POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

CreScan: A Rule-Based Security Auditing and Remediation Tool for Crestron Smart Home Systems

Supervisors

Prof. Fulvio CORNO

Dr. Luca MANNELLA

Candidate

Federico SCHIAVOLINI

Academic Year 2024/2025

CreScan: A Rule-Based Security Auditing and Remediation Tool for Crestron Smart Home Systems

Federico Schiavolini

Abstract

As the increasing adoption of Crestron home automation systems in smart environments continues, the strong cybersecurity of such installations has become a top concern. The problem of this concern is addressed by this master's thesis by proposing and developing a new tool that is intended to provide automated security scans for installation professionals of Crestron-based smart home systems. The suggested tool conducts an automatic audit, inspecting the configuration settings, network design, and communication protocols adopted within the Crestron setup.

It does this through an approach that employs a rule-based solution that infuses different cybersecurity standards and regulations applicable in a smart home environment. It scans the installation against these pre-configured rules, flagging possible security gaps and vulnerabilities. The tool further categorizes the severity of each vulnerability, providing technicians with a prioritized list of issues to tackle.

The output of the tool is an organized and detailed report that presents findings and recommendations to enhance security within the Crestron home automation system. The report is presented in simple language that is easily understandable by technical and non-technical stakeholders, facilitating successful communication of security vulnerabilities and proposed actions to counter them.

Additionally, the tool offers corrective action recommendations, suggesting best practices and configurations to neutralize detected vulnerabilities. This proactive approach not only secures the Crestron installation but also educates technicians on security best practices, making the smart home system more secure and resilient.

This tool is a significant contribution to cybersecurity engineering, providing a viable solution to enhancing the security of Crestron-based smart home installations. While the IoT appears to be growing more pervasive, this study will focus on reinforcing secure smart environment foundations, encouraging the implementation of best cybersecurity practices in installing and even upgrading Crestron automation systems for residences.

Table of Contents

List of Figures

List of Tables

Acronyms

AD Active Directory.

AES Advanced Encryption Standard.

AMSN Agence Monégasque de Sécurité Numérique.

ANSSI Agence Nationale de la Sécurité des Systèmes d'Information.

ARP Address Resolution Protocol.

BACnet Building Automation and Control Network.

BAS Building Automation System.

BEAST Browser Exploit Against SSL/TLS.

CA Certification Authority.

CIP Common Industrial Protocol.

CISO Chief Information Security Officer.

COV Change of Value.

CVE Common Vulnerabilities and Exposures.

CSS Cascading Style Sheets.

CTP Crestron Toolbox Protocol.

DDoS Distributed Denial of Service.

DoS Denial of Service.

ICMP Internet Control Message Protocol.

IDS Internet Detection System.

IoT Internet of Things.

IP Internet Protocol.

IPS Internet Prevention System.

IPsec Internet Protocol Security.

IPv4 Internet Protocol Version 4.

IPv6 Internet Protocol Version 6.

HSM Hardware Security Module.

HTML HyperText Markup Language.

HTTP HyperText Transfer Protocol.

HTTPS HyperText Transfer Protocol Secure.

HVAC Heating, Ventilation, and Air Conditioning.

MAC Media Access Control.

MITM Men In The Middle.

NTP Network Time Protocol.

OS Operating System.

PASSI Prestataire d'Audit de la Sécurité des Systèmes d'Information.

PoE Power over Ethernet.

POODLE Padding Oracle On Downgraded Legacy Encryption.

PKI Public Key Infrastructure.

RMS myCrestron Residential Monitoring Service.

SNTP Simple Network Time Protocol.

SSH Secure Shell.

SSL Secure Sockets Layer.

TLS Transport Layer Security.

TPM Trusted Platform Module.

VLAN Virtual Local Area Network.

YABE Yet Another BACnet Explorer.

Chapter 1

Introduction

This chapter introduces the context, motivations, and challenges that frame the research work presented in this thesis. Section 1.1 analyses the evolution and diffusion of smart home technologies, particularly focusing on the growing complexity and security concerns brought by the integration of IoT systems into residential environments. Section 1.2 outlines the concrete cybersecurity risks affecting smart homes, while section 1.3 define the goals of this study—namely, the development of a tool to assess and improve the security of Crestron-based installations. This foundation establishes the relevance and necessity of the work within the broader scope of cybersecurity and smart environments.

1.1 Context and Motivation

Smart home automation systems are advancing due to a necessity for better comfort, safety, and energy efficiency within our daily lives. With the proliferation of the IoT, smart homes operate a variety of devices - cameras, door locks, lighting systems, and environmental sensors - that are collectively aimed at delivering a simple, intelligent lifestyle. The rapid integration of technology brings serious security risks that we must face to protect these systems from attacks by malicious agents.

In the past, the early implementation of smart home technology dealt with serious problems, like complicated setup procedures, high prices, and absent standardization, which relegated their use to only technology hobbyists. Throughout the years, developments in technology, especially processing power and communication protocols, have dropped costs and broadened the functional capabilities of smart home systems. This contributed to their greater ease of access for the layperson. Modern smart homes are capable of executing many functions, like tracking energy consumption, overseeing home appliances, managing home security, and automating common tasks remotely using either mobile applications or cloud-based platforms [1, 2].

These improvements notwithstanding, security is still one of the most pressing challenges in smart home systems. The interconnected elements of smart home gadgets boost the attack surface, enabling attackers to access the full system through flaws found in just one device. These systems generally encompass sensitive personal

information, such as routines performed every day, monitor feeds from security cameras, access codes for door locks, and also financial and health data. After a breach, attackers can manipulate these devices from afar, resulting in either privacy violations, financial losses, or even dangers to physical health [3].

The protection offered to communication protocols used by IoT devices is a significant security issue in smart homes. Devices that are smart commonly communicate over wireless technologies such as WiFi, Bluetooth, ZigBee, or Z-Wave, and vulnerable protocols might allow attackers to bypass defenses and enter for data interception or control of devices. In IoT systems, replay attacks and man-in-the-middle attacks tend to happen commonly due to the application of weak encryption or authentication schemes [2]. In addition, as a growing number of smart home systems connect to the internet, the danger of remote attacks escalates, allowing potential attackers to target residences from almost anywhere worldwide [4].

A further serious challenge is the bias toward smart home applications, under which apps receive substantial permissions that go beyond their required functionality. This overvaluing permits avenues for unauthorized entry into important systems. As an illustration, an application that looks benign but actually monitors battery levels in smart devices could gain control over more sensitive systems, including door locks and security alarms, because of ineffective permission handling in the smart home framework [5, 2]. This kind of vulnerability could enable an attacker to cut off the security systems, unlock entryways, or perhaps create phony alarms, affecting the safety of the residence.

Through a user lens, the difficult task of managing these systems introduces probable risks. A large fraction of homeowners lacks the technical understanding crucial for the correct configuration and security of their smart home devices. The exploitation of weaknesses from misconfigurations, poor passwords, or antiquated software by attackers becomes much easier. Differential to corporate networks, smart home systems generally lack system administrators who are dedicated to the continuous monitoring and updating of the network for security threats [2]. Such results mean that attackers usually experience an easier time targeting these environments, which are less apt to be properly secured.

Due to these challenges, there is a surging need for complete security solutions designed especially for smart home scenarios. Experts in the field and developers are thoroughly examining numerous technologies to strengthen security, such as blockchain for device authentication that is decentralized, fog and edge computing for processing data locally, and machine learning to detect unusual behavior in smart homes [4, 6, 7]. The goal of these technologies is to supply more reliable, scalable, and efficient frameworks for handling the enormous volume of data produced by IoT devices, all while reducing security breach risks.

As smart home system adoption increases rapidly, securing them will be fundamental to preserving user confidence and protecting personal privacy.

1.2 Problem statement

The combination of IoT devices within smart homes raises a variety of security issues that call for timely focus [8, 9, 10]. As the number of home network devices—such as cameras, smart locks, thermostats, and even kitchen appliances—increases, the chances of cyberattacks have escalated considerably [2]. All devices, whether they function for home security, energy management, or entertainment use, add a new weakness point. Due to the diverse characteristics of these devices, which communicate using protocols such as Wi-Fi, Bluetooth, and Zigbee, they create a broad and complex attack surface. If the security of communication protocols is inadequate, they may become a vector for harmful actors to exploit, potentially putting the entire smart home system in danger [11, 12].

A notable issue is that many IoT devices, which are part of smart homes, have insufficient security capabilities. Designed to cut down costs, such devices usually lack enough computational power to easily integrate powerful security measures, such as advanced encryption and ongoing monitoring. As a consequence, smart home systems are often at risk of unauthorized entry, data robbery, and system interference. Ineffective security protocols that allow for default or weak passwords, untimely firmware, and uncorrected vulnerabilities enable attackers to exploit them. Inside the system, attackers might seize control of important functions, for example, unlocking doors, disabling or accessing security cameras without authorization, or making changes to environmental settings, all of which can happen without the homeowner realizing it [11].

It's not just theory; cyberattacks targeting smart homes are occurring on a daily basis [11]. As a single example, DDoS attacks can overwhelm a smart home network with data, consequently limiting the use of devices and disrupting significant services. Still another usual threat is the MITM attack, which takes place when an attacker captures and alters the communication flow between two distinct devices. In the case of this kind of attack, a hacker might change the instructions meant for devices, possibly disabling alarms or monitoring security cameras, all without the homeowner noticing. Also, malware can enter the system via compromised devices, spreading across the network and allowing the attacker complete control [12].

Complicating the scenario is a shortage of user awareness, along with the inconsistent security protocols across diverse devices. A large number of users do not see the critical importance of securing their smart home devices, often neglecting to update firmware or to change their default security settings. Not having an abundance of proactive security approaches makes smart homes markedly more vulnerable to hostile attacks. The situation deteriorates because various IoT manufacturers do not emphasize security in their design strategies, concentrating instead on ways to cut costs and improve user convenience. As a result, a lot of smart devices feature inadequate security features straight from the factory [12, 13].

To address these dangers, manufacturers need to prioritize the implementation of powerful security measures and develop equipment capable of performing more

advanced security tasks, all within the tight confines of an IoT environment. including advanced encryption technology, guaranteeing that firmware updates occur automatically, and improving access control systems. Also, smart home users ought to be more informed about the threats of connected devices and should take initiative to secure their home networks by changing the default passwords, turning on two-factor authentication, and routinely updating software [11, 12].

Also, industry standardization is important. Given that so many manufacturers are producing IoT devices, each with its unique protocols and security procedures, developing a consistent security framework is quite hard. Using a unified security standards approach similar to standard IT infrastructures would probably reduce vulnerabilities and create more reliable security against cyber threats [12].

In the end, the fast incorporation of IoT solutions in intelligent households gives rise to many benefits, but it also presents substantial security risks. The extensive attack surface, along with the inferior security functions of many devices, results in a vulnerable ecosystem that needs immediate engagement from both users and manufacturers. Implementing more robust security features and increasing user understanding can markedly decrease risks, which allows smart homes to be both safe and secure and to respect user privacy [12, 13].

1.3 Research objectives

The goals of this thesis are two in number. The primary focus is on researching and analyzing the security complications caused by bringing together IoT devices in smart homes, focusing on the recognition of vulnerabilities and threat vectors that threaten both user privacy and user safety. The objective of this thesis also encompasses creating a specialized audit tool to measure the cybersecurity of Crestron home automation systems. The tool will conduct a full analysis of the security status of a Crestron installation, allowing technicians responsible for system installation to reveal weaknesses, comply with industry best practices, and elevate the overall security of the system. The thesis provides both theoretical insights and practical application of an audit tool to better security standards and deliver practical insights for developers, integrators, and technicians involved in smart home system implementation.

This research will only consider Crestron smart home systems, with a specific aim of assessing and improving their cybersecurity capabilities. The findings and methodologies developed within this thesis will be adapted to the special considerations of Crestron systems, yet they may also give important clarity to the larger smart home ecosystem field. The findings related to vulnerabilities and the security methodologies suggested could theoretically apply to different smart home platforms, giving rise to a standard for improving security in multiple IoT situations. The immediate relevance of these results to various systems might need further modification and confirmation.

1.4 Document Structure

The rest of this thesis is structured as follows.

- **Chapter 2** presents the technological and theoretical background of the research, offering an overview of smart home cybersecurity challenges and the architectural elements of Crestron automation systems. It also introduces regulatory frameworks and relevant technologies such as BACnet and Cresnet.
- **Chapter 3** outlines the methodology adopted for designing the proposed tool. It describes the rationale behind the rule-based approach, the modular structure of the system, the data acquisition strategies, and the logic governing the automated reporting and remediation process.
- **Chapter 4** details the design and implementation of CreScan, focusing on test scenarios, rule definitions, vulnerability checks, and the corresponding security domains. It also includes performance considerations and validation strategies.
- **Chapter 5** presents the evaluation metrics and performance results of the implemented system, highlighting the effectiveness of the auditing process and the practical impact of the autofix capabilities.
- **Chapter 6** discusses the results obtained, linking them to the initial objectives and interpreting them in the context of real-world deployments.
- **Chapter 7** concludes the work, summarizing the main contributions, limitations, and future directions for research and development.

Chapter 2

Background

This chapter provides the theoretical and technological foundations required to understand the research work presented in this thesis. It begins in section 2.1 with an overview of cybersecurity aspects in smart home environments, focusing on how interconnected IoT devices introduce new vulnerabilities and risks. Section 2.2 then describes the Crestron home automation ecosystem, illustrating its architecture, main components, and communication principles that enable integrated control within residential environments. The discussion continues in section 2.3, which examines existing security challenges and currently available mitigation strategies. Subsequently, section 2.4 presents the regulatory frameworks and standards that guide the secure deployment of smart home technologies. Finally, section 2.5 introduces the Crestron Security Audit Tool, a reference implementation developed to assess and enhance the security posture of Crestron-based installations.

2.1 Cybersecurity in Smart Home Environments

The nature of IoT devices, intricacy in their ecosystems, and usually low security measures across the network make cybersecurity a complicated challenge in smart home environments.

These range from simple motion sensors and cameras to connected locks and thermostats in today's smart home that improve user comfort and control. However, while these devices bring innovations and conveniences, they also introduce new risks and amplify the existing vulnerabilities. Most IoT devices can barely adopt strong security measures due to their limited processing power and resources. Besides that, due to the fact that these are mass-produced consumer products, devices are seldom updated on a regular basis to deal with newly occurring cyber threats. The lack of updates, combined with rapid and often cost-driven development, introduces vulnerabilities that malicious actors can exploit. The most frequent ones are hardcoded passwords, poor encryption protocols, and insecure channels for communication that can lead to system compromise and user privacy breach [10, 14]. Most of the IoT devices are heterogeneous in nature and use a wide variety of communication protocols such as Zigbee, V-Wave, and Wi-Fi. This acts as a big challenge for

implementing the same security practice throughout. Devices often communicate in an insecure manner, while certain uniform security standards are lacking. Many IoT applications associated with smart homes lack mutual authentication or end-to-end encryption, which provides scope for intercepting data during transmission. This is further exacerbated by vendors developing devices with rapid development and cost efficiency in mind rather than security, thus leaving devices open to vulnerabilities and relying on the end user for the cybersecurity aspect [3, 4].

A good example is the SmartThings framework, which, through design flaws, leads to "overprivilege," including installed applications that have been granted more access than what is actually needed. This would, therefore, imply that even an apparently benign application could access sensitive data such as door lock codes without the user's knowledge and exploit the excess privileges it has been granted. Such a kind of security vulnerability is very insidious since third-party applications may be allowed to perform certain actions not necessarily requested by the user, and thus increasing the chances of security breaches [1]. Some of the future solutions to these security challenges also involve technologies around edge computing, fog computing, and blockchain. Fog and edge computing, in particular, are effective within smart homes, in that this technology provides the ability to process data locally within or nearby the devices themselves, thus further reducing the dependency on any central server, which is where most attackers usually target. This better improve the detection and mitigation of threats in real time.

These decentralized approaches allow for real-time threat detection and response right at the device, which is critical in resource-limited IoT deployments. Another promising direction is the use of machine learning algorithms to monitor device activity for anything unusual, which could help identify patterns that may indicate compromise or indicate DDoS attack vectors, limiting potential damage well in advance [14, 15, 16].

Similarly, blockchain technology is being integrated into the modes of operation to develop the integrity and authenticity of data in various smart home systems. Other smart home systems apply decentralized data storage and impose trust to prevent tampering and unauthorized access. However, blockchain remains an emerging area of research since it is resource-intensive and hence hardly scalable across IoT environments.

But at the same time, one of the most important roles in securing the smart home falls to the user. Poor configurations or lack of awareness of best practices—such as changing default passwords or application restrictions—can turn into security issues. Because smart home devices often collect sensitive personal data, users seek a balance between convenience and security, though this is sometimes not accompanied by guidance from the manufacturer on clear guidelines or easily configurable security tools [5, 10]. Conversely, while smart homes offer numerous advantages, they demand heightened cybersecurity awareness. Securing smart home environments calls for an integrated approach encompassing safer design practices, advanced protection frameworks, user education, and emerging technologies like edge computing and

machine learning to counter increasingly sophisticated and persistent threats [17].

2.2 Crestron Home Automation Systems

With its comprehensive capabilities, the Crestron Home system commands several home technologies like audio/video and climate management. The system delivers a customizable workspace that facilitates easy monitoring using voice commands. This system allows coupling with external devices to provide versatility for a wide range of installations. Crestron Home's easy-to-use interface improves both usability and safety so homeowners can customize the system to their individual requirements.

Crestron Home OS is a sophisticated, scalable home automation system that provides centralized control over a wide array of connected devices in smart homes. The core of the Crestron Home OS consists of control processors, such as the CP4-R, MC4-R, and DIN-AP4-R, each of which is optimized for homes of various sizes and levels of complexity. These processors manage communication and automation between all integrated devices within a home, allowing for real-time control and monitoring of everything from lighting and climate systems to security cameras and multimedia entertainment setups.

The system operates through a combination of wireless and wired protocols, including Bluetooth, Wi-Fi, ZigBee, and proprietary Crestron protocols like Cresnet and infiNET EX. This flexibility allows the Crestron Home OS to integrate a wide variety of devices, including those from third-party manufacturers, such as Sonos for audio systems, Lutron for lighting and shading, and Honeywell for security systems. The processors serve as the central hub for device communication, ensuring seamless operation of all connected systems.

Crestron Home OS provides two primary user interfaces: the Crestron Home App and the Crestron Home Setup App. The Home App is designed for end-users to control home devices through a unified interface available on mobile devices, touchscreens, and computers. This app allows users to manage lighting, audio, climate control, security, and more from a single point. The Setup App is intended for technicians to configure and set up the system, streamlining the installation process and enabling quick adjustments onsite or remotely.

Security is a central concern in the Crestron Home ecosystem, and the system includes features such as secure device pairing, user authentication, and encrypted communication. Technicians can remotely monitor systems using the RMS, which offers diagnostic tools, firmware updates, and logs that help ensure the smooth operation of Crestron installations [18].

2.2.1 BACnet

BACnet is a peer-to-peer, object-oriented protocol that manages the communication between devices within a BAS. Released as an ASHRAE/ANSI standard in 1995 and later ISO-certified, BACnet allows devices to share information and execute

commands across a network, enabling automation [19]. However, the protocol was not originally designed with robust security in mind, as many early implementations were on isolated networks, which has led to significant security concerns as systems become more interconnected [20].

Within the Crestron Home OS, BACnet plays a crucial role in facilitating the integration of third-party HVAC systems with the smart home automation platform. The protocol allows for standardized communication between devices from different manufacturers, ensuring that complex systems like thermostats, sensors, and other climate control devices can be easily managed through the Crestron system.

Crestron Home OS supports BACnet through its control processors, which act as intermediaries between the smart home network and BACnet-enabled devices. These processors can communicate with BACnet objects, allowing for control of heating and cooling setpoints, fan modes, and even advanced functions like humidity control. The system can interact with BACnet devices using features like subscribing to COV notifications and setting object priorities, scan rates, and decimal places to fine-tune communication with the BACnet objects.

BACnet over IP allows Crestron to control and receive feedback from BACnet-enabled devices through the network. For example, the system can integrate with thermostats using this protocol, providing both manual control and automation features. A BACnet explorer, such as the YABE, is often used during setup to navigate the BACnet object properties and assign the appropriate settings within the Crestron Home interface.

This capability makes Crestron Home OS a flexible solution for environments that require complex HVAC integrations, allowing for seamless communication between third-party building control devices and the smart home ecosystem.

2.2.2 Cresnet

The Cresnet system [1] is a specialized communications protocol developed by Crestron for use in smart home and building automation environments. It is designed to handle communication between Crestron devices that do not require the high speeds of Ethernet, while still allowing robust data and power distribution over a single network.

- **Cresnet Control Systems**: The central processors in the system (such as Crestron's CP4 or MC4 units) manage communication with Cresnet client devices. These processors, also referred to as Cresnet servers, provide both power and data to the connected devices via Cresnet ports over a shared bus network. Cresnet control systems supports up to 25 client devices on a single network using four wires: two for power and two for data.
- **Cresnet Servers**: Cresnet servers distribute 24 VDC power and manage data transmission between devices on the network. They control all bidirectional

[1] `https://docs.crestron.com/en-us/9272/Content/Topics/Home.htm`, last visited on 13 October 2025.

communication, ensuring seamless interactions between the server and the client devices. The Cresnet bus handles both power and data distribution, allowing up to 25 client devices to function efficiently within the same network architecture.

- **Cresnet Clients**: These devices are connected to the network to receive both power and data from the Cresnet server. Cresnet clients include keypads, sensors, lighting controls, shade motors, thermostats, and occupancy sensors. Each client device is assigned a unique Cresnet ID, enabling precise communication and control. The four-wire system used by Cresnet ensures that all client devices are powered and can communicate effectively over the shared bus network.

The Cresnet protocol supports several network topologies:

- **Home-Run Topology**: In this configuration, each client device (e.g., lighting controls, thermostats, or sensors) is connected directly to the Cresnet server through individual, dedicated cabling runs. This type of topology offers a high level of control and straightforward management because each device has its own independent connection to the server. It simplifies troubleshooting, as faults can be isolated to a specific run without affecting other devices. However, it can require a significant amount of cabling, especially in larger installations, as each device must be wired separately to the server, leading to potentially higher installation costs and increased complexity in large systems.

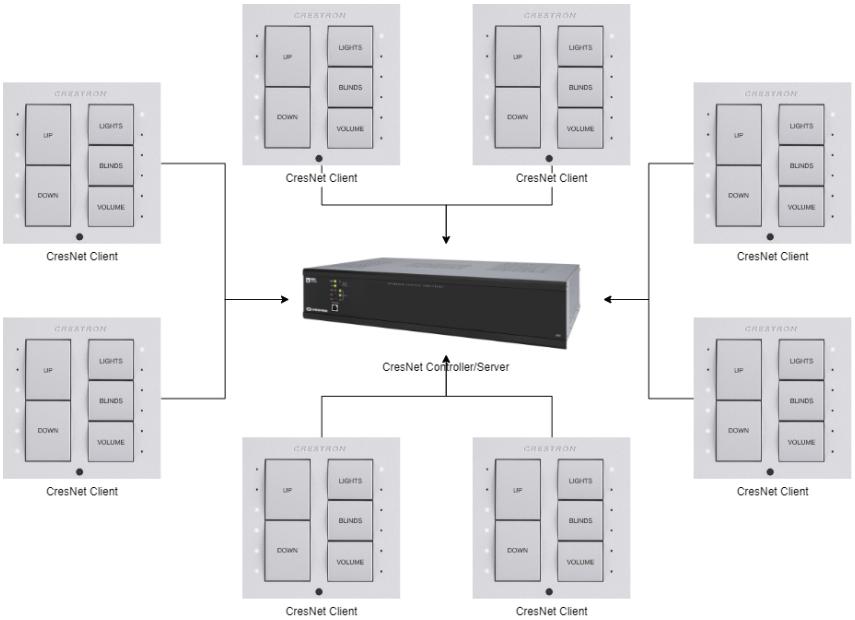

Figure 2.1: Home-Run Topology

- **Daisy-Chain Topology**: In this design, devices are connected in series, with the cabling running from the Cresnet server to the first device, then from that device to the next, and so on, forming a chain. This topology reduces the

amount of cabling needed compared to the Home-Run approach, as fewer cables are required to connect multiple devices. However, a major drawback is that if one device or connection in the chain fails, it could disrupt communication for all devices downstream of the failure point. Additionally, diagnosing issues becomes more complex, as the fault could originate from any device in the chain.

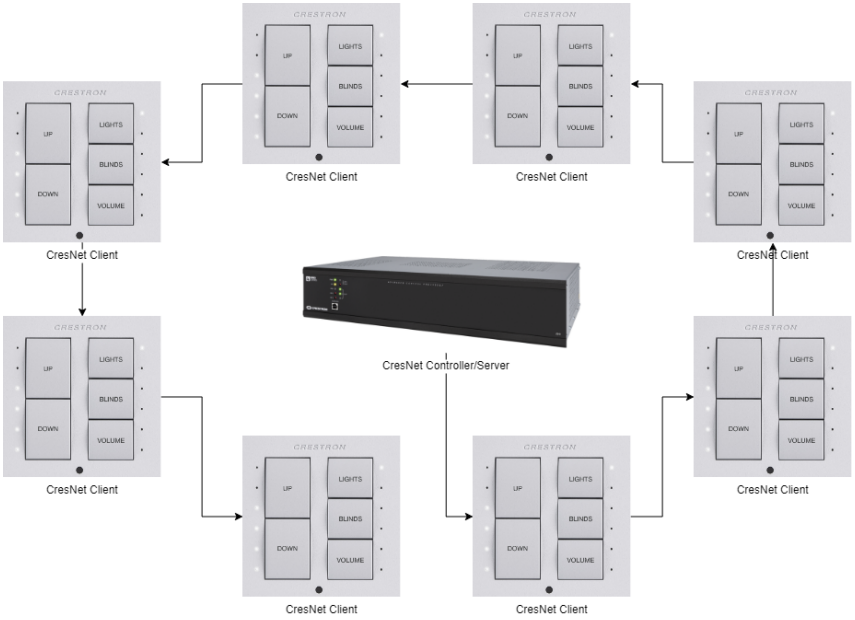

Figure 2.2: Daisy-Chain Topology

- **Star-Network Topology**: In this topology, a central Cresnet hub connects to multiple client devices or even sub-networks of devices, distributing power and data from the hub to various locations. The devices are connected independently to the hub, much like spokes on a wheel. This approach is ideal for larger or more complex systems where devices are spread across different rooms or areas, as it allows for more centralized management. It also provides a level of fault isolation similar to the Home-Run topology, where issues in one connection do not affect other parts of the network. While it can require a more complex setup with a central hub, it strikes a balance between cabling efficiency and fault tolerance.

Each device on the network must be assigned a unique Cresnet ID, which functions similarly to an IP address in networked systems. Cresnet IDs can be assigned using the Crestron Toolbox software, ensuring no two devices share the same ID within the same network segment. Devices are typically assigned IDs in groups based on their function.

The Cresnet network can be expanded through the use of Cresnet Hubs and Ethernet-to-Cresnet Bridges. A single hub can increase the number of devices and the overall cabling length of the network. For instance, a Cresnet hub can support up to 25 additional client devices per segment, expanding the network to as many as 90

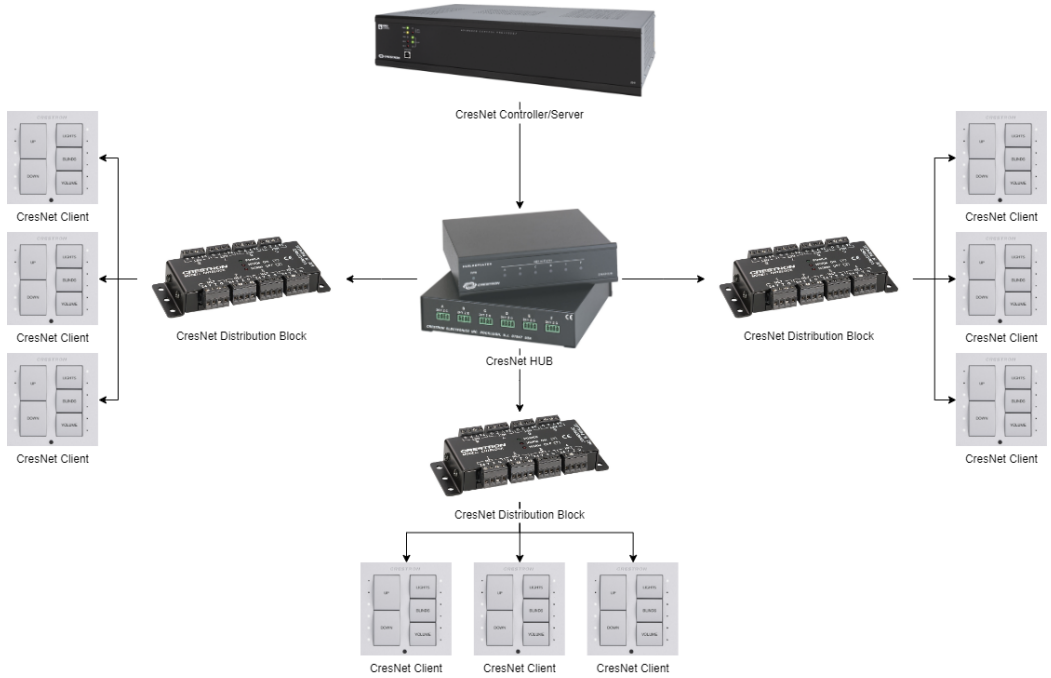

Figure 2.3: Star-Network Topology

devices. The Ethernet-to-Cresnet bridges are used when connecting Cresnet devices over long distances or when the network needs to be integrated into an existing Ethernet infrastructure. Power Distribution and Requirements

One of the defining features of Cresnet is its ability to supply both power and data over the same wiring. A typical Cresnet cable includes two wires for power and two wires for data. Crestron provides various tools, such as the Cresnet Power Calculator, which can help determine the total power consumption of all connected devices. This is important because exceeding power limits can cause devices to malfunction.

Within Crestron Home OS, there are several types of passwords that control different levels of access and functionality within the system. The Admin Password is the most critical password, granting full access to the setup and configuration of the Crestron Home processor. The password can be installed by a dealer during the installation process or when a reset is implemented; it allows the user to add devices, remove them, change some of the settings in the system, and arrange accounts for other users. It is really important for the paperwork and management of an entire system.

The other password would be the Advanced User Password, which gives holders more advanced options in configuring the setup without necessarily having full privileges like that of access through Admin. This password would be applied by the homeowners or any other users desiring more access and, at the same time, control over the system, but not wanting to possess total administrative access. It offers a more controlled means of managing the system compared to the Admin password.

The password of the User Interface Device refers to the user interface devices, which are the Crestron touch screens, handheld remotes, and mobile devices. There-

fore, only the authorized user interface device has access for control of the system. Any modifications to the password are then synchronized among all those devices that are connected to provide an authentic level of access across the system.

Additionally, it authenticates any of the various devices needing secure access within the Crestron Home system through the Common Device Password. It is a device password, common between devices using authentication, so that new devices can then get added into the system and secured. Upon updating the Common Device Password, all devices using it immediately begin re-authentication to automate multi-device security maintenance.

Each one of these passwords serves an important function in system protection, access rights management, and keeping the system from being used by others who should not be. Rules for password usage—like the minimum character number—have been enhanced in the recent updates in order to improve security.

2.3 Existing Security Challenges and Solutions

The term IoT today encompasses a wide variety of devices, ranging from pacemakers to industrial systems, as well as consumer and professional smart home equipment. Such devices present risks that can be perceived as new due to their ability to produce "physical" effects outside information systems or systemic effects resulting from massive deployments. They are also subject to traditional cybersecurity threats, which can be exacerbated by the limited resources available to ensure their security or by the difficulties of maintaining unmanaged systems.

The security of connected systems cannot be addressed through a single, one-size-fits-all approach, since devices differ in architecture, communication protocols, and operational contexts [1]. This heterogeneity necessitates a context-aware methodology, beginning with a detailed security analysis that identifies potential attack vectors, assets to protect, and the security functions that must be enforced. Such an analysis ensures that general security recommendations can be meaningfully adapted to each specific scenario.

The goal of this section is therefore twofold: on the one hand, to facilitate the conduct of a security analysis by listing relevant potential threats; on the other hand, to propose technical recommendations to address these threats whenever it can be done generically.

The scope of interest is the realm of connected objects—referred to hereafter as "connected devices"—and systems implementing sets of connected devices communicating with each other, which will be more simply referred to as "connected systems." A device is considered connected if it has a digital connection to its environment, typically in the form of a protocol that is not exclusively point-to-point and supports multiple functions (e.g., data transfer, as well as control, updates, etc.).

Connected devices are often capable of interacting physically with their environment through sensors and/or actuators. Lastly, connected devices often have a high level of autonomy, which generally entails: Limited or nonexistent interven-

tion capacity for troubleshooting, reconfiguration, etc. Limited processing, storage (volatile and non-volatile), and communication capabilities due to cost or energy consumption constraints. These limited capabilities may restrict software update possibilities or prevent the use of network protocols or cryptographic standards that are not specifically designed for this context.

2.4 Regulatory Frameworks and Standards

Cybersecurity in IoT systems and smart homes starts with well-defined regulatory frameworks and standards. These frameworks serve as essential tools for harmonizing practices, defining security requirements, and establishing accountability across all stakeholders, including device manufacturers, software developers, and end-users.

The following sections focus on the main legislative and normative instruments targeting the mitigation of cybersecurity risks in the context of IoT systems. Further, we will analyze regulations and standards that provide structured methodologies for ensuring confidentiality, integrity, and availability for interconnected devices and data.

Specific regulations and standards will be elaborated in the following sections, which will show their importance, how they are put into practice, and the consequences this has on the smart home cybersecurity landscape.

In this thesis, the focus will primarily be on the cybersecurity guidelines [21] provided by the ANSSI [22] in France and the AMSN [23] in Monaco. These organizations play a critical role in establishing robust standards and best practices for securing information systems across various domains.

The company where this thesis was conducted, MVE [24], is a PASSI-certified entity. Hence, they need a strict observance of the regulations and recommendations emanating from the ANSSI and the AMSN. Being a PASSI company implies that MVE has to apply very strict compliance rules in ensuring that its operations are according to the best cybersecurity standards as dictated by these agencies.

By centering on the guidelines from ANSSI and AMSN, this thesis aims to explore their applicability and relevance in enhancing the security and resilience of information systems. It will also demonstrate how these regulations influence and guide the practices of PASSI-certified companies like MVE, ensuring that their methodologies and operations meet stringent regulatory and security expectations.

2.4.1 ANSSI guide

The ANSSI is France's national authority for the security of information systems. It operates under the authority of the Prime Minister and is tasked with protecting state, public, and critical private sector networks. A key aspect of ANSSI's work is the publication of comprehensive guidelines and technical recommendations aimed at enhancing the cybersecurity posture of organizations and systems across different sectors.

ANSSI publishes detailed guides that provide actionable insights, technical requirements, and best practices tailored to specific contexts, including information system security, cryptography, incident management, and the security of connected devices. These guides are not mandatory but serve as a reference for compliance with national and international standards. They are widely adopted by government agencies, private companies, and cybersecurity professionals to design, implement, and maintain secure systems.

The guides are categorized based on their intended audience, ranging from system developers and administrators to CISO and end-users. ANSSI's publications often include step-by-step methodologies, such as risk analysis frameworks, secure system architecture designs, cryptographic recommendations, and practical measures for protecting sensitive data.

One of ANSSI's notable contributions is its focus on lifecycle security, emphasizing the importance of security from the initial design phase of a system through its deployment, operation, and eventual decommissioning. This ensures that organizations address cybersecurity risks proactively and systematically.

Figure 2.4 represents the general architecture of an IoT system, based on guidelines given by ANSSI. The model describes main components, interactions, and data flows that can be met in a usual IoT ecosystem [21].

This generalization is drawn from the intent to provide a standardized framework for analyzing security risks and deploying protective measures in connected systems. By decomposing an IoT system into its basic components, such as devices, infrastructure, cloud services, and user interactions, ANSSI gave a clear structure for assessing critical points of vulnerability and needed safeguards.

The architecture provides a reference model that guides IoT deployments to observe best practices in cybersecurity: secure communications, lifecycle management, and user privacy. Below is an explanation of each element and its role within the system.

1. **Central Infrastructure**: Represents a central server or backend infrastructure managing data and services. It handles the core processing and data management functions for the connected devices.

2. **Connected Device**: A smart device or hub that interacts with other devices, the environment, and external systems. This is the main interface for commands, data collection, and interactions.

3. **Internet Services**: External internet-based services or platforms that the connected device interacts with. These could include APIs, third-party integrations, or online dashboards.

4. **Cloud Services (4)**: Represents cloud-based storage or computational services. The connected device or central infrastructure exports data to the cloud for storage, processing, or analysis.

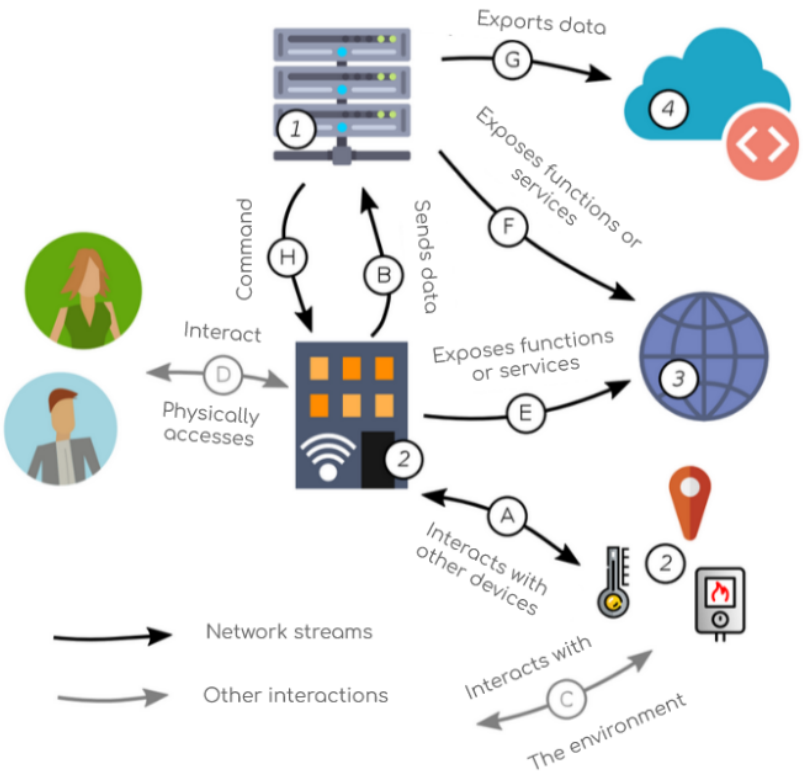

Figure 2.4: ANSSI Iot Architecture

- **A - Interacts with other devices**: The connected device communicates with other devices (e.g., sensors, actuators) within the system for operational purposes.
- **B - Sends data**: The connected device sends data to the central infrastructure for processing or storage.
- **C - Interacts with the environment**: Physical or environmental interactions occur via sensors (e.g., detecting temperature, humidity) or actuators (e.g., controlling lights or thermostats).
- **D - Interacts**: Users interact with the connected device via an interface (e.g., a mobile app, voice command, or direct physical interaction).
- **E - Exposes functions or services**: The connected device provides services or functionalities accessible via the internet.
- **F - Exposes functions or services**: The central infrastructure exposes higher-level functions or services to external systems or applications.
- **G - Exports data**: Data is exported to cloud services for storage or further processing, such as analytics or machine learning.
- **H - Command**: The central infrastructure sends commands to the connected device to control or configure its operations.

The ANSSI provides a comprehensive list of security recommendations for connected systems. For the scope of this study, the following recommendations have been selected, as we focus exclusively on the installation and configuration of Crestron systems, excluding any considerations related to software development.

Secure Configuration

Device Security

- **Initial Credential Setup (R19)**:
  - **Associated Risk**: The use of default or shared credentials makes brute force attacks and unauthorized access easier, especially if default credentials are widely shared across devices.
  - **How the Recommendation Mitigates Risk**: Unique credentials for each device reduce the likelihood of large-scale brute force attacks, ensuring that compromising one device does not expose others.
- **Disable Unused Features (R26)**:
  - **Associated Risk**: Unnecessary features (e.g., open ports or debugging interfaces) can be exploited by attackers to access the device, execute unauthorized code, or obtain sensitive information.
  - **How the Recommendation Mitigates Risk**: Disabling unused features reduces the attack surface, minimizing opportunities for exploiting vulnerabilities.

Communication Protection

- **Encryption for Connections (R23)**:
  - **Associated Risk**: Unencrypted communications can be intercepted through man-in-the-middle (MITM) attacks, compromising the confidentiality and integrity of transmitted data.
  - **How the Recommendation Mitigates Risk**: Cryptographic protocols ensure that data in transit is accessible only to authorized entities, protecting against eavesdropping and tampering.
- **Authentication Between Devices (R18, R20)**:
  - **Associated Risk**: The absence of authentication allows attackers to impersonate legitimate devices, enabling spoofing and unauthorized communication.
  - **How the Recommendation Mitigates Risk**: Robust authentication verifies the identities of communicating entities, preventing impersonation and MITM attacks.

- **Replay Protection (R5)**:
  - **Associated Risk**: Replay attacks reuse valid messages (e.g., commands or tokens) to gain unauthorized access or execute repeated actions.
  - **How the Recommendation Mitigates Risk**: Mechanisms like unique nonces or monotonic counters make every message unique, preventing attackers from reusing valid packets.

Logging and Monitoring

- **Logging Security Events (R41)**:
  - **Associated Risk**: The lack of logs prevents post-incident analysis and delays the detection of compromises or security issues.
  - **How the Recommendation Mitigates Risk**: Logs provide visibility into device activity, enabling the monitoring of suspicious events and rapid identification of breaches.
- **Remote Logging (R43)**:
  - **Associated Risk**: Locally stored logs can be altered or deleted by attackers who have compromised the device.
  - **How the Recommendation Mitigates Risk**: Remote logging ensures the integrity and availability of logs, even if the device is compromised.

Network Configuration

Segmentation and Isolation

- **Network Segmentation (OBJ 1, R29)**:
  - **Associated Risk**: Non-isolated IoT devices can serve as entry points for broader attacks on the corporate network.
  - **How the Recommendation Mitigates Risk**: Segmenting IoT devices into dedicated VLAN limits the impact of a compromise, preventing lateral movement by attackers.
- **Radio Interface Security (R35)**:
  - **Associated Risk**: Unprotected wireless interfaces can be exploited for eavesdropping or unauthorized access to devices.
  - **How the Recommendation Mitigates Risk**: Using modern encryption protects wireless communications, making it harder for attackers to intercept or manipulate traffic.

Resilience Against Network Attacks

- **Rate-Limiting and DoS Protection (R30)**:
  - **Associated Risk**: DoS or brute force attacks can exhaust IoTdevice resources, rendering them unusable.
  - **How the Recommendation Mitigates Risk**: Limiting the number of requests per time unit prevents attackers from overwhelming devices with illegitimate traffic.
- **Traffic Monitoring (R43)**:
  - **Associated Risk**: Suspicious activities on the IoT network may go unnoticed, allowing attackers to maintain persistence.
  - **How the Recommendation Mitigates Risk**: Integration with IDS or IPS systems detects and blocks anomalous traffic, improving network visibility and response capabilities.

Key and Certificate Management

- **Key Management (R11)**:
  - **Associated Risk**: Poorly managed or unprotected keys can be stolen, enabling unauthorized access or data tampering.
  - **How the Recommendation Mitigates Risk**: Secure storage (e.g., HSM or TPM) protects keys from unauthorized access, ensuring a high level of security.
- **Key Lifespan and Revocation (R16)**:
  - **Associated Risk**: Static or compromised keys can be exploited for extended periods without the ability to revoke access.
  - **How the Recommendation Mitigates Risk**: Rotation and revocation mechanisms limit the exposure of compromised keys and reduce the attack window.

Update Management

- **Authenticated Updates (R29)**:
  - **Associated Risk**: Unauthenticated updates can be used to install malicious firmware or exploit vulnerabilities.
  - **How the Recommendation Mitigates Risk**: Authenticated and cryptographically verified updates ensure the integrity and legitimacy of distributed software, preventing malicious code from being introduced.

2.4.2 ISO-IEC-27400-2022

The ISO/IEC 27400:2022 standard [25] outlines guidelines for managing cybersecurity and privacy in IoT systems, addressing the unique challenges posed by these highly distributed and diverse systems. IoT environments are characterized by extensive attack surfaces due to the large number of interconnected devices and the variety of stakeholders involved. The standard emphasizes the importance of implementing robust security controls throughout the IoT system lifecycle, from design to decommissioning.

For IoT developers and service providers, the standard recommends integrating security measures such as user authentication, secure firmware updates, malware protection, and redundancy mechanisms to enhance resilience. These measures are vital to ensure both the integrity and availability of IoT systems. Privacy concerns are addressed through compliance with established frameworks like ISO/IEC 27701 [26], advocating for a "privacy by design" approach to safeguard Personally Identifiable Information (PII) throughout the system's operation.

Standardization within IoT systems is guided by a domain-based reference model, which divides the IoT ecosystem into distinct domains, such as user domains, physical and control domains, and operational management domains. Each domain is associated with specific risks and recommended controls, promoting a structured approach to security and privacy management. The roles of key stakeholders—IoT developers, service providers, and users—are clearly defined, with shared responsibilities to ensure security and privacy compliance. Developers are tasked with creating secure devices and services, service providers must implement and maintain these measures, and users are encouraged to stay informed and apply updates.

The standard integrates a risk-based approach, aligning with frameworks such as ISO 31000 [27] for risk management and ISO/IEC 27005 [28] for cybersecurity. It stresses the need for comprehensive risk assessments that consider vulnerabilities, human errors, malicious threats, and even natural disasters. By applying this structured approach, ISO/IEC 27400:2022 [25] aims to create a secure and trustworthy IoT environment, fostering collaboration among stakeholders while addressing the complexities of modern IoT ecosystems.

2.4.3 Standard Cybersecurity Features in Crestron Systems

Crestron itself has established cybersecurity guidelines to safeguard its systems. These guidelines focus on essential principles, including the use of encrypted communication, secure user authentication, and regular system updates. For instance, Crestron utilizes protocols like 802.1x [29] authentication to ensure that only authorized devices can connect to networks, AES [30] encryption for secure data transmission, and PKI-based authentication [31] for strong identity verification. The company also incorporates centralized credential management through AD and supports secure communication protocols such as TLS [32], SSH [33], and HTTPS [34]. While these measures are vital for maintaining basic security, they primarily highlight general best practices

instead of offering detailed technical standards or comprehensive frameworks to tackle complex cybersecurity issues. As stated in Crestron's commitment to security [35], the recommendations aim to assist users in implementing fundamental protective measures but do not explore the intricacies of advanced threat mitigation or lifecycle security management. Therefore, while these guidelines are useful, they are more appropriate for general deployments and may not provide the specificity required for highly sensitive or regulated environments.

Crestron has a dedicated Security Advisories section on its website [35], where it shares information about identified vulnerabilities, including their technical details and potential security implications. This resource is designed to help users stay informed about risks and implement necessary measures to maintain system integrity.

The vulnerabilities are documented using the CVE [36] system, an international standard for identifying and cataloging publicly disclosed cybersecurity vulnerabilities. A CVE entry provides a unique identifier for each vulnerability, along with a brief description and references to additional information or patches. By adopting the CVE system, Crestron ensures transparency and alignment with industry best practices for vulnerability management, enabling users to track issues efficiently and access relevant updates. CVE identifiers are widely used in cybersecurity reporting and allow for consistent tracking across different platforms and tools [36]. This structured approach helps Crestron reinforce its commitment to improving cybersecurity while fostering collaboration between researchers, vendors, and end-users in mitigating risks.

One notable vulnerability is CVE-2023-6926 [37], which affects the AirMedia 300 series devices. The issue arises from the handling of the edidload command, which is used to manage Extended Display Identification Data (EDID). An attacker could exploit this vulnerability by creating a malicious script and uploading it through the command interface, potentially leading to remote code execution on the underlying operating system. This type of vulnerability is critical, as it could give attackers full control over the device, enabling them to execute arbitrary commands, change configurations, or carry out malicious actions across the network.

Another significant issue, CVE-2023-38405 [38], was found in the BACnet protocol implementation of 3-Series Control Systems. BACnet is a widely used protocol for building automation and control. The vulnerability involved a specific malformed packet that could trigger an infinite processing loop, effectively causing a DoS by making the device unresponsive. This vulnerability highlights how improperly handled inputs in commonly used protocols can lead to operational disruptions in critical environments.

CVE-2022-40298 [39] impacted the AirMedia Windows Application and involved a privilege escalation scenario. An attacker with low-level access could exploit file handling weaknesses by initiating a repair operation on the application, allowing them to escalate their privileges to SYSTEM level. This vulnerability emphasizes the necessity of securely managing system-level processes and permissions in software design, especially for applications that interact with user-controlled environments.

Additionally, older vulnerabilities like CVE-2018-11228 [40] highlighted risks associated with the CTP. This vulnerability enabled unauthenticated attackers to execute remote code through command injection in the protocol interface. Such issues illustrate the urgent need for strong authentication mechanisms and input validation in proprietary communication protocols to prevent unauthorized access and exploitation.

These advisories highlight the technical challenges of securing interconnected devices and systems. Crestron's proactive approach to disclosing and mitigating these vulnerabilities demonstrates its commitment to enhancing cybersecurity across its product ecosystem. More information on these vulnerabilities can be found on Crestron's Security Advisories page [35]. The company has also developed a dedicated tool designed to help administrators detect and resolve security issues within their networks.

2.5 Reference Tool: Crestron Security Audit Tool

This utility is integrated into the *Crestron Toolbox* software suite and is commonly used by technicians and system integrators to verify compliance with Crestron's own best practices for system security.

The Crestron Audit Tool evaluates a fixed set of configuration parameters. Its goal is to verify whether the system conforms to a number of predefined security recommendations, such as enforcing authentication, enabling SNTP, limiting login attempts, and disabling outdated protocols. Each configuration item is marked as either compliant or non-compliant.

One feature of this tool is its use of four security levels, each representing an increasing depth and strictness of checks:

- **Level 1** – This level validates only essential protections such as authentication. It serves as a minimal compliance baseline. As shown below, only a handful of checks are executed, and most categories are omitted.
  - **Recommended Firmware Version** – Verifies if the device firmware is up to date.
  - **SSH Enabled** – Confirms that SSH access is enabled.
  - **Authentication Enabled** – Checks that user login is required.
  - **Block IP Lockout Time** – Ensures a delay is enforced after failed login attempts.
  - **Password Max Attempts** – Verifies the maximum number of failed login attempts.
  - **Password Min Length** – Confirms a minimum length is set for passwords.
  - **SSL Enabled** – Checks that HTTPS is enabled.

  - **SNTP Enabled** – Verifies that Simple Network Time Protocol is active.

- **Level 2** – Adds slightly stricter checks on password policies, SNTP activation, and SSL configuration. This level corresponds to a typical mid-range deployment where basic best practices are expected.

  - **Password Policy – Mixed Case** – Requires both uppercase and lowercase characters in passwords.
  - **Password Policy – Numeric Characters** – Requires digits in passwords.
  - **User Idle Time** – Confirms that sessions time out after inactivity.
  - **SSL Fallback Disabled** – Prevents fallback to older, insecure SSL versions.
  - **SSL CA Certificate** – Verifies use of a trusted (not self-signed) certificate.
  - **Secure Gateway Mode** – Ensures the gateway mode enforces secure communication.

- **Level 3** – At this level, the tool evaluates a more comprehensive set of rules, including password complexity (mixed case, numeric, special characters), user idle time, and certificate validation. This is often the recommended level for enterprise deployments.

  - **802.1X Mode Enabled** – Ensures 802.1X authentication is active.
  - **802.1X Server Validation Enabled** – Validates the 802.1X authentication server.
  - **Web Server Disabled** – Checks if the embedded web server is turned off.
  - **Audit Log Enabled** – Verifies that audit logging is active.
  - **Audit Log Level Sufficient** – Ensures logging includes security-relevant actions.
  - **Local Users** – Displays and verifies number of local users configured.
  - **Active Directory Groups** – Checks for any AD group integration.

- **Level 4** – The most exhaustive level, Level 4 enforces strict password policies, disables fallback options, and requires CA-signed SSL certificates, audit logging, and 802.1X authentication. This level is aligned with the highest security standards and is suitable for sensitive environments. The full report structure is visible below.

  - **Password Policy – Special Characters** – Requires symbols in passwords.

- **802.1X Certificate Authentication** – Confirms certificate-based 802.1X login is enforced.
- **System Log Enabled** – Ensures system-level event logging is active.
- **System Log Security Level Sufficient** – Verifies minimum required security level for logs.
- **System Log Includes Audit Log** – Ensures audit logs are included in the system log stream.
- **AutoDiscovery Disabled** – Confirms the device does not broadcast its presence on the network.

Each level is cumulative, meaning higher levels include all checks from the previous ones. However, it is important to note that the checks remain relatively static across firmware versions, and the tool does not dynamically adapt to evolving security standards or network architectures.

Despite offering a structured output, the audit tool lacks several features that are critical for advanced cybersecurity auditing, such as remediation suggestions, vulnerability severity classification, or standardized compliance mappings (e.g., NIST, CIS). Moreover, it does not offer automatic fixes or in-depth inspection of network-facing services, which limits its utility in professional security assessments.

This limitation is what inspired the development presented in this thesis: the creation of a tool capable not only of assessing the security level of a Crestron system, but also of automatically remediating identified vulnerabilities. Additionally, the proposed solution aims to provide both an overall risk score and detailed risk levels for each vulnerability, enabling technicians to gain a clearer understanding of the system's security level and the criticality of each issue.

Chapter 3

Methodology

This chapter outlines the methodological approach adopted for the design and development of the CreScan tool. It provides a comprehensive overview of the principles and decisions that guided its architecture, implementation, and operational workflow. The development process, described in section 3.1, follows a pragmatic and security-oriented approach tailored specifically for Crestron environments, emphasizing modularity, precision, and usability. The rationale behind the adoption of a rule-based evaluation mechanism is detailed in section 3.2, where the use of deterministic and auditable logic is explained as a foundation for consistent and transparent security assessments. Subsequently, section 3.3 describes how CreScan collects and processes configuration data through both HTTPS and SSH channels, ensuring a reproducible and non-invasive audit workflow. This section also details the data analysis pipeline, the automated report generation process, and the structure of the HTML-based deliverables produced for end users, as well as introducing the autofix remediation subsystem, which bridges the gap between diagnosis and action by enabling secure, user-validated corrections directly on target devices.

3.1 Tool Development Framework

The development process for the CreScan tool was guided by a pragmatic field-based approach, focused on creating a tool that could be integrated into the smart home techs' workflow with little disruption. The key assumption was to deliver an automated security auditing system that is Crestron environment specific, which is notorious for being closed, complex, and difficult to evaluate with generic tools [41]. The system must be both secure and lightweight: secure enough to return accurate evaluation based on security principles, but lightweight enough to run on standard technician hardware on stand-alone networks.

Rather than starting from a common template, the software was designed from the ground up to accommodate the specific operation semantics and security needs of Crestron devices. This meant functioning within the constraints of limited protocol support, the necessity for SSH-based diagnostic procedures, and the inability to work with centralized management platforms. CreScan was intended as a specialized tool

for a particular but important niche: embedded Crestron controllers in residential or corporate IoT networks.

The initial step in the development process was to determine the list of functionality that would be needed within a beneficial security assessment. It was establishing a minimum set of network behavior and configuration items mandated by security standards or usually ignored in smart home deployments. These were HTTPS certificate expiration [34, 42], presence of outdated TLS versions [43], ICMP [44] response on or off, IPv6 [45] configuration, password settings, and authentication. All of these were mapped to a distinct functional requirement that was programmatically testable.

Python [46] was utilized as the base language due to its flexibility, quick prototyping ability, and widespread application in cybersecurity scripting. Its capacity to interact with SSH [33] libraries, system command wrappers, and templating engines also placed it well to combine some of the pieces of the auditing process together. The implementation is based heavily on the Paramiko [47] library for secure shell-based communication, which allows the tool to remotely query devices for configurations and execute commands in a controlled, isolated manner.

The system was built with a modular architecture, wherein each security test was contained in an individual self-contained module. All such modules shared a common interface and data format and could be plugged into the normal execution flow seamlessly. Such a structure significantly reduces coupling between components and makes extensibility simple. New modules may be added without altering the core code, as long as they support the expected input/output schema.

Every module performs a specific audit action—checking for outdated TLS versions, weak password policies, or disabled firmware verification—and reports an ordered result that includes the severity score, human-readable description, and (where applicable) remediation guidance. The severity score is on a normalized 0-to-10 scale, and thresholds have the labels of low, medium, high, and critical problems. The output of all modules is aggregated and fed into a centralized report generator that translates technical outcomes into a visually cohesive HTML report. The report includes the following core components:

- **Risk Gauge** – a graphical indicator summarizing the overall security score.
- **Severity Grading** – classification of issues into Low, Medium, and High risk levels.
- **Category Breakdown** – detailed listing of each tested configuration domain, with specific results and recommendations.

The visual form of the report was a significant design choice. Taking cues from expert auditing software like PingCastle [48], the HTML report displays a balance of technical richness and usability. Technicians are presented with a risk dashboard containing a radial gauge to display the overall severity score. Individual results are shown in collapsible panels, providing the underlying cause, a plain English

explanation, and, where appropriate, a recommendation. This allows users to address the most critical issues without being bombarded with raw data.

From a development workflow perspective, a test-first approach was employed. Each module was validated against secure as well as misconfigured environments to ensure that its diagnostic reasoning was accurate. Extreme caution was exercised to prevent false positives and to ensure the tool never provided recommendations based on assumptions or unsubstantiated outcomes. The reliability of each check was a top priority, especially considering that the target user is not necessarily a cybersecurity professional, but a technician requiring timely and actionable results.

The second significant feature of the tool is its "autofix" feature, which can be optionally enabled. There are some problems that the tool can automatically fix using SSH with pre-defined commands that adjust the controller's configuration (e.g., enabling or disabling 802.1X, or suppressing ICMP responses). This required thoughtful management of SSH sessions, parsing of output, and error recovery so as not to create undesired side effects. A precise mapping was designed to map every fixable category to a corresponding command set, and metadata indicating whether a reboot is required. To guard against abuse, autofix is always optional, and the user is warned before any change can be made. Table 3.1 summarizes the current mapping between each vulnerability category, the associated SSH command (when applicable), and the availability of automated remediation.

Table 3.1: Audited Vulnerabilities, SSH Commands, and Autofix Availability

Vulnerability	**SSH Command**	**Autofix Available**
Firmware signature	AUVERIFYSIGNATURE	Yes
802.1X authentication	8021XAUThenticate	Yes
Password policy	GETPAsswordrule	No
SNTP configuration	SNTP	Yes
IPv6 configuration	IPV6	Yes
ICMP configuration	ICMP	Yes
Excessive login attempts	SETLOGINAttempts	Yes
Brute-force lockout time	SETLOCKOUTTIME	Yes
Global authentication	AUTHentication	Yes
TLS version	-	No
X.509 Certificate	-	No
Remote Syslog	REMOTESYSlog	No
Audit Log	AUDITLogging	Yes

Usability was another main driver. While the tool was designed for non-hobbyist use, it needed to be accessible to system integrators, auditors, and technicians who may or may not have programming backgrounds. The command line needed to be as simple and intuitive as possible, with minimal arguments required to satisfy most usage cases. Parameters like controller IP address, SSH user, verbosity level, and

output directory are sufficient to run a full scan. If the credentials are missing or incorrect, the tool queries the user interactively rather than failing silently. When run in verbose mode, the tool logs all important actions, including the raw output of the checks and the commands executed.

The software was designed to operate completely offline. Everything is checked on the local network, and nothing is relayed to third-party servers. This is very crucial in enterprise setups where data minimization and confidentiality are essential. Additionally, no credentials are stored or cached by the application, and everything entered is safely processed in memory. The decision not to keep sensitive data available via logging is based on the security-first methodology to create the software.

Rather than returning a single application, the codebase was structured to mirror the form of a security audit. Every check script is found within an individual module, each returning JSON-like dictionaries that may be dealt with generically. New scripts may then be created by other developers or auditors by copying an existing script and reusing the audit functionality. This plug-and-play extensibility not only makes CreScan possible as an end product but as a living product that can evolve along with the ecosystem that it is designed to monitor.

When Crestron updates with new firmware or introduces new config capabilities, new security validations can be implemented without rewriting the entire engine. The modularity allows the auditing system to naturally scale over time. This is particularly applicable to integrators that may be made to comply with evolving standards, such as those released by ANSSI [21], ISO/IEC 27400 [25], or corporate internal guidelines. CreScan can be readily adapted to meet new compliance requirements and is therefore an asset for the long term rather than a throwaway script.

Generally, the design of CreScan was a balance between flexibility and precision. It combines extensive in-field audit expertise into a single, cohesive tool, but is sufficiently flexible to be able to adapt to future regulatory change or hardware evolution. It is not empowered by generic scanning, but by being precisely aligned with the actual-world implementation realities of Crestron-based intelligent spaces. What began as a specialist diagnostic tool evolved into a complete audit aide, having the potential to obscure the disconnect between security theory and practice for technicians in a critical domain of modern-day IoT infrastructure.

3.2 Rule-Based Approach

The choice of a rule-based solution within the CreScan tool was based on a combination of pragmatism, flexibility, and auditability. Rule-based systems provide a systematic and deterministic way of judging the configuration status of a device and, as such, are ideally suited to applications where standards must be enforced and regulation compliance easily identifiable.

Fundamentally, a rule-based model [49] operates by applying a predefined set of conditions, which are enforced either sequentially or independently. A rule is a security best practice, policy mandate, or technical config standard. For the purposes

of this thesis, rules were written to encapsulate both general security principles (e.g., encryption in transit, authentication hardening) and Crestron-specific configurations (e.g., 802.1X authentication status or firmware signature verification). This structure allows the tool to produce reproducible and traceable results, which is essential when the target audience is security auditors, techs, and even non-technical stakeholders.

The idea of using a rule-based approach was not new in itself. In fact, there was drawing from other audit platforms such as PingCastle [48], a mature security audit tool for Microsoft Active Directory. PingCastle uses a collection of domain-specific rules to tokenize and examine configuration information, producing a human-readable, prioritized danger report. This concept was taken and reinterpreted inside the CreScan environment, having rules focused on the particular smart home controller environment and the way that it works. Rather than reinventing the audit model from scratch, CreScan recontextualized it within an environment with much less tooling available.

One of the greatest strengths of the rule-based approach is that it is open. Any given diagnostic result can be traced back to one single rule that started it off. For instance, if the report indicates firmware verification disabled, it is immediately clear what command was executed to verify this, what should have been observed in return, and what actually was observed. This traceability provides both user assurance and audit defensibility, two key factors in a working professional cybersecurity environment.

Also, the rules application allows for expansion module-wise. New requirements can be introduced simply by introducing new rules as new modules, without modifying the base logic engine. For example, whenever a new ANSSI recommendation is published or Crestron introduces new configuration parameters for network segmentation, pertinent checks can be scripted and introduced to the system without disturbing the existing rule set. This makes the tool future-proof and versatile.

From a deployment perspective, each rule is encapsulated in a dedicated Python script that embodies the logic required to check for a single control or configuration option. The outcome of each rule is not only a pass/fail determination, but a severity rating, a text description of the issue, and a recommended fix. This approach ensures that outcomes are not only technically accurate but readable and actionable. Each rule is an independent diagnostic unit, part of a bigger report architecture.

In addition, the design of the tool supports a form of prioritization. Rules are assigned severity scores, which are then used to contribute to the aggregated risk figure presented in the final report. This means technicians can identify what issues must be addressed immediately and what may have to wait. It also places the tool in alignment with best practice in managing risk, where risk is not simply binary but graduated.

Another significant benefit of rule-based design is that it is well-suited to environments of access limitation. As Crestron systems are commonly deployed in private, segmented networks, the software must be able to work independently of cloud dependency and remote examination. An in-box rules engine allows all the

tests to occur in the local environment, minimizing data exposure and ensuring that audits can be performed safely in segmented environments.

Lastly, the rule-based methodology presents a balance between elegance and rigor. It promotes structured analysis with the flexibility to adapt to change.

3.3 Data Collection and Analysis

A fundamental part of the operation of CreScan is its ability to collect, process, and analyze data from a Crestron controller in a reproducible and consistent fashion. CreScan's data collection process was carefully designed to enhance audit quality, ensuring accurate diagnoses without adding overhead or introducing security risks.

The utility relies on two main data channels: HTTPS and SSH. For public-facing setups, e.g., TLS version support and certificate validation, CreScan directly establishes HTTPS socket connections to the controller. This enables it to retrieve certificate metadata—such as expiration times, issuer chains, and validity windows—using libraries like `ssl` [50] and `cryptography` [51]. This passive, read-only inspection avoids modifying the controller in any way during these tests, keeping system integrity and avoiding side effects.

More detailed configuration data, however, is accessed via SSH. After the presentation of credentials, the tool uses the `paramiko` library to open an encrypted session with the device. In this session, it fires diagnostic commands that inform on the system's security stance, for example, the status of 802.1X, login policy, ICMP response behavior, firmware signature check enforcement, SNTP settings, and IPv6 exposure. Each command is carefully selected to be non-invasive, and no configuration changes are performed in the auditing process unless the user explicitly decides to utilize the autofix capability.

Once collected, this data is put into a uniform format for examination. Every check module examines raw output and extracts significant indicators relevant to the rule that it is enforcing. For instance, the 802.1X check will ascertain whether authentication is `ON`, `OFF`, or has indefinite results. In indeterminate cases, the result is still logged, although with decreased seriousness and a message that manual verification may be required. This conservative approach ensures the report to be safe but not at the expense of omitting any potential misconfigurations.

Apart from formal environments, the software also gathers unstructured data where necessary. For example, in gathering hardware configuration data, it retrieves multi-line terminal dumps and reconstructs them as a structured HTML table and integrates it into the final report. Such integration of structured and semi-structured data is made possible so that technician-specific information lost in strict JSON-style reporting is retained.

A key goal in development was to render data gathering resilient to network volatility and device inconsistency. Crestron controllers, despite how standardized they may be, are still going to differ in firmware level and capability supported. Therefore, each test was designed to handle missing commands gracefully. If a command

does not succeed or returns nothing, the module logs the failure internally, and the resulting report signals the doubt by lowering the severity score and recommending verification of the setting manually.

CreScan also provides network scanning functionality by utilizing the Nmap [52] tool. By specifying a subnet via the `-network` option, the software performs an initial host discovery phase in conjunction with ICMP pings and ARP resolution to identify live systems. The action is not necessary, but it helps deliver engineers to a quick understanding of connected devices and their MAC addresses, which can be helpful in those environments where device inventory is not well-maintained. The outcome is optionally stored for future reference.

Once the data is collected and processed, it is passed on to the report generation module. Here, Jinja2 [53] templating is used to restructure the raw diagnostics output into an interactive HTML report. The data is placed inside pre-formatted sections, each dedicated to a specific rule or configuration domain. There is a final risk score that is derived as a weighted average of the separate severity scores, and this score is presented graphically on the report in the form of a radial gauge. This provides a bird's-eye view without compromising access to low-level detail.

The second major aspect of the analysis stage is the flagging of issues that are eligible for correction. While not all issues are actionable, CreScan is designed to flag some checks as "autofix eligible". These tags are then used in the final report to allow users to decide whether or not to use the autofix module. This is an optional, local choice, but the analysis includes enough metadata to make it possible: original command output, expected values, and the suggested command to fix the issue.

SSH-based checks require valid user credentials to execute privileged diagnostic commands on the controller. The password is requested securely during runtime and is never printed, stored, or cached by the system. When typing in the password, it is masked to not display it in the terminal and, more importantly, not printed to the screen or written to the bash history. This configuration is mindful that authentication data does not persist beyond the active session, which is a good credential hygiene practice.

Contrarily, the HTTPS-based verifications entail no type of authentication. These verifications are solely based on publicly visible metadata revealed through the TLS handshake mechanism of the controller. This renders them secure, fast, and suitable even when administrator credentials are lacking.

Finally, the data collection and analysis pipeline of CreScan is designed to be deterministic, reproducible, and security-aware. Every effort has been made to ensure that the data retrieved is accurate, relevant, and immediately applicable to the operational needs of auditors and technicians.

3.3.1 Report Generation and Structure

One of the most valuable deliverables of the CreScan tool is the automatic HTML report that consolidates the results of all of the security tests run in one location

in a brief and visually clear document. The purpose of the report is not only to make the user aware of the vulnerabilities that exist, but to facilitate quick response, documentation, and communication to the stakeholders.

Technically, the report is generated using the Jinja2 template engine, which combines all structured data collected by the tool into an HTML template in a dynamic way. The use of Jinja2 enables clear separation between data and presentation logic, making it easy to maintain and develop the visual presentation of the report in the long run. The output is an HTML document, readable by any contemporary web browser, independently of external dependencies or an internet connection.

The report begins with a title with the generation date and time, and a collapsible box displaying detailed hardware information for the scanned controller. The box, an HTML table, includes the system name, device modules, and other platform-dependent identifiers.

Below the title, a "Summary" box shows three key measurements:

- **Overall Score** (from 0 to 10), the average of the severity score of all checks.
- **Risk Level**, which is categorized as follows:
  - **Low**: Score of 0.0 to 3.9
  - **Medium**: Score of 4.0 to 6.9
  - **High**: Score of 7.0 to 8.9
  - **Critical**: Score of 9.0 or higher
  - **Number of Critical Issues**, counting the number of findings that exceeded the 8.0 threshold of severity.

This summary is followed by a radial gauge chart, drawn using the Canvas Gauges JavaScript library [54]. The gauge offers the instant gut feeling of the overall security posture of the system and is a graphical restatement of the numeric score. A graphical representation of the described summary is shown in Figure 3.1.

The report body is divided into individual findings, each one shown in an expandable/collapsible "Vulnerability" element. Each item includes:

- Configuration or check name that was examined.
- Severity label and score, color-coded by severity.
- Short text description of the issue.
- Full technical details, i.e., error messages or command output.
- Recommendation on a line by itself, explaining how to resolve the issue.

A graphical representation of the generated report is available in Figure 3.2. Each vulnerability can be expanded to read more details. Specifically, Figure 3.3 shows the details related to the first two vulnerabilities shown in Figure 3.2.

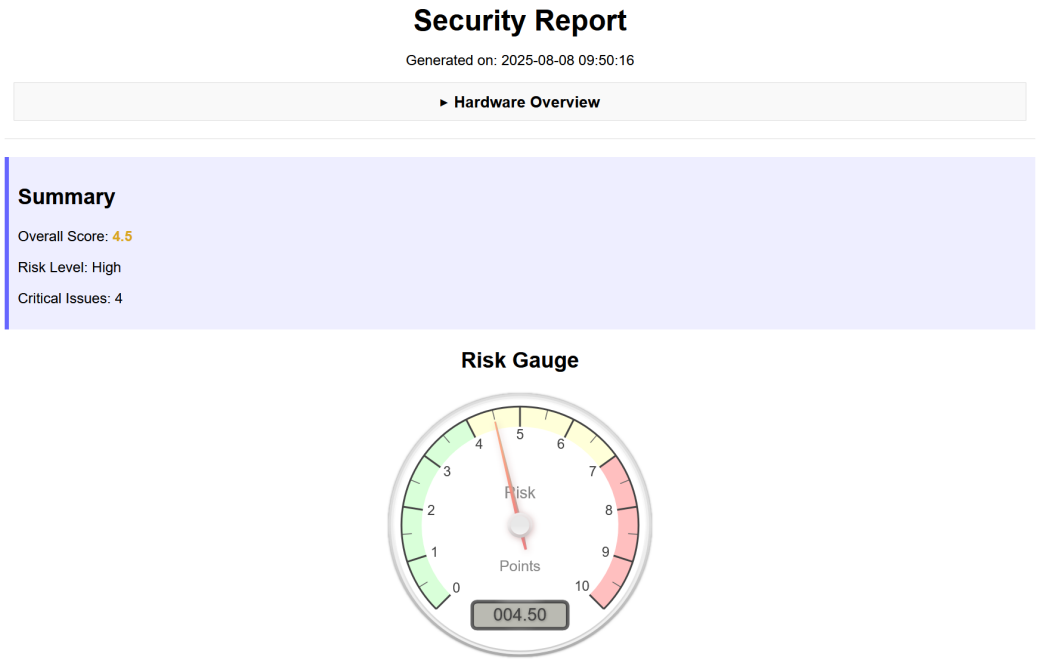

Figure 3.1: Report Summary

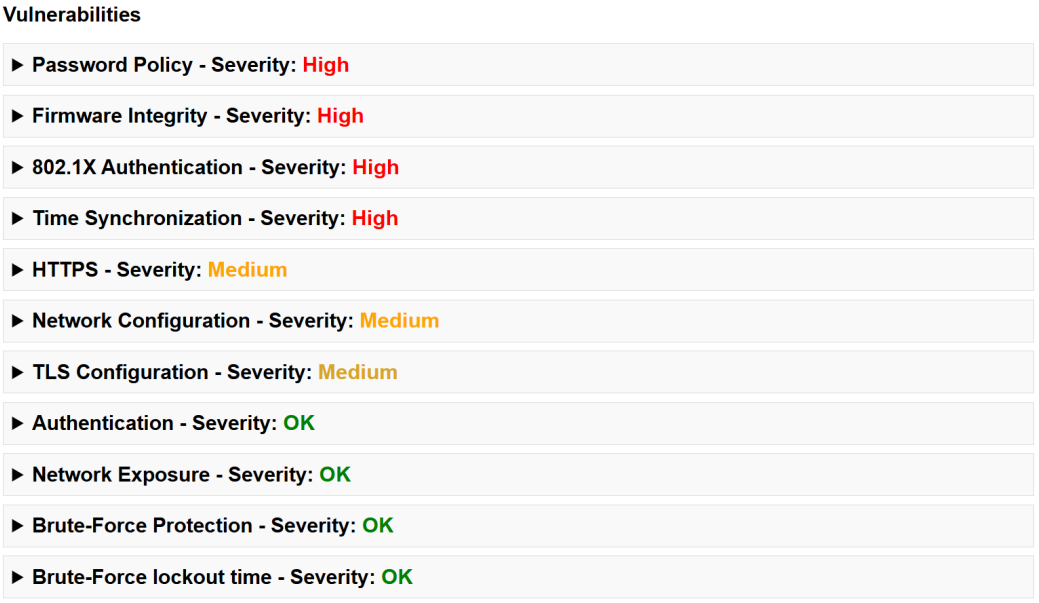

Figure 3.2: Example of a possible vulnerability list.

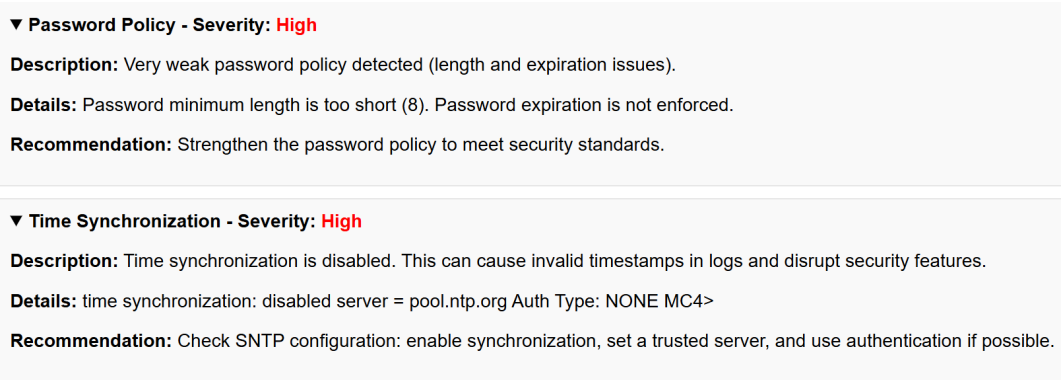

Figure 3.3: Vulnerability Details.

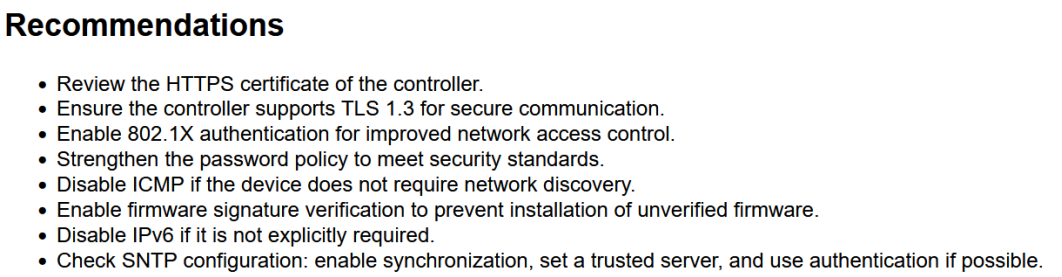

Figure 3.4: Example of possible recommendations

Where the autofix module was enabled and autofix was successfully applied, the original severity score is shown with a strikethrough, and a "Fixed" tag is appended. This allows auditors to track both the initial vulnerability and also remediation afterwards, giving them a history of risk exposure and mitigation.

At the end of the report, a "Recommendations" section provides a brief listing of all suggested remediations. The section is formatted as a to-do list for technicians, allowing security enhancements to be easily reviewed and prioritized. The recommendations are copied from each finding's internal remediation field and are presented in plain English, using as little technical jargon as possible.

Overall, the report was written with usability, traceability, and accessibility in mind. The sections are written to facilitate both high-level and detailed reading, and the file format facilitates long-term sharing and archiving without reliance on cloud storage or proprietary file formats. This final deliverable is both an engineering diagnosis and a compliance artifact, and it closes the gap between managerial oversight and engineering detail.

3.3.2 Autofix Remediation System

As already said, CreScan is equipped with an "autofix" system designed to automatically correct specific types of vulnerabilities identified during the scanning process. The rationale behind this feature is to bridge the gap between diagnostic and remediation, enabling technicians not only to understand the weaknesses in a Crestron installation but also to act on them immediately, without requiring manual

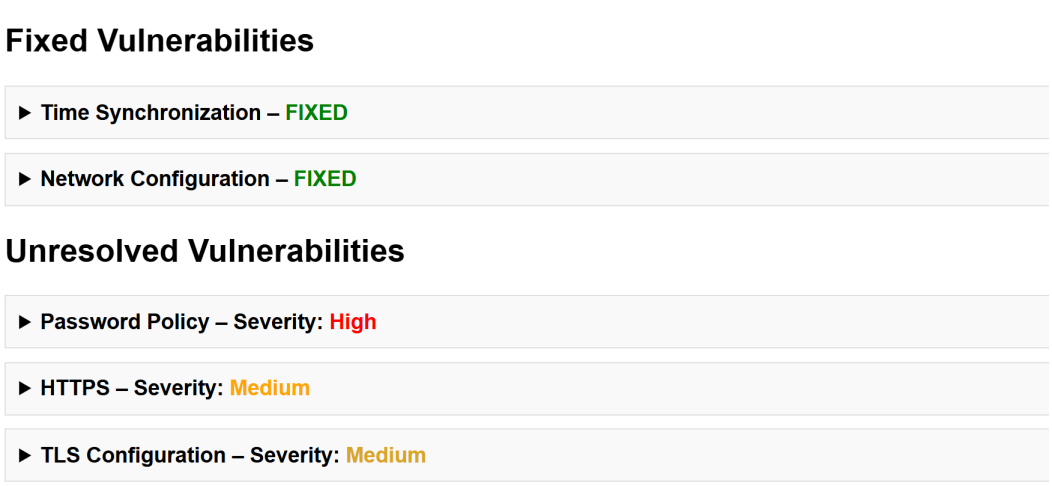

Figure 3.5: Example of an HTML report with a couple of vulnerabilities fixed.

intervention via third-party interfaces.

The autofix feature operates post-analysis and under user validation. The user must either invoke the tool with the `-autofix` flag or explicitly confirm remediation when prompted at the end of a standard scan. This ensures that no configuration changes are made without informed consent, aligning with best practices in controlled environments where device configuration must be managed with caution.

Internally, the autofix system is implemented as a mapping between vulnerability categories and their associated correction commands. For example:

- For the "802.1X Authentication" check, the fix consists of executing `8021XAUThenticate ON`.
- For disabled firmware signature validation, the command `AUVERIFYSIGNATURE ON` is issued.
- To disable ICMP responses, `ICMP OFF` is applied.
- To enforce a login attempt policy, `SETLOGINAttempts 5` is used.

Each fixable issue is therefore directly linked to a deterministic set of SSH commands, which are sent to the controller using the same secure session established for data collection. After execution, the system checks for success based on the output and exit status, and updates the internal report structure accordingly.

In the final report, issues that have been automatically resolved are clearly marked as such. The original severity score is displayed with a strikethrough, and the updated score is set to zero. Additionally, a confirmation message is shown next to each fixed item, indicating which command was executed and the result.

A design consideration of high importance was the safe handling of potentially disruptive fixes. Some configuration changes (such as enabling firmware signature checks or disabling IPv6) might require a reboot to take full effect. In such cases, the autofix logic tracks whether a reboot is recommended and prompts the user accordingly. If consent is given, the tool will reconnect to the controller and issue a

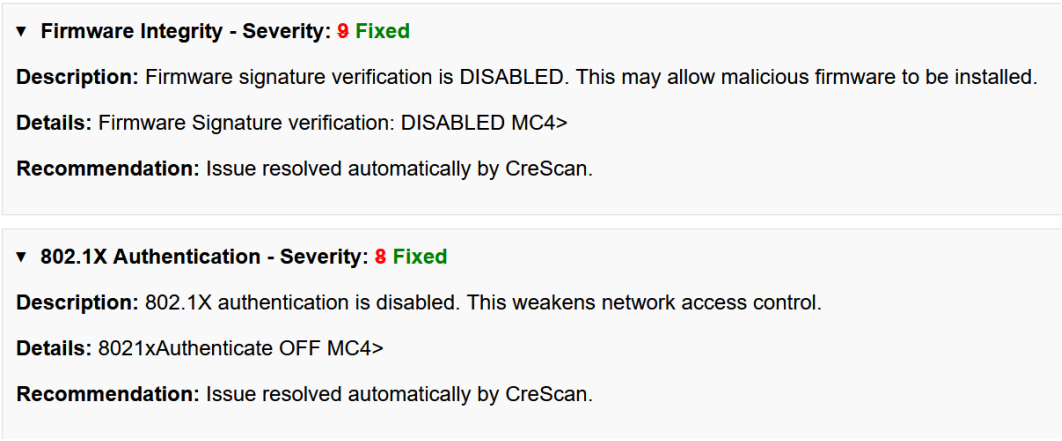

Figure 3.6: Fixed Vulnerability details

`REBOOT` command. Otherwise, the user is informed that a reboot should be performed manually at a convenient time.

To prevent accidental misuse, all commands executed during autofix are transparently printed in verbose mode, and any error during execution is logged and gracefully handled. The tool never continues applying fixes if a failure is encountered mid-process; this ensures configuration integrity and avoids partial states that could cause operational inconsistencies.

The scope of the autofix system is deliberately limited to high-confidence remediations. It avoids changes that could compromise network accessibility or require external dependencies. The philosophy is to provide safe, local, and targeted fixes for the most common and impactful misconfigurations encountered in the field.

The autofix remediation system adds significant operational value to CreScan. It transforms the tool from a passive assessment utility into an active security hardening assistant. By offering automated, controlled corrections for well-defined vulnerabilities, it empowers technicians to close critical security gaps quickly, efficiently, and with full awareness of what changes are being made to the system.

The implementation of CreScan, including the complete source code and documentation, is publicly available on GitHub.[1]

[1] `https://github.com/FedericoSchiavolini/CreScan`, last visited on 13 October 2025.

Chapter 4

Design and Implementation

This chapter presents a comprehensive evaluation of CreScan's diagnostic capabilities through a structured suite of test cases designed to replicate real-world deployment conditions and security configurations. Each test case focuses on a specific security control or configuration domain, illustrating how the tool identifies vulnerabilities, assesses compliance, and proposes corrective measures. The analysis encompasses key areas of embedded system security, including authentication mechanisms, firmware integrity, network hardening, and cryptographic configurations. For each scenario, CreScan's detection logic, verification approach, and remediation strategies are discussed in detail, highlighting the tool's ability to perform reproducible, standards-aligned audits. Finally, the chapter introduces the risk scoring model used to quantify the severity of identified issues, enabling prioritized remediation and facilitating compliance with established cybersecurity frameworks such as CIS Benchmarks, ISO/IEC 27001, and NIST guidelines.

4.1 Test Cases and Scenarios

To evaluate the diagnostic effectiveness and depth of coverage offered by the CreScan tool, an extensive suite of test cases was constructed to mirror real-world conditions and security configurations. Each test case corresponds directly to a specific security domain, rule, or configuration area tested by the tool. The following categories encapsulate the full spectrum of checks performed during the audit.

4.1.1 Password Policy

Password policies play a pivotal role in preventing unauthorized access to networked devices. A weak password policy is one of the most common misconfigurations in embedded environments and is often the first vulnerability exploited during lateral movement or brute-force attacks. From a security engineering standpoint, a robust password policy should enforce a minimum character length, a mix of character classes (uppercase, lowercase, numeric, and special characters), and a mandatory expiration interval to rotate credentials periodically [21].

CreScan evaluates the controller's enforcement settings by inspecting parameters that dictate the allowed complexity and lifecycle of user credentials. Systems that allow short passwords (fewer than 10 characters) or that lack expiration mechanisms fail to comply with most industry standards, including CIS Benchmarks [55] and ISO/IEC 27001 guidelines [56]. Weak credentials drastically reduce entropy and make the system vulnerable to dictionary attacks, especially in the absence of login rate limiting or account lockout features. The tool flags these misconfigurations with a high severity score and recommends tightening the policy in alignment with NIST SP 800-63B [57] requirements.

4.1.2 Firmware Integrity

The integrity of firmware is foundational to any trust model in embedded systems. If an attacker can inject unauthorized firmware into a controller, they gain the ability to circumvent all operating system-level protections, since firmware operates below the application stack. Modern secure development lifecycles mandate that firmware updates must be signed using asymmetric cryptographic keys, and that runtime environments validate these signatures before execution.

Crestron systems implement this via the `AUVERIFYSIGNATURE` control, which enables or disables signature enforcement during firmware loading. When signature validation is not active, any binary—malicious or benign—can be flashed onto the device. This opens the door to persistent threats such as implanted backdoors, surveillance tools, or logic bombs. CreScan validates this configuration state through secure shell access and considers any device with signature validation disabled to be at severe risk. The tool can automatically re-enable this protection and, if supported, prompt the user to reboot the system to finalize the configuration change.

4.1.3 802.1X Authentication

IEEE 802.1X [29] is a port-based network access control protocol that forms the basis for secure wireless and wired LAN infrastructures. It is a member of the IEEE 802 suite that prevents unauthorized devices from being connected to the network by enforcing authentication using a central RADIUS [58] server.

802.1X operates in practice by placing a switch port in "blocking" mode until a connected device (supplicant) provides valid credentials. Once authenticated, the port enters an authorized state, granting the device access to the controller network services. On Crestron controllers, this feature is regulated using the `8021XAUThenticate` interface. When off, any connected device plugged into the network port can use the services of the controller even without specific authorization. In hostile environments, this allows for impersonation, rogue DHCP servers, and could allow attackers to intercept or alter network traffic.

CreScan retrieves this setting and checks it against enterprise deployment best practices. If the feature is turned off, it is considered a critical misconfiguration. 802.1X not only secures the perimeter but also introduces accountability by linking

MAC addresses to authenticated users.

4.1.4 Time Synchronization (SNTP)

The validity of system logs, authentication time stamps, and certificate verification routines all depend entirely on synchronized timekeeping. SNTP is a less complex version of the more intricate NTP, and while it is less precise, it is very prevalent in embedded systems since it has lower overhead.

If SNTP is not implemented, devices will drift from their expected time values. This will invalidate the X.509certificates [42], interfere with authentication handshakes, and create inconsistencies in audit logs that complicate incident response. Accurate time is also required to identify replay attacks or correlate events across multiple systems.

CreScan determines whether SNTP synchronization is activated and whether a valid time source is configured. Vulnerable systems that have SNTP disabled or incorrectly configured are flagged with high priority, and the tool can reinstate synchronization with default or user-defined NTP servers if needed. SNTP tends to be minimized in significance, yet from a cyberengineering perspective, it is among the foundations of non-repudiation and traceability [59].

4.1.5 HTTPS Certificate Validation

The use of HTTPS provides encryption for communication with the client (e.g., technician or web interface) and controller. However, if the device provides an expired, self-signed, or otherwise untrusted certificate, very little actual protection is provided by the encryption. This opens doors for attackers to carry out MITM attacks via intercepted and manipulated traffic without notice.

CreScan conducts a TLS handshake with Python's crypto libraries and checks the presented certificate against a list of trusted certificate authorities. The tool extracts the certificate chain, expiry date, and subject. If the certificate is invalid or cannot be validated, the tool marks the HTTPS channel as untrusted.

Secure system design here relies on PKI and certificate validation to ensure authenticity. Every system that possesses self-signed certificates must ensure the certificate fingerprint is securely sent out-of-band. Otherwise, they may be spoofed and masqueraded. CreScan warns of such misconfigurations and recommends that the certificate be replaced by one signed by a trusted CA.

4.1.6 IPv6 Network Configuration

While IPv6 is a significant leap in routing enhancement and address assignment, its usage within an environment that is not fully IPv6-aware can lead to uncontrolled exposure. Firewall rule sets are primarily IPv4-centric, and having IPv6 enabled without corresponding inspection can mean open attack channels.

CreScan queries the OS configuration whether IPv6 is enabled and associated with any network interfaces. If it is enabled in a location where IPv6 is not required,

the utility recommends disabling it. IPv6 can be exploited by attackers to bypass conventional NAT-based impediments, discover what adjacent devices are present, and exploit protocol-specific weaknesses.

From a system hardening perspective, IPv6 must be specifically enabled when it is needed. Otherwise, the principle of least functionality [26] requires it to be removed in order to minimize the attack surface. The software applies the required commands automatically to disable IPv6 when it is abused.

4.1.7 TLS Configuration

TLS is the de facto standard for secure IP network communications. However, TLS 1.0 and 1.1 are no longer supported since they have multiple exploitable vulnerabilities in cipher suites and handshakes. Even if TLS 1.2 remains widely used, TLS 1.3 offers improved performance and security by eliminating weak algorithms.

CreScan attempts to establish a secure connection using TLS 1.3, the most recent and secure version of the protocol. If the controller does not support TLS 1.3, CreScan automatically falls back to the next highest version supported by both the client and the controller, usually TLS 1.2. Older versions such as TLS 1.1 or 1.0 are avoided because they are known to be vulnerable to attacks such as POODLE [60] and BEAST [61].

Since OWASP [32] and NIST SP 800-52r2 [62] recommend the use of TLS 1.3 and the restriction of cipher suites to strong algorithms, CreScan attempts to establish connections using TLS 1.3 and disables legacy versions such as TLS 1.0 and 1.1. TLS 1.3 provides simplified handshakes, mandatory forward secrecy, and the removal of weak ciphers and compression methods. Furthermore, configurations should prioritize secure cipher suites such as AES-GCM, which ensures both confidentiality and integrity [63], and ECDHE, which provides ephemeral key exchange and forward secrecy [64]. CreScan identifies potential misconfigurations and exposes these risks according to the aforementioned security guidelines.

4.1.8 ICMP exposure

ICMP is useful for diagnostics, but its openness can be abused by attackers during reconnaissance. ICMP echo requests (commonly known also as pings) allow attackers to determine which hosts are online, and ICMP timestamp or address mask requests can yield additional metadata.

CreScan checks whether the device responds to various ICMP types. If enabled and not rate-limited or filtered by a firewall, it represents a low-effort path for attackers to map network topology. Furthermore, ICMP can be exploited in reflective DDoS attacks, making it both a discovery and amplification vector.

Security frameworks like CIS Controls [65] recommend disabling ICMP where not explicitly required. The tool identifies this exposure and disables it automatically if permitted by the Crestron controller.

4.1.9 Authentication Requirement

Requiring authentication to access a device's management interface is non-negotiable in any secure system. Without this, an unauthenticated actor can access logs, change configurations, or reset device states.

CreScan examines the global authentication flag using SSH. If authentication is disabled, the system effectively operates in an open mode. This violates all modern access control models and is typically only acceptable in lab or demo environments. If found disabled, the tool issues a critical severity finding and offers a remediation pathway through the autofix mechanism.

4.1.10 Brute-Force Protection

Defending against brute-force attacks involves limiting how many times a user can fail authentication within a given time frame. Without such controls, attackers can script infinite credential attempts until the correct one is found [21].

CreScan reads the login policy parameters and evaluates whether the number of failed attempts is capped—usually at five. This is consistent with CIS Benchmarks [55] and many vendor hardening guides. A properly configured threshold significantly reduces the feasibility of brute-force attacks, particularly when combined with a sufficient password policy.

4.1.11 Brute-Force Lockout Time

The lockout timer complements brute-force defense by introducing a delay after the threshold is reached. A long delay increases the cost of attack and reduces the likelihood of success through automation.

As recommended by NIST SP 800-63B [57] and OWASP Authentication Guidelines [32], account lockout durations should be sufficient to prevent brute-force attacks while avoiding permanent denial of service for legitimate users. Accordingly, CreScan evaluates whether the lockout time is set to a minimum of 15 minutes.

4.1.12 Audit Log Configuration

Audit logs are crucial for tracking user actions, system changes, and security-relevant events on the controller. Without proper logging, incident investigations and forensic analyses are severely hindered, and compliance with industry standards such as ISO/IEC 27001 [56] or NIST SP 800-92 [59] cannot be achieved.

As emphasized also by ISO/IEC 27002:2022 [66], audit logging is critical to ensure traceability, non-repudiation, and detection of security incidents. Accordingly, CreScan inspects whether audit logging is enabled and whether logs capture all critical events, such as configuration changes, authentication attempts, and firmware updates. Systems that do not generate audit logs or limit logging to non-critical events are flagged with a high severity score.

4.1.13 Remote Syslog Transmission

Sending audit logs to a remote syslog server enhances security monitoring and centralizes log management, reducing the risk of log tampering or loss due to device compromise. Remote logging is also a prerequisite for SIEM integration and automated alerting.

CreScan verifies the configuration of remote syslog settings, including server address, port, protocol (TCP/UDP), and minimum severity level. If remote logging is disabled or misconfigured, the system is considered at elevated risk because local logs alone may not provide timely detection of incidents.

4.1.14 Risk Level Assessment Criteria

Each vulnerability identified by CreScan is assigned a numerical risk score, which serves as a quantitative representation of its severity. The scoring scale ranges from 0 to 10, where higher values indicate more critical security concerns that demand immediate attention. This approach allows for prioritization of remediation actions based on technical risk rather than simple rule matching.

Risk scores are calculated using predefined criteria that account for configuration values, feature status, and compliance with security standards. While some vulnerabilities are binary in nature (e.g., authentication enabled or not), others allow for gradation, where intermediate configurations lead to moderate risk levels. For example, a password policy that enforces length but not complexity is considered less severe than a policy that enforces neither.

Table 4.1 summarizes the evaluation logic used to classify each test case, providing insight into how each configuration issue is translated into a risk score.

Table 4.1: Risk Evaluation Criteria for Each Vulnerability.

Vulnerability	**Risk Conditions and Evaluation Criteria**	**Score**
Authentication Requirement	Management interface is accessible without login.	10
Brute-Force Protection	Missing: High.	8
	Only partially enforced: Moderate.	6
802.1X Authentication	Network access allowed without authentication.	5
HTTPS Certificate Validation	Expired or self-signed certificate presented.	9
Password Policy	Length < 8 and no complexity: High.	9
	Only one weakness: Moderate.	6
TLS Configuration	Only TLS 1.0/1.1 supported: High.	10
	Fallback to TLS 1.2 allowed: Moderate.	2
Firmware Integrity	Signature verification is disabled during firmware updates.	5
ICMP Exposure	Device responds to ping or other ICMP queries.	5
IPv6 Network Configuration	IPv6 enabled in unmanaged or unnecessary environments.	5
Brute-Force Lockout Time	Lockout missing	8
	Less than 10 minutes	2
Time Synchronization (SNTP)	SNTP disabled	8
	Misconfigured server	5
Audit Log Configuration	Audit Logs disabled: no traceability on the system	8
Remote Syslog Transmission	Remote syslog server not configured, reduced centralized monitoring capability.	6

Chapter 5

Experimental Setup

This chapter describes the experimental setup and validation process adopted to assess the performance, reliability, and coverage of the CreScan tool. The evaluation combines controlled laboratory tests with real-world audits on operational Crestron systems, ensuring that both functional correctness and practical applicability are thoroughly verified.

section 5.1 details the testing environment, including the hardware platforms, network configuration, and laboratory conditions used to perform reproducible experiments across multiple controller models. section 5.2 defines the evaluation criteria and metrics used to quantify tool performance, focusing on execution time, rule coverage, autofix effectiveness, and device compatibility. Finally, section 5.3 presents a series of representative use case scenarios designed to emulate real deployment conditions, illustrating how CreScan identifies misconfigurations, prioritizes risks, and validates remediations across different security postures.

5.1 Testing Environment

To ensure the relevance and reliability of the experimental results, the CreScan tool was tested across multiple Crestron controller models and deployment scenarios. In terms of hardware, the evaluation encompassed three different models of Crestron control processors: the CP4 [67], its predecessor the CP3 [68], and the compact MC4 [69]. These devices differ significantly in terms of computational capabilities, memory, and networking features, which provided a diverse hardware base for testing.

The detailed technical specifications of each controller model are presented in Tables 5.1, 5.2, and 5.3, respectively. This information is relevant to contextualize the performance results, particularly in relation to execution speed and protocol compatibility.

All tests, including scanning, remediation, and report generation, were executed from a dedicated auditing workstation. This host was connected to the same subnet as the target Crestron controllers, ensuring optimal communication conditions over SSH and HTTPS. The system was equipped with modern hardware and a reliable operating environment to avoid performance bottlenecks during experimentation.

Table 5.1: Technical specifications – Crestron CP3

Specification	Value
Operating System	Linux-based
CPU	3-Series real-time engine
RAM	512 MB SDRAM
Flash Memory	4 GB
Networking	10/100 Mbps Ethernet
Supported IP Versions	IPv4/IPv6
Power Consumption	Not specified

Table 5.2: Technical specifications – Crestron CP4

Specification	Value
Operating System	Linux-based
CPU	4-Series multicore processor
RAM	2 GB SDRAM
Flash Memory	8 GB
Networking	Gigabit Ethernet
Supported IP Versions	IPv4/IPv6
Power Consumption	15 W

The technical specifications of the auditing machine are reported in Table 5.4.

In real-world installations, the CP4 model was deployed in three distinct environments: a residential smart home, an enterprise meeting room, and a luxury yacht. These production setups offered valuable insight into the variability of real deployments, where configurations are influenced by system integrator practices, firmware versions, and contextual requirements. The CP4 controller used in the field is shown in Figure 5.2.

For laboratory-based experiments, a CP3 and an MC4 unit were used to simulate various configurations. In some cases, actual project files from deployed Crestron systems were imported into the test devices to replicate realistic operational states. In other cases, specific misconfigurations were intentionally introduced to assess how CreScan handles edge-case conditions and security lapses. The lab equipment is shown in Figures 5.1 and 5.3, respectively. These controlled experiments enabled repeatability, precision measurement, and safe testing of critical scenarios.

All tests were conducted over SSH and HTTPS communication channels. During laboratory testing, the Crestron controllers were connected to an isolated subnet with no external traffic, ensuring a controlled environment and minimizing side-channel variability.

5.2 Evaluation Metrics

To rigorously evaluate the security auditing tool developed in this thesis, it is essential to define metrics that reflect both its functional correctness and its operational

Table 5.3: Technical specifications – Crestron MC4

Specification	Value
Operating System	Linux-based
CPU	4-Series multicore processor
RAM	1 GB SDRAM
Flash Memory	4 GB
Networking	Gigabit Ethernet
Supported IP Versions	IPv4/IPv6
Power Consumption	7 W (PoE)

Table 5.4: Specifications of the auditing workstation

Component	Specification
Processor	Intel Core i5, 12^{th} Generation
RAM	16 GB DDR4
Storage	1 TB SSD
Network Interface	1 Gbps Ethernet
Operating System	Windows 11 Pro

practicality. The selected performance metrics are designed to capture how effectively and efficiently the tool performs its intended tasks in real-world Crestron environments. Each metric offers a distinct perspective on the tool's value from both a technical and cybersecurity standpoint.

Execution Time. The time required to complete an audit cycle is a fundamental metric for determining whether the tool is viable for practical use in field deployments. Excessively long execution times may discourage adoption by technicians or disrupt service availability during audits. Measuring the average execution time helps to assess the tool's responsiveness and identify potential optimizations in command processing and report generation. A tool that provides timely results is more likely to be adopted as part of routine maintenance and security checks.

Rule Coverage. This metric captures the number of security rules that are actively implemented and tested during an audit session. Since the tool is built on a rule-based architecture, its effectiveness depends heavily on the breadth of its rule set. A high rule coverage indicates that the tool is capable of addressing a wide spectrum of known configuration vulnerabilities and aligns with established security frameworks. In contrast, poor coverage may leave critical issues undetected, reducing the audit's reliability and comprehensiveness.

Autofix Effectiveness. One of the key differentiators of the tool is its ability to not only detect but also resolve certain vulnerabilities. The autofix effectiveness metric evaluates how many of the identified, fixable issues were successfully corrected by the system. This helps validate the reliability of the remediation commands and the correctness of the logic that triggers them. A high autofix success rate implies that the tool can serve not just as a diagnostic instrument, but also as a proactive defense mechanism—significantly reducing technician workload and human error.

Figure 5.1: Crestron CP3 control processor used in laboratory testing.

Figure 5.2: Crestron CP4 control processor used in real-world installations.

Device Compatibility. Finally, it is important to ensure that the tool works reliably across different Crestron controller models. This metric evaluates whether all components of the tool—auditing, parsing, and remediation—function as expected regardless of firmware differences or hardware constraints. Compatibility is a practical requirement for adoption, especially in environments where mixed device generations are deployed. A tool that fails to operate uniformly across supported platforms risks losing credibility and limiting its real-world applicability. Testing was focused on three representative devices — CP3, CP4, and MC4 — selected to cover both the 3-Series and 4-Series architectures and to reflect the most common deployments found in professional AV and automation systems. Specifically, the CP3 represents the previous generation of 3-Series controllers, widely used in educational and corporate environments; the CP4 embodies the modern 4-Series hardware with higher computational capabilities and updated network interfaces; and the MC4 provides a compact, PoE-powered alternative suited for small to medium installations. This combination ensures that the evaluation includes both legacy and current architectures, offering a realistic assessment of tool compatibility across heterogeneous Crestron systems.

5.3 Use Case Scenarios

To validate the effectiveness and coverage of CreScan under realistic operational conditions, a series of tests were conducted on three real-world installations: one residential smart home, one enterprise meeting room, and one luxury yacht. In addition to these production environments, five use case scenarios were carefully designed to reflect typical deployment patterns, configuration oversights, and varying levels of security awareness commonly encountered in the field. These scenarios were defined based on hands-on experience with Crestron environments, observed technician behaviors, and recurring misconfigurations reported during audits. The

Figure 5.3: Crestron MC4 control processor used in laboratory testing.

goal is to test the tool's ability to detect security weaknesses, recommend corrective actions, and deliver value beyond what is achievable with traditional diagnostic tools that do not focus on cybersecurity.

5.3.1 Use Case 1: Default Installation

This scenario simulates the most basic real-world condition: a freshly deployed Crestron controller using default factory settings. It reflects a situation where an inexperienced installer, or one under time pressure, connects the device to the network without applying security hardening measures. Since different firmware versions and hardware models may ship with slightly varying default configurations, the specific vulnerabilities present in this case are not assumed a priori. The objective is to observe whether the tool can autonomously identify and prioritize misconfigurations out-of-the-box, without any manual input or assumptions. This represents a critical baseline, as many real-world systems are deployed in this state and remain operational for long periods before receiving any review.

Table 5.5 lists the vulnerabilities present in the Default Installation scenario (UC1).

Table 5.5: UC1 — Default installation: vulnerabilities detected

Vulnerability	**Severity**	**Description**
Firmware signature	High	Signature check not enabled; firmware integrity at risk
802.1X authentication	High	Network access allowed without 802.1X authentication
HTTPS certificate validation	High	Self-signed or expired certificate in use
IPv6 configuration	Low	IPv6 enabled unnecessarily
ICMP exposure	Medium	Device responds to ping; potential information disclosure
Excessive login attempts	Medium	Brute-force protection missing
Brute-force lockout time	Medium	Lockout threshold not configured
Audit Logs	High	Audit logs not configured
Remote Syslog	Medium	Remote Syslog not configured

5.3.2 Use Case 2: Debug Leftovers

In professional maintenance or troubleshooting contexts, technicians may temporarily disable certain protections—such as authentication enforcement or TLS restrictions—to perform debugging or firmware updates. However, these changes are sometimes not reverted once maintenance is complete. This use case captures such a scenario, where the system remains in a weakened state due to forgotten or untracked modifications. CreScan is expected to identify these residual risks, distinguishing between temporary debugging conditions and actual misconfigurations.

In this configuration, missing authentication controls, outdated certificates, and unconfigured audit mechanisms are evident, reflecting an environment where functionality was prioritized over security hardening.

Table 5.6: UC2 — Debug Leftovers: vulnerabilities detected

Vulnerability	**Severity**	**Description**
802.1X authentication	High	Network access allowed without 802.1X authentication
HTTPS certificate validation	High	Self-signed or expired certificate in use
TLS version	High	TLS 1.0 still enabled; requires firmware update
ICMP exposure	Medium	Device responds to ping; potential information disclosure
Excessive login attempts	Medium	Brute-force protection missing
Brute-force lockout time	Medium	Lockout threshold not configured
Audit Logs	High	Audit logs not configured
Remote Syslog	Medium	Remote Syslog not configured

5.3.3 Use Case 3: Partial Hardening

This scenario represents installations where security is applied inconsistently—either because of limited awareness or incomplete adherence to internal guidelines. Basic protections such as authentication and password management may be correctly implemented, but secondary controls like network filtering, time synchronization, or logging are often overlooked. CreScan's ability to recognize these gaps demonstrates its granularity: not only identifying missing configurations, but also evaluating whether implemented controls are complete and aligned with best practices.

Table 5.7 shows a limited set of vulnerabilities, mainly related to network exposure and missing monitoring capabilities, typical of environments that were hardened only at a superficial level.

Table 5.7: UC3 — Partial Hardening: vulnerabilities detected

Vulnerability	Severity	Description
SNTP synchronization	Medium	NTP synchronization failed; clock drift may affect event correlation
IPv6 configuration	Low	IPv6 enabled unnecessarily
ICMP exposure	Medium	Device responds to ping; potential information disclosure
Remote Syslog	Medium	Remote Syslog not configured

5.3.4 Use Case 4: Legacy Firmware Deployment

Many production environments continue operating with outdated firmware, often due to concerns about compatibility or stability. This scenario represents such a system, running on a legacy firmware version that lacks modern security capabilities. In this case, vulnerabilities are not the result of poor configuration, but of inherent software limitations. CreScan's role is to identify and clearly report these structural weaknesses, providing evidence of where security cannot be improved without a firmware upgrade. The results reveal missing cryptographic enforcement, outdated TLS versions, and absent brute-force protection mechanisms—typical of legacy deployments where critical security functions were not yet part of the platform design.

Table 5.8 shows a broader set of vulnerabilities, primarily linked to outdated firmware limitations and missing core protections, which are characteristic of legacy deployments that predate modern security standards.

Table 5.8: UC4 — Legacy Firmware: vulnerabilities detected

Vulnerability	Severity	Description
Firmware signature	High	Signature check not enabled; firmware integrity at risk
802.1X authentication	High	Network access allowed without 802.1X authentication
HTTPS certificate validation	High	Self-signed or expired certificate in use
TLS version	High	TLS 1.0 still enabled; requires firmware update
ICMP exposure	Medium	Device responds to ping; potential information disclosure
Excessive login attempts	Medium	Brute-force protection missing
Brute-force lockout time	Medium	Lockout threshold not configured
Audit Logs	High	Audit logs not configured
Remote Syslog	Medium	Remote Syslog not configured

5.3.5 Use Case 5: Fully Compliant Reference Configuration

This scenario was deliberately designed as a controlled reference case to represent an enterprise environment configured according to best practices. Unlike the previous scenarios, its purpose was not to expose weaknesses, but to validate CreScan's ability to correctly recognize a secure configuration and produce a clean result. All relevant protections—authentication enforcement, TLS configuration, certificate validity, and brute-force prevention—were intentionally implemented to reflect an ideal compliance state. The absence of vulnerabilities in this case therefore confirms that CreScan can accurately distinguish between misconfigurations and correctly secured systems, ensuring that no false positives are reported when the environment meets the expected security baseline.

Chapter 6

Results and Findings

This chapter presents the comparative evaluation of CreScan against the official Crestron Security Audit Tool across CP3, CP4, and MC4 controllers. The analysis focuses on performance, coverage, and functional behavior under identical testing conditions and at the highest security profile offered by the vendor utility.

section 6.1 examines execution times across scenarios and controller families, demonstrating CreScan's improved efficiency and runtime consistency compared to the vendor tool. section 6.2 evaluates rule coverage, highlighting differences in applicability and structural design between the two auditing frameworks. section 6.3 analyzes the effectiveness of CreScan's automated remediation subsystem, quantifying its impact on reducing manual intervention. section 6.4 discusses cross-device compatibility, focusing on how both tools handle variations in firmware capabilities and platform features. Finally, section 6.5 provides an overall comparison of scope, reporting structure, and diagnostic granularity, establishing a normalized domain-based mapping between the Crestron and CreScan rule sets.

6.1 Execution Time

This section analyses execution time across controllers and deployment contexts. Results are shown per run, paired by scenario and controller, as illustrated in Figure 6.1, and summarized numerically in Table 6.1.

CreScan completes its workflow with consistently shorter runtimes than the vendor utility across all runs. As reported in Table 6.1, the average execution time of CreScan is 28.7 s, compared to 41.7 s for the Crestron Security Audit Tool, corresponding to an average improvement of approximately **31% faster execution**. The difference remains consistent across controller families: on CP3, CreScan averages 26.8 s against 43.4 s for the vendor utility (a 38% reduction); on MC4, 26.2 s compared to 40.4 s (35% faster); and on CP4 in real deployments, 27.7 s compared to 40.7 s (32% faster).

The gap is visible both on legacy and modern controllers, with the largest differences observed when no remediation is needed (e.g., well-configured environments), where CreScan approaches its baseline execution. The execution time of UC5 shows that, when the remediation phase is triggered, CreScan introduces a small and pre-

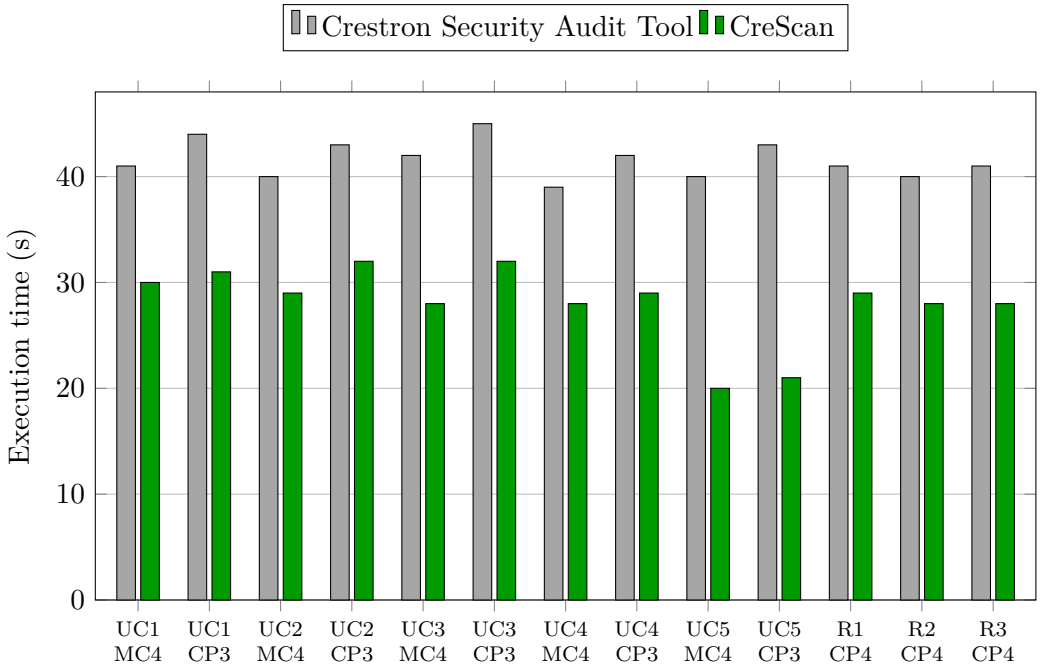

Figure 6.1: Execution time per scenario–controller pair. CreScan (green) consistently completes faster than the Crestron utility (gray).

dictable overhead, yet remains faster overall. In this specific use case, however, the execution time is considerably lower than the average(8s faster), as UC5 contains no detected vulnerabilities. This observation suggests that the reduction in processing time is directly related to the absence of the autofix phase, since no corrective actions are required once the auditing and parsing stages are completed. Consequently, the total runtime in this scenario reflects only the baseline performance of the scanning and analysis components, without the additional computational cost introduced by automated remediation routines. Controller-specific effects are limited: the vendor utility shows slightly longer runs on CP3, reflecting reduced feature support and compatibility handling, while CreScan maintains stable timings across CP3, CP4, and MC4. Real-world CP4 installations exhibit the same pattern as the controlled environments, indicating that network and platform variability do not materially alter the relative performance between the two tools.

Overall, the data in Table 6.1 and Figure 6.1 demonstrate that CreScan achieves a substantial and consistent reduction in execution time, averaging roughly one-third faster than the vendor utility without compromising completeness or accuracy.

6.2 Rule Coverage

Rule coverage is assessed along two complementary dimensions: the applicability of checks on each platform and the number of rules effectively evaluated. Results are interpreted by security domain to ensure a like-for-like comparison between the granular profile of the Crestron Security Audit Tool and the consolidated rule set of CreScan.

Table 6.1: Execution time comparison between CreScan and Crestron Utility across all test scenarios

ID	Controller	CreScan (s)	Crestron Utility (s)
UC1	MC4	30	41
UC1	CP3	31	44
UC2	MC4	29	40
UC2	CP3	32	43
UC3	MC4	28	42
UC3	CP3	32	45
UC4	MC4	28	39
UC4	CP3	29	42
UC5	MC4	20	40
UC5	CP3	21	43
R1	CP4	29	41
R2	CP4	28	40
R3	CP4	28	41
Average	**(all runs)**	**28.7**	**41.7**

Three points are noteworthy. First, CreScan preserves full applicability across CP3, CP4, and MC4, indicating that its checks are formulated at a stable abstraction level and are resilient to firmware and platform differences. Second, the vendor tool shows a noticeable drop on the CP3 controller, where a subset of Level 4 security checks is no longer available. This reduction does not indicate a failure of the Crestron tool itself, but rather reflects the hardware and firmware limitations of the CP3, which is an older and now unsupported device. As a result, the number of security checks applicable to the CP3 is inherently lower compared to more recent controllers such as the CP4 and MC4, where the latest firmware supports the full set of advanced verification routines. This difference is summarized in Table 6.2, which reports the number of security checks performed by each tool on the tested controllers. As shown, the Crestron Utility performs up to twenty checks on the MC4 and CP4, but only eighteen on the CP3, while CreScan maintains a consistent set of thirteen checks across all platforms, ensuring uniform coverage regardless of hardware generation. Third, differences in absolute rule counts stem from design granularity rather than domain gaps: Crestron expands policy areas into multiple atomic items, while CreScan groups parameters under single domain checks. In practice, CreScan's constant applicability reduces variance and simplifies remediation planning, whereas the vendor tool requires normalizing results by domain to avoid over-weighting non-applicable items on legacy hardware.

Table 6.2: Number of security checks performed by each tool per controller type

Controller	Crestron Utility (checks)	CreScan (checks)
MC4	20	13
CP3	18	13
CP4	20	13

6.3 Autofix Effectiveness

CreScan implements automated remediation whenever a finding can be corrected locally on the controller (e.g., disabling unnecessary ICMP responses, enforcing lockout policies, enabling time synchronization, deactivating unused IPv6). Issues that depend on external infrastructure or organizational policy (e.g., enterprise 802.1X or replacement of untrusted HTTPS certificates) are reported but left for manual remediation. The Crestron Security Audit Tool does not provide autofix capabilities and is therefore excluded from this part of the analysis.

Table 6.3 reports, for each scenario (UC1–UC5 and R1–R3), the number of vulnerabilities detected by CreScan and the subset that were automatically remediated. Not all vulnerabilities can be addressed through the autofix mechanism. Certain findings, such as disabled audit logs or the presence of an active IPv6 configuration, represent binary settings that can be safely toggled by the tool without additional information. Others require manual intervention by a technician because they depend on environment-specific parameters, for example the IP address of the syslog or NTP server, which cannot be generically inferred or applied. Each count of detected issues is expressed with respect to the complete set of checks executed by the tool, which, as detailed in Section *Rule Coverage*, maintains full applicability across all controller families. This representation highlights not only the absolute number of misconfigurations found but also their proportion within a consistently evaluated rule set, allowing a direct comparison between scenarios without bias from platform-dependent variations.

Across all simulated and real-world scenarios, CreScan identified an average of 4.8 misconfigurations per run and automatically corrected an average of 3.5, yielding an overall mean autofix rate of approximately 69%. The highest remediation success is seen in controlled use cases (UC1–UC4), where the majority of vulnerabilities that could be fixed locally were resolved. In real deployments (R1–R3), the rate is lower due to external dependencies and safety constraints preventing full automation. This confirms that CreScan effectively translates detection into tangible hardening improvements when device-local conditions allow, substantially reducing the manual effort required to achieve compliance.

Table 6.3: CreScan: detected and automatically remediated vulnerabilities per scenario

Scenario ID	Detected	Auto-fixed	Fixed (%)
UC1	7/13	6	85.7
UC2	6/13	6	100.0
UC3	7/13	3	42.9
UC4	3/13	3	100.0
UC5	0/13	0	–
R1	4/13	2	50.0
R2	4/13	2	50.0
R3	3/13	2	66.7
Average	4.8	3.5	69.0

6.4 Device Compatibility

Device compatibility was assessed across the CP3, CP4, and MC4 controller families, focusing on how each platform exposes security-relevant configuration primitives and how the tools behave when features are missing or implemented differently. The discussion below summarizes the observed behavior without relying on platform-specific heuristics or ad-hoc exceptions. On modern platforms (MC4 and CP4) both tools execute their intended security checks without functional gaps. On legacy CP3, the Crestron Security Audit Tool shows a reduced set of applicable checks at the highest profile, where a few controls are flagged as not supported and skipped. CreScan, in contrast, keeps uniform applicability of its consolidated rule set across all three families because rules are expressed at the level of security domains (for example password policy, transport security, time synchronization) rather than depending on version-specific toggles. Compatibility also emerges in the remediation path. Controls that are purely local to the controller are supported in a consistent manner and are reliably remediated by CreScan on CP3, CP4, and MC4: enforcing authentication, configuring lockout thresholds and timings, disabling unnecessary ICMP responses, disabling unused IPv6, and enabling stable SNTP synchronization. Items that depend on external infrastructure or organizational policy remain intentionally non-automated on all platforms, such as enabling 802.1X where a NAC backend is absent or replacing untrusted HTTPS certificates without access to the site PKI. Residual findings therefore reflect environmental dependencies rather than device incompatibilities. A small subset of differences is attributable to firmware capabilities rather than hardware families. In the legacy-firmware configuration, the usual constraints appear: lack of configurable brute-force policy primitives, partial or deprecated TLS negotiation that prevents full deprecation of older protocol versions, and the absence of a runtime switch for firmware signature enforcement on certain builds. CreScan detects and reports these conditions uniformly; where a control cannot be enforced locally, the

report escalates with a clear recommendation to upgrade firmware or adjust policy. All controllers were audited over SSH and HTTPS. Across families, CreScan exhibited stable behavior with predictable execution characteristics and no transport-specific fallbacks. The vendor utility also operated reliably, but on CP3 it short-circuits or omits checks that are flagged as unsupported, which affects comparability unless results are interpreted with applicability in mind. From an operational perspective, CreScan's device-agnostic rule set simplifies rollout in heterogeneous fleets: the same policy definitions apply to CP3, CP4, and MC4, and the reporting structure remains identical regardless of platform age. Where incompatibilities arise, they are explicitly tied to firmware features or environmental prerequisites, which clarifies ownership for remediation (software upgrade versus infrastructure change). The Crestron Security Audit Tool remains useful for confirming vendor-specific settings on supported platforms, but its reduced applicability on legacy devices requires additional interpretation to avoid understating risk in mixed deployments.

6.5 Comparison with the Crestron Security Audit Tool

To establish a comprehensive and technically equal evaluation, the comparison between CreScan and the official Crestron Security Audit Tool is based exclusively on Security Level 4, the most complete profile offered by the latter. As discussed earlier, Level 4 includes all lower-level checks and enforces advanced measures such as 802.1X certificate-based authentication, audit log integration, and complex password policies.

While both tools aim to assist technicians in securing Crestron environments, their approach, scope, and outputs differ significantly.

One major distinction lies in the concept of remediation. CreScan includes an automated remediation system, which attempts to fix detected misconfigurations whenever safe and possible. By contrast, the Crestron tool does not offer any form of autofix. The only post-scan action available is a shortcut to open the device's configuration page through the Toolbox interface, leaving the burden of interpretation and correction entirely to the technician.

Another key difference concerns reporting. The Crestron Audit Tool generates a plain CSV file listing each check, its pass/fail status, and a timestamp. While useful for internal documentation, this format lacks essential cybersecurity context—such as a qualitative risk level, severity ranking, or structured remediation guidelines. Industry standards like the Common Vulnerability Scoring System (CVSS) explicitly recommend assigning a severity class to each vulnerability as a way to support risk communication and remediation prioritization [70].

By contrast, CreScan assigns a severity level to each detected issue based on security impact and criticality, making it easier to identify which vulnerabilities require urgent intervention. For example, a technician using CreScan will immediately see that missing authentication or disabled firmware signature verification are marked as "Critical," while less urgent issues—such as ICMP exposure—may be classified as

Column1	Column2	Column3	Column4	Column5	Colum	Column7
Hostname / IP	Model	S/N	FW Version	Security Setting	Status	Resolve By
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Recommended FW	Passed	The Firmware meets the Recommended Vers
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	SSH	Passed	SSH enabled
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Authentication	Passed	Authentication enabled
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Active Directory Group	Failed	There must be at least 1 Active Directory Grou
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Local Users	Passed	Number of Local Users is sufficient
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Block IP Lockout Time	Passed	Lockout Time is sufficient
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Passwords - Max Atter	Failed	The maximum number of login attempts mus
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Passwords - Min Lengt	Passed	Minimum Password Length is sufficient
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Passwords - Mixed Ca	Failed	The Password Policy should require mixed ca
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Passwords - Numeric	Failed	The Password Policy should require numeric
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	Passwords - Special C	Failed	The Password Policy should require special c
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	User Idle Time	Passed	User Idle Time is sufficient
172.16.130.179	MC4	="2330JBH07318"	2.8005.00(	SSL	Passed	SSL enabled

Figure 6.2: Example of the Crestron Audit Tool output in CSV format.

"Medium." This prioritization mechanism allows for faster decision-making and more efficient resource allocation during audits.

At first glance, the Crestron Audit Tool—particularly at Security Level 4—appears to conduct a larger number of individual checks compared to CreScan. As detailed earlier, Level 4 includes a total of 28 separate checks. In contrast, CreScan operates on a set of 11 well-defined test cases.

However, this numerical disparity is largely due to the structural design of the Crestron audit system. Many of the 28 checks are highly granular or redundant in nature. For instance, password policy enforcement is broken down into separate checks for minimum length, use of mixed case, numeric characters, and special characters—each treated as an individual result. Similarly, 802.1X authentication is evaluated through three distinct rules, covering mode activation, server validation, and certificate authentication.

In contrast, CreScan adopts a more structured and semantically grouped rule set. It consolidates related parameters into unified checks that reflect actual security domains, such as "Password Policy" or "802.1X Authentication." This not only reduces redundancy but aligns better with how vulnerabilities are typically modeled in security engineering and compliance frameworks. Furthermore, several Crestron checks included in Level 4 pertain to operational state or configuration hygiene, such as firmware version recommendation, active directory group listing, or local user enumeration, which, while informative, do not directly represent exploitable security flaws.

CreScan deliberately omits these non-critical checks in favor of focusing exclusively on configuration aspects that expose the device to real attack vectors. Additionally, CreScan introduces severity classification and remediation logic features entirely missing from the Crestron tool.

To ensure fairness and consistency during the evaluation, a normalization strategy was adopted to compare the outputs of the Crestron Audit Tool (Level 4) and the CreScan tool. Since the Crestron audit performs a more granular series of checks—some of which are operational rather than security-critical—a one-to-one comparison would be misleading. Instead, the comparison focuses on logical security domains.

This normalization allows for a domain-based evaluation, measuring coverage

in terms of security functions rather than raw test count. Table 6.4 presents the mapping between grouped Crestron checks and corresponding CreScan test cases.

Table 6.4: Mapping of Crestron Audit checks (Level 4) to CreScan test cases

Crestron Audit Checks (Level 4)	**Equivalent CreScan Test Case**
Password Min Length Password Max Attempts Password Policy – Mixed Case Password Policy – Numeric Password Policy – Special Characters	Password Policy (unified check)
Authentication Enabled	Authentication Requirement
Block IP Lockout Time User Idle Time	Brute-Force Lockout Time
SSL Enabled SSL CA Certificate SSL Fallback Disabled	HTTPS Certificate Validation and TLS Version
SNTP Enabled	SNTP Configuration
802.1X Mode Enabled 802.1X Server Validation 802.1X Certificate Authentication	802.1X Authentication
AutoDiscovery Disabled	ICMP Exposure
Web Server Disabled	Not checked by CreScan
Local Users Active Directory Groups	Not checked by CreScan (informational only)
Audit Log Enabled Audit Log Level Sufficient System Log Enabled System Log Security Level Sufficient System Log Includes Audit Log	Audit Logs Remote Syslog
Recommended Firmware Version	Not checked by CreScan (not a vulnerability)

Chapter 7

Discussion

CreScan was designed to address a practical challenge: auditing and improving the security posture of Crestron controllers in real-world deployments. Unlike vendor-provided tools, which often focus on granular configuration checks or legacy compliance, CreScan adopts a domain-oriented approach that better reflects how technicians and auditors assess risk in practice. Each rule is mapped to a clear security domain, such as authentication, time synchronization, transport security, or legacy protocol exposure. This mapping avoids the fragmentation seen in tools that split policies into multiple sub-checks, reducing noise in the results and maintaining consistency across controller generations. On older platforms, vendor tools may skip unsupported features silently, which can lead to incomplete or misleading outputs. CreScan ensures uniform applicability across CP3, CP4, and MC4 platforms, as demonstrated in Chapter 6.

A key strength of CreScan is its ability to perform automatic remediation where feasible. Local issues—such as enabling lockout thresholds, disabling ICMP, or configuring SNTP—are corrected and verified automatically. Vulnerabilities that require external infrastructure, like RADIUS servers or trusted certificates, are detected and reported with actionable guidance. This balance ensures that configurations improve without risking service disruption or exceeding operational boundaries.

Table 7.1 quantifies the impact of automated remediation. Across all scenarios, average risk dropped from 4.13 to 2.18, representing a reduction of approximately 47%. Even in simulated or real deployments, residual risk is substantially lowered, confirming that CreScan not only identifies weaknesses but actively strengthens security posture.

CreScan emphasizes high-impact configuration states directly affecting the attack surface. While vendor tools may report additional operational or informational items, these rarely change practical risk assessment but can increase output volume. By concentrating on authentication enforcement, time synchronization integrity, transport security, and legacy service exposure, CreScan provides clear, actionable insights with minimal noise. Execution is consistent across platforms, with predictable timing and low variance, even when remediation is invoked. Vendor tools may experience longer, less predictable runs, particularly on older hardware or in compatibility modes.

Table 7.1: Average risk per scenario before and after CreScan autofix

Scenario ID	Risk Before Autofix	Risk After Autofix	Risk Reduction (%)
UC1	5.00	2.90	42%
UC2	6.00	3.00	50%
UC3	3.20	1.40	56%
UC4	5.90	3.20	46%
UC5	0.00	0.00	0%
R1	4.20	2.30	45%
R2	5.00	2.80	44%
R3	3.80	1.90	50%
Average	4.13	2.18	47%

Cross-platform results are stable: CP4 and MC4 devices yield homogeneous findings, while CP3 occasionally shows reduced applicability in vendor tools; CreScan preserves logical coverage consistently. Environmental factors, such as time-source reachability or certificate trust chains, influence specific outcomes without altering overall trends, as demonstrated in Chapter 6.

Finally, CreScan's reporting format enhances practical utility. Findings are structured by severity, separated into resolved and unresolved issues, and include explicit remediation guidance. This allows technicians to act promptly and auditors to trace changes over time. In contrast, vendor tools provide flat listings with no prioritization or remediation hints, requiring additional triage.

CreScan deliberately avoids risk-prone operations or unnecessary complexity. It does not scan the network, fuzz protocols, or attempt full compliance scoring. By focusing strictly on controller configuration, it remains safe, predictable, and easily integrated into maintenance workflows.

Chapter 8

Conclusion

Smart home systems and the IoT are becoming more and more common, and they bring a lot of benefits. Homes are more comfortable, automation is easier, and managing the environment can be done more efficiently. But at the same time, all these advantages come with serious cybersecurity issues. Most connected devices have limited processing power and memory, they use different types of communication protocols, and often they come with weak default settings. This makes them easy targets for attackers. It is possible, for example, that an attacker could get access to the system without authorization, read sensitive information, or even take control of important devices like cameras, locks, or alarms. Things get even worse because security updates are not always done regularly, and many users don't really understand how to keep their systems safe. International standards like ISO/IEC 27400:2022 suggest that the best approach is to combine secure configurations, constant monitoring, strong authentication, and careful handling of credentials, but in practice, this is not so easy to do.

To try and solve these problems, this thesis developed CreScan, a tool made for auditing and fixing security issues in Crestron systems. CreScan works in a modular way and is rules-based. It checks system settings, communication protocols, and other security configurations following established best practices. The tool can run automated scans, find vulnerabilities, give them a risk level, and produce reports that are detailed but also understandable even for people who are not security experts. One feature that stands out is the autofix option, which can automatically correct some important configuration issues. This feature helps reduce risks and makes the whole system more secure.

Based on the results of the tests conducted, CreScan showed that it can effectively detect and classify security vulnerabilities. With its detailed checks, risk classification, and practical suggestions, it gives technicians a much clearer idea of what is going on and allows them to act faster in comparison with Crestron security audit tool. It also makes the auditing process simpler and quicker, which is very useful when dealing with complex smart home installations. In general, CreScan improves on existing solutions and addresses many of the security concerns discussed in the background, while also promoting better cybersecurity habits.

8.1 Future Work

Even if this thesis has already demonstrated how CreScan can be a valuable tool for security assessment, several improvements could further increase the tool's flexibility and value in the field.

First, mapping CreScan rules to well-known frameworks (such as CIS Controls or ISO standards) would allow organisations to reuse results for formal compliance reporting. Second, optional integrations with external systems—like checking NTP reachability, certificate trust chains, or switch-level 802.1X configuration—would reduce false positives caused by missing prerequisites.

Another useful extension would be support for policy templates, so organisations can define security baselines across multiple sites and track drift over time. The remediation engine could also be enhanced with safer execution models, such as dry-run previews, transactional changes, and rollback on failure.

Future work will include extending the testing campaign to additional Crestron controllers, such as the RMC4, CP4-R, and the virtual VC-4 platform. Broader validation across these devices will further strengthen the reliability and portability of the tool, ensuring consistent operation regardless of hardware generation or deployment environment.

Finally, broader device support and better PKI tooling would help close remaining gaps. This includes assisting with CSR generation and certificate installation, and extending compatibility to other Crestron product lines or adjacent ecosystems with similar management models.

These additions would build on CreScan's core strengths: a focused rule set, reliable remediation, clean reporting, and compatibility across devices—without sacrificing the simplicity and predictability that make it effective in practice.

Bibliography

[1] George Vardakis, George Hatzivasilis, Eleftheria Koutsaki, and Nikos Papadakis. "Review of Smart-Home Security Using the Internet of Things". In: *Electronics* 13.16 (2024). Accessed: 2025-10-10, p. 3343 (cit. on pp. 1, 7, 13).

[2] Arun Cyril Jose and Reza Malekian. "Smart Home Automation Security: A Literature Review". In: *The Smart Computing Review* (Aug. 2015). DOI: 10.6029/smartcr.2015.04.004 (cit. on pp. 1–3).

[3] Earlence Fernandes, Jaeyeon Jung, and Atul Prakash. "Security Analysis of Emerging Smart Home Applications". In: Institute of Electrical and Electronics Engineers Inc., Aug. 2016, pp. 636–654. ISBN: 9781509008247. DOI: 10.1109/SP.2016.44 (cit. on pp. 2, 7).

[4] Vikas Hassija, Vinay Chamola, Vikas Saxena, Divyansh Jain, Pranav Goyal, and Biplab Sikdar. *A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures*. 2019. DOI: 10.1109/ACCESS.2019.2924045 (cit. on pp. 2, 7).

[5] Bin Yuan, Jun Wan, Yu-Han Wu, et al. "On the Security of Smart Home Systems: A Survey". In: *Journal of Computer Science and Technology* 38.6 (2023). Accessed: 2025-10-10, pp. 1209–1237 (cit. on pp. 2, 7).

[6] Amity University, Institute of Electrical, and Electronics Engineers. *10th International Conference on Cloud Computing, Data Science & Engineering : proceedings of the Confluence 2020 : 29-31 January 2020, Amity University, Uttar Pradesh, India.* ISBN: 9781728127910 (cit. on p. 2).

[7] Parma Nand. Astya. *IEEE International Conference on Computing, Communication and Automation (ICCCA 2017) : proceeding : on 5th-6th May, 2017.* IEEE, 2017. ISBN: 9781509064717 (cit. on p. 2).

[8] Hichem Mrabet, Sana Belguith, Adeeb Alhomoud, and Abderrazak Jemai. *A survey of IoT security based on a layered architecture of sensing and data analysis*. July 2020. DOI: 10.3390/s20133625 (cit. on p. 3).

[9] Drushti Desai and Hardik Upadhyay. *Security and Privacy Consideration for Internet of Things in Smart Home Environments* (cit. on p. 3).

[10] Ryan Heartfield, George Loukas, Sanja Budimir, Anatolij Bezemskij, Johnny R.J. Fontaine, Avgoustinos Filippoupolitis, and Etienne Roesch. *A taxonomy of cyber-physical threats and impact in the smart home*. Sept. 2018. DOI: 10.1016/j.cose.2018.07.011 (cit. on pp. 3, 6, 7).

[11] Abdulbasit Darem, Asma A Alhashmi, and Jemal H A. "Cybersecurity Threats and Countermeasures of the Smart Home Ecosystem". In: *IJCSNS International Journal of Computer Science and Network Security* 22 (3 2022), p. 303. DOI: 10.22937/IJCSNS.2022.22.3.39. URL: https://doi.org/10.22937/IJCSNS.2022.22.3.39 (cit. on pp. 3, 4).

[12] Aaesha Aldahmani, Bassem Ouni, Thierry Lestable, and Merouane Debbah. "Cyber-Security of Embedded IoTs in Smart Homes: Challenges, Requirements, Countermeasures, and Trends". In: *IEEE Open Journal of Vehicular Technology* 4 (2023), pp. 281–292. ISSN: 26441330. DOI: 10.1109/OJVT.2023.3234069 (cit. on pp. 3, 4).

[13] Shafiq Ul Rehman and Selvakumar Manickam. "A Study of Smart Home Environment and its Security Threats". In: *International Journal of Reliability, Quality and Safety Engineering* 23 (3 June 2016). ISSN: 02185393. DOI: 10.1142/S0218539316400052 (cit. on pp. 3, 4).

[14] Daniele Canavese, Luca Mannella, Leonardo Regano, and Cataldo Basile. "Security at the Edge for Resource-Limited IoT Devices". In: *Sensors* 24 (2 Jan. 2024). ISSN: 14248220. DOI: 10.3390/s24020590 (cit. on pp. 6, 7).

[15] Engineering & Management Sciences Balochistan University of Information Technology, Institute of Electrical, Electronics Engineers. Karachi Section, Institute of Electrical, and Electronics Engineers. *2016 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube) : proceedings : 11-12 April 2016, at BUITEMS Takatu Campus, Airport Road, Quetta, Pakistan*. ISBN: 9781509012527 (cit. on p. 7).

[16] Arun Cyril Jose, Reza Malekian, and Ning Ye. "Improving Home Automation Security; Integrating Device Fingerprinting into Smart Home". In: *IEEE Access* 4 (2016), pp. 5776–5787. ISSN: 21693536. DOI: 10.1109/ACCESS.2016.2606478 (cit. on p. 7).

[17] Crestron Electronics. *Crestron Security Tool*. Accessed: 2025-10-10. URL: https://help.crestron.com/toolbox/#Security_Checker_Tool/About_the_Security_Checker_Tool.htm (cit. on p. 8).

[18] Crestron. *Crestron Home Documentation*. Accessed: 2025-10-10. 2024. URL: https://docs.crestron.com/en-us/8525/Content/Topics/Home.htm (cit. on p. 8).

[19] BACnet Committee. *BACnet Committee*. Accessed: 2025-10-10. 2024. URL: https://bacnet.org/ (cit. on p. 9).

[20] Matthew Peacock, Michael N. Johnstone, and Craig Valli. "An exploration of some security issues within the BACnet protocol". In: *Communications in Computer and Information Science*. Vol. 867. Springer Verlag, 2018, pp. 252–272. ISBN: 9783319933535. DOI: 10.1007/978-3-319-93354-2_12 (cit. on p. 9).

[21] Agence Nationale de la Sécurité des Systèmes d'Information (ANSSI). "Recommandations Relatives à la Sécurité des (Systèmes d') Objets Connectés". In: *ANSSI Publication* (Aug. 2021). Documento originale redatto da ANSSI, Francia. URL: https://cyber.gouv.fr/sites/default/files/2021/09/anssi-guide-securite_des_systemes_objets_connectes_iot-v1.0.pdf (cit. on pp. 14, 15, 28, 37, 41).

[22] *ANSSI — Agence nationale de la sécurité des systèmes d'information*. Accessed: 2025-10-10 (cit. on p. 14).

[23] *AMSN — Academy of Medical-Surgical Nurses*. Accessed: 2025-10-10 (cit. on p. 14).

[24] *MVE — Experts in Cybersecurity, Home Automation and AI*. Accessed: 2025-10-10 (cit. on p. 14).

[25] "ISO/IEC 27400:2022 — Cybersecurity — Guidelines for cyber security and privacy". In: (2022). Accessed: 2025-10-10. URL: https://www.iso.org/standard/80395.html (cit. on pp. 20, 28).

[26] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). *ISO/IEC 27701:2019 — Security techniques — Extension to ISO/IEC 27001 and ISO/IEC 27002 for privacy information management — Requirements and guidelines*. https://www.iso.org/standard/71670.html. Accessed: 2025-10-10. 2019 (cit. on pp. 20, 40).

[27] International Organization for Standardization (ISO). *ISO 31000:2018 — Risk management — Guidelines*. https://www.iso.org/standard/65694.html. Accessed: 2025-10-10. 2018 (cit. on p. 20).

[28] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC). *ISO/IEC 27005:2018 — Information security, cybersecurity and privacy protection — Information security risk management*. https://www.iso.org/standard/75281.html. Accessed: 2025-10-10. 2018 (cit. on p. 20).

[29] P. Congdon, B. Aboba, A. Smith, G. Zorn, and J. Roese. *RFC 3580: IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines*. https://datatracker.ietf.org/doc/html/rfc3580. Accessed: 2025-10-10. 2003 (cit. on pp. 20, 38).

[30] R. Housley and J. Schaad. *RFC 3394: Advanced Encryption Standard (AES) Key Wrap Algorithm*. https://datatracker.ietf.org/doc/html/rfc3394. Accessed: 2025-10-10. 2002 (cit. on p. 20).

[31] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. *RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. https://datatracker.ietf.org/doc/html/rfc5280. Accessed: 2025-10-10. 2008 (cit. on p. 20).

[32] OWASP Foundation. *Transport Layer Protection Cheat Sheet*. Accessed: 2025-10-10. 2025. URL: https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html (cit. on pp. 20, 40, 41).

[33] T. Ylonen and C. Lonvick. *RFC 4251: The Secure Shell (SSH) Protocol Architecture*. https://datatracker.ietf.org/doc/html/rfc4251. Accessed: 2025-10-10. 2006 (cit. on pp. 20, 26).

[34] Eric Rescorla. *HTTP Over TLS*. RFC 2818. May 2000. DOI: 10.17487/RFC2818. URL: https://www.rfc-editor.org/info/rfc2818 (cit. on pp. 20, 26).

[35] Crestron Electronics. *Crestron Security*. Accessed: 2025-10-10. 2024. URL: https://www.crestron.com/security (cit. on pp. 21, 22).

[36] *CVE - Common Vulnerabilities and Exposures*. Accessed: 2025-10-10. URL: https://cve.mitre.org/ (cit. on p. 21).

[37] MITRE Corporation. *CVE-2023-6926: Crestron AirMedia 300 Series – edidload Command Remote Code Execution Vulnerability*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2023-6926. Accessed: 2025-10-10. 2023 (cit. on p. 21).

[38] MITRE Corporation. *CVE-2023-38405: Crestron 3-Series Control Systems – BACnet Protocol Denial of Service (DoS) Vulnerability*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2023-38405. Accessed: 2025-10-10. 2023 (cit. on p. 21).

[39] MITRE Corporation. *CVE-2022-40298: Crestron AirMedia Windows Application – Privilege Escalation Vulnerability*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-40298. Accessed: 2025-10-10. 2022 (cit. on p. 21).

[40] MITRE Corporation. *CVE-2018-11228: Crestron – Legacy System Vulnerability*. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-11228. Accessed: 2025-10-10. 2018 (cit. on p. 22).

[41] Luis P. Rondon et al. "LightningStrike: (In)secure Practices of E-IoT Systems in the Wild". In: *Proceedings of the ACM* (2021). Accessed: 2025-10-10. URL: https://dl.acm.org/doi/pdf/10.1145/3448300.3467830 (cit. on p. 25).

[42] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and David Cooper. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC 5280. May 2008. DOI: 10.17487/RFC5280. URL: https://www.rfc-editor.org/info/rfc5280 (cit. on pp. 26, 39).

[43] Kathleen Moriarty and Stephen Farrell. *Deprecating TLS 1.0 and TLS 1.1*. RFC 8996. Mar. 2021. DOI: 10.17487/RFC8996. URL: https://www.rfc-editor.org/info/rfc8996 (cit. on p. 26).

[44] J. Postel. *RFC 792: Internet Control Message Protocol (ICMP)*. RFC 792. Sept. 1981. DOI: 10.17487/RFC0792. URL: https://www.rfc-editor.org/info/rfc792 (cit. on p. 26).

[45] Dr. Steve E. Deering and Bob Hinden. *Internet Protocol, Version 6 (IPv6) Specification*. RFC 8200. July 2017. DOI: 10.17487/RFC8200. URL: https://www.rfc-editor.org/info/rfc8200 (cit. on p. 26).

[46] Python Software Foundation. *Python 3 Programming Language*. https://www.python.org/. Version 3.x, Accessed: 2025-10-10. 2025 (cit. on p. 26).

[47] Jeff Forcier et al. *Paramiko: Python implementation of the SSHv2 protocol*. https://www.paramiko.org/. Accessed: 2025-10-10. 2025 (cit. on p. 26).

[48] Ping Castle SAS. *PingCastle: Rule-based Active Directory Security Audit Tool*. Accessed: 2025-09-02. 2025. URL: https://www.pingcastle.com/ (cit. on pp. 26, 29).

[49] B. Moore. *RFC 3198: Terminology for Policy-Based Management*. https://www.rfc-editor.org/info/rfc3198. Accessed: 2025-10-10. 2001 (cit. on p. 28).

[50] Python Software Foundation. *Python Standard Library: ssl — TLS/SSL wrapper for socket objects*. https://docs.python.org/3/library/ssl.html. Accessed: 2025-10-10. 2025 (cit. on p. 30).

[51] Python Cryptographic Authority (PyCA). *cryptography: Python Cryptographic Library (PyCA)*. https://cryptography.io/. Accessed: 2025-10-10. 2025 (cit. on p. 30).

[52] Nmap Project. *Nmap (Network Mapper): Open-source Network Exploration and Security Auditing Tool*. Accessed: 2025-09-02. 2025. URL: https://nmap.org/ (cit. on p. 31).

[53] Pallets Projects. *Jinja2: A Modern and Designer-Friendly Templating Language for Python*. https://jinja.palletsprojects.com/. Accessed: 2025-10-10. 2025 (cit. on p. 31).

[54] Mikhus. *Canvas Gauges: HTML5 Gauge Widgets for JavaScript*. Accessed: 2025-10-10. 2025. URL: https://canvas-gauges.com/ (cit. on p. 32).

[55] Center for Internet Security. *CIS Benchmarks*. Accessed: 2025-10-10. 2025. URL: https://www.cisecurity.org/cis-benchmarks (cit. on pp. 38, 41).

[56] International Organization for Standardization (ISO). *ISO/IEC 27001:2013 — Information security, cybersecurity and privacy protection — Information security management systems — Requirements*. https://www.iso.org/isoiec-27001-information-security.html. Accessed: 2025-10-10. 2013 (cit. on pp. 38, 41).

[57] National Institute of Standards and Technology (NIST). *NIST SP 800-63B: Digital Identity Guidelines – Authentication and Lifecycle Management*. https://pages.nist.gov/800-63-3/sp800-63b.html. Accessed: 2025-10-10. 2020 (cit. on pp. 38, 41).

[58] C. Rigney, S. Willens, A. Rubens, and W. Simpson. *RFC 2865: Remote Authentication Dial In User Service (RADIUS)*. https://datatracker.ietf.org/doc/html/rfc2865. Accessed: 2025-10-10. 2000 (cit. on p. 38).

[59] National Institute of Standards and Technology (NIST). *NIST SP 800-92: Guide to Computer Security Log Management*. https://csrc.nist.gov/pubs/sp/800/92/final. Accessed: 2025-10-10. 2006 (cit. on pp. 39, 41).

[60] Bodo Möller, Thai Duong, Krzysztof Kotowicz. *POODLE (Padding Oracle On Downgraded Legacy Encryption)*. https://www.openssl.org/~bodo/ssl-poodle.pdf. Accessed: 2025-10-11. 2014 (cit. on p. 40).

[61] Thai Duong, Juliano Rizzo. *BEAST (Browser Exploit Against SSL/TLS)*. https://www.cve.org/CVERecord?id=CVE-2011-3389. Accessed: 2025-10-11. 2011 (cit. on p. 40).

[62] National Institute of Standards and Technology. "NIST SP 800-52r2: Guidelines for the Selection, Configuration, and Use of TLS Implementations". In: (2019). Accessed: 2025-10-10. URL: https://doi.org/10.6028/NIST.SP.800-52r2 (cit. on p. 40).

[63] D. McGrew and K. Igoe. *AES Galois/Counter Mode (GCM) Cipher Suites for TLS*. https://datatracker.ietf.org/doc/html/rfc5288. Accessed: 2025-10-11. 2008 (cit. on p. 40).

[64] E. Rescorla. *RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3*. https://datatracker.ietf.org/doc/html/rfc8446. Accessed: 2025-10-11. 2018 (cit. on p. 40).

[65] Center for Internet Security. *Center for Internet Security (CIS)*. Accessed: 2025-10-10. 2025. URL: https://www.cisecurity.org/ (cit. on p. 40).

[66] International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC). *ISO/IEC 27002:2022 – Information Security, Cybersecurity and Privacy Protection – Controls*. https://www.iso.org/standard/75652.html. Accessed: 2025-10-10. 2022 (cit. on p. 41).

[67] Crestron Electronics, Inc. *Crestron CP4 Control System*. Accessed: 2025-10-10. 2025. URL: https://www.crestron.com/Products/Catalog/Control-and-Management/Control-System/Rack-Mount/CP4 (cit. on p. 44).

[68] Crestron Electronics, Inc. *Crestron CP3 Control System*. Accessed: 2025-10-10. 2025. URL: https://www.crestron.com/Products/Catalog/Inactive/Discontinued/C/CP3 (cit. on p. 44).

[69] Crestron Electronics, Inc. *Crestron MC4 Control System*. Accessed: 2025-10-10. 2025. URL: `https://www.crestron.com/Products/Catalog/Control-and-Management/Control-System/Wall-Mount/MC4` (cit. on p. 44).

[70] Forum of Incident Response and Security Teams (FIRST). *CVSS v3.1 Specification Document*. `https://www.first.org/cvss/specification-document`. Accessed: 2025-10-10. 2019 (cit. on p. 57).