AMD Family 10h revision B2 processors suffer from an issue in the

processor TLB known as erratum 298. Erratum 298 is documented in

a forthcoming update to the Revision Guide for AMD Family 10h

Processors (PID 41322). The workaround in the Revision Guide

document is intended to be applied by BIOS. The BIOS workaround

has performance implications which can be avoided by having the

OS directly workaround the issue. A Linux 64-bit patch was developed

for 2.6.23.8 by AMD's OSRC team and will be posted to this list by

Joerg Roedel. The patch is for demonstration purposes and is NOT

being recommended to be applied upstream.

Erratum 298 will be described as follows: "The processor operation

to change the accessed or dirty bits of a page translation table

entry in the L2 from 0b to 1b may not be atomic. A small window of

time exists where other cached operations may cause the stale page

translation table entry to be installed in the L3 before the modified

copy is returned to the L2. In addition, if a probe for this cache

line occurs during this window of time, the processor may not set

the accessed or dirty bit and may corrupt data for an unrelated

cached operation. The system may experience a machine check event

reporting an L3 protocol error has occurred. In this case, the MC4

status register (MSR 0000\_0410) will be equal to B2000000\_000B0C0F

or BA000000\_000B0C0F. The MC4 address register (MSR 0000\_0412) will

be equal to 26h."

The L2 Eviction Linux kernel performance patch re-enables the

registers set for the BIOS workaround described in the Revision

Guide document. It then prevents the processor from performing the

operation that can trigger erratum 298. The patch works by emulating

the Accessed and Dirty bits.

The basis for the kernel patch solution depends on the root cause of

the L2 eviction problem. The only exposure for the problem is when

the TLB needs to set an A or D bit in a page table entry. If the TLB

never needs to set an A or D bit, the bug cannot occur. By emulating

the A and D bits with the help of the Present and Writable bits, the

patch will ensure the real A and D bits are always preset. It works

by forcing a page fault when the first access is made to a page with

the emulated A bit not set, and when the first write access is made

to a writable page with the emulated D bit not set. Emulated A and D

bits are stored in bits generally available to the OS in the page

table entry.

The fix here is to check for the present bit and simply not update

> the pte if the page is not present anymore. If the page is not present

> then the fault handler should run next which will take care of the problem

> by bringing the page back and then mark the page dirty or move it onto the

> active list.