eBPF-based Mandatory Access Control for IoT Devices

MSc Research Project
Cloud Computing

Alexander Mamani
Student ID: 23329823

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Alexander Mamani
Student ID:	23329823
Programme:	Cloud Computing
Year:	2025
Module:	MSc Research Project
Supervisor:	Vikas Sahni
Submission Due Date:	20/08/2025
Project Title:	eBPF-based Mandatory Access Control for IoT Devices
Word Count:	XXX
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Alexander Mamani
Date:	9th June 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

eBPF-based Mandatory Access Control for IoT Devices

Alexander Mamani
23329823

Abstract

The increasing deployment of Internet of Thing (IoT) devices has raised a serious problem about the lack of scalable and adaptable system-level access control in environments with limited resources. Traditional Mandatory Access Control (MAC) systems like SELinux and AppArmor are often either too static or heavyweight for embedded Linux platforms. This research proposes a lightweight, programmable MAC framework using Extended Berkeley Packet Filter (eBPF) and Linux Security Modules (LSM) hooks to enforce security policies within the Linux kernel. This approach will support policy updates, and low latency; the system aims to overcome existing MAC models like AppArmor and SELinux. It follows a design science methodology and will be implemented and evaluated on a physical environment (Raspberry Pi). The expected contribution is a reference framework in-kernel access control in modern IoT systems.

1 Introduction

The fast adoption of IoT in multiple sectors have led to companies to deploy thousands of devices under limited hardware and software constraints (Hossain et al.; 2015). These devices run lightweight Linux-based distributions and these are expected to run continuously for days in hostile environment wherein save resources like power, memory, CPU consumption, and so on are crucial. So, their limited computational power and memory capacity makes them unable to support traditional security mechanisms like in conventional cloud services like AWS EC2 instances, or Azure virtual servers. As a result, IoT devices often lack strong system-level access control and are vulnerable to unauthorized access, malware infections and policy violations.

Traditional access control mechanisms such as Discretionary Access Control (DAC) and Role-Based Access Control (RBAC) are not sufficient to secure actual IoT systems due to these models rely strongly on user identity or predefined roles and offer limited enforcement granularity especially against internal or process level threats (Malik et al.; 2020; Ravidas et al.; 2019). Mandatory Access Control (MAC) implemented through the Linux Security Module (LSM) interface is a more robust enforcement model. However, the most commonly used MAC frameworks such as AppArmor, SELinux, and TOMOYO are not suitable for embedded and IoT environments because they rely on static, precompiled policies, complex configuration and high memory usage.

To address these challenges, this research explores the use of Extended Berkeley Packet Filter (eBPF) technology combined with LSM hooks to create an eBPF-based MAC system. This eBPF approach allows verified programs to run in kernel space, responding to system events with minimal performance overhead (Song and Li; 2024). These combination enables the creation of lightweight, programmable access control mechanisms that are compatible with real-world IoT constraints. These enforcement mechanisms can adapt to changing system conditions, be updated at runtime and avoid the rigidity of traditional static policy models.

Research question: How can an eBPF-based Mandatory Access Control system ensure high availability, low overhead, and compatibility with kernel-based IoT devices?

This project proposes the design and implementation of an eBPF-based MAC system integrated with LSM hooks to enforce access control policies within the Linux kernel. The system will support policy definition, modification and removal without requiring system restarts or kernel recompilation. The architecture will be optimized for low-latency enforcement, small memory footprint and practical deployment on platforms like Raspberry Pi system.

This research contributes to mandatory access control for embedded IoT systems by showing how programmability and low overhead can coexist in kernel level security. By using eBPF and LSM hooks the system sets a precedent for enforcing policies in real time without relying on traditional policy compilers. It offers a reference implementation for future researchers exploring in-kernel, event-driven access control and lies the ground work for more intelligent mechanisms. It also provides a new way to evaluate MAC systems in resource constrained environments.

2 Related Work

2.1 MAC Systems for IoT

Miki et al. (2023) performed a evaluation wherein compares four MAC systems (SELinux, AppArmor, TOMOYO, and Smack) on IoT environments. The evaluation was focused on evaluate default policy effectiveness, malware resilience in specifically against Mirai (Alrawi et al.; 2021), system compatibility, and resource consumption. While SELinux offered a strong security out-of-the-box by managing to block Mirai attacks, it also introduced the highest resource overhead, reaching up to use 85% of the memory in test scenarios; making it unsuitable for limited IoT hardware. AppArmor and TOMOYO show better efficiency but required high customized policy development to achieve the same protection. Smack failed because it was unable to prevent malware propagation due to limitations in its label and capability model.

Guo et al. (2013) proposed MPdroid, a hybrid solution combining Smack (kernel-level MAC) with role-based access control in the Android framework. MPdroid reached to block malware behaviors (unauthorized SMS and data access) through dynamic post-installation role assignment. However, MPdroid has a strongly dependency to the platform and it was tightly integrated with Android's internal system services, making it unsuitable to general-purpose or embedded Linux distributions.

These findings are consistent with earlier efforts by Nakamura and Sameshima (2008) and Nakamura et al. (2015), who worked in reducing SELinux's memory consumption using SEEdit and SPDL. While they achieved to reduce in over 90%, SELinux still required several megabytes of RAM and significant CPU usage for policy evaluation; unsuitable for typical IoT devices running minimal Linux distributions like BusyBox. These understanding of the past reinforce that high resource overhead is still a critical limitation, despite years of optimization.

Across these researches, there is no solution that provides low-overhead, dynamic, and portable MAC enforcement suitable for general-purpose, kernel-based IoT platforms. This opens the niche for a runtime-adaptable, programmable approach using eBPF and LSM hooks.

2.2 Policy Customization

Miki et al. (2024) extended their earlier research by exploring the effectiveness that involved customizing MAC policies across multiple enforcement strategies: focusing on deny-listing based on system commands, process context isolation, and origin-based file access restrictions. They concluded that AppArmor as the easiest to customize due to its intuitive path-based rules, support for deny directives, and user-space tools like aa-genprof, which help to reduce the policy complexity and configuration time, but it still requires manual tuning and careful rule specification to guarantee accurate coverage. Otherwise, TOMOYO, due to its process-based control, demanded a large volume of rules and manual editing. Smack, in contrast, showed serious limitations because of its lack of native deny rules; they also highlight the complexity in managing label combinations in larger systems.

In contrast, Bessler et al. (2024) deployed a real-world industrial IoT gateway using TOMOYO (LSM-based Mandatory Access Control) and CaitSith (Rule-Based Access Control) to enforce layered security policies. In the evaluation, their framework managed to defend against exploits like PwnKit on the filesystem, process, and network control layers. Their implementation prevents access to critical services (sshd, SNMP, VPN); Privilege escalation attempts (PwnKit, shell escape) failed due to policy enforcement and filesystem restrictions; and CaitSith rules explicitly prevented Living-off-the-Land attacks (unauthorized SNMP config overrides). Their results confirm the effectiveness of LSM-based MAC systems in blocking known threats such as privilege escalation attacks and misconfigured services. However, they clarify that the system's protection depends on carefully layered and static policies; they also acknowledge the lack of scalability across multiple devices or dynamic environments.

While these works demonstrated that MAC customization can improve IoT security, they all rely on static policy models that must be manually defined and pre-configured. No current solution allows policies to be defined, adjusted, or learned at runtime, neither they support dynamic adaptation to context or behavior. A runtime-programmable MAC system, such as one based on eBPF and LSM hooks, would fill this gap.

2.3 Real-Time Constraints

Ko et al. (2019) established that in environments with sensitive latency such as IoT and robotics, MAC enforcement can severely break down operation by introducing unpredictable delays in system call execution. To address this problem they propose a Priority-Type Enforcement (Priory-TE), a tuned SELinux that prioritizes access control decisions for real time processes. To do that, it reorders type identifiers to optimize rule lookup and reduce SELinux Access Vector Cache (AVC) misses. Their evaluation showed that Priority-TE reduced system call latency by nearly 50% for open() and remove() operations; while also improving thread periodicity at 5 ms intervals. Standard SELinux could not sustain that level of performance under load. Yet, Priority-TE still relies on SELinux's static policy structure. That means we need to manually prioritize types and tune cache parameters. That can be difficult to maintain and scale in dynamic or resource-constrained environments. That limitation highlights the need for a MAC more flexible and lightweight enforcement mechanisms.

That is where the gap lies: Priority-TE shows what is possible with MAC systems in real-time environments but it does not give us the dynamic enforcement or in-kernel policy logic that modern IoT systems really need. A MAC system built on eBPF could change that. By attaching logic to LSM hooks, you could get low-latency enforcement without needing to manually tune policies.

2.4 Embedded Optimization

Nakamura and Sameshima (2008) addressed SELinux's lack of suitability for consumer electronics (CE) and embedded devices by doing a lot of kernel and userland optimizations and creating SEEdit, a simplified policy editor. SEEdit uses Simplified Policy Description Language (SPDL) so developers can write more concise and readable policies than traditional SELinux syntax. Their tuning reduced SELinux's memory footprint by over 90% (from 5.3 MB to 465 KB) and policy file size from 2.3 MB to 211 KB. This allowed SELinux to run on hardware with as little as 64 MB of RAM and 32 MB of flash, which is common in CE and IoT devices at that time.

Building on this, Nakamura et al. (2015) formalized the concept of Embedded SELinux, further tuned the kernel and policy and contributed patches to BusyBox, libselinux, and SELinux userland utilities to make it work better with minimal Linux distributions. They introduced hash size tuning for Access Vector Cache (AVC), dynamic table allocation based on policy size and optional stripping of unused policy types to reduce runtime overhead and improve scalability. Benchmarking showed syscall latency reduction up to 50% and sustained policy enforcement with no performance degradation under typical embedded workloads.

Although these optimizations improve the feasibility of SELinux on embedded devices, they do not offer programmable, context-aware enforcement logic. All decisions must still be predefined and deployed statically. This limitation leaves open the need for a lightweight MAC system capable of making access control decisions dynamically at runtime, which eBPF is uniquely positioned to support.

2.5 Research Niche

This research fills a unexplored gap in access control for IoT systems: a lightweight Mandatory Access Control (MAC) system in the kernel space using eBPF and LSM hooks. Existing MAC solutions like SELinux, AppArmor, TOMOYO and Smack are effective in static environment but not suitable for modern IoT environments as they rely on precompiled and high resource consumption. Even in their embedded forms (e.g. Embedded

SELinux) they require manual rule specification and pre-deployment configuration; all of which are limiting in fast evolving, resource constrained IoT ecosystems.

This project defines a new space by proposing an eBPF based enforcement mechanism that hooks directly into the Linux kernel's LSM interface. Unlike traditional MAC systems, eBPF allows small, verifiable programs to be loaded and updated at runtime, enabling context aware, event driven security policies without rebooting or recompiling the kernel. This programmability allows you to define dynamic rules like "deny file access if the process comes from an untrusted namespace", a type of conditional logic not possible in static MAC systems

In this section you need to situate your work in the academic literature; this entails a critical (positive, negative, helpful) review of similar work. If you can't find similar work, you haven't looked hard enough. Ideally, you want to be reading around 50 papers; of which at least 25 should appear in the paper itself. Note that urls are not references, they are footnotes.[1]

You are expected to provide a critical/analytic overview of the significant literature published on your topic. Comment on the strength and weakness/limitation of work in each reviewed paper.

The literature review should end in a paragraph that summarises the findings from the state of the art, why the previous solution are not adequate and justifies the need for your research question.

The content sections of your report should of course be structured into subsections. Note that here there are 2 subsections subsection 2.6 and subsection 2.7.

2.6 Subsection 1

Lorem ipsum dolor sit amet, ut veri deleniti eloquentiam sea (**?**). Ea commodo aperiam complectitur pri, usu et case dolore. **?** ad quidam regione percipitur, est ut possit bonorum persecuti. Quis utinam offendit eu usu, eu accumsan disputando per, id cibo reprehendunt sit (**??**). In melius legendos corrumpit pro. Eos dico dignissim voluptatibus et, duo nisl cibo ut. Diceret periculis posidonium cum eu. **?** regione nam ex. Vix id viris phaedrum. Pri augue cetero probatus ut.

A nice little way of leaving yourself notes and reminders:

(Write Lit Review in English) **ToDo**

2.7 Subsection 2

In Table 1 an example table is provided.

Table 1: A table caption.

Animal	**Description**	**Price ($)**
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

[1] Like this one: `http://www.ncirl.ie`

References

Alrawi, O., Lever, C., Valakuzhy, K., Court, R., Snow, K., Monrose, F. and Antonakakis, M. (2021). The circle of life: A Large-Scale study of the IoT malware lifecycle, *30th USENIX Security Symposium (USENIX Security 21)*, USENIX Association, pp. 3505–3522.
URL: *https://www.usenix.org/conference/usenixsecurity21/presentation/alrawi-circle*

Bessler, M., Sangster, P., Upadrashta, R. and OConnor, T. (2024). Hardening the internet of things: Toward designing access control for resource constrained iot devices, *Proceedings of the 17th Cyber Security Experimentation and Test Workshop*, CSET '24, Association for Computing Machinery, New York, NY, USA, p. 1–7.
URL: *https://doi.org/10.1145/3675741.3675744*

Guo, T., Zhang, P., Liang, H. and Shao, S. (2013). Enforcing multiple security policies for android system.

Hossain, M. M., Fotouhi, M. and Hasan, R. (2015). Towards an analysis of security issues, challenges, and open problems in the internet of things, *2015 ieee world congress on services*, IEEE, pp. 21–28.

Ko, J.-Y., Lee, S.-G. and Lee, C.-H. (2019). Real-time mandatory access control on selinux for internet of things, *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1–6.

Malik, A. K., Emmanuel, N., Zafar, S., Khattak, H. A., Raza, B., Khan, S., Al-Bayatti, A. H., Alassafi, M. O., Alfakeeh, A. S. and Alqarni, M. A. (2020). From conventional to state-of-the-art iot access control models, *Electronics* **9**(10).
URL: *https://www.mdpi.com/2079-9292/9/10/1693*

Miki, M., Yamauchi, T. and Kobayashi, S. (2023). Evaluation of effectiveness of mac systems based on lsm for protecting iot devices, *2023 Eleventh International Symposium on Computing and Networking (CANDAR)*, pp. 161–167.

Miki, M., Yamauchi, T. and Kobayashi, S. (2024). Effectiveness of mac systems based on lsm and their security policy configuration for protecting iot devices, *J. Internet Serv. Inf. Secur.* **14**: 293–315.
URL: *https://api.semanticscholar.org/CorpusID:272633061*

Nakamura, Y. and Sameshima, Y. (2008). Selinux for consumer electronics devices, *2008 Linux Symposium*, pp. 125–134.

Nakamura, Y., Sameshima, Y. and Yamauchi, T. (2015). Reducing resource consumption of selinux for embedded systems with contributions to open-source ecosystems, *Journal of Information Processing* **23**: 664–672.

Ravidas, S., Lekidis, A., Paci, F. and Zannone, N. (2019). Access control in internet-of-things: A survey, *Journal of Network and Computer Applications* **144**: 79–101.
URL: *https://www.sciencedirect.com/science/article/pii/S108480451930222X*

Song, L. and Li, J. (2024). ebpf: Pioneering kernel programmability and system observability - past, present, and future insights, *2024 3rd International Conference on Artificial Intelligence and Computer Information Technology (AICIT)*, pp. 1–10.