Exploring Operating System Mechanisms for Practical and Resilient Confidential Computing

Background. Cloud computing underpins critical services in healthcare, finance, and AI. Yet, it faces a constant threat of cyberattacks and data breaches. In 2024, these cybercrimes exposed over 280 million records in the US's health sector alone, and in 2025, they and cost up to $4.4 billion in remediation on average, worldwide. This not only devastate financially but also severely erode public trust in cloud.

To restore trust in cloud computing systems, cloud providers have started to isolate their customers' data from their own platforms by adopting Trusted Execution Environments (TEEs), built on top of modern CPU features like AMD SEV. TEEs protect customers' data against a strong threat model, resistant to even attacks from a cloud administrator. TEEs form the foundation of Cloud Confidential Computing (CCC), a paradigm shift toward trustworthy private data processing in cloud.

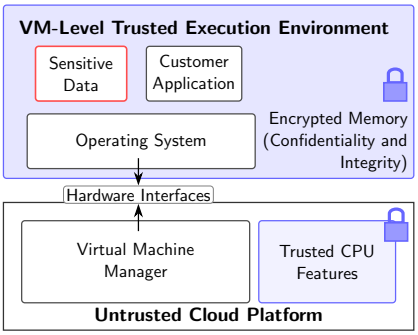

Figure 1: VM-Level TEEs in Cloud Confidential Computing

The latest TEEs can now fully isolate virtual machines (VMs), made possible by technologies like AMD SEV (Figure 1). This improves usability by safeguarding existing workloads with minimal modifications and boosts performance by reducing interactions with the host system. However, this advancement means that TEEs must now secure the entire software stack of the VM – or its Trusted Computing Base (TCB). In this picture, the Operating System (OS) is particularly problematic. As of 2025, the Linux kernel contains around 40 million lines of code – seeing over 3,500 vulnerabilities in 2024 that attackers can leverage. Furthermore, the OS's complex interactions with the underlying hardware – now being virtualized by the untrusted platform – further complicate efforts to secure CCC.

Proposal overview. This proposal seeks to make CCC more robust and practical, building on the observation that the industries' shift toward VM-level TEEs also exposes a golden opportunity: the OS can now actively shape the security of CCC. My doctoral studies demonstrated the potential of this direction by designing INCOGNITOS, a side-channel mitigation technique that harnessed OS components such as memory management and task scheduling. The work provided a proof of concept, showing that OS abstractions can be employed for principled and deployable security. Building on that foundation, my postdoctoral research will develop a broader class of OS-supported techniques that address the practical challenges of CCC deployments.

At UBC, I will pursue this agenda under the supervision of Prof. Aastha Mehta; her expertise in both systems security and OS will be invaluable to the research. Moreover, I plan to enrich this research with perspectives from UBC's researchers, especially systems researchers at Systopia Lab and cloud computing researchers at CIRRUS Lab. Plan for these collaborations will be detailed in the remainder of this proposal, along with the challenges that will be the focus of my research: *side-channel attacks* and *private data processing*.

Focus 1: Side-channel attacks. *Side channels* are critical and persistent vulnerabilities at the software-hardware boundary of TEEs. They are indirect signals – such as the time difference measured when accessing a memory location – through which an attacker can infer private information. Such attacks are widely acknowledged as a formidable threat to CCC (AWS, Azure, IBM, and Alibaba). Efforts to completely remove side channels often run into roadblocks; some are buried deep in the processor's internal circuitry requiring massive hardware redesigns to eliminate, others come from everyday cloud resource management operations and removing them would impair flexibility.

A more practical, less intrusive approach is to mitigate these side-channel by transforming the sensitive application to obscure side-channel signals. Unfortunately, current state-of-the-art mitigation techniques fall short; systems such as Klotski and Obelix are difficult to adopt as they require developers to rewrite or recompile their entire software. Even when applied, they can slow applications by hundreds or even

thousands of times. Worse, to remedy the severe performance hit, security trade-off are often made, but they are ad hoc, without a sound theoretical basis.

My postdoctoral work will tackle the side-channel challenge from an OS vantage point. The prototype introduced during my Doctoral studies has limited deployability, relying on a minimal OS. I aim to push this direction further by incorporating side-channel mitigation into the commodity Linux OS to support diverse cloud workloads. This direction especially synergizes with Aastha Mehta's profound experiences on side-channel attacks and defenses. Further, the work of userspace memory management by Margo Seltzer and Alexandra Fedorova will play an essential role in making the side-channel mitigation technique using memory management practical. The large TCB of commodity OSes also makes side-channel elimination particularly challenging – it requires tracking potential side-channel leaks across millions of lines of code. To this end, the study of cross-system information flow tracking, pioneered by Margo Selter and Thomas Pasquier, makes it a tractable problem.

Focus 2: Private data processing. A common scenario for CCC deployments is private data processing in the cloud that involve multiple stakeholders, as shown in Table 1. Take personal data analytic for example; users of services like 23andMe are required upload private data to the service server, which runs some health analytic program on the data nd returns the results to the user. On the one hand, the user wants to verify that the service provider is using their data responsibly; in other words, keep the data secret and does not leak it else where. On the other hand, to prove this, the service provider is required to publicize their data processing code for the user to verify, which might contain proprietary business logic. While CCC excludes cloud providers from the threat model, the conflicts between service users and providers persist.

Stakeholder	Sensitive Asset	Interest
Service users	Personal data	Ensure data confidentiality
Service providers	Proprietary business logic	Keep business logic confidential

Table 1: Private data processing stakeholders, assets and interests.

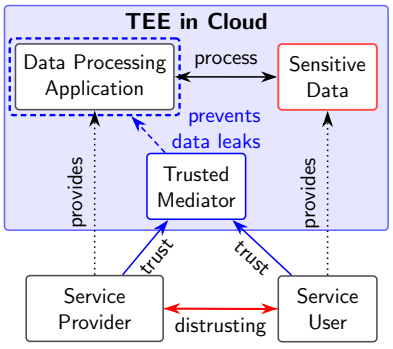

Figure 2: How CCC resolves stakeholders' conflicts with a mediator.

Existing deployments employs a TEEs that functions as a mediator that is trusted by both service users and providers. The service provider first send the data processing program to the trusted mediatior, which verify that the program maintain data confidentiality. Because the users trust the mediator, they now send private data to the mediator, on which it will run the data processing program on. Still, most research prototypes fail to introduce a deployable solution applicable to many cloud programs. Most solutions enforce strict information flow control, e.g., requiring that the data processing program must not write personal data to a file. State-of-the-art solutions like PAVE and Erebor enforce this by placing the program inside a sandbox, constraining its actions, such as system calls, making them incompatible with many workloads.

I will tackle this problem from an OS-first perspective. Particularly, within the OS community, the practice of tracking whole-system provenance – tracking the flow of information across all software layers – is common. While existing systems prevent any potentially dangerous actions with the operating system, I aim to mitigate only the point at which sensitive data is about to be exported to the outside. For instance, almost all previous systems prevent the private data processing program from accessing the file system; a provenance-aware solution could permit file accesses, as long as it does not transmit the information outside.

1 Appendix: Guide (REMOVE IN FINAL)

The proposal should be written in clear, non-technical language that allows a non-specialist to comprehend the overall content and importance of the work. Members of the Killam Postdoctoral Fellowships and Prizes Committee are from abroad range disciplines and may not have expertise in your area of study.

Although the fellowship may be used to extend or expand upon doctoral work, it must be made clear that you are not intending to use the award to wrap up a thesis. While it is expected that a postdoctoral fellow will be taking the next step beyond the PhD thesis, you must differentiate clearly between the postdoctoral project and the thesis research. Feel free to include hyperlinks to provide links to additional information.

As applicants must make UBC their base, it is important – particularly for applicants whose primary research materials are elsewhere – to indicate what travel is involved, to where, and for how long. You should describe how you will deal with the remoteness of the primary materials.

The adjudication committee is very interested in "fit" with the selected UBC department or unit and the university's research programs. You must provide information on how your research relates to that of specific campus programs and advisors. If a colloquium is envisioned, a possible title should be proposed. If you will visit in classes, please suggest which ones. Since inter-disciplinarity is often a valuable dimension (the Killam Trusts declares that a candidate shall not be "a one-sided person"), specific details on proposed inter-departmental connections are welcome.