Leveraging Hardware Capabilities for Building Dependable Many-Core Operating Systems

Nathan S. Rutherford[1], Darren Hurley-Smith[2], Dan O'Keeffe[3]

[1]Centre for Doctoral Training in Cyber Security for the Everyday, [2]ISG Smart Card and IoT Security Centre, [3]Software and System Security Lab (S3Lab), Information Security Group, Royal Holloway, University of London

ROYAL HOLLOWAY UNIVERSITY OF LONDON

Objectives

- Distributed kernel design to support fault tolerance
- Treat specialised processors as first-class citizens in the operating system
- Provide cross-domain process communication (XPC) and enforce core isolation and security policies using hardware capabilities
- Flexible abstraction model to support hardware accelerators

Introduction

Post-Moore's Law computer architectures are moving towards:

- Specialised Processor Technology (SPT) [7]
- General Processor Technology (GPT) that increase the number of cores
- Operating System (OS) designs that support heterogeneity [6]

Multikernel architectures like Barrelfish [2] emerged to support growing core count:

- Treats multi-core systems as a distributed system running a separate kernel per core
- Not suitable for specialised devices that cannot run an OS Kernel e.g. GPUs [5]

Most research centred around performance and scalability

- Little focus on dependability

Background

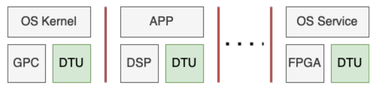


Figure 1: Architecture diagram of the M^3 system adopted from [1]

M^3 [1]

- Network-on-a-chip level isolation
- Common hardware interface via Data Transfer Unit (DTU)
- Extended to support distributed capabilities model [3]

CHERI [8]

- Capabilities part of the chips Instruction-Set-Architecture (ISA)
- Security policy enforced by the hardware
- Built-in spatial memory safety
- Temporal safety possible

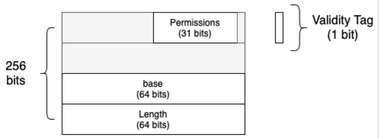


Figure 2: Original CHERI capability visualisation from [8]

Early System Design

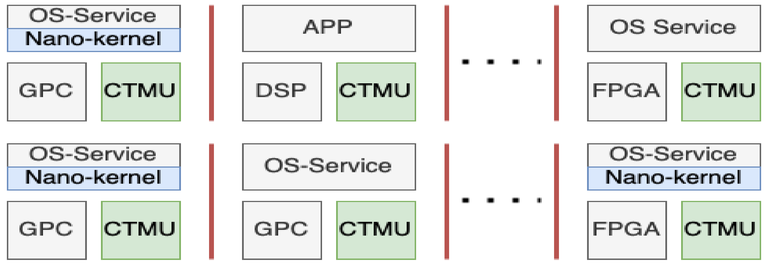


Figure 3: Early system diagram with a distributed nano-kernel for OS management and a Capability Transfer and Management Unit (CTMU) for supporting communication between heterogeneous cores and enforcing fine-grained security policy

Overview

Distributed kernel architecture supported by a CHERI-aware hardware chip, treats kernel management as a multi-core consensus problem

Capability Transfer and Management Unit (CTMU)

- Manages enforcement of security policy
- End-point for cross-core communications

Nano-kernel

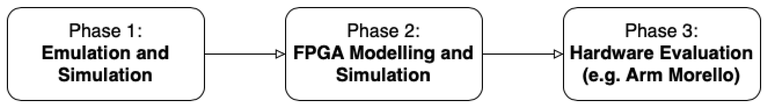
- Optional component as not all cores may support it
- Resource management function
- Replicated across all nano-kernel instances via consensus
- Quick recovery time on failure, or in periods of high utilization in a region

Flexible Abstraction Model

- Null-kernel composition [4]
- Treat hardware accelerators as first-class citizens

Evaluation Plan

Two-phase evaluation, with an optional third phase depending on hardware availability



Key Performance Metrics:

- Fault Tolerance & Recoverability
- Latency

Research Opportunities

- 5G Infrastructure
- Native interfaces for cryptography accelerators
- Media encoding
- Datacenters

Conclusions

- Most many-core designs treat the OS as a distributed system, however few focus on dependability
- We propose a many-core OS design leveraging hardware capabilities to distribute the kernel across cores, allowing higher levels of fault-tolerance and enabling rapid recoverability

References

[1] N. Asmussen, M. Völp, B. Nöthen, H. Härtig, and G. Fettweis.
M3: A Hardware/Operating-System Co-Design to Tame Heterogeneous Manycores.
In *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, pages 189–203, New York, NY, USA, 2016. Association for Computing Machinery.
event-place: Atlanta, Georgia, USA.

[2] A. Baumann, P. Barham, P.-E. Dagand, T. Harris, R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania.
The multikernel: a new OS architecture for scalable multicore systems.
In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles - SOSP '09*, page 29, Big Sky, Montana, USA, 2009. ACM Press.

[3] M. Hille, N. Asmussen, P. Bhatotia, and H. Härtig.
SemperOS: A Distributed Capability System.
In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 709–722, Renton, WA, July 2019. USENIX Association.

[4] J. Litton, D. Garg, P. Druschel, and B. Bhattacharjee.
Composing Abstractions using the null-Kernel.
In *Proceedings of the Workshop on Hot Topics in Operating Systems*, pages 1–6, Bertinoro Italy, May 2019. ACM.

[5] C. J. Rossbach, J. Currey, M. Silberstein, B. Ray, and E. Witchel.
PTask: operating system abstractions to manage GPUs as compute devices.
In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 233–248, 2011.

[6] J. Shalf.
The future of computing beyond Moore's law.
Philosophical Transactions of the Royal Society A, 378(2166):20190061, 2020.
Publisher: The Royal Society Publishing.

[7] N. Thompson and S. Spanuth.
The decline of computers as a general purpose technology: why deep learning and the end of Moore's Law are fragmenting computing.
Available at SSRN 3287769, 2018.

[8] R. N. Watson, J. Woodruff, P. G. Neumann, S. W. Moore, J. Anderson, D. Chisnall, N. Dave, B. Davis, K. Gudka, B. Laurie, S. J. Murdoch, R. Norton, M. Roe, S. Son, and M. Vadera.
CHERI: A Hybrid Capability-System Architecture for Scalable Software Compartmentalization.
In *2015 IEEE Symposium on Security and Privacy*, pages 20–37, San Jose, CA, May 2015. IEEE.