WHITE PAPER

Data Center
Hyperscale Cloud Data Center

Engineering Practice to Reduce Server Crash Rate from DDR Uncorrectable Errors (UCE) in Hyperscale Cloud Data Center

Tencent 腾讯

Business:

Tencent is one of the biggest cloud solution providers in China with a presence throughout three continents.

Tencent Seafront Towers in Shenzhen, China

Tencent: Yongkai Wu, Jingle Jiang, Eric Li, Alan Chen, Lidong Chen

Intel: Alex Zhou, Youquan Song, Fang Yuan, Xiaoguo Liang, Theodros Yigzaw, Subhankar Panda, Nishi Ahuja, Chris Fong

1. Background

In today's hyper scale Cloud Data centers, server crash rate has been one critical metric to measure RAS (Reliability, Availability and Serviceability) and a top concern to meet SLAs (Service Level Agreement) for cloud end users.

Of all the factors causing server crash such as fan failure, PSU failure, storage disk failure and so on, DDR associated errors including correctable errors (CE) and uncorrectable errors (UCE) are increasingly becoming one key contributor (especially DDR UCE causes more often server crash).

Considering today's sever system with more and more CPU cores (or Virtual Machines - VM) and DDR DIMMs operating on it, the impact of server crash caused by DDR UCEs is much bigger than before. Thus, finding effective engineering solutions to address them becomes essential.

1.2. Coverage of This Document

This paper is mainly focused on the discussion of mitigation solutions of DDR UCE errors on the Purley platform (for the First and Second Generations of the Intel® Xeon® Processor Scalable Family CPUs) and mainly to cover:

1. Intel available technologies to mitigate impact of DDR UCEs.
2. Tencent Cloud's practice case study on some detailed work to address DDR UCEs on Purley servers, especially figuring out a workaround to address those fatal errors during VM migration that still cannot be recovered even after a MCA (Machine Check Architecture) is deployed.
3. Outlook of future work.

1.3. Target User of This Document

The target audience for this document are users who already have familiarity with Intel® Xeon® Processors RAS technologies, and DDR failure modes in CSP environments:

1. System engineers
2. RAS/BIOS engineers
3. OS kernel engineers
4. Platform architects/RAS architects

1.4. Notice

There are some OS patches as workaround solutions to enhance MCA recovery, to address more DDR UE cases in this document. It should be noted that those patches are customized, and may NOT upstream, or eventual upstream patches may look very different.

2. Intel Tech to Reduce Server Crash Rate from DDR UCE

If the memory error cannot be corrected by a Purley ECC code, it will become an UCE. The Intel® Xeon® Processor silicon of the Purley platform supports the following RAS features to mitigate the impact of uncorrectable errors; meaning, they reduce server crash rate from DDR UCE.

1. MCA recovery
2. Address range mirroring
3. Post Package Repair (PPR)
4. Patrol scrub

In addition, always upgrading to the latest Intel BKC for the platform and conducting some advanced memory test at manufacturing stage, will help mitigate the impact of DDR UCEs as well.

Note: This paper is not intended to describe all RAS features on the Purley platform, but focus on the previous points mentioned. For a full list of RAS features, refer to the Intel document #563361.

2.1. Updating to the Latest Intel Platform BKC

It's always a best practice to keep the servers configured to the latest BKC, considering it may contain fixes and optimizations relevant for your configuration. Be sure to contact your Intel field representative to get the latest version of BKC before planning to take other actions.

2.2. Conducting Advanced Memory Tests

Other good practice is to run an advanced memory test tool (either from Intel or known third party vendors) to filter out any marginal or weak DDR DIMMs at manufacturing stage. Intel is working to release a tool for that purpose and is also considering consolidating third party test coverage into the tool.

Before a one stop solution is in place, it's still recommended to run both Intel's and a third-party tool to make sure a more complete coverage of memory stress test.

2.3. MCA Recovery

MCA recovery is an Intel advanced RAS feature that involves Silicon Hardware, Firmware, and OS application working together to recover from uncorrectable data errors to keep the physical machine operational.

There are several pre-conditions that need to be met to ensure a successful recovery:

1. Memory UCE is non-Fatal error.
2. Memory failure address isn't in kernel space.
3. The impacted application can be killed.

2.3.1. Difference Between Fatal/SRAR/SRAO/UCNA

Talking about MCA recovery technology, we need to continue to classify memory errors into these types:

1. Fatal error with memory UCE–**PCC (Processor Context Corruption)** happened, kernel need to be panic immediately.
2. SRAR (Software Recoverable Action Required) – **Action Required**: OS/App requires to take action (for example, Offline failure page/Kill failure thread) to recover this uncorrectable error.
3. SRAO (Software Recoverable Action Optional) – **Action Optional**: OS/App is optional to take action (for example, Offline failure page/Kill failure thread) to recover this uncorrectable error.
4. UCNA (Uncorrectable Error No Action required). No MCE interrupt assert, typically no action needs to be taken.

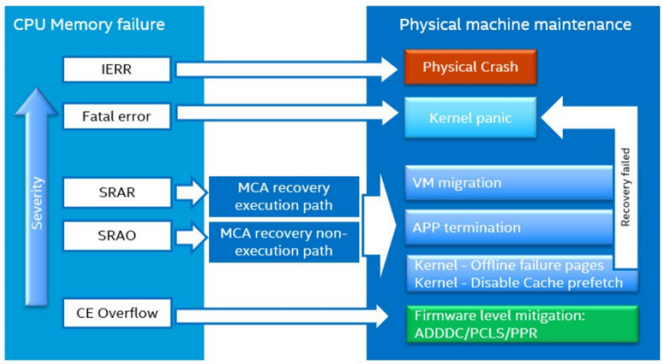
   The maintenance policy for these different types of errors is detailed in the next section.

Table 2-1. Classification of Different Types of DDR Errors

	Mci_STATUS							MCG_STATUS						
								Errored Thread		Other Threads				
	Val ID	UC	PCC	Service	AR (Action Required)	ADDRV	MISCV	RIPV	EIPV	RIPV	EIPV	Signaling	ADDR in Kernel Space	SW Action
Uncorrected Errors	1	1	1	x	x	x	x	x	x	x	x	MCERR	x	System crash.
SRAR – Instruction	1	1	0	1	1	1	1	0	0	1	0	MCERR	No	Take specific recovery action
SRAR – Instruction	1	1	0	1	1	1	1	0	0	1	0	MCERR	Yes	Kernel Panic
SRAR – Data Load	1	1	0	1	1	1	1	1	1	1	0	MCERR	No	Take specific recovery action
SRAR – Data Load	1	1	0	1	1	1	1	1	1	1	0	MCERR	Yes	May Kernel Panic
SRAO	1	1	0	1	0	1	1	1	0	1	0	MCERR	x	Optional for recovery action
UCNA	1	1	0	1	0	1	1	x	x	x	x	CMCI	x	Log the error and Optional for recovery action
CE	1	0	0	1	0	1	1	x	x	x	x	CMCI	x	Log the error and no corrective action required

2.3.2. Customer's Data Center Maintenance Policy

Figure 2-1. Data Center Maintenance Policy



Tencent's data center maintenance team will take specific actions based on the following failures descriptions:

IERR – When IERR happens, the physical machine is crashed. Data center maintenance need to harvest a crash dump log from BMC to help identify which component encounter failure later. After replacing the failed device, they will restart the system to bring the workload back online as soon as possible.

Fatal error – Fatal errors will result in kernel panic. Data center maintenance need to reboot the system and put the workload back online, then collect a Kdump and MCA log to identify which component encounters the failure later.

SRAR/SRAO – When a SRAR/SRAO happens, MCA recovery will be triggered. If the kernel can perform a successful recovery, the system can stay online as long as there is no additional UCE. At the same time, Data center maintenance will collect a failure log to identify which DIMM failed, and then trigger a VM migration to migrate the VMs to another healthy machine; the failure DIMM will be replaced after that. If the kernel can't perform a successful recovery, kernel panic will happen. Data center maintenance then needs to reboot the system and put the workload back online, then collect a Kdump and MCA log to identify which component encounter the failure.

CE overflow – When CE overflow happens, it means there is a hard memory failure for a specific DIMM; data center maintenance will check the failure log in BMC SEL to identify which DIMM needs to be repaired. The failure DIMM will be replaced after the VM is migrated to another healthy machine.

For UNC mentioned in the above section, the maintenance team needs a plan to migrate VM to another physical node and replace the failed DIMM

2.4. Patrol Scrub UCE Downgrade

To avoid that soft errors accumulate and become an uncorrectable error, the Intel silicon has a built in memory scrub engine to scrub the memory and eliminate soft errors that might accumulate over time.

Another benefit is, if the patrol scrubber finds an uncorrectable error, it will poison the cache line and report an SRAO event. Because SRAO errors happen outside of program execution, finding them early enough may help in avoiding some of the unrecoverable cases on the execution path (SRAR).

Notice that the second benefit is not attainable if patrol scrub events are downgraded to correctable events. Other implications are:

- The MCA overwrite rules state that corrected errors can overwrite other corrected errors. Therefore, there exists a window between a downgraded patrol scrub error getting logged and signaled via CMCI, and the CMCI handler getting engaged, where another corrected error might overwrite the downgraded patrol scrubbing error.
- Depending on the implementation, FRU isolation might be complicated as well. The patrol scrubbing engine will always poison the affected cache line and write it back to memory (recall that cache lines no longer report an error once poisoned). If the platform cannot intercept the corrected error log in time, then it will never be reported again, losing the details of the source of the Poison.

Patrol scrub errors are SRAO, which are by default signaled with MCE. These events are broadcast to all sockets/cores. One conflict case is a scenario where the patrol scrubber signals an SRAO event while at the same time one (or more) cores are handling an SRAR event. If the timing is right, it is possible for such a conflict to cause a shutdown.

For old generation platform (before Grantley) without MCA recovery, the system will crash immediately when memory UCE happens even if UCE could be found by patrol scrub.

For a new generation platform with MCA recovery, it's also found that multi SRAO UCE may cause nested MCE interrupts and finally become an IERR. Hence, It is possible to downgrade the patrol scrub SRAO to UCNA or other correctable error in the logging/signaling behavior and signal CMCI only. This would eliminate any possible conflict cases.

OS patches could be adopted to handle CMCI events caused by patrol scrub UCE, similarly as a MCA recovery event. It should be noted that since these get logged as correctable errors, they observe all the rules of correctable errors. They can be overwritten by other correctable errors. If "cloaking" is enabled, then these errors will not be observable from OS context.

Patrol Scrubbing is typically set for a 24-hour scrubbing interval (meaning, each address is scrubbed once every 24 hours).

2.5. PPR (Post Package Repair)

Boot Time Post Package Repair (PPR) provides a mechanism to repair a failed DDR4 DRAM row after manufacturing (that is, in the field). This is an optional feature for 4 Gb DRAMs, but required for 8 Gb and above densities.

Per JEDEC* specification, DDR4 DRAM can minimally correct 1 row per bank group which is identifiable via datasheet and module SPD. Intel® Xeon® processor-based systems Hard PPR (hPPR).

Boot Time PPR can be applied at several points, including:

- System manufacturing, if tests run at that time find row(s) that need to be repaired. PPR can be applied to replace the bad row(s).

- In-field, tests find a failed row. Advanced MemTest is a memory testing vehicle that can be run by MRC (Memory Reference Code) at boot time to locate DRAM row failures. If a bad row is encountered, it is repaired at that time if there is a spare row available.
- During runtime, BIOS gathers UCE logs (Uncorrectable Error Logs) to identify failures that are repairable via PPR. If so, BIOS can store that information until the next reboot and perform PPR at that time. In addition, BIOS may gather corrected error information and forecast a row failure in the future. Hence, it can employ PPR preemptively at the next reboot.

Intel's current Memory Reference Code (MRC) has the capabilities of logging correctable errors, then on the subsequent reboot, doing PPR on those correctable errors, should the frequency of correctable errors exceed a set threshold.

Intel is working to include additional capabilities to our MRC that will allow PPR on uncorrectable errors following a reboot, as well as a more stressful "Advanced MemTest" as described previously.

2.6. Address Range Mirroring

Address Rang Mirroring is one advanced memory RAS feature from the Intel Haswell server platform, which shows improvement on the Purley platform in comparison. It offers high granularity of mirroring, and can offline-configure the amount of mirrored memory size upon real demand . Even with some extra memory capacity consumed, this feature can help the server kernel to survive from DDR UCE in kernel space.

Address Range Mirroring (ARM) allows a portion of the address space to be mirrored, thus providing resiliency through redundancy. The purpose of ARM is to reduce the cost of mirroring by placing only the most critical data (such as the OS kernel) in the mirrored region. The FM-OS interface that allows the OS to request the FW to set up the range to be mirrored is specified in doc# 553716, *Platform Firmware-OS Interface Specification for Address Range Partial Memory Mirroring*. ARM is set up and the range to be mirrored specified (at TAD granularity – that is, 64 MB) at boot time.

With ARM, the given address range is paired within two memory channels (a primary and secondary) such that writes are duplicated to both channels, and reads are serviced from the primary channel as long as no uncorrected errors is detected. If an uncorrected error is detected, then the read is serviced from the secondary channel. In the event that the primary channel is deemed too unreliable, the mirror is broken (an event known as Mirror Failover).

A Mirror Failover event is considered a "service state", but the time scale where the service is considered necessary depends on the particular usage and customer experience. In highly reliable systems, a Mirror Failover event may trigger workload migration from that system to another to prevent possible interruption of service due to another uncorrectable data error in the (previously) mirrored region.

A system extracts the biggest improvement in resiliency when ARM is coupled with MCA Recovery. Considering MCA Recovery is typically not possible from uncorrectable errors in kernel space, this region can be provided resiliency through ARM. The rest of the address space can then participate in MCA Recovery, thus reducing the "blast radius" of any uncorrectable errors from memory.

For CSP customers, optimization of memory size used by the kernel space is also of great importance to minimize extra memory capacity consumed if applying this RAS feature.

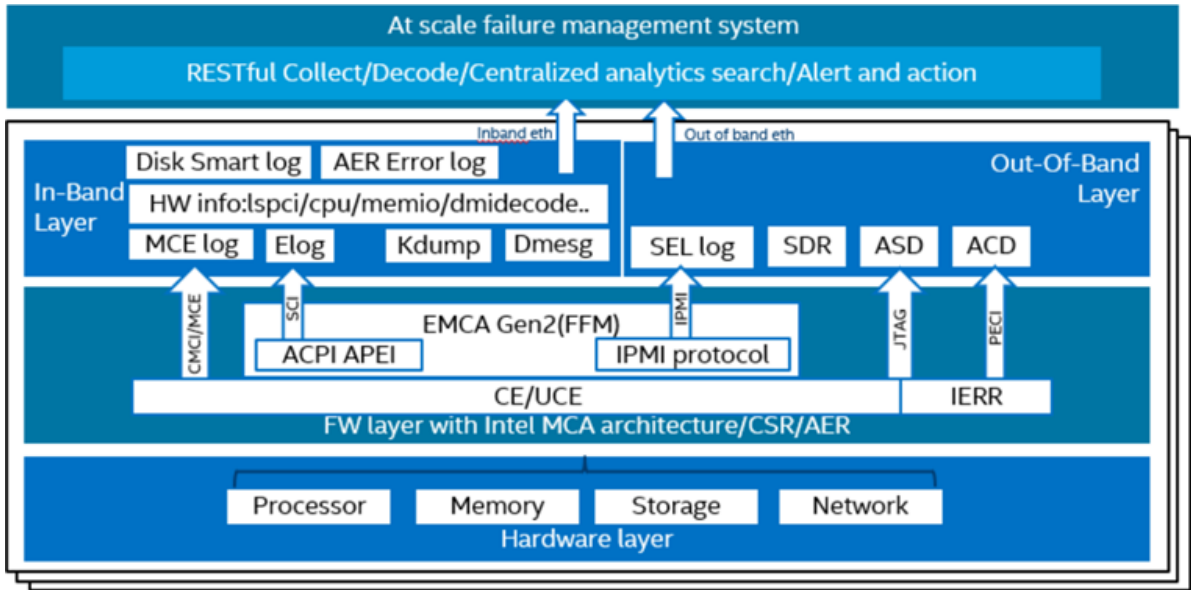
Note: Enabling both Address Range Mirroring and ADC/ADDDC at the same time is currently not a supported configuration.

3. Tencent Cloud Case Study on Purley Server Clusters

3.1. Original Memory UCE Failure Rate

In late Q4 of 2018, Tencent Cloud reported that about 50% of their Purley Cloud VM servers crashes came from memory UCE associated errors, and then closely work together with Intel to address this serious problem by pilot deployment of a series of solutions.

Figure 3-1. Failure Monitor System Diagram



This figure illustrates a high-level diagram of the customer's failure monitor system. The customer captured detail memory failure data with their failure monitor system.

3.2. Systematic Solutions

Based on the customer's failure data statistics, Intel proposed the solutions described in Table 3-1

It should be noted that, in practice, it's kind of difficult to breakdown how much improvement each action contributes because verification of each action takes several months to monitor its effectiveness. The aim here was to implement all solutions as soon as possible and keep monitoring the failure data report in a large-scale data center to analyze the overall effectiveness of improvement.

Table 3-1. Systematic Actions to Improve DDR UCE

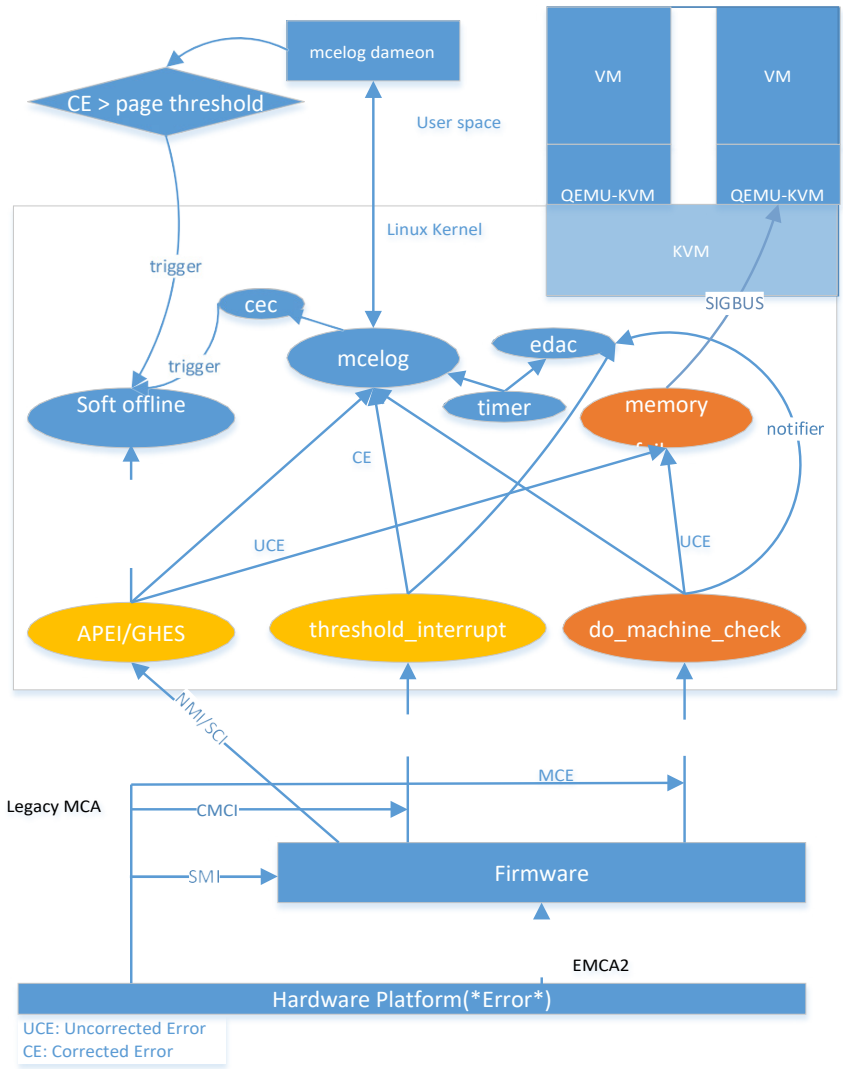
	Actions	Details	Deployment Status
#1	Replace marginal DIMMs (UCE and CE)	Tencent studied the history statistic memory failure data of deployed servers and adopted memory stress tool to screen those weak DIMMs	Deployed
#2	New BKC upgrade (UCE and CE)	Updated the Intel® Xeon® Processor Scalable Family BKC to 2018_WW14 (PLR6) or above.	Deployed
#3	MCA recovery enabling (UCE)	Enabled on all newly installed Purley servers (see Section 3.3 for details).	Deployed
#4	Patrol Scrub UCE downgrade (UCE)	Enabled on all newly installed Purley servers, Purpose is to improve cases where multiple MCA faults can occur due to conflicts between SRAO and SRAR errors, making possibly unrecoverable errors recoverable.	Deployed
#5	Improve VM Migration Success Rate	Deployed Linux* OS patches as workaround to fix several types of fatal errors (see Section 3.3 and further for details).	Partly deployed
#6	Enhanced DDR stress test (UCE and CE)	Used the Intel MCA error injection tool and third party vendor tool.	Deployed

3.3. MCA Recovery Detail

MCA recovery is an Intel® Xeon® Processor Scalable Family advanced RAS feature deployed in this case. MCA Recovery is effective and allows the system to stay operational in the presence on uncorrected data errors. However, error recovery is not possible under all conditions, such as those cases of #5 in Table 3-1. A good practice is to find the error address and try as possible to kill the application which is accessing that error address.

More details will be explained in a dedicated session in the future. Figure 3-2 is the block Diagram of MCA recovery implementation in the customer side.

Figure 3-2. Block Diagram of MCA Implementation



As shown in Figure 3-2, Firmware First Model (also known as eMCA Gen2) where CMCI and MCE are morphed into CSMI and MSMI respectively, allows UEFI-FW to decide whether the error needs to be reported to the OS or not. The decision is based on whether OS action is required for the error, or such recovery action could be performed by the UEFI-FW itself.

When the OS kernel receives the MCE/UCE events, the `memory _ failure` function (`HWPoison` handler) analyzes the log to verify if recovery is feasible. It then takes actions to offline the affected memory page and logs the event in the `mcelog` and the possible results of the actions appear to be ignoring, recovery, delay, and failure.

When an SRAR event is received, the `HWPoison` function starts the recovery action by isolating the affected page and flagging it with a "poisoned" tag to disallow any reuse of the page. It then triggers a "SIGBUS" event to notify the application for further recovery action. The application has a choice to either reload the data and resume normal execution, or terminate the application to avoid crashing the entire system. In the case of an SRAO event, the OS records the event to the event ring and delays taking action for recovery. When the OS receives the Correctable Error (CE) via CMCI, it will trigger the `mcelog` daemon to record the event.

If the `HWPoison` function finds that the affected address maps to a memory page associated with `qemu-kvm`, the SIGBUS will be routed to the `qemu-kvm` and then pseudo MCE/UCE is delivered to the VM, where the it will handle the MCE/UCE in the same way as the Host.

To avoid the possibility of future uncorrectable faults in the same page, we can copy the data to a new page and mark the original page as offline (or retired) when correctable event threshold reaches. This is the mechanism used by Memory Predictive Failure Analysis (PFA).

Firmware requirement

To support MCA recovery, Data Poisoning (also known as Corrupt Data containment) needs to be enabled by the FW by writing to the `MSR 0x178 – (MSR _ MCG _ CONTAIN)` register, bit 0. This can be done from the BIOS configuration utility. According to the architecture, this bit is meant to be set at boot time, considering that setting it during runtime in an OS like Linux introduces a risk that an uncorrected UCE might happen while the POISON bits are being set. If the

customer wants to take this risk and set it on the OS runtime, they need to set the Linux hot patch with minimal memory transaction.

Table 3-2. Poison Enable Bit

(178h) MSR_MCG_CONTAIN				
Machine Check Containment Mode Register. This register is used to configure Containment Mode capabilities of the machine check architecture of the processor. Before attempting to access this register, the SW must test the Software Error Recovery Support [24] in the IA32_MCG_CAP register (MSR 179h). If the Software Error Recovery Support bit is not set to 1, a #GP exception may be raised on access to this register.	Core	0h	RW	**POISON_ENABLE** — Enable Poison Mode. When set to 1, enables Poison. **Mode** - Erroneous data coming from memory will be Poisoned. Errors may be reported in several places. When set to 0 (default), indicates Legacy Mode - No poisoning available.

Notice: 1. It's recommended to use the latest BKC for MCA recovery practice, UEFI FW contains several fixes related to MCA handling issue in SMM.
2. To minimize the MCA interrupt impact for the processor, it's recommended to enable LMCE in BIOS setup (Note: Local MCE only applies to SRAR events).

3.3.1. Linux requirement

Linux environment: CentOS 7x and some RAS related patches are back-ported and developed, and some patches are out of the upstream Linux kernel tree, configuration used to open the following items for validation and verification.

Configurations in Linux kernel

```
CONFIG _ X86 _ MCE=y
CONFIG _ ACPI _ APEI=y
CONFIG _ ACPI _ APEI _ GHES=y
CONFIG _ ACPI _ APEI _ MEMORY _ FAILURE=y
COFIG _ ARCH _ SUPPORTS _ MEMORY _ FAILURE=y
CONFIG _ MEMORY _ FAILURE=y
CONFIG _ X86 _ MCE _ INTEL=m
CONFIG _ ACPI _ APEI _ EINJ=m CONFIG _ HWPOISON _ INJECT=m
```

With MCA recovery validation and online failure analysis after deployment, the customer still encountered several failures in a specific scenario where recovery was not successful . Intel developed and proposed over 10 kernel patches to the customer. With these patches, MCA recovery success rate improved significantly.

Table 3-3. MCA Recovery Kernel Support List

	Kernel version	RHEL			SUSE*		Cent OS			Ubuntu*	
	V3.14	7.2	7.3	7.4	11.04	12	6.5	7.2	7.3	14.04	16.04
MCA2.0 Recovery-Execution path		x	x	x	x	x		x	x	x	x
MCA2.0 Recovery-Non-Execution path	V3.14	x	x	x	x	x		x	x	x	x
Local Machine Check (LMCE) based Recovery	V4.2		x	x					x		x
Address Range/Partial Mirroring	V4.6			x							x

3.3.2. Validation Consideration

A test utility was developed to verify Intel's MCA recovery feature on CSP's servers. The test utility uses Linux EINJ error injection interface* to inject memory errors. The test includes:

1. CE/UCE error injection test in serial.
2. CE/UCE error injection in parallel.
3. SRAO error injection stress.
4. SRAR error injection stress.

After applying several FW and kernel patches, the servers were able to keep running even after 10K + UCE errors combined with CE storm. Refer to Section 5 for details on how a MCA error injection is processed.

3.4. Server crash rate reduction from Memory UCE Failure with the previous solutions

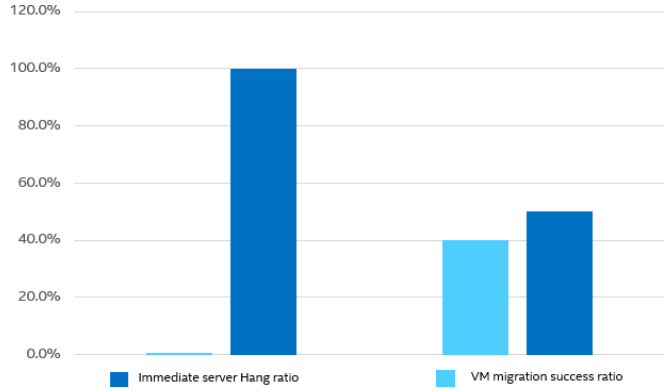
Using the group of those 50% crash servers due to Memory UCE as a base, the overall effectiveness of these actions is shown in Figure 3-3: the server immediate crash rate was significantly reduced from 100% to 50% and the VM migration success ratio was increased from 0% to 40%.

This brings up two observations: first, the big improvement in server immediate crash rate offers more possibility for VM migration and proves the effectiveness of the engineering action (especially MCA recovery) described so far; second, there are still server crashes (even without immediate crash after UCE accrued) before a successful VM migration completes, which is caused by some fatal errors that MCA recovery was not effective against.

Addressing these fatal errors is critical to ensure an eventual high VM migration success ratio, which is a fundamental metric for CSPs customer.

Using the entire Purley server cluster as a base, the improvement rate will becomes: : server immediate crash rate is reduced from 50% to 25% and VM migration successful ratio is increased from 0% to 20%. OS patches are used to address some types of fatal errors and it's expected further ~5% of VM migration successful ratio could be achieved. In total, the server crash of the entire Purley Server cluster could be reduced by roughly ~25%, in other words, almost half of the original crash could be addressed through the holistic engineering solution described here.

Figure 3-3. Key Improvement of Server Crash from Memory UCE



3.5. Enhancement for MCA Recovery

After the deployment of previous series of solutions, it was observed that the VMs' immediate crash rate was significantly reduced.

However, some servers that did not crash immediately were experiencing a system crash still after some certain period of time (for example, within 2 hours). The error log files were analyzed and the failures could be classified into several types, as described next.

3.5.1. Kernel Space Access with SRAR (30%)

If the UCE error was triggered while executing OS kernel code, then the UCE error will be fatal. As an example, when applications write data to disk by OS file system or networking application, such as VM live migration transferring data from one machine to another machine via OS TCP/IP stack, the user space application invokes the system call and goes to kernel to copy data from user space.

The `copy_user_generic (alternative_call : copy_user_enhanced_fast_string / copy_user_enhanced_fast_string/ copy_user_generic_unrolled)` action is executed by the kernel, but at the same time, if the copied data at the user space is found to be corrupted, MCE/SRAR will be triggered to notify the OS.

The OS kernel MCE handler considers this case as fatal (non-recoverable) since it happened during kernel execution and it will cause kernel panic. But if the application that initiated the copy and owned corrupt data can be easily identified by the kernel, it is possible to isolate the corrupt data by marking the affected page with the 'poison' tag and terminating the initiator/impacted applications to stop the corrupt data spreading.

Implementation against latest kernel: First, `ex_has_uaccess_handler` and other handlers are added to help detect the process that the kernel accesses from the user space, like copy data. Second, when MCE/SRAR are triggered and it is in the kernel execution context (but from a newly added handler), it can detect if it is a kernel copy of data from the user space, so it will be marked as `IN_KERNEL_RECOV` error, not the previous `IN_KERNEL`, which means it can be recoverable though it was encountered in kernel context.

If it is memory MCE/SRAR marked as `IN_KERNEL_RECOV`, it will perform recovery by isolating the corrupt memory page and terminating the impacted applications. Customer kernel will `invoke do_memory_failure` to perform a recovery in the MCE handler, `do_machine_check` in this situation.

Approximately six related patches are backported to Cent OS 7.x based kernel:

Speculative processor accesses may refer to any memory that has a valid page table entry. While a speculative access won't generate a machine check, it will log the error in a machine check bank. That could cause escalation of a subsequent error because the overflow bit will be then set in the machine check bank status register.

Strong Uncacheable (UC) -System memory locations are not cached. All reads and writes appear on the system bus and are executed in program order without reordering. No speculative memory accesses, pagetable walks, or prefetches of speculated branch targets are made. The customer must mark the linear address as 'UC' to make sure it does not log more errors because of speculative access to the page.

Approximately four related patches are backported to Cent OS 7.x based kernel:

3.5.2. Fatal Error after SRAO Overflow (4%)

Kernel 5.3 and earlier versions do not perform SRAO machine checks, if they are logged with the overflow bit set to 1 in the machine check bank status register. This is overly conservative. The customer found that some cases of SRAO+OVER turned into fatal errors if they were not handled by the kernel.

There are two cases where we could end up with an SRAO+OVER log based on the SDM volume 3 overwrite rules in "Table 15-8. Overwrite Rules for UC, CE, and UCR Errors":

1. A corrected error is logged, then the SRAO error overwrites. The second error overwrites the first because uncorrected errors have a higher severity than corrected errors.
2. The SRAO error was logged first, followed by a corrected error. In this case the first error is retained in the bank.

In either case, the machine check bank will contain the address of the SRAO error. It can process that even if the overflow bit was set.

One OS patch was backported.

3.5.3. SRAR without Valid Address (17%)

A DDR UCE with signature of SRAR, but whit out a valid address of error memory segment, cannot be fixed by the SW (kernel patch) to offline the error page due to lack of the address. This is an known issue in Purley CPUs, and will be fixed in Whitley silicon and beyond. For this issue, no work around is available.

3.5.4. Spare Copy Fatal UCE (4%)

If DDR UCE results during a failure DIMM device data replacement/copy, the OS cannot recover this kind of error. Workaround is to disable the spare copy function (SDDC+1, ADDDC or other similar feature using spare copy). This stays a problem if CSP customers want to enable the SDDC+1 function for a robust CE error handling.

3.5.5. IERR+PCC with Memory UCE (55%)

DDR UCE causes CPU IERR and crashes the sever, with no converged root cause and it's case by case. This type of errors accounts for almost ¼ of total fatal errors, but analysis takes too much time and thus, is pushed for future work.

3.5.6. Patrol Scrub Downgrade Handling (without statistic of %)

Generally speaking, after the patrol scrub detects memory uncorrected data errors, it will report the OS by MCE interrupt 18.

Intel® Xeon® Processor-based platforms have an `UCE _ TO _ CE _ DOWNGRAGE` mode where the users can request the memory controller to report UCE found by the patrol scrubber as a corrected type. It is also called 'downgrading patrol scrub UCE/SRAO to CE'. Those errors are signaled by using CMCI, a process less disruptive than a machine check and thus helps avoid double MCE interrupts to crash the system.

In addition, on the platform without supporting corrupt data containment and MCA recovery, the patrol scrub UCE will lead to kernel panic directly. If it is reported as a CE, the machine still can operate because the correctable error can be identified as SRAO by checking the MCE back registers, isolating the bad page for recovery.

Kernel Implementation: the Intel® Xeon® Processor Scalable Family has the `UCE _ TO _ CE _ DOWNGRAGE` mode. After the mode is detected and set on, it will enable a quirk handler to adjust the severity level of MCE/CMCI errors.

In the CMCI process handler, we will check the MCE bank registers and error code to tell if the CMCI is an uncorrected patrol scrub error from one of the memory controller banks. If so, it will adjust the severity level of the error to `MCE _ AO _ SEVERITY`, record the error in the MCE event pool and then wait for MCE work queue schedule to process the events in the pool, similarly like the UCE/SRAO process.

One OS patch was backported here.

3.6. Future Work

There are more options to explore, to further reduce server crash rate from DDR UCE:.

1. Study relationships between DDR CE and UCE, and figure out proactive mitigation to those CE errors that can directly cause UCE errors.
2. Figure out more solutions to address those open fatal errors. Especially some UCE with IERR, which need a deep dive to analyze if there is any commonality.
3. Exploring other advanced RAS features (like Memory Mirroring) to check its feasibility of deployment.
4. Applying other features, like the PPR feature, to replace a failure row in a DIMM for fatal memory error.

4. Conclusions

This technical paper covers a brief introduction of Intel available technologies to mitigate the impact of server crash from DDR UCE errors in cloud IDC. Tencent Cloud's case is deep dived, and some workaround solutions are presented to address those non-recoverable fatal errors are discussed in detail.

The data indicates that MCA is an effective way to recover DDR UCE together with other engineering solutions. However, addressing those non-recoverable UCE fatal errors is still even more challenged. A holistic engineering work ranging from weak DIMM sorting to advanced RAS features validation and deployment (from fatal errors handling to final VM successful migration) is needed to achieve a significant improvement on server crash from DDR UCE.

5. Appendix

5.1. MCA Validation Process

1. RAS Integration and Validation guide – See Chapter 23, *MCA Recovery* for validation detail:

   https://cdrdv2.intel.com/v1/dl/getContent/563361

   - RAS Technology Integration and Validation Guide - Purley Platform.
   - This document provides guidelines on how to integrate and validate RAS features within the servers requiring BIOS, OS, and/or application support. More specifically it describes 'how to configure', 'how to verify the status', and 'how to validate' various RAS features. Scope of this document does not cover component level or platform level specification which is delivered for example through 'External Design Specification (EDS)'.
2. Automatic MCA Recovery Injection Tool and Verification Process.

   Automatic MCA Recovery Injection Tool and Verification Process are available. Details are available upon demand.
3. Fish tool to inject memory errors to customer Application.
4. Open source RAS tool for error injection in Linux.

   https://git.kernel.org/pub/scm/linux/kernel/git/aegl/ras-tools.git/

5.2. Kernel Patches

Contact Intel for details regarding OS patches used for this work.

6. Acronyms and References

6.1. Acronym List

Term	Definition
ADDDC	Adaptive Double Device Data Correction
ADC	Adaptive Data Correction
ACPI	Advanced Configuration and Power Interface
ARM	Address Range Mirroring
BMC	IPMI compliant Baseboard Management Controller
BIOS	Basic Input/Output System
BKC	Best Known Configuration
CE	Correctable Error
CSMI	Corrected Machine Check SMI
CMCI	Corrected Machine Check Interrupt

Term	Definition
CPGC	Converged Pattern Generation and Checking
CSP	Cloud Service Provider
DDDC	Double Device Data Correction
DDR4	Double Data Rate (DRAM) version 4
DIMM	Dual Inline Memory Module
eMCA	Enhanced MCA(Machine Check Architecture)
ECC	Error Correction Code
FRU	Field Replaceable Units
IERR	Internal Error
MCA	Machine Check Architecture
MSMI	Machine Check SMI
MRC	Memory Reference Code
OS	Operating System
PPR	Post Package Repair
PCC	Processor Context Corruption
RAS	Reliability, Availability, Serviceability
SRAR	Software Recoverable Action Required
SRAO	Software Recoverable Action Optional
UCNA	Uncorrectable error no action required
SDDC	Single Device Data Correction
SDDC+1	Single Device Data Correction plus one
SDM	Software Developers Manual
SLA	Service Level Agreement
SMI	System Management Interrupt
TCP/IP	Transmission Control Protocol/Internet Protocol
UEFI	Unified Extensible Firmware Interface
UCE	Uncorrectable Errors
VM	Virtual Machine

6.2. References

1. Intel® Xeon® Scalable Processor Family/Cascade Lake Server External Design Specifications (EDS) Vol1 (#558771), Vol2A (#610949), Vol2B (#610950).
2. Purley RAS Integration and Validation Guide (#563361).
3. RAS 2018 Workshop (#604663).
4. Autonomous Crash Dump (#610610).
5. Intel® 64 and IA-32 Architectures Developer's Manual

   https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32- architectures-software-developer-vol-3b-part-2-manual.html

   (MCA architecture detail, ref: Chapter 15 machine-check architecture)
6. Purley Platform Design Guide [PDG] (#546835).
7. Advanced Configuration and Power Interface Specification

   https://uefi.org/specifications
8. Address Range Partial Memory Mirroring

   https://software.intel.com/en-us/articles/address-range-partial-memory- mirroring
9. Improving Memory Reliability via Intel® Xeon® Processor RAS Features DRAM Failures, Rev 1.0 (#615791).
10. Intel® Xeon® Processor MCA Recovery-Supplemental Documentation, Rev 1.0.

Intel Confidential

Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.