A Rising Tide Lifts All Boats: How Memory Error Prediction and Prevention Can Help with Virtualized System Longevity

Yuyang Du, Hongliang Yu, Yunhong Jiang*, Yaozu Dong*, Weimin Zheng
Tsinghua University
Intel Corporation*
Outline

- Background and motivation
  - Memory errors in the wild
  - Virtualized systems
- Our approach
  - Memory error characteristics
  - Prediction and prevention model
- System level design
- Conclusion and future work
Memory Error (1)

- Memory in computer

Memory Controller

CPU

Memory modules
Memory Error (2)

Memory error causes
- Service unavailable, data loss, etc.
- Reboot and replacement

Memory error in the wild
- Make up the largest fraction of system failures among all possible reasons (both software and hardware)
- Take up the top place of component replacement
Memory Error (3)

- Conventional solutions
  - Fault tolerance
    - adding certain form of redundancy (by hardware or software) to the system is necessary
  - Error Correction Code (ECC)
    - detecting and correcting one bit error
- Difficult to extensively apply or not enough to commodity computers in datacenters
Virtualized Systems (1)

- Virtual machines and Hypervisor

Diagram showing the relationship between physical host hardware, operating system, device drivers, and virtual machines. The diagram illustrates how virtual machines (VMs) are abstracted from the physical host hardware by the virtual machine monitor (VMM).

- Physical machine, OS, and applications
- VMs and Hypervisor
Virtualized Systems (2)

- A cloud business model
  - The providers and consumers adopt a “paying for what you use policy”
    - The provider guarantees some service level agreement, such as system uptime percentage
    - The consumers use and pay for the provisioned resources
  - Lack the opportunity to protect themselves from hardware failure, such as memory errors

- VM based high availability and fault tolerance
  - VM state replication, and record and replay
Vulnerabilities in virtualization

- Higher utilization due to server consolidation and live migration leads to an increase in memory error rate
- Many proprietary and legacy OSes cannot handle memory errors
- The “eggs in a basket” effect needs hypervisor to be more robust and resilient to system failures
- Memory errors can be used to attack system security
Our Goal

- **Cost–effective**
  - Can be applied to large scale

- **Best–effort**
  - Not as complete as fault tolerance
  - Alleviate the problem

- **Virtualization–aware**
  - Small footprint
  - Transparent to OS and applications
Outline

- Background and motivation
  - Memory errors in the wild
  - Virtualized systems
- Our approach
  - Memory error characteristics
  - Prediction and prevention
- System design
- Conclusion and future work
Our Solution

- This paper proposes memory error prediction and prevention model
  - Predict using memory error characteristics and log events
  - Prevent using page/DIMM replacement and live migration
Memory error category

- **Soft error vs. hard error**
  - Soft errors randomly flip memory bits but without permanent physical damage (e.g., particles strike on the silicon chip)
  - Hard errors repeatedly corrupt bits due to device defect (e.g., device wear-out)

- **Correctable vs. uncorrectable**
  - If the errors can be corrected by hardware (i.e., Error Correcting Codes), then software is oblivious to such events and can continue running
  - If the errors can not be corrected, system failure is unavoidable.
Memory error characteristics

- Hard errors occur more often than soft errors, which are unfortunately concerned by most previous studies.
- Strong correlation between correctable and uncorrectable errors.
- High system utilization (CPU and memory) increase memory errors.

Memory subsystem virtualization

- Physical page frame redirection

```
<table>
<thead>
<tr>
<th></th>
<th>VM1</th>
<th>VM2</th>
<th>VM3</th>
<th>VM4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
```

- Machine Physical Memory
- Guest Pseudo Physical Memory
Memory subsystem virtualization

- MMU Virtualization
  - Direct page table
  - Virtual TLB
  - Shadow page table
  - Hardware assisted translation tables (i.e., Extended page table)
    - Guest can have full control over its page tables and events
    - VMM controls Extended Page Tables
Outline

- Background and motivation
  - Memory errors in the wild
  - Virtualized systems
- Our approach
  - Memory error characteristics
  - Prediction and prevention
- System design
- Conclusion and future work
System Design (1)

- System overview

Diagram:
- Page Replacement
- DIMM Replacement
- VM Live Migration
- Operator
- Memory Error Prediction & Prevention
- Bookkeeping
- Risk Assessor
- Error Log
- Error Handler
- Event Collector
- Utilization Monitor
Components
- Event Collector
- Error Log
- Utilization Monitor
- Risk Assessor
  - Error Prediction
  - Hotspot Avoidance
- Operator
  - Page replacement
  - DIMM replacement
  - Live migration
System Design (3)

- Unexpected uncorrectable errors
  - The memory is probably owned by guest VMs, because hypervisor has very small footprint
    - Destroys the guest VM and restarts it, or invokes guest’s error handler to deal with the error by passing a simulated MCE event to the guest
  - If the hypervisor owns the faulty memory, however error handler will probably have no choice but reboot the system
System Design (4)

- Hardware requirement
  - Intel Machine Check Architecture (MCA)
    - Corrected Machine Check Interrupt (CMCI)
    - Machine Check Exception (MCE)
Implementing on Xen

- Replacing a physical page for a guest VM requires the updating of all the page mappings from guest physical address to host physical address, and vice versa.
- The overhead of page replacement is very small, especially when Expanded Page Table (EPT) is used.
Outline

- Background and motivation
  - Memory errors in the wild
  - Virtualized systems
- Our approach
  - Memory error characteristics
  - Prediction and prevention
- System design
- Conclusion and future work
This paper advocates predicting memory errors and preventing them from affecting system longevity by using memory error characteristics and taking as input the error events and system utilization.

We focus on virtualized systems, and try to make it cost–effective to be applied to data centers easily.
Future Work

- We are still on the way to fulfill our work and collaborate with industry to take quantitative measurements to fine-tune the predictive model and evaluate the effects
Thanks!

Questions?