BlazeVMM: Fast Startup of ARM CCA MicroVMs for Confidential Serverless Computing

Jiamei Ren*
School of Cyber Science and Technology
University of Science and Technology of China
Hefei, Anhui, China
renjiamei@mail.ustc.edu.cn

Hang Yang
School of Cyber Science and Engineering
Southeast University
Nanjing, Jiangsu, China
220235292@seu.edu.cn

Yong Liao
School of Cyber Science and Technology
University of Science and Technology of China
Hefei, Anhui, China
yliao@ustc.edu.cn

Abstract

To enhance the trustworthiness of cloud computing, confidential computing leverages hardware-supported Trusted Execution Environment (TEE) to provide security assurances. However, the additional initialization overhead of traditional confidential virtual machines makes them unsuitable for the high-performance demands of serverless platforms. We propose BlazeVMM, a lightweight VMM built upon Firecracker that supports launching ARM CCA microVMs. Additionally, by utilizing a pre-measurement mechanism, we have optimized the initialization measurement process during Realm startup, avoiding the overhead incurred within the RMM. We show that BlazeVMM improves cold start performance of ARM CCA VMs over current methods.

CCS Concepts

• **Security and privacy** → Systems security; Operating systems security.

Keywords

ARM CCA, Firecracker, microVM, fast startup, confidential serverless computing

ACM Reference Format:
Jiamei Ren, Hang Yang, and Yong Liao. 2025. BlazeVMM: Fast Startup of ARM CCA MicroVMs for Confidential Serverless Computing. In *The 4th International Conference on Computer, Artificial Intelligence and Control Engineering (CAICE 2025), January 10–12, 2025, Heifei, China.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3727648.3727789

1 Introduction

Serverless computing [1], also referred to as Function-as-a-Service (FaaS), represents a cloud computing execution model. In Serverless computing, the cloud provider dynamically handles the allocation and provisioning of servers, and developers can use services without managing the underlying infrastructure. Due to it reduce operational complexity and has lower costs, serverless computing has gained widespread attention.

As cloud services continue to grow in popularity, there is increasing concern about data confidentiality and runtime security in the cloud. Especially in serverless scenarios, user must trust to untrusted cloud providers. Confidential computing aims to make cloud computing more confidential by creating services on hardware-supported Trusted Execution Environments (TEE) [2-4]. TEE runs sensitive data and workloads in the isolated enclaves so that the untrusted cloud provider cannot access the data.

However, confidential virtual machines do not meet the high performance requirements of the serverless platform [5]. It is a challenging task to improve the startup speed of confidential VM, because traditional virtual machine optimization methods [6] have no effect. To ensure the integrity and confidentiality, confidential computing need data encryption, integrity measurement, and remote attestation. While these processes are critical for security, they also introduce substantial resource consumption and time overhead.

In this paper, we have adopted microVM [7] as the foundation for confidential serverless computing. MicroVMs consume minimal resources while providing the same level of isolation as traditional virtual machines [8]. By analyzing existing methods, we found that Firecracker [7] has the potential to reduce startup times because it does not require the full boot process of system. Firecracker currently does not support the ARM CCA platform. Motivated by these insights, we propose BlazeVMM, which can leverage Firecracker technology to launch ARM CCA confidential microVMs. This is the first time firecracker has been applied to ARM CCA, we believe it can take new possibilities for confidential computing on serverless platforms. Through the analysis of the Realm startup process, we also found that the Realm initialization measurement account for a large porion of the startup time [8]. The measurement process is always complex, involving many components and encryption algorithms. The large size of components that need to be measured, such as the kernel and initrd, also make it slow. We design a Pre-measurement Mechanism to avoid using time-consuming components and pre-calculating hash values outside the RMM.

2 Background

2.1 Firecracker

The traditional VM always runs on QEMU/KVM. QEMU has many functions and it offers flexibility, but it has disadvantages in security,

*Corresponding author

CAICE 2025, Heifei, China
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1264-7/2025/01
https://doi.org/10.1145/3727648.3727789

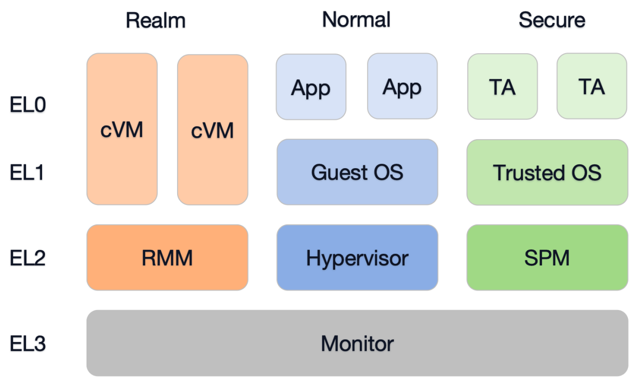

Figure 1: ARM Confidential Compute Architecture.

performance, and startup speed. Firecracker is a lightweight virtualization technology based on KVM and it can launch microVMs. Compared to regular virtual machines, Firecracker microVMs have a lower memory footprint and faster startup speeds. Compared to containers, Firecracker offers strong security isolation. Firecracker is implemented in Rust, so it can provide memory safety and thread safety at the level of language.

2.2 ARM Confidential Compute Architecture (ARM CCA)

In the trend of confidential computing in cloud service platforms, ARM has introduced a confidential computing architecture (CCA) within the ARMv9. Figure 1 illustrates the overall architecture of CCA. This confidential computing architecture allows the deployment of applications or virtual machines, preventing privileged software from accessing critical data and registers. The VMM is responsible for allocating and managing memory of virtual machines without unsafe behavior. ARM has introduced hardware extensions called Realm Management Extension (RME) and software extensions called Realm Management Monitor (RMM) [9]. While TrustZone has the normal world and the secure world, ARM has added two new domains: the Root world and the Realm world. In the exploration of the startup process for confidential virtual machine (cVM), the concept Realm introduced within the ARM architecture caught people‘s attention. Realm provides a protected execution environment for cVMs, and the entire process of launching a Realm is controlled by the VMM. In addition to providing isolation, Realm offers remote attestation services.

2.3 The Boot Process of Realm

We show the boot process of Realm in Figure 2. The VMM is initialized first and then configure key features, including SVE, PMU, and hash algorithms. Additionally, a Realm Personalization Value (RPV) is specified to authenticate the identity of the Realm. The Realm Descriptor (RD) with its parameters passed from KVM to the RMM. The RD contains various attributes of the Realm, including page table, IPAS information, the Realm’s current state, VMID and authentication-related data.

The Realm is then populated, and the initialization of IPA is completed based on the configuration information. The page table mapping and data copying are handled by the rmi_data_create and rmi_rtt_create functions, ensuring the correct mapping from Internal Physical Address (IPA) to Physical Address (PA) and copying the necessary data. Additionally, the Realm Entry Context (REC) is created, it is context established by the RMM when launching the Realm, used to manage the runtime status. Finally, the cVM is activated by setting its status to ACTIVATE and executing KVM_RUN.

The owner must gain trust before configuring any confidential information in the Realm. They need to know that the Realm is correctly constructed and it is running on the ARM CCA hardware. RMM is responsible for measurement during Realm startup and runtime. It will generate an attestation token based on the measurement information. A attestation report contains a CCA attestation token, which consists of two parts: the Realm token and the CCA platform token. In this article, we focus on the Realm token, including Realm Initial Measurement (RIM) and Realm Extensible Measurement (REM). The RIM is a hash of the initial state that can reflect features of VM. The four domains of REM are hashes of the firmware, bootloader, operating system, and applications running in the Realm. When the Realm is in the ACTIVE state, the RIM value becomes fixed.

3 Method: BlazeVMM

BlazeVMM is a comprehensive rapid startup solution for confidential microVMs based on ARM CCA. Figure 3 shows an overview of BlazeVMM. We microservice the startup process of confidential virtual machines and design a pre-measurement mechanism to enhance the startup speed of confidential microVMs.

3.1 Hypervisor Microservices Mechanism

Through our observation, it is feasible to use Firecracker to run the ARM CCA microVMs. But we need to make some changes to support the confidential virtual machine features. We use the KVM API for CCA to use the system call so that Firecracker can interact more deeply with other components and improve the virtualization performance. We implemented a entire process from configuration to creation, and the management of the Realm lifecycle until its destruction. Due to ARM CCA introduce the stage 2 and Memory Fault mechanism, and they are crucial for memory management, we have implemented these functions in BlazeVMM.

3.2 Pre-measurement Mechanism

Given that measurement is a critical but time-consuming step in the Realm startup process, we are exploring mechanisms for fast cold startup of microVMs. While the measurement and attestation implemented by RMM provides a strong defense against potential attacks, the measurements are performed on the critical path of the boot, significantly increasing the launch overhead. To address this issue, we found that the initial measurement process can be decoupled and executed separately on the CPU rather than on the RMM. This eliminates the need for the RMM to compute hashes during the startup process.

We design a pre-measurement calculator that can calculates the initial measurement values of a VM at startup and pass the

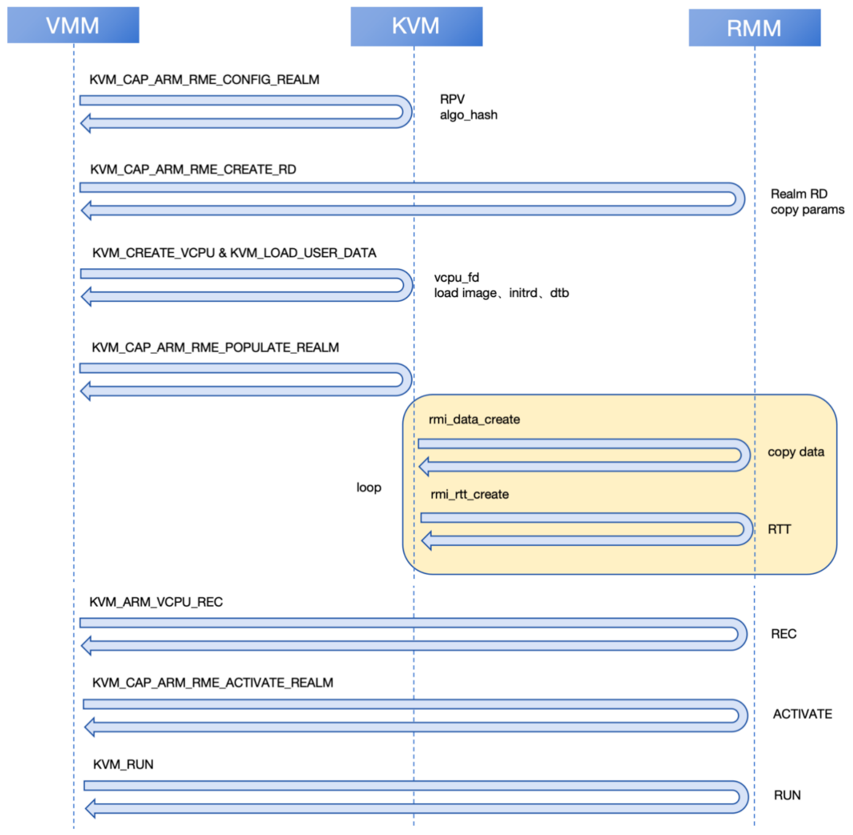

Figure 2: The boot process of Realm.

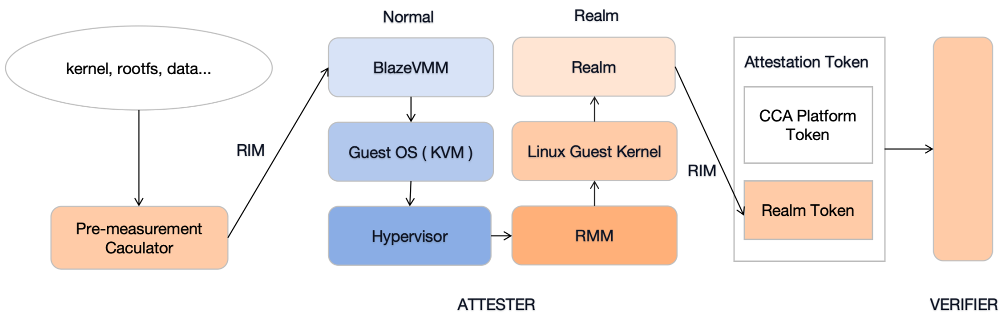

Figure 3: Overview of BlazeVMM.

calculated RIM directly to RMM from VMM. In addition, this tool can reuse the RIM values from existing cVMs, because the same cVM configuration has the same RIM and RIM does not alter after startup.

Figure 4 illustrates the process of generating RIM using pre-measurement caculator. The initial value of RIM is calculated based on some parameters of the Realm. During the construction of the Realm, the RIM value is extended in a predefined order: first the initial RIPAS of the guest RAM, then data granules in ascending order, then the RECs. When an operation that requires the expansin of RIM is performed, the RMM constructs a structure of measurement descriptor that contains the current RIM value. The calculation of RIM is a dynamic process that is continuously expanded and updated during the construction of the Realm, ensuring the integrity and consistency of Realm parameters.

To further optimize the RIM calculation process, we have introduced selective layered measurement based on the pre-measurement mechanism, allowing users to choose the level of measurement according to their needs. As shown in Figure 4, we divide the RIM calculation process into four steps. We found that

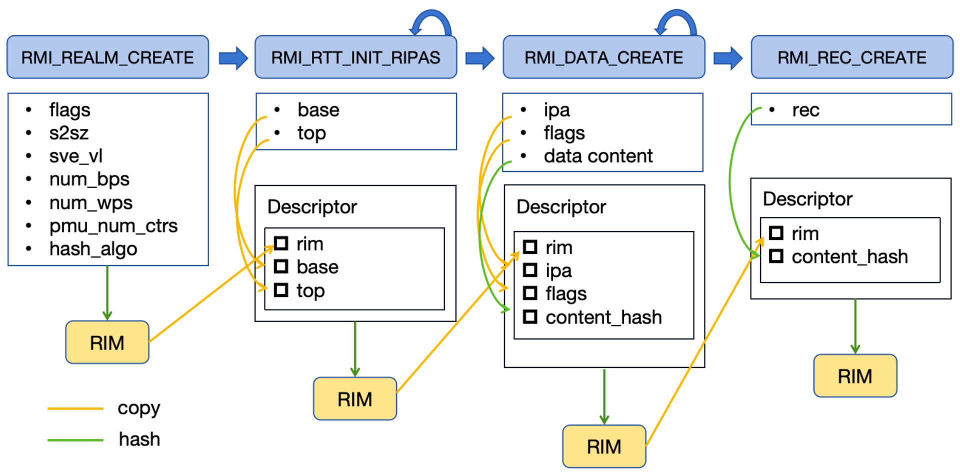

Figure 4: The calculation process of RIM.

some steps can be omitted to improve calculation speed. For example, the first and second step depend on configuration parameters. If the same parameters are provided, they will always produce the same hash value. We can pre-store the commonly hash values and skip these computation steps by using the stored hash values directly. However, the last two steps involve checking that the data and components are correctly loaded, so we do not recommend skipping them.

3.3 Implementation

We implemented the support for launching Realm cVM guests in Firecracker v1.10.0, which is open source and implemented in Rust. Our modifications do not affect the boot process of a normal VM. The code we added to support CCA is mostly contained in its own Rust module, and spans over 1000 lines of code. We also developed a pre-measurement caculator module that can calculate the initialization measurements for cVM startup outside the Realm environment. This module generates the RIM based on the provided VM configuration information, and following the specified order defined by CCA Realm. The final RIM value is passed to the RMM, which then generates the corresponding attestation token and completes remote attestation process.

4 Evaluation

4.1 Experimental Setup

Since CCA is still under active development and a CCA-compatible board has not been released yet, we build a prototype of BlazeVMM on QEMU-CCA [10], which is provided by Linaro for building an RME stack for QEMU. QEMU, in comparison to FVP, focus more on fast simulation, offering better performance and features. QEMU is not cycle-accurate because it does not attempt to simulate how much time an instruction takes on real hardware. But QEMU is instruction-accurate and allows for instruction counting during execution. This allows us to compare the overhead of starting cVMs with different VMMs by tallying the instruction counts [11], and demonstrate the feasibility of BlazeVMM through experimental

Table 1: Number of instructions during cVM startup with VMMs.

VMM	Number of Instructions (Millions)
QEMU	63,772
Kvmtool	48,294
Cloud-Hypervisor	42,034
BlazeVMM	39,023

results. Additionally, we will provide the data of startup times, which also offers a reference value for the overhead.

All experiments were conducted on a machine equipped with an 8-core ARM processor and 16GB of memory (OS: Ubuntu 24.04 LTS). All VMMs load a 44MB Linux 4.20 kernel and a 256MB root filesystem. The guest cVMs are configured with a single vCPU and 256MB of memory.

4.2 cVM startup

Table 1 presents number of instructions during cVM startup with our BlazeVMM and existing VMMs. It demonstrates that our design can reduce the startup time of confidential virtual machines.

The chart contrasts four different VMMs: QEMU [12], Kvmtool, Cloud-Hypervisor[13] and our BlazeVMM. QEMU has a long startup time because it needs to simulate and initialize a large number of hardware devices at startup. Kvmtool is a lightweight VMM designed for KVM, which is lighter than QEMU in some aspects. Cloud-hypervisor is a lightweight VMM designed for cloud environments, aiming to provide high-performance and high-density virtualization. Our BlazeVMM is based on Firecracker, which is further optimized to enable faster startup of microVM. To provide a more complete comparison, we provide four specific times for VMM to start the cVM, QEMU takes 74.59s, Kvmtools takes 60.95s, Cloud-Hypervisor takes 53.08s, and BlazeVMM's startup time is 49.93s, which is far less than the other three cases.

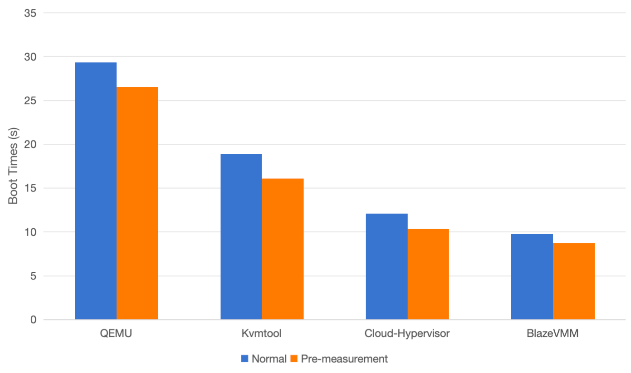

Figure 5: Comparative analysis of boot times for various VMMs with and without Pre-Measurement Mechanisms.

4.3 Pre-measurement

To evaluate the impact of the pre-measurement mechanism on the startup performance of virtual machines, we implemented our pre-measurement mechanism for four different VMMs and compared the effects with and without using this mechanism.

By comparing the results of two measurements, we can find that the measurement data is consistent, provding the correctness of this method. Additionally, we analyze the number of instructions and startup times, and found that the introduction of the pre-measurement mechanism significantly reduced the measurement startup time for the VMMs. Since the pre-measurement mechanism primarily reduces the measurement process in the RMM, and different VMMs only have minor differences in configuration information, the number of instructions reduced by optimization among different VMMs is similar. We found that the pre-measurement mechanism reduced about 1,241~1,530 million instructions during the startup of the VMM. In Figure 5, we also presented the startup times for the four VMMs, which were reduced 1.24~2.77s. These results demonstrate the effectiveness of the pre-measurement mechanism.

4.4 Security Analysis

Since the modifications of BlazeVMM only involve the VMM and RMM, without affecting the host environment or other components, our security analysis focuses on potential issues during the microVM startup process.

The VMM is not a part of the CCA's Trusted Computing Base (TCB), which means it is considered untrusted. While the VMM can initialize the Realm, it has no capability to access the Realm's data or memory. Our BlazeVMM is built on Firecracker, and preserve its original functionality and VM isolation. Furthermore, when faced with a malicious VMM, ARM CCA's measurement and attestation mechanisms ensure that the confidential environment remains secure.

We also examine the security of the RMM. Executing the pre-measurement mechanism outside the RMM not only reduces the size of the TCB but also conform the security principles. Due to the RIM value is generated based on the configuration information of the cVM, the same configuration have the same RIM. To distinguish between cVMs with the same configuration but different behaviors, ARM CCA provides a value, RPV, which is provided by the host. RPV is a separate value in the Attestation Token. When we verify the identity of a Realm, the RPV can be used to distinguish whether it is what we want. ARM CCA also offers runtime measurement mechanism to ensure that the Realm can not be tampered with.

5 Conclusion

BlazeVMM is a comprehensive rapid startup solution for confidential microVMs based on ARM CCA. We microservice the startup process of cVMs by utilizing Firecracker tecnology. Through pre-measurement mechanisms, we further optimize and enhance the startup speed of cVMs. With these two mechanisms, BlazeVMM significantly accelerates the startup of confidential serverless computing.

References

[1] Apache Software Foundation. Apache openwhisk: Open source serverless cloud platform. http://openwhisk.apache.org/. Accessed on 2021-01-04.
[2] Intel®Software Guard extensions (Intel®SGX). 2015. https://www.intel.com/content/www/us/en/developer/videos/intelsoftware-guard-extensions-sgx.html?wapkw$=$intel+sgx.
[3] Advanced Micro Devices (AMD). AMD secure encrypted virtualization (SEV). https://developer.amd.com/sev/, Dec 2022.
[4] Arm confidential compute architecture. https://developer.arm.com/documentation/den0125/0200/?lang$=$en. Referenced December 2022.
[5] Yanqi Zhang, Íñigo Goiri, Gohar Irfan Chaudhry, Rodrigo Fonseca, Sameh Elnikety, Christina Delimitrou, and Ricardo Bianchini. Faster and cheaper serverless computing on harvested resources. In Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles, SOSP '21, page 724–739, New York, NY, USA, 2021. Association for Computing Machinery.
[6] Carlos Segarra, Tobin Feldman-Fitzthum, Daniele Buono, and Peter Pietzuch. 2024. Serverless Confidential Containers: Challenges and Opportunities. In Proceedings of the 2nd Workshop on Serverless Systems, Applications and MEthodologies (SESAME '24). Association for Computing Machinery, New York, NY, USA, 32–40. https://doi.org/10.1145/3642977.3652097
[7] Alexandru Agache, Marc Brooker, Alexandra Iordache, Anthony Liguori, Rolf Neugebauer, Phil Piwonka, and Diana-Maria Popa. Firecracker: Lightweight virtualization for serverless applications. In USENIX Symposium on Networked Systems Design and Implementation, Santa Clara, CA, February 2020.
[8] Benjamin Holmes, Jason Waterman, and Dan Williams. 2024. SEVeriFast: Minimizing the root of trust for fast startup of SEV microVMs. In Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24), Vol. 2. Association for Computing Machinery, New York, NY, USA, 1045–1060. https://doi.org/10.1145/3620665.3640424
[9] 2023. Realm Management Monitor specification. https://developer.arm.com/documentation/den0137/latest
[10] 2024. Building an RME stack for QEMU. https://linaro.atlassian.net/wiki/spaces/QEMU/pages/29051027459/Building+an+RME+stack+for+QEMU.
[11] Sandra Siby, Sina Abdollahi, Mohammad Maheri, Marios Kogias, and Hamed Haddadi. 2024. GuaranTEE: Towards Attestable and Private ML with CCA. In Proceedings of the 4th Workshop on Machine Learning and Systems (EuroMLSys '24). Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3642970.3655845
[12] https://git.codelinaro.org/linaro/dcap/qemu. 2023.
[13] Intel Cloud Hypervisor. https://www.cloudhypervisor.org/. (Accessed on 2023-01-10)