM DEBS*, 2023.

[31] E. Eryilmaz *et al.*, "Quality-Aware Service Selection Approach for Adaptive Context Recognition in IoT," in *ACM IoT Conf.*, 2019.

[32] X. Zeng *et al.*, "Distream: Scaling Live Video Analytics with Workload-Adaptive Distributed Edge Intelligence," in *ACM SenSys*, 2020.

[33] H. Hajj Hassan *et al.*, "PlanIoT: A Framework for Adaptive Data Flow Management in IoT-enhanced Spaces," in *IEEE/ACM SEAMS*, 2023.

[34] F. Nausheen and S. H. Begum, "Healthcare iot: Benefits, vulnerabilities and solutions," in *2018 2nd International Conference on Inventive Systems and Control (ICISC)*. IEEE, 2018, pp. 517–522.

[35] H. Muccini, M. Sharaf, and D. Weyns, "Self-adaptation for cyber-physical systems: a systematic literature review," in *Proceedings of the 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 75–81.

[36] X. Fan, L. Liu, R. Zhang, Q. Jing, and J. Bi, "Decentralized trust management: Risk analysis and trust aggregation," *ACM Comput. Surv.*, vol. 53, no. 1, Feb. 2020. [Online]. Available: https://doi.org/10.1145/3362168

[37] K. Kaynar, "A taxonomy for attack graph generation and usage in network security," *Journal of Information Security and Applications*, vol. 29, pp. 27–56, Aug. 2016.

[38] K. Zenitani, "Attack graph analysis: an explanatory guide," *Computers & Security*, 2022.

[39] M. Bertoli, G. Casale, and G. Serazzi, "Jmt: performance engineering tools for system modeling," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 4, pp. 10–15, 2009.

[40] D. Gross, J. Shortle, J. Thompson, and C. Harris, *Fundamentals of queueing theory*. John Wiley & Sons, 4th edition, 2008.

[41] G. Gonzalez-Granadillo, S. Dubus, A. Motzek, J. Garcia-Alfaro, E. Alvarez, M. Merialdo, S. Papillon, and H. Debar, "Dynamic risk management response system to handle cyber threats," *Future Generation Computer Systems*, vol. 83, pp. 535–552, 2018.

[42] A. Palma and M. Angelini, "It is time to steer: A scalable framework for analysis-driven attack graph generation," in *Computer Security – ESORICS 2024*, J. Garcia-Alfaro, R. Kozik, M. Choraś, and S. Katsikas, Eds. Cham: Springer Nature Switzerland, 2024, pp. 229–250.

[43] P. Haslum, N. Lipovetzky, D. Magazzeni, C. Muise, R. Brachman, F. Rossi, and P. Stone, *An introduction to the planning domain definition language*. Springer, 2019, vol. 13.
[44] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM CS*, vol. 35, no. 2, Jun. 2003.
[45] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker, "A survey on engineering approaches for self-adaptive systems," *Pervasive and Mobile Computing*, vol. 17, pp. 184–206, 2015.
[46] F. D. Macías-Escrivá, R. Haber, R. Del Toro, and V. Hernandez, "Self-adaptive systems: A survey of current approaches, research challenges and applications," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7267–7279, 2013.
[47] D. Weyns, *An introduction to self-adaptive systems: A contemporary software engineering perspective*. John Wiley & Sons, 2020.
[48] H. Hu, J. Liu, Y. Zhang, Y. Liu, X. Xu, and J. Tan, "Attack scenario reconstruction approach using attack graph and alert data mining," *Journal of Information Security and Applications*, vol. 54, p. 102522, 2020.
[49] T. G. Robertazzi, *Computer Networks and Systems: Queueing Theory and Performance Evaluation*. Springer Science & Business Media, 2000.
[50] K. L. Lim, J. Whitehead, D. Jia, and Z. Zheng, "State of data platforms for connected vehicles and infrastructures," *Comunication in Transportation Research*, vol. 1, 2021.
[51] M. Fox and D. Long, "Pddl2. 1: An extension to pddl for expressing temporal planning domains," *Journal of artificial intelligence research*, vol. 20, pp. 61–124, 2003.
[52] J. Hoffmann, "Extending ff to numerical state variables," in *ECAI*, vol. 2. Citeseer, 2002, pp. 571–575.
[53] A. Amro, V. Gkioulos, and S. Katsikas, "Assessing cyber risk in cyber-physical systems using the att&ck framework," *ACM Transactions on Privacy and Security*, vol. 26, no. 2, pp. 1–33, 2023.
[54] H. I. Kure, S. Islam, and M. A. Razzaque, "An integrated cyber security risk management approach for a cyber-physical system," *Applied Sciences*, vol. 8, no. 6, p. 898, 2018.