Smart Building Control System Emulation Platform for Security Testing

Xiaoqi Weng
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
s2310019@jaist.ac.jp

Razvan Beuran
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
https://orcid.org/0000-0002-4109-3763

***Abstract*—Smart buildings play a crucial role in advancing the smartness of cities, with technologies like automated control, smart sensors, and communication networks becoming increasingly complex. In addition, ensuring the effectiveness and reliability of these technologies requires continuous testing and improvement. However, evaluating them in real buildings is costly, risky, and resource constrained. Furthermore, as cyberattack techniques evolve and digital transformation accelerates, the security threats to smart buildings are also growing. To tackle these challenges, in this paper we introduce the Smart Building Control System Emulator (SBCSE). Developed based on real building log data, this platform emulates control systems, IoT devices, and communication protocols for thorough testing and evaluation purposes. SBCSE can also be used to analyze potential threats in various security scenarios, and to validate effective countermeasures. Consequently, our platform can improve testing efficiency and safety, reduce costs, and support system design and maintenance. Moreover, by simulating different network risk scenarios, it helps identify security threats and provides actionable solutions.**

***Index Terms*—smart building, control system, security testing, emulation, simulation, MQTT**

I. Introduction

As smart buildings evolve, the integration of Building Automation Systems (BAS) [1], Building Management Systems (BMS) [2], smart sensors, and IoT devices is becoming more complex. The effectiveness of these technologies depends on continuous testing, but real-world evaluations face challenges such as high costs, risks, and limited resources. Therefore, simulation tools provide a more flexible, cost-effective, and efficient alternative.

Several simulators have been developed, such as Open-SBS [3], a cross-platform open-source smart building simulator. However, it mainly focuses on smart home scenarios and does not fully address the integration of smart building operating systems with IoT devices. Other systems, such as HVAC simulations [4], primarily focus on energy management. We conclude that, despite these advancements, smart building simulators are still in the early stages, facing challenges in interoperability, real-time performance, and scalability with building operating systems and IoT devices.

Moreover, with the advancement of cyberattack techniques and the acceleration of digital transformation, smart buildings face numerous cybersecurity risks. To address these challenges, researchers have proposed an IoT security framework for smart infrastructures [5] and an ontology-based cybersecurity framework for IoT [6]. However, research specifically focused on cybersecurity in smart buildings remains relatively limited. Given the complexity and diversity of smart buildings, more comprehensive strategies are required, such as security testing and countermeasure analysis.

In this paper, we present SBCSE, which aims to address the cybersecurity challenges in smart buildings. SBCSE was designed and implemented as a platform that integrates management systems—focusing on network communication and device motion for IoT devices such as robots and elevators—with a security module that simulates attack scenarios. This makes it possible for users to assess the attack impact and validate countermeasures. Thus, we seek to advance smart building control systems and provide tools to ensure their security and reliability against evolving cybersecurity threats.

The remainder of this paper is organized as follows. In Section II, we introduce the architecture of SBCSE. Section III presents the evaluation and analysis of the experimental results from the SBCSE. In Section IV, we discuss the attack scenario design and preliminary experiments. The paper ends with the conclusion and references.

II. Proposed System

A. System Components

SBCSE was designed based on a smart building prototype from a certain construction company. Figure 1 shows the main components of the system: (i) Elevators (ELV); (ii) Building Operating System (BOS); (iii) Robot Platform (RPF); (iv) Robots (ROB); (v) MQTT-Broker. BOS represents the building "operating system" that facilitates the collaboration among building equipment, IoT devices, and various applications. RPF is a robot control platform that enables remote robot control and collaborative work among multiple robots. MQTT-Broker acts as a middleware agent enabling communication between different devices and services via the MQTT protocol.

2473-3105/24/$31.00 ©2024 IEEE
DOI 10.1109/PRDC63035.2024.00041

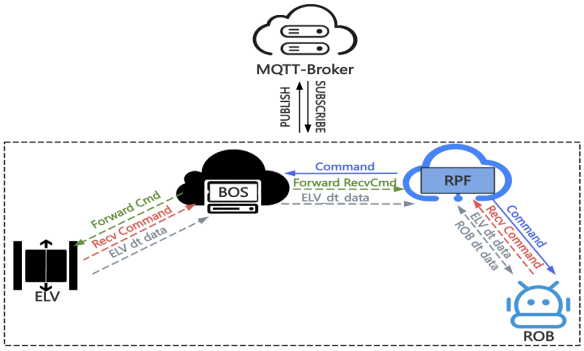

Fig. 1. Target smart building system components.

B. SBCSE Architecture

We designed and developed SBCSE based on the analysis of the communication log data of the real smart building components, and the motion trajectory log data of robots from that building. Figure 2 shows the architectural design of the platform, which includes: network communication module, device motion module, RPF control protocol module, utilities module, simulator modules, and security attack module.

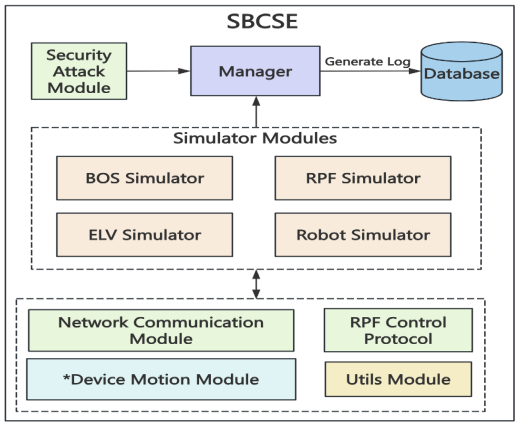

Fig. 2. Overview of the SBCSE architecture.

1) Network Communication Module: This module emulates the communication between building components. We adopted the MQTT protocol [7], which transmits messages between devices via a central broker. The MQTT-based communication module manages the protocol and processes data, and includes components for defining and managing communication logic, data models, and structures.

2) Device Motion Module: This module was mostly developed by partners to simulate IoT device movement, and it supports the creation and management of IoT device instances. The module is used to define the motion patterns and behaviors for robots and elevators, such as the routes for robots and the floor-to-floor movement for elevators.

3) RPF Control Protocol Module: This module is responsible for enacting the control protocol within the RPF component. Its implementation is based on automaton principles, and for each task to be completed a fixed communication sequence takes place.

4) Utilities (Utils) Module: This module includes general functions, such as log formatting and simulation time control, with time acceleration support added for improved efficiency.

5) Simulator Modules: The simulator modules integrate the functionality of the various modules mentioned so far to implement simulators for the different components, including BOS, RPF, elevators and robots.

6) Security Attack Module: This module is used to evaluate the security characteristics of the system components within SBCSE by simulating different cyberattack scenarios.

7) User Interface: As shown in Figure 3, we developed an intuitive and user-friendly graphical interface to enhance the user experience. This interface, built with Python and Kivy, is cross platform and high performance. Clicking on the "Start" button initiates the simulation; upon completion, detailed logs are displayed in the right panel for easy access.

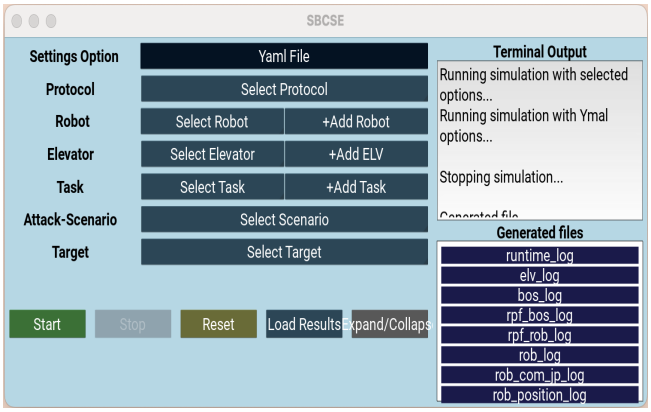

Fig. 3. SBCSE graphical user interface.

III. SBCSE Evaluation

A. Evaluation Method

Since SBCSE was developed based on real building logs, we evaluated it by analyzing its internal communication logs. For this purpose, we examined the communication flow of SBCSE to assess the overall process and inter-component communication, and compared its communication flow with that of the actual system. Our goal with this evaluation was to verify that the communication matches the actual system as closely as possible, proving SBCSE's accuracy and reliability in providing a simulation that is very close to the real system.

B. Performance Assessment

The results we obtained are presented next. First, regarding communication consistency, Figure 4 displays a comparison of the communication flows for the real building and SBCSE. The comparison confirms that the communication process of SBCSE matches that of the actual system, ensuring the accuracy of the emulation.

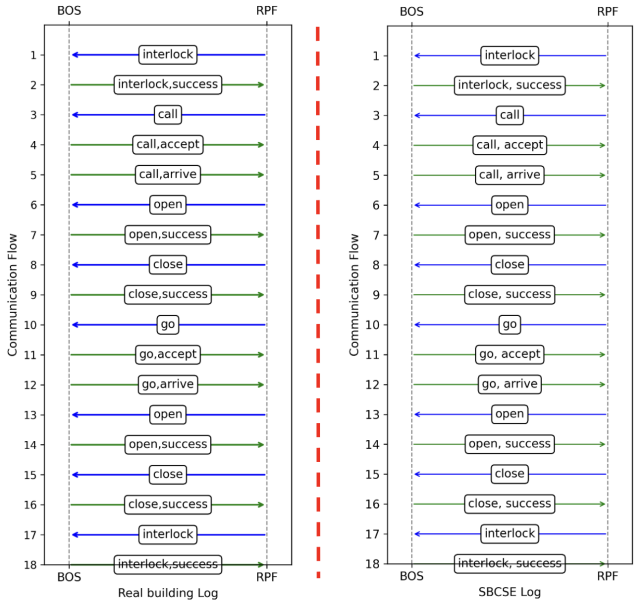

Fig. 4. Communication flow: real building log versus SBCSE log.

Due to the implementation constraints in the real system, the communication logs of the real building mainly involve the communication between the elevator, BOS, and RPF. However, SBCSE also includes in the communication module implementation the communication between the robot and RPF as an extension of the available logs, further enhancing SBCSE functionality. Figure 5 shows the entire communication flow according to SBCSE logs.

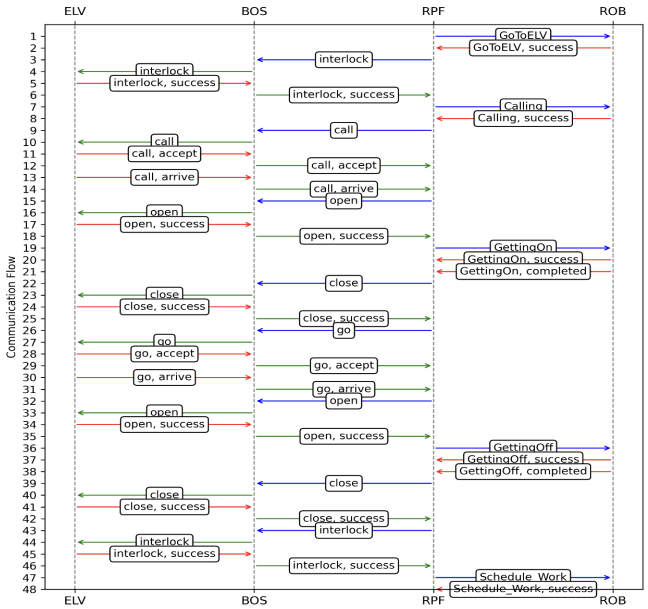

Fig. 5. Entire communication flow according to SBCSE logs.

Based on the communication flow shown in the figure, we confirmed that we were able to successfully reproduce the communication within the real system. This demonstrates that SBCSE is reliable for conducting experiments in conditions similar to real environments, and validates its practical utility.

IV. Attack Scenarios

A. Risk Analysis

In order to support the implementation of the attack scenarios, we first conducted a comprehensive risk analysis:

- Step 1: Component Analysis – Analyzed the functionality of each component and the data they store and process.
- Step 2: Categorizing Risks – Categorized the components using labels to facilitate subsequent risk classification and management.
- Step 3: Attack Surface for Components – Assessed the components in smart buildings based on reference standards [8]–[10] using the labels assigned at Step 2 to determine whether they have potential risks.
- Step 4: Threat Analysis and Scenario Design – Analyzed the potential impacts and associated threats based on the risk classifications identified in Step 3. Through this analysis, we identified five scenarios that are of high significance for the system components:
  1) Attacks on information transmitted over the network (man-in-the-middle attack)
  2) Broken Access Control (Unauthorized Access)
  3) Distributed Denial-of-Service (DDoS) attacks using IoT botnets
  4) Attacks involving malware
  5) Attacks based on human error and social engineering techniques

B. Attack Scenario Design

This section introduces the prototype man-in-the-middle (MITM) attack scenario designed for our system. The MITM attack [11] typically involves an attacker secretly intercepting and manipulating communication between two parties. The components in our system that are most vulnerable to eavesdropping are RPF and BOS. An attacker typically follows the next steps to execute the MITM attack:

- Step 1: Information Gathering – Collect information about the target systems (BOS/RPF) and the communication protocols used by the broker to identify potential vulnerabilities.
- Step 2: Initial Positioning – Position the device controlled by the attacker within the communication path between the BOS/RPF and the MQTT broker.
- Step 3: Interception – Intercept and capture the communication between the BOS/RPF and the broker.
- Step 4: Manipulation – Manipulate the intercepted communication to alter the behavior of the elevator and robot.
- Step 5: Exfiltration – Extract sensitive data from the intercepted communication.
- Step 6: Clearing Traces – Remove all malicious software and clean system logs to hide the evidence of the attack.

C. Preliminary Experiments

This section presents preliminary experiments with an MITM attack aimed at BOS. For this purpose, we added an MITM attack module in SBCSE. However, to minimize execution risks, we skipped initial steps and assumed successful scanning and eavesdropping (steps 1 through 3 above).

As shown in Figure 6, the MITM attack scenario involves RPF sending an "Open" command to the elevator through BOS. The attacker intercepts this communication, preventing BOS from forwarding the "Open" command to ELV, and fakes an "Open success" message to RPF. This causes RPF to mistakenly send a "GettingOn" command to the robot, leading the robot to try to enter the elevator even if the door is closed.

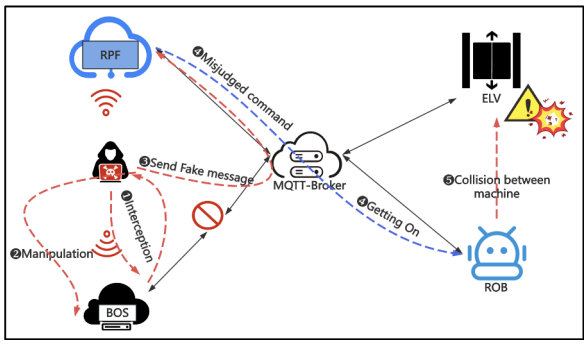

Fig. 6. Summary of the MITM attack scenario.

Figure 7 shows the log analysis of a successful MITM attack, where the robot fails to enter the elevator. To prevent a collision and physical damage, we considered that the simulated robot is equipped with a camera for detecting the door status, which enables it to report the failure and its reason.

As security countermeasure, we considered encrypting the communication channel, and implemented support for the MQTTS protocol with encryption to prevent MITM attacks. The implementation of further countermeasures is ongoing.

V. Conclusion

This study addressed the shortcomings of current smart building control system simulation platforms. By developing SBCSE, we provided a tool to test and evaluate the performance and security of smart building systems. The platform offers a user-friendly interface, and a security module that can simulate attack scenarios, helping to identify potential vulnerabilities and propose effective security measures.

Overall, this study provides a valuable tool for the smart building research field, and lays the foundation for future research and development. First of all, we will focus on further improving the experiment support aspects to address some of the current limitations. For example, SBCSE currently focuses on normal communication and lacks handling for packet loss, timeouts, and errors.

In addition, expanding the SBCSE functions to address evolving cybersecurity threats is also important. Thus, while

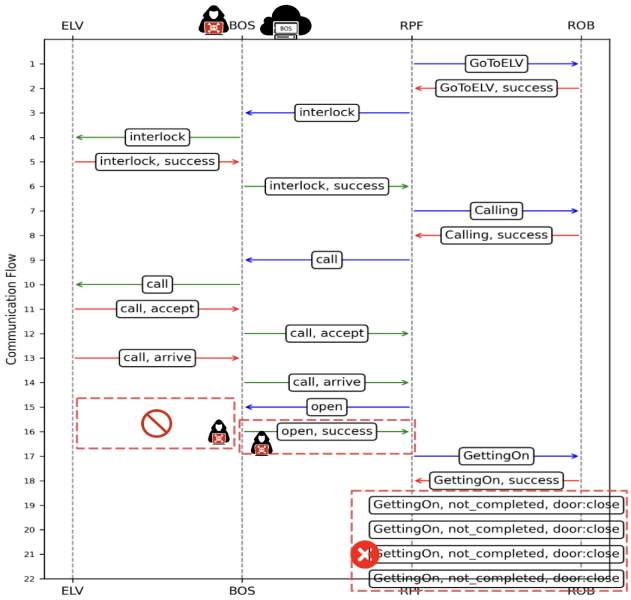

Fig. 7. MITM experimental log.

the current MITM attack scenario illustrates possible security challenges in smart buildings, many other security issues need to be addressed. Moreover, the scope of smart building control systems is broad; future research could simulate multiple robots, add other IoT devices, energy systems, etc.

References

[1] W. Kastner, G. Neugschwandtner, S. Soucek, and H. M. Newman, "Communication systems for building automation and control," Proc. IEEE, vol. 93, no. 6, pp. 1178-1203, 2005.

[2] S. Wang and J. Xie, "Integrating Building Management System and facilities management on the internet," Automation in Construction, vol. 11, no. 6, pp. 707–715, 2002.

[3] H. E. Degha, F. Z. Laallam, O. Kazar, I. Khelfaoui, B. Athamena, and Z. Houhamdi, "Open-SBS: Smart Building Simulator," in Proc. 2022 Int. Arab Conf. Inf. Technol. (ACIT), pp. 1-15,2022

[4] R. Zhang and T. Hong, "Modeling of HVAC operational faults in building performance simulation," Appl. Energy, vol. 202, pp. 178-188, Sep. 2017.

[5] J. Pacheco and S. Hariri, "IoT security framework for smart cyber infrastructures," in Proc. 2016 IEEE 1st Int. Workshops on Found. Appl. Self Syst. (FASW), pp. 242-247, Sep. 2016

[6] B. A. Mozzaquatro et al., "An ontology-based cybersecurity framework for the internet of things," Sensors, vol. 18, no. 9, p. 3053, Sep. 2018.

[7] M. B. Yassein, M. Q. Shatnawi, S. Aljwarneh, and R. Al-Hatmi, "Internet of Things: Survey and open issues of MQTT protocol," in Proc. 2017 Int. Conf. Eng. & MIS (ICEMIS), pp. 1-6, May 2017

[8] European Union Agency for Cybersecurity, "Good Practices for Security of Internet of Things in the context of Smart Manufacturing," enisa.europa.eu, 2018.

[9] Ministry of Economy, Trade and Industry (METI), "Cyber-Physical Security Measures Guideline for Building Systems,", 2023. Available: https://www.meti.go.jp/policy/netsecurity/wg1/bill_gideline_2.pdf. [Accessed: Aug. 26, 2024].

[10] OWASP, "OWASP Top 10 - 2021," Available: https://owasp.org/Top10/. [Accessed: Aug. 26, 2024].

[11] M. Conti, N. Dragoni, and V. Lesyk, "A survey of man in the middle attacks," IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2027-2051, 2016.