

Article

Not peer-reviewed version

Enhancing Automotive Intrusion Detection Systems with CHERI-Based Memory Protection

[Chathuranga Sampath Kalutharage](#)^{*}, Saket Mohan, [Xiaodong Liu](#), [Christos Chrysoulas](#)

Posted Date: 21 December 2024

doi: 10.20944/preprints202412.1781.v1

Keywords: Automotive Cybersecurity; IP Spoofing; Memory Protection






Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Enhancing Automotive Intrusion Detection Systems with CHERI-Based Memory Protection

Chathuranga Sampath Kalutharage ^{1,*}, Saket Mohan ¹, Xiaodong Liu ²
and Christos Chrysoulas ³

¹ Secure Elements, Coventry, UK
² Edinburgh Napier University, Scotland, UK
³ Heriot-Watt University, Scotland, UK
* Correspondence: sampath.kalutharage@secureelements.co.uk

Abstract: The automotive sector is changing fast with more integration of advanced communication technologies and further connectivity. The modern vehicle is already a collection of diverse Electronic Control Units (ECU) communicating over interconnected networks that decide critical functionalities such as engine control, braking, and entertainment. However, this increasing complexity also introduces major cybersecurity risks, including network vulnerabilities like IP spoofing, message replay, and denial-of-service(DoS) attacks, besides software vulnerabilities due to coding errors in unsafe languages like C/C++. These are serious threats to vehicle operational reliability, passenger safety, and data integrity, making robust automotive security a critical concern. This paper explores the application of CHERI(Capability Hardware Enhanced RISC Instructions) in enhancing the security of Intrusion Detection Systems(IDS) in automotive networks. CHERI introduces fine-grained memory protection mechanisms that mitigate software vulnerabilities by enforcing spatial memory safety and preventing unauthorized access to critical data. Moreover, CHERI secures IDS rule configurations from network-based threats, such as manipulation of rules and spoofing attacks, by utilizing strict memory bounds and capability-based access controls. This work experimentally demonstrates that CHERI-enhanced IDSs are highly effective in identifying and mitigating spoofing and IDS rule manipulation attacks, ensuring the integrity of rules even against attackers using forged traffic with legitimate-looking source IP addresses. The results highlight CHERI's hardware-enforced security model as a robust solution for preventing network and software-level exploits without compromising performance while maintaining compatibility with automotive-friendly programming languages like C/C++. This study underscores the critical importance of integrating CHERI and other hardware-based security frameworks into connected and autonomous vehicles to address emerging cybersecurity challenges and build a safer automotive ecosystem.

Keywords: automotive cybersecurity; IP spoofing; memory protection

1. Introduction

The automotive industry is undergoing transformative innovation, driven by rapid technological advancements and the advent of autonomous vehicles. Modern vehicles are no longer purely mechanical systems; they have evolved into complex mechatronic platforms, integrating numerous Electronic Control Units (ECUs) interconnected through sophisticated communication networks [1]. These ECUs manage critical vehicle functions, such as engine control, braking, and driver assistance, as well as secondary systems like lighting, infotainment, and connectivity services. However, this increased interconnectivity introduces a broad attack surface for cyber threats, posing significant risks to vehicle security, operational reliability, and passenger safety.

One of the critical components in ensuring vehicle cybersecurity is the Intrusion Detection System (IDS). IDSs monitor network activity to identify and mitigate threats, such as unauthorized access, data manipulation, and protocol violations. Despite their importance, traditional IDSs face significant challenges in addressing sophisticated attack vectors targeting automotive systems. Common threats include IP spoofing, where attackers forge source IP addresses to impersonate legitimate devices, and rule manipulation, where IDS configurations are altered to bypass detection mechanisms.

Other notable attack vectors include denial-of-service (DoS) attacks, which flood networks with excessive traffic to overwhelm IDS processing capabilities, and message replay attacks, where previously captured valid messages are resent to disrupt operations. Payload tampering involves injecting or modifying data packets to exploit vulnerabilities, while man-in-the-middle (MITM) attacks enable malicious actors to intercept, alter, or inject messages between ECUs. Timing attacks, which exploit delays in message processing, and protocol-specific exploits targeting communication standards like CAN, CAN-FD, or SOME/IP, further complicate the security landscape. These vulnerabilities underscore the limitations of traditional IDS solutions and the critical need for robust, hardware-based security measures to safeguard IDS functionality.

As vehicles become increasingly connected and autonomous, the potential for cyberattacks targeting their ECUs grows exponentially. Attackers can exploit network vulnerabilities to inject malicious packets or spoof legitimate IP addresses, bypassing conventional IDS mechanisms. Furthermore, unauthorized access to IDS configurations can enable attackers to modify detection rules, undermining the system's ability to flag suspicious activity. Without strong protections, these threats compromise both vehicle safety and security.

To address these challenges, this research investigates the application of Capability Hardware Enhanced RISC Instructions (CHERI), a novel hardware architecture that provides fine-grained memory protection. CHERI enforces strict access controls at the hardware level, ensuring critical components, such as IDS rule sets, remain isolated from unauthorized processes. By leveraging CHERI's memory compartmentalization capabilities, this study demonstrates how IDSs can be made resilient against IP spoofing, rule manipulation, and other advanced attack vectors. The findings highlight how CHERI-enhanced IDSs preserve their integrity and functionality even under sophisticated attack scenarios, offering a robust solution for modern automotive networks.

This study investigates the efficacy of CHERI-based memory protections in enhancing the security of IDSs within automotive networks. Specifically, it seeks to:

- Assess the ability of a CHERI-enhanced IDS to detect and mitigate IP spoofing attacks.
- Evaluate how CHERI's hardware-level protections prevent unauthorized access and manipulation of IDS rules.
- Provide insights into the integration of CHERI capabilities in automotive cybersecurity frameworks, highlighting their practical implications for securing ECUs and In-vehicle network communication.

The remainder of this paper is organized as follows: Section 2 provides the Background, outlining key challenges in automotive cybersecurity, IDS vulnerabilities, and an introduction to CHERI technologies. Section 3 presents the Literature Review, discussing related work on IDS frameworks, CHERI applications, and automotive network security. Section 4 explains the Methodology, detailing the experimental setup, CHERI-enhanced IDS framework, and attack simulation scenarios. Section 5 covers the Results and Analysis, highlighting the effectiveness of CHERI in mitigating IP spoofing and rule manipulation attacks. Finally, Section 6 concludes the paper, offering insights into CHERI's potential in real-world automotive applications and recommendations for future research directions.

2. Background

2.1. CHERI Project Overview

The CHERI (Capability Hardware Enhanced RISC Instructions) project is a research initiative that integrates fine-grained memory protection and software compartmentalization into modern processor architectures. It was originally developed as a collaboration between the University of Cambridge, SRI International, and ARM Research, funded by the DARPA-sponsored CRASH and MRC programs [2,3].

At its core, CHERI extends traditional architectures with a capability-based model, replacing standard pointers with capabilities that carry metadata, including base, bounds, and access permissions. This design enforces spatial memory safety, preventing out-of-bounds memory access, and reduces the risk of

vulnerabilities such as buffer overflows and code injection attacks [4]. The CHERI model is compatible with conventional C/C++ programming, making it particularly suitable for automotive systems, where these languages dominate due to their low-level control and performance efficiency [5,6].

The project also emphasizes hardware-enforced least privilege, allowing software components to operate with minimal permissions, thereby limiting the impact of potential compromises. The ARM Morello board is one of the notable implementations of the CHERI architecture, demonstrating its practical applicability in real-world scenarios [7].

CHERI's ability to enhance memory safety and enforce strong isolation mechanisms has garnered attention in the field of automotive cybersecurity, where protecting critical systems from sophisticated attacks such as rule manipulation and IP spoofing is paramount. By adopting CHERI capabilities, automotive applications can achieve robust security by design, aligning with emerging standards such as ISO/SAE 21434 [8,9]

2.2. CHERI and ARM Morello Technologies

The CHERI (Capability Hardware Enhanced RISC Instructions) architecture represents a transformative approach to addressing memory safety vulnerabilities, one of the most critical challenges in modern computing systems. By integrating fine-grained memory protection and scalable software compartmentalization directly into hardware, CHERI provides a robust mechanism for securing both traditional and modern computing environments.

2.2.1. CHERI Architecture Overview

CHERI is built upon two foundational principles[10]:

1. **Principle of Least Privilege:** Each component in a system operates with only the permissions it requires, minimizing the potential impact of a compromise. For example, a pointer in a CHERI system is restricted to accessing only the memory region it is explicitly authorized for, reducing the risks posed by buffer overflow vulnerabilities.

2. **Principle of Intentional Use:** Actions performed by a program must explicitly state the permissions utilized. This eliminates ambiguity in authorization and mitigates risks such as the confused deputy problem, where a component inadvertently misuses its privileges due to unclear permissions.

CHERI capabilities are hardware-enforced tokens that combine memory addresses with metadata, such as bounds and permissions, ensuring that memory access is strictly controlled. Each memory region is protected by a unique capability, and these capabilities are immutable in ways that increase their security. For instance, CHERI employs tagged memory to distinguish valid capabilities from arbitrary data, preventing their corruption or unauthorized modification.

2.2.2. CHERI Enhancements and Implementation

CHERI enhances conventional architectures like MIPS, RISC-V, and ARM by introducing a capability-based memory model that coexists with existing software [11] [12]. This hybrid approach facilitates the gradual adoption of CHERI technologies, enabling legacy code to operate alongside CHERI-enabled applications. The architecture extends pointers to include additional metadata, such as base, bounds, and permissions, ensuring that only authorized memory regions are accessed.

Key features of CHERI include:

- **Memory Protection:** By attaching bounds and permission bits to pointers, CHERI ensures spatial memory safety, preventing unauthorized access beyond designated memory regions.
- **Software Compartmentalization:** CHERI supports the isolation of software components, enabling secure interaction between mutually untrusting programs or processes.
- **Tagged Memory:** This feature ensures that capabilities cannot be forged or corrupted, as the hardware tracks their validity independently of their location in memory.

2.2.3. ARM Morello: A Prototype for Commercial Application

ARM Morello, a prototype implementation of the CHERI architecture, extends the ARMv8-A architecture with CHERI capabilities to evaluate its feasibility in commercial processors. Developed in collaboration with the UK government, Morello explores the integration of CHERI's fine-grained memory protection mechanisms into mass-market hardware.

The Morello board incorporates several CHERI advancements:

- **Compatibility with Existing Software:** Morello provides a hybrid execution environment, enabling unmodified ARMv8-A applications to coexist with CHERI-enhanced programs. This compatibility reduces adoption barriers for developers and organizations.
- **Enhanced Security Properties:** By integrating CHERI capabilities, Morello strengthens security guarantees for memory safety, compartmentalization, and access control at the hardware level.
- **Performance Considerations:** ARM Morello demonstrates that CHERI's additional security features can be implemented with minimal performance overhead, making it viable for real-time systems like those in automotive networks.

2.3. IP Spoofing and IDS Rule Manipulation

IP spoofing is a common tactic used in cyber attacks to impersonate a legitimate entity and bypass network security measures [13]. In an automotive context, IP spoofing can allow an attacker to introduce malicious commands or data into the network, masked as a legitimate ECU communication. Traditional IDS systems, especially those reliant on IP-based filtering, struggle to accurately detect spoofed IP packets without high rates of false positives. Attackers can exploit this weakness to gain further access to the vehicle network, potentially leading to unauthorized data collection or manipulation of vehicle functions.

Once attackers establish a foothold within the network, they may attempt to alter IDS rules to avoid detection in future attacks. This manipulation could involve disabling specific detection rules or creating exceptions for their activity [14]. Without robust memory protection for IDS configurations, rule manipulation remains a persistent vulnerability, as rule-based detection alone cannot prevent access to the IDS's internal logic if an attacker gains sufficient privileges.

3. Literature Review

Automotive networks have evolved from isolated, closed systems to highly interconnected networks due to the inclusion of wireless interfaces such as Wi-Fi, Bluetooth, and cellular connectivity [1]. These interfaces, while providing benefits like over-the-air (OTA) updates and telematics services, expose vehicles to numerous cyber risks [15,16]. One of the primary concerns is the complexity and distributed nature of automotive software, with modern vehicles containing over 100 million lines of code distributed across ECUs. These ECUs interconnect through the Controller Area Network (CAN) and Ethernet networks, using protocols such as Scalable service-Oriented Middleware over IP (SOME/IP) to enable efficient communication.

To enhance communication security in automotive electrical and electronic architectures, various approaches have been developed. One notable example is SecOC, standardized by AUTOSAR, which employs symmetric cryptography to secure communication over the Controller Area Network (CAN) [17]. Lee et al. in [18] suggest a hybrid framework leveraging blockchain technology to improve network security within vehicles. Nevertheless, the framework does not include a comprehensive explanation of the protocol's interaction process or demonstrate the practicality of the proposed solution. Islam et al. in [19] present a CAN-based scheme designed to mitigate multiple attack scenarios. They assert that the method ensures security without requiring changes to the underlying CAN protocol. Additionally, validation experiments are conducted, demonstrating the effectiveness of the proposed approach in preventing and defending against a range of attacks. Hafeez et al. in [20,21] employ a neural network approach to authenticate messages within vehicle networks, including the conventional CAN protocol. They assert that this method significantly enhances the accuracy of electronic control unit (ECU) communication.

While the study includes analysis and validation of the approach, it does not provide a detailed assessment of its security. Woo et al. in [22] propose a security architecture leveraging the CAN with Flexible Data Rate (CAN-FD) network to establish a secure transmission environment. They introduce a hierarchical encryption transmission technique for the vehicle network, which has been validated but lacks a formal verification process to confirm its feasibility. Lodge et al. [23] analyze vulnerabilities in CAN and CAN-FD protocols, such as replay, injection, and DoS attacks, emphasizing the need for robust automotive security frameworks. They propose solutions like lightweight cryptography, blockchain-based key provisioning, and multi-layer architectures to enhance communication security. While promising, these methods face challenges in compatibility with legacy systems and increased complexity, highlighting the importance of advanced IDS for Ethernet-based networks.

Kreissl et al. in [24] explored methods for securing SOME/IP-based vehicular communication networks. The authors suggest a central entity to manage key material distribution through (D)TLS during the event group subscription process. However, the "service offer" and "find offer" steps remain unprotected. To support broadcast communication, SOME/IP is enhanced with the TESLA protocol. Despite its advantages, TESLA's delayed authentication for broadcast messages requires modifications to the SOME/IP process flow, such as caching messages until verification data is received. This introduces latency, making it unsuitable for time-sensitive data transmission. Iorio et al. [25] propose a new framework to enhance the security of SOME/IP. While the framework includes security protocol modeling and verification, offering strong assurance of the desired security properties, it imposes significant network overhead and lacks a clear and intuitive formal verification proof. Herold et al. [26] propose an Intrusion Detection System (IDS) for SOME/IP using Complex Event Processing (CEP) to detect attacks. The system identifies threats such as malformed packets, protocol violations, and timing issues. However, like other IDS solutions, it faces challenges with false positives and false negatives. Zorman et al. [27] propose a firewall implementation for embedded automotive systems, utilizing rule-based security and deep packet inspection (DPI) to validate SOME/IP payloads. The solution demonstrates feasibility in resource-constrained environments, highlighting its potential for enhancing security in automotive Ethernet communication. Qi Liu et al. [28] proposed a method that combines deep learning models with residual self-attention to analyze Ethernet traffic, aiming to detect cyberattacks like DDoS and MITM, as well as random hardware failures. This framework demonstrates high accuracy and real-time detection capabilities, highlighting the importance of robust cybersecurity solutions for modern Ethernet-based automotive communication systems.

Intrusion Detection Systems (IDS) in automotive environments monitor network traffic to identify suspicious activities that may indicate an attack. Common threats include IP spoofing, where attackers modify IP headers to disguise their identity. While IDS can flag such anomalies, conventional approaches are often limited in countering spoofing if attackers effectively mask their actions. Additionally, if attackers compromise IDS configurations, they could manipulate detection rules to allow undetected future attacks. Notably, none of the existing solutions address IP spoofing and rule manipulation attacks with a security-by-hardware level design approach.

4. Methodology

This section describes the methodology and experimental setup, including the network and components simulated, the configuration of the IDS, and the process of executing spoofing and rule manipulation attacks. The experiment evaluates the CHERI capabilities in restricting unauthorized access and protecting the IDS configuration in an automotive network.

4.1. CHERI in Automotive Security Applications

The integration of CHERI (Capability Hardware Enhanced RISC Instructions) into automotive security offers a transformative approach to mitigating critical vulnerabilities in modern vehicles. CHERI introduces fine-grained memory protection by replacing traditional pointers with capabilities, which include metadata such as base, bounds, and permissions. This architecture prevents unau-

thorized access and memory safety violations, addressing issues such as buffer overflows and code injection—common attack vectors in automotive systems.

In automotive applications, CHERI can enhance the security of Ethernet-based protocols like SOME/IP by safeguarding IDS configurations and protecting critical ECUs from malicious tampering. By embedding CHERI capabilities, automotive systems can enforce hardware-level isolation for IDS rules, preventing rule manipulation and ensuring secure anomaly detection. Additionally, CHERI mitigates risks associated with IP spoofing by ensuring that memory access is explicitly authorized, eliminating the possibility of attackers altering network configurations.

This approach aligns with the principles of security by design, enabling robust protections against advanced cyberattacks in the dynamic and interconnected environment of modern vehicles. CHERI’s capability-based model ensures that automotive security solutions remain both effective and resilient, providing a foundational framework for safeguarding next-generation vehicle networks.

4.2. CHERI Capability Usage

Memory safety violations often result from coding errors in unsafe languages like C and C++, making software susceptible to security vulnerabilities and unauthorized access. CHERI introduces capabilities that enforce spatial memory safety, effectively addressing these issues. Figure 1 illustrates a comparison between the behavior of ISO C and CHERI C in handling pointer arithmetic and memory access.

In ISO C, two variables, data and critical_data, are declared:

- data is allocated at address 0x20 with a value of 5.
- critical_data is allocated at address 0x24 with a value of 2023.

A pointer, ptr, is initialized to point to data. In ISO C, pointer arithmetic allows ptr to be incremented to point to 0x24, accessing and leaking the value of critical_data unintentionally. This behavior is described mathematically in Equation 1.

In CHERI C, pointers are replaced with capabilities that include metadata such as base, length, and permissions. The capability for ptr is confined to the memory bounds of data (0x20 to 0x24). Any attempt to dereference ptr outside these bounds (e.g., at 0x24) triggers a hardware trap, preventing unauthorized access to critical_data. This behaviour is defined in Equation 2. CHERI enforces the principle of least privilege by ensuring ptr cannot access memory beyond its defined bounds, maintaining memory safety by design

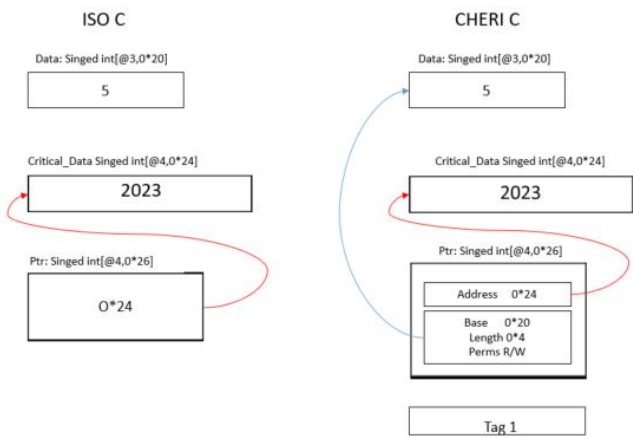


Figure 1. Fine-Grained Memory Protection: Comparing ISO C with CHERI C. In ISO C, the flawed program allows undefined behavior, leading to the potential leakage of critical_data due to unrestricted pointer arithmetic. In contrast, CHERI C enforces strict memory bounds through capabilities, ensuring that any out-of-bounds memory access triggers a hardware exception, thereby preventing unauthorized access to critical_data.

Memory safety violations often arise from coding errors in unsafe languages like C and C++, exposing software to security vulnerabilities. CHERI introduces **capabilities** to enforce **spatial memory safety**, addressing these issues effectively. The diagram compares the behavior of ISO C and CHERI C in handling pointer arithmetic and memory access.

In **ISO C**, two variables, `data` and `critical_data`, are declared:

- `data` is allocated at address `0x20` with a value of `5`.
- `critical_data` is allocated at address `0x24` with a value of `2023`.

A pointer, `ptr`, is initialized to point to `data`. In ISO C, pointer arithmetic allows `ptr` to be incremented to point to `0x24`, accessing and leaking the value of `critical_data` unintentionally. This behavior is described mathematically in Equation 1.

Variable `data` : signed int, allocated at address `0x20`, value `5`.

Variable `critical_data` : signed int, allocated at address `0x24`, value `2023`.

ISO C: Pointer `ptr` : signed int*, initialized to address of `data`. (1)

After increment: $ptr = ptr + 1 \implies ptr = 0x24$.

Dereferencing: $y = *ptr \implies y = critical_data = 2023$.

In **CHERI C**, pointers are replaced with capabilities that include metadata such as `base`, `length`, and `permissions`. The capability for `ptr` is confined to the memory bounds of `data` (`0x20` to `0x24`). Any attempt to dereference `ptr` outside these bounds (e.g., at `0x24`) triggers a hardware trap, preventing unauthorized access to `critical_data`. This behavior is defined in Equation 2. CHERI enforces the **principle of least privilege** by ensuring `ptr` cannot access memory beyond its defined bounds, maintaining memory safety by design.

Variable `data` : signed int, allocated at address `0x20`, value `5`.

Variable `critical_data` : signed int, allocated at address `0x24`, value `2023`.

CHERI C: CHERI C Pointer `ptr` : signed int* with metadata as a CHERI capability: (2)
Address: `0x20`, Base: `0x20`, Length: `0x4`, Permissions: R/W, Tag: Valid.

Dereferencing: $y = *ptr \implies$ Bounds check by CHERI: Access Denied if $ptr > 0x20 + 0x4$.

By integrating CHERI capabilities into the program, the hardware ensures that memory safety violations, such as accessing `critical_data` through `ptr`, are impossible. This approach not only protects against common vulnerabilities like buffer overflows and out-of-bounds memory access but also mitigates intentional attacks aimed at exploiting memory corruption.

This demonstrates how CHERI implements **memory safety by design** on our IDS, transforming insecure C code into a secure execution model without significant performance overhead or extensive code changes. As C remains a dominant and automotive-friendly language due to its low-level hardware access, efficiency, and real-time capabilities, CHERI's compatibility with C ensures that security can be integrated into existing automotive systems without compromising these critical advantages.

4.3. Memory Protected IDS

The proposed methodology integrates CHERI (Capability Hardware Enhanced RISC Instructions) to fortify automotive Ethernet-based Intrusion Detection Systems (IDS) against advanced attack scenarios, such as IP spoofing and rule manipulation. By leveraging CHERI's capability-based memory protection, the framework ensures that IDS rules and configurations are isolated from unauthorized access, even in the event of system compromise. This section explains the approach with examples, attack scenarios, and code-level defences.

In IP spoofing attacks, an attacker forges the source IP address in network packets to mimic a legitimate ECU, enabling unauthorized access or manipulation of in-vehicle communication. For example, A spoofed packet may claim to originate from 192.168.1.11 (a trusted ECU) but carry malicious payloads targeting safety-critical systems. This can be defense Using CHERI, IDS rules are stored with strict memory bounds, ensuring only authenticated packets can access or modify ECU configurations. Spoofed packets are detected by comparing their source IP and payload against predefined rules confined to CHERI-protected memory as Listing 1.

```

1 int check_ecu_rule(const char *src_ip, const struct someip_packet *someip_pkt)
2 {
3     for (int i = 0; i < 5; i++) {
4         const char *ecu_ip = (const char *)ecu_memory[i]->ecu_ip; // Access
5         // Match source IP and packet attributes
6         if (strcmp(src_ip, ecu_ip) == 0 &&
7             ntohs(someip_pkt->method_id) == ecu_memory[i]->method_id &&
8             ntohs(someip_pkt->client_id) == ecu_memory[i]->client_id &&
9             ntohs(someip_pkt->session_id) == ecu_memory[i]->session_id) {
10                return i; // Rule matched
11            }
12    }
13    return -1; // No match found (potential spoofed packet)
14 }

```

Listing 1: Function to Check ECU Rules Using CHERI Capabilities.

CHERI ensures that **ecu_memory** is bounded and immutable, so spoofed packets cannot tamper with or bypass these rules. If the source IP and attributes do not match any defined ECU rule, the packet is flagged as a potential spoofing attempt.

In rule manipulation attacks, an attacker gains unauthorized access to the IDS configuration, modifying detection thresholds or disabling certain rules entirely. This could allow malicious packets to bypass the IDS undetected. The IDS leverages CHERI to isolate IDS rules within protected memory regions as Listing 2. Only authorized processes can modify these rules, ensuring they remain intact even during a system compromise.

```

1 void initialize_ecu_memory() {
2     for (int i = 0; i < 5; i++) {
3         // Allocate memory with CHERI capabilities
4         ecu_memory[i] = (struct ecu_rule *__capability)malloc(sizeof(struct
5             ecu_rule));
6         if (!ecu_memory[i]) {
7             perror("Failed to allocate memory for ECU rule");
8             exit(EXIT_FAILURE);
9         }
10    }
11
12    // Assign rules and enforce bounds
13    ecu_memory[0]->ecu_ip = (char *__capability)cheri_setbounds("192.168.1.11"
14        , sizeof("192.168.1.11"));
15    ecu_memory[0]->method_id = 0x1234;
16    ecu_memory[0]->client_id = 0x5678;
17    ecu_memory[0]->session_id = 0x9abc;
18 }

```

Listing 2: Function to Initialize ECU Memory with CHERI Capabilities.

The **cheri_setbounds** function ensures that IDS rules are stored in tightly bounded memory regions. This prevents unauthorized processes from accessing or altering these rules. Any attempt to modify the rules outside authorized contexts results in a deterministic hardware trap.

Furthermore, the IDS must monitor and analyze network traffic for potential anomalies in real time, including spoofed packets and unauthorized rule manipulations. During live packet capture, the IDS compares incoming packets against the CHERI-protected rule set. CHERI ensures that only validated rules are accessed during this process, flagging spoofed or tampered packets. Packets are captured and validated against the CHERI-protected rule set, as shown in Listing 3. Any anomaly, such as a spoofed IP or a rule mismatch, triggers a warning, enabling the IDS to respond in real time. The **libpcap** library is used to capture network traffic in real time. Additionally, the libraries **arpa/inet.h**, **netinet/ip.h**, and **netinet/udp.h** are utilized to parse and analyze specific fields in the captured packets, such as IP and UDP headers.

```

1 void process_packet(u_char *args, const struct pcap_pkthdr *header, const
  u_char *packet) {
2     struct ip *ip_hdr = (struct ip *)(packet + 14); // Parse IP header
3     char src_ip[INET_ADDRSTRLEN];
4
5     inet_ntop(AF_INET, &(ip_hdr->ip_src), src_ip, INET_ADDRSTRLEN);
6
7     if (ip_hdr->ip_p == IPPROTO_UDP) { // Check for UDP packets
8         const struct someip_packet *someip_pkt = (struct someip_packet *) (
          packet + 14 + ip_hdr->ip_hl * 4);
9         int rule_index = check_ecu_rule(src_ip, someip_pkt);
10
11        if (rule_index >= 0) {
12            printf("Valid packet matched with ECU%d rules\n", rule_index + 1);
13        } else {
14            printf("Unmatched packet from %s! Possible attack detected.\n",
              src_ip);
15        }
16    }
17 }

```

Listing 3: Function to Process Packets and Validate Against ECU Rules.

4.4. Experimental Setup

The experimental setup includes several simulated components representing elements of an automotive network, such as ECUs, an IDS, a packet sniffer, and an attacker model, as shown in Figure 2. The network layout mimics a typical vehicular communication structure, where the IDS monitors network traffic between ECUs and detects potential anomalies.

- **ECU Components (ECU1, ECU2, ECU3, ECU4 and ECU5):** These scripts simulate legitimate ECUs within the vehicle's network, transmitting standard, authorized packets. ECU1 and ECU2 generate and send routine communication packets across the network to emulate regular vehicular functions.
- **Intrusion Detection System (IDS):** This IDS script monitors the network for any suspicious packets or anomalies. With CHERI capabilities enabled, the IDS memory is compartmentalized, ensuring that only authorized processes can access or modify its rule set. The IDS is configured to detect IP spoofing attempts and to log unauthorized access attempts to the IDS memory or rule configurations.
- **Attacker Model (Attacker):** The attacker model is designed to send spoofed IP packets, using legitimate IP addresses to disguise malicious activities. The attacker's objective is to bypass the

IDS by impersonating a legitimate ECU. If successful, the attacker aims to modify IDS rules to prevent future detections and alter the rules to allow malicious data to be sent to critical ECUs.

- **Packet Sniffer (Sniff):** This component captures network traffic for analysis and logs all packets, enabling verification of whether spoofed packets are correctly identified by the IDS.

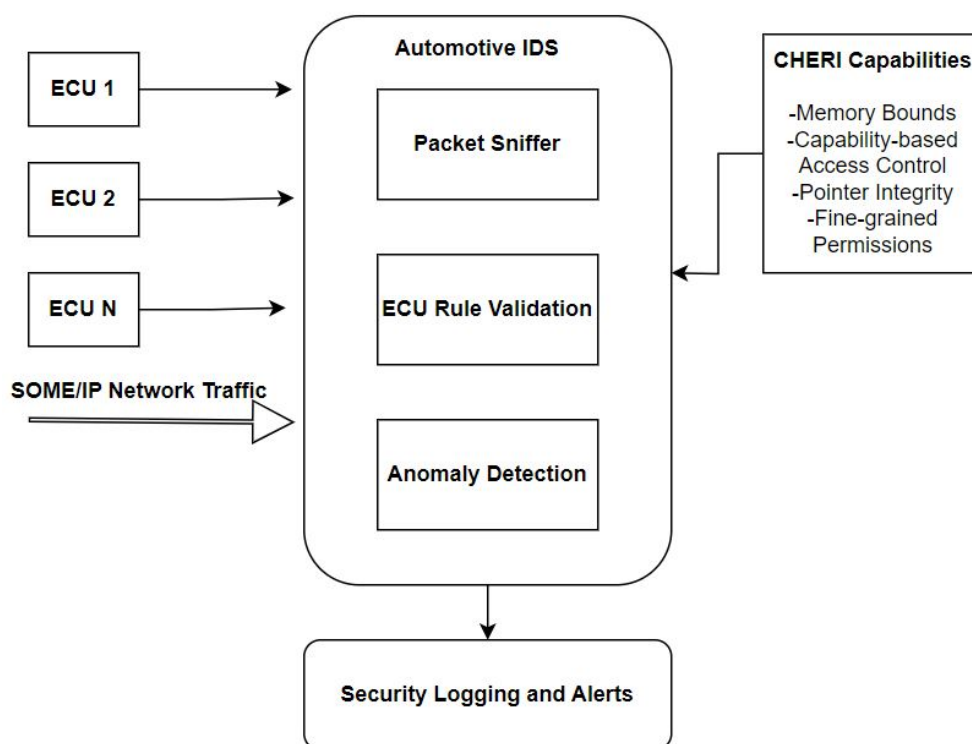


Figure 2. Automotive Intrusion Detection System (IDS) with CHERI Capabilities Experimental Setup: The diagram illustrates the experimental setup of an IDS architecture for automotive networks. SOME/IP traffic from multiple ECUs, along with potential attacker scripts masquerading as random ECUs, is processed through a pipeline comprising packet sniffing, ECU rule validation, and anomaly detection. CHERI capabilities enhance the IDS by enforcing memory bounds, capability-based access control, pointer integrity, and fine-grained permissions. These features ensure robust protection against threats such as spoofed packets, rule manipulation, and unauthorized ECU impersonation. Detected anomalies are logged and trigger alerts, maintaining the integrity and security of the network.

4.5. Execution of Attack

The Attacker: The attacker injects packets with spoofed IP addresses, mimicking the IP addresses of ECU1 and ECU2 to evade detection. The IDS monitors network traffic, comparing expected and actual packet source data to detect anomalies. CHERI capabilities within the IDS memory ensure that unauthorized packets cannot influence IDS functionality beyond their initial reception. Subsequently, the attacker attempts to access and modify the IDS rule set. By spoofing legitimate ECU IP addresses, the attacker seeks to gain access to IDS rule configurations and alter them to disable detection for future spoofing activities. However, the CHERI-enhanced IDS enforces strict memory protection, preventing unauthorized access or modification of its rules. Any access violation attempt is flagged by CHERI's memory bounds checks, and the IDS logs the incident while ensuring the attacker's script is terminated.

4.6. CHERI-Enhanced Memory Protections

CHERI capabilities introduce fine-grained memory protections to the IDS. In this experiment, the CHERI memory model is applied to compartmentalize IDS configurations, ensuring that:

- **Access Permissions:** Only authorized memory regions can be accessed by the IDS. Unauthorized attempts to modify IDS rules or configurations trigger CHERI alerts.
- **Memory Bounds:** The memory boundaries around IDS rules are strictly enforced, preventing any modification or access from processes outside of predefined bounds.
- **Logging Violations:** Each unauthorized memory access attempt is logged, providing detailed records of any access violations. This functionality allows for verification of CHERI's effectiveness in blocking unauthorized access to IDS configurations.

By structuring the IDS with CHERI-enabled protections, the experiment tests how effectively CHERI prevents spoofed packets from bypassing IDS detection and protects against unauthorized rule manipulation.

5. Results and Analysis

The results section presents the findings from attack simulation, demonstrating the impact of CHERI capabilities on IDS effectiveness in detecting spoofed IP packets and blocking unauthorized access attempts.

The IDS successfully flagged 100% of anomalous packets originating from undefined ECUs, demonstrating a high detection rate for known attacks. In our tests, 150 undefined IPs were used to repeatedly send multiple packets over a 30-minute timeframe, all of which were accurately detected. Furthermore, packets from known IPs were sent with altered features that deviated from the predefined rules assigned to each ECU and IP. In each scenario, the IDS accurately flagged the anomalies. The CHERI capabilities enabled the IDS to maintain rule integrity, preventing unauthorized packets from influencing detection mechanisms. This demonstrates that CHERI-enhanced memory isolation significantly enhances detection robustness, even against attackers using legitimate IPs as a disguise. The latency of the IDS response to anomalous packets was also measured to evaluate the performance impact of CHERI protections. Observed latency increases were minimal, indicating that the CHERI enhancements do not significantly affect IDS performance. This minimal overhead makes CHERI a viable solution for real-time applications in automotive IDS contexts.

5.1. IDS Rule Integrity and Unauthorized Access Blocking

During the IDS rule manipulation attempt, the attacker sought to alter the IDS configuration to disable specific detection rules.

- **Memory Access Violation Logs:** The CHERI-enabled IDS logged multiple unauthorized access attempts by the attacker as they tried to modify the IDS rules. Each access violation was flagged by CHERI's capability checks, which block any process outside the authorized scope from modifying memory. This illustrates CHERI's effectiveness in maintaining the integrity of the IDS rule set, as all modification attempts were successfully prevented.
- **Rule Integrity Preservation:** The IDS maintained its rule configurations intact throughout the attack scenario, as evidenced by the lack of any successful modifications to the IDS rules. CHERI's memory protection mechanisms effectively isolated the IDS configuration, preventing the attacker from altering detection logic. This result demonstrates the value of CHERI capabilities in protecting critical security configurations, ensuring consistent IDS performance even under attack.

5.2. Analysis of CHERI's Impact on IDS Security

The experimental results indicate that CHERI's fine-grained memory protections significantly enhance IDS security in the following ways:

- **Increased Resilience to Spoofing Attacks:** CHERI's hardware-enforced memory compartmentalization makes the IDS resilient to IP-based spoofing attacks. Although the attacker could mimic legitimate IPs, CHERI's memory protection mechanisms prevented unauthorized influence on the IDS logic.
- **Enhanced Rule Protection Against Manipulation Attempts:** The compartmentalized memory model prevented the attacker from modifying IDS rules, effectively securing the IDS from rule manipulation attacks. The unauthorized access logs demonstrate that CHERI successfully restricted all access attempts to the IDS configuration, ensuring rule integrity.
- **Implications for Real-Time Security Applications:** The minimal performance overhead observed in this experiment suggests that CHERI is suitable for real-time security applications in automotive contexts, where high-speed detection and low latency are critical.

The analysis of these results supports the conclusion that CHERI-based memory protection is an effective method for bolstering IDS security against IP spoofing and rule manipulation in automotive networks. By enforcing strict memory access boundaries, CHERI prevents attackers from bypassing IDS detection mechanisms and ensures rule integrity, highlighting its potential for broader automotive cybersecurity applications.

6. Conclusions

The results highlight the effectiveness of CHERI capabilities in securing automotive networks against advanced threats such as IP spoofing and rule manipulation. CHERI's fine-grained memory protection mechanisms, particularly suited for systems developed in C/C++, enhance resilience while preserving performance, a critical requirement in automotive environments. C/C++ remain the dominant programming languages in the automotive domain due to their efficiency and close-to-hardware capabilities, making CHERI's compatibility with these languages a significant advantage for adoption in vehicle systems.

Although the results are promising, implementing CHERI in automotive systems requires careful consideration of performance trade-offs, hardware costs, and integration complexity. Future research should focus on optimizing CHERI configurations for resource-constrained environments and extending its applications to other critical automotive components beyond IDS. Testing under diverse cyber-physical attack scenarios and aligning CHERI-based solutions with automotive standards, such as ISO/SAE 21434, would further validate its role in secure automotive design. Additionally, deploying CHERI on lightweight automotive-friendly platforms like the SONATA board could demonstrate its feasibility in real-world automotive systems.

This study underscores that CHERI-based memory protection significantly enhances IDS resilience against IP spoofing and rule manipulation attacks. By leveraging CHERI's fine-grained memory control, this work showcases a hardware-based security solution that maintains IDS rule integrity under attack. The findings suggest that adopting CHERI could strengthen the cybersecurity of critical vehicle control systems while seamlessly integrating with the C/C++-centric automotive software ecosystem.

Author Contributions: Conceptualization, C.S.K. and S.M.; methodology, C.S.K.; software, C.S.K.; validation, C.S.K., S.M., X.L. and C.C.; formal analysis, C.S.K.; investigation, C.S.K.; resources, C.S.K. and S.M.; data curation, C.S.K.; writing—original draft preparation, X.X.; writing—review and editing, S.M., X.L. and C.C.; visualization, C.S.K.; supervision, S.M., X.L. and C.C.; project administration, S.M.; funding acquisition, C.S.K. AND S.M. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: Dataset available on request from the authors.

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AUTOSAR	Automotive Open System Architecture
CAN	Controller Area Network
CAN-FD	Controller Area Network with Flexible Data Rate
CEP	Complex Event Processing
CHERI	Capability Hardware Enhanced RISC Instructions
DDoS	Distributed Denial of Service
DPI	Deep Packet Inspection
ECU	Electronic Control Unit
IDS	Intrusion Detection System
ISO	International Organization for Standardization
MITM	Man-in-the-Middle
OTA	Over-the-Air
PCAP	Packet Capture
SOME/IP	Scalable service-Oriented Middleware over IP
TESLA	Timed Efficient Stream Loss-tolerant Authentication

References

1. Luo, F.; Zhang, X.; Hou, S. Research on cybersecurity testing for in-vehicle network. In Proceedings of the 2021 International Conference on Intelligent Technology and Embedded Systems (ICITES). IEEE, 2021, pp. 144–150.
2. University of Cambridge. The CHERI Project. <https://www.cl.cam.ac.uk/research/security/ctsr/cheri/>. Accessed: 2024-12-02.
3. Watson, R.N.M.; Moore, S.W.; Neumann, P.G.; Sewell, P. CHERI: A RISC design with hardware-enforced capabilities. *IEEE Micro* **2014**, *34*, 10–23. <https://doi.org/10.1109/MM.2014.20>.
4. Davis, B.; Watson, R.N.M.; Neumann, P.G.; Woodruff, J. CheriABI: Enforcing spatial memory safety within the POSIX C runtime. In Proceedings of the IEEE Symposium on Security and Privacy. IEEE, 2019, pp. 941–958. <https://doi.org/10.1109/SP.2019.00078>.
5. Grisenthwaite, R. A Safer Digital Future, by Design. <https://www.arm.com/blogs/blueprint/digital-security-by-design>, 2019.
6. Embedded.com. Why C/C++ Dominate in Automotive Systems. <https://www.embedded.com/>. Accessed: 2024-12-02.
7. ARM. Introducing the Morello Prototype Board. <https://www.arm.com/morello>. Accessed: 2024-12-02.
8. ISO/SAE 21434: Road Vehicles – Cybersecurity Engineering. <https://www.iso.org/standard/70918.html>, 2020. Accessed: 2024-12-02.
9. Zorman, E.H.; Lodge, C.; Miucic, R.; Bazzi, R. Real-Time Monitoring of Hardware Functional Safety and Cybersecurity with In-Vehicle SOME/IP Ethernet Traffic. *Sensors* **2024**, *24*, 1–20. <https://doi.org/10.3390/s24010015>.
10. Saltzer, J.H. Protection and the control of information sharing in Multics. *Communications of the ACM* **1974**, *17*, 388–402.
11. Kho, N.M.D.; Uy, R.L. MIPSers: MIPS extension release 6 simulator. In Proceedings of the 2017IEEE 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM). IEEE, 2017, pp. 1–6.
12. Grisenthwaite, R. A Safer Digital Future, by Design. <https://www.arm.com/blogs/blueprint/digital-security-by-design>, 2019. SVP, Chief Architect & Fellow, Arm.
13. Song, S.; Manikopoulos, C.N. IP Spoofing Detection Approach(ISDA) for Network Intrusion Detection System. In Proceedings of the 2006 IEEE Sarnoff Symposium, 2006, pp. 1–4. <https://doi.org/10.1109/SARNOF.2006.4534792>.
14. Parameshwarappa, P.; Chen, Z.; Gangopadhyay, A. Analyzing attack strategies against rule-based intrusion detection systems. In Proceedings of the Proceedings of the Workshop Program of the 19th International Conference on Distributed Computing and Networking, 2018, pp. 1–4.

15. Aliwa, E.; Rana, O.; Perera, C.; Burnap, P. Cyberattacks and countermeasures for in-vehicle networks. *ACM computing surveys (CSUR)* **2021**, *54*, 1–37.
16. Ghadi, M.; Sali, Á.; Szalay, Z.; Török, Á. A new methodology for analyzing vehicle network topologies for critical hacking. *Journal of Ambient Intelligence and Humanized Computing* **2021**, *12*, 7923–7934.
17. AUTOSAR. Specification of Module Secure Onboard Communication - CP. Technical report, AUTOSAR, 2017. Release 4.2.2.
18. Lee, T.Y.; Lin, I.A.; Liao, R.H. Design of a FlexRay/Ethernet gateway and security mechanism for in-vehicle networks. *Sensors* **2020**, *20*, 641.
19. Islam, R.; Refat, R.U.D. Improving CAN bus security by assigning dynamic arbitration IDs. *Journal of Transportation Security* **2020**, *13*, 19–31.
20. Hafeez, A.; Topolovec, K.; Awad, S. Ecu fingerprinting through parametric signal modeling and artificial neural networks for in-vehicle security against spoofing attacks. In Proceedings of the 2019 15th International Computer Engineering Conference (ICENCO). IEEE, 2019, pp. 29–38.
21. Hafeez, A.; Malik, H.; Irtaza, A.; Uddin, M.Z.; Noori, F.M. Enhancing ECU identification security in CAN networks using distortion modeling and neural networks. *Frontiers in Computer Science* **2024**, *6*, 1392119.
22. Woo, S.; Jo, H.J.; Kim, I.S.; Lee, D.H. A practical security architecture for in-vehicle CAN-FD. *IEEE Transactions on Intelligent Transportation Systems* **2016**, *17*, 2248–2261.
23. Lodge, N.; Tambe, N.; Saqib, F. Addressing Vulnerabilities in CAN-FD: An Exploration and Security Enhancement Approach. *IoT* **2024**, *5*, 290–310.
24. Kreissl, J. Absicherung der some/IP kommunikation bei adaptive autosar. Master's thesis, 2017.
25. Iorio, M.; Reineri, M.; Risso, F.; Sisto, R.; Valenza, F. Securing SOME/IP for in-vehicle service protection. *IEEE Transactions on Vehicular Technology* **2020**, *69*, 13450–13466.
26. Herold, N.; Posselt, S.A.; Hanka, O.; Carle, G. Anomaly detection for SOME/IP using complex event processing. In Proceedings of the NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2016, pp. 1221–1226.
27. Zorman, E.H.; Isoaho, J.; Mohammad, T. Implementation of a SOME/IP Firewall with Deep Packet Inspection for automotive use-cases. PhD thesis, Ph. D. Thesis, University of Turku, 2024.
28. Liu, Q.; Li, X.; Sun, K.; Li, Y.; Liu, Y. SISSA: Real-time Monitoring of Hardware Functional Safety and Cybersecurity with In-vehicle SOME/IP Ethernet Traffic. *IEEE Internet of Things Journal* **2024**.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.