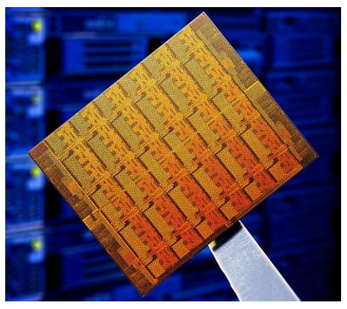
ETH zürich
Department of Computer Science

# Isolation on Many-core Architectures

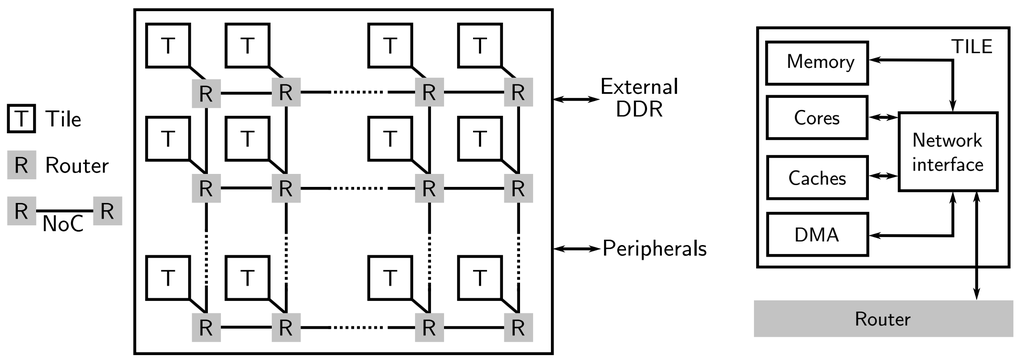
Ramya Jayaram Masti, Devendra Rai, Claudio Marforio, Srdjan Čapkun

## Abstract

The use of many-core platforms like Intel's Single-chip Cloud Computer (Intel's SCC) in cloud-like environments, requires them to support security guarantees found in common multi-core platforms. In this work we explore the problem of how to isolate execution of sensitive processes on many-core platforms. In particular, we identify the desirable properties of a security kernel that enables isolation on such platforms. We design a centralized security kernel that achieves isolation and assumes small hardware changes to Intel's SCC. We prototype our design and report the time needed to setup and execute isolated Linux instances.



## Background




**Many-core systems architecture**

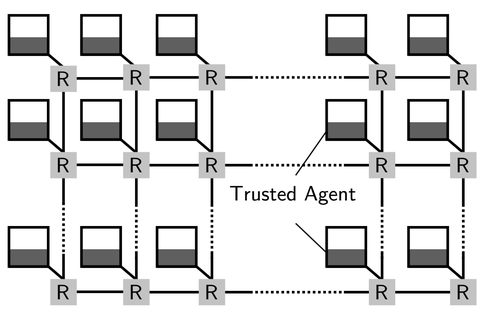
- Tiles communicate over a network-on-chip (NoC)
- Each tile consists of one or more cores, caches, on-tile memory and DMA controller
- The network consists of one router per tile

## Isolation

**Desirable Properties**

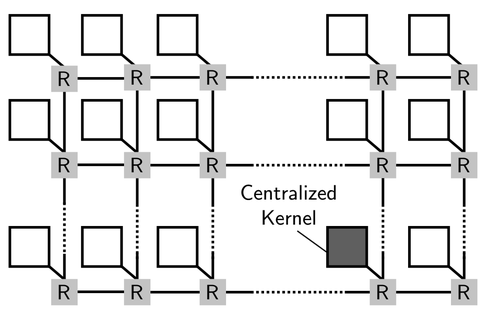
- **Small Security Kernel**
  Minimize interaction with co-resident (potentially malicious) software.
  Scheduling and resource management (disengaged).
- **Restricted Security Kernel Capabilities**
  Minimize the impact of its compromise.
  Must only be able to terminate a process and not schedule it (DoS).
- **Context Awareness**
  Mechanism to learn system configuration (e.g., sharing of resources).
  Preferably without interaction with the Security Kernel.

## Design Alternatives




**Distributed Security Kernel**

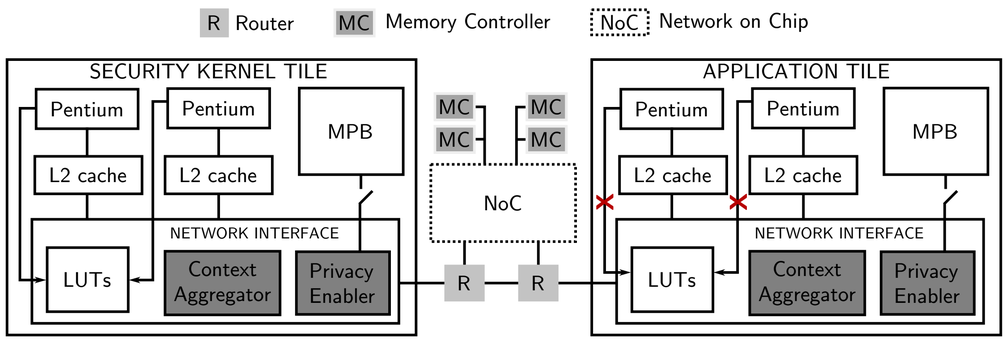
- Avoid single point of failure
- Implementable on current hardware
- Requires coordination between components




**Centralized Security Kernel**

- Better disengagement
- Less intrusive (e.g., for clouds)
- Requires hardware support

## Intel's SCC



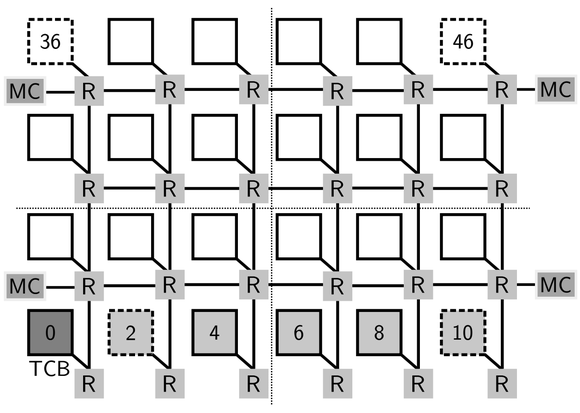

**Centralized Solution**

- Key intuition: LUTs control access to all system resources
- In its current implementation, each core can modify all LUTs in the system

**Required hardware changes:**

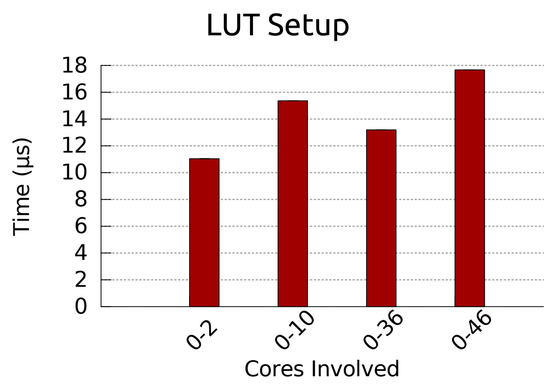
- Only the security kernel can modify LUTs
- Context Aggregator collects the status of LUTs
- Privacy Enabler prevents access to on-tile resources from other tiles/peripherals

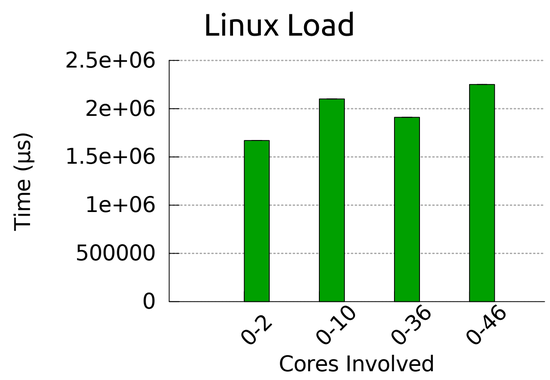
## Experiments

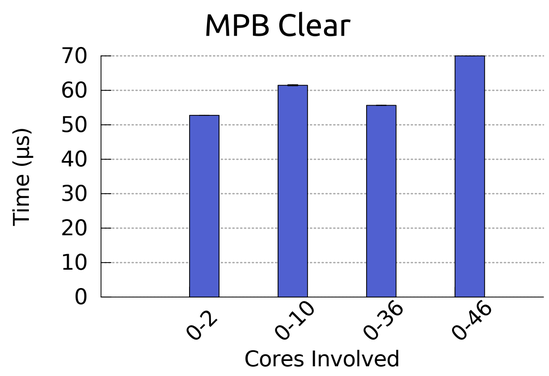


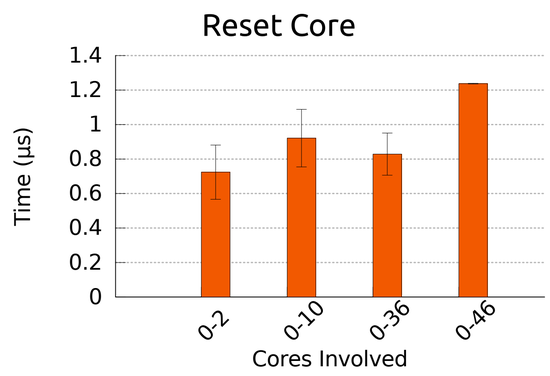

**Linux Setup Time**

1. Lookup Table Setup
2. Load executable (i.e., Linux image)
3. Clear on-tile memory (i.e., MPB)
4. Reset core to start execution




## Future Work

- Explore other security properties enabled by many-core systems
- Implement and compare distributed and centralized solutions for Intel's SCC
- Evaluate other commercially available architectures (e.g., Adapteva's Epiphany, Tilera's TilePro)

**References**

Intel Corporation, "SCC External Architecture Specication (EAS)", https://communities.intel.com/servlet/JiveServlet/previewBody/5852-102-1-9012/SCC EAS.pdf
S. Lukovic and N. Christianos, "Enhancing Network-on-chip Components to Support Security of Processing Elements", in Proceedings of the 5th Workshop on Embedded Systems Security, WESS'10, 2010
S. Peter, T. Roscoe, and A. Baumann, "Barrelsh on the Intel Single-chip Cloud Computer", http://www.barrelfish.org/TN-005-SCC.pdf, 2013

ramya.masti@inf.ethz.ch
Institute of Information Security

devendra.rai@tik.ee.ethz.ch
Computer Engineering and Networks Laboratory

claudio.marforio@inf.ethz.ch
Institute of Information Security

srdjan.capkun@inf.ethz.ch
Institute of Information Security