Hypervisor-Assisted Dump

November 2007

The goal of hypervisor-assisted dump is to enable the dump of a crashed system, and to do so from a fully-reset system, and to minimize the total elapsed time until the system is back in production use.

As compared to kdump or other strategies, hypervisor-assisted dump offers several strong, practical advantages:

-- Unlike kdump, the system has been reset, and loaded with a fresh copy of the kernel. In particular, PCI and I/O devices have been reinitialized and are in a clean, consistent state.
-- As the dump is performed, the dumped memory becomes immediately available to the system for normal use.
-- After the dump is completed, no further reboots are required; the system will be fully usable, and running in its normal, production mode on its normal kernel.

The above can only be accomplished by coordination with, and assistance from the hypervisor. The procedure is as follows:

-- When a system crashes, the hypervisor will save the low 256MB of RAM to a previously registered save region. It will also save system state, system registers, and hardware PTE's.

-- After the low 256MB area has been saved, the hypervisor will reset PCI and other hardware state. It will *not* clear RAM. It will then launch the bootloader, as normal.

-- The freshly booted kernel will notice that there is a new node (ibm,dump-kernel) in the device tree, indicating that there is crash data available from a previous boot. It will boot into only 256MB of RAM, reserving the rest of system memory.

-- Userspace tools will parse /sys/kernel/release_region and read /proc/vmcore to obtain the contents of memory, which holds the previous crashed kernel. The userspace tools may copy this info to disk, or network, nas, san, iscsi, etc. as desired.

   For Example: the values in /sys/kernel/release-region would look something like this (address-range pairs).
   CPU:0x177fee000-0x10000: HPTE:0x177ffe020-0x1000: /
   DUMP:0x177fff020-0x10000000, 0x10000000-0x16F1D370A

-- As the userspace tools complete saving a portion of dump, they echo an offset and size to

/sys/kernel/release_region to release the reserved
memory back to general use.

An example of this is:
"echo 0x40000000 0x10000000 > /sys/kernel/release_region"
which will release 256MB at the 1GB boundary.

Please note that the hypervisor-assisted dump feature
is only available on Power6-based systems with recent
firmware versions.

Implementation details:

During boot, a check is made to see if firmware supports
this feature on this particular machine. If it does, then
we check to see if a active dump is waiting for us. If yes
then everything but 256 MB of RAM is reserved during early
boot. This area is released once we collect a dump from user
land scripts that are run. If there is dump data, then
the /sys/kernel/release_region file is created, and
the reserved memory is held.

If there is no waiting dump data, then only the highest
256MB of the ram is reserved as a scratch area. This area
is *not* released: this region will be kept permanently
reserved, so that it can act as a receptacle for a copy
of the low 256MB in the case a crash does occur. See,
however, "open issues" below, as to whether
such a reserved region is really needed.

Currently the dump will be copied from /proc/vmcore to a
a new file upon user intervention. The starting address
to be read and the range for each data point in provided
in /sys/kernel/release_region.

The tools to examine the dump will be same as the ones
used for kdump.

General notes:

Security: please note that there are potential security issues
with any sort of dump mechanism. In particular, plaintext
(unencrypted) data, and possibly passwords, may be present in
the dump data. Userspace tools must take adequate precautions to
preserve security.

Open issues/ToDo:
---
 o The various code paths that tell the hypervisor that a crash
   occurred, vs. it simply being a normal reboot, should be
   reviewed, and possibly clarified/fixed.

 o Instead of using /sys/kernel, should there be a /sys/dump
   instead? There is a dump_subsys being created by the s390 code,
   perhaps the pseries code should use a similar layout as well.

- Is reserving a 256MB region really required? The goal of reserving a 256MB scratch area is to make sure that no important crash data is clobbered when the hypervisor save low mem to the scratch area. But, if one could assure that nothing important is located in some 256MB area, then it would not need to be reserved. Something that can be improved in subsequent versions.

- Still working the kdump team to integrate this with kdump, some work remains but this would not affect the current patches.

- Still need to write a shell script, to copy the dump away. Currently I am parsing it manually.