Firmware-Assisted Dump, a kdump alternative to kernel dump capturing mechanism

Hari Bathini, Linux Technology Center, IBM

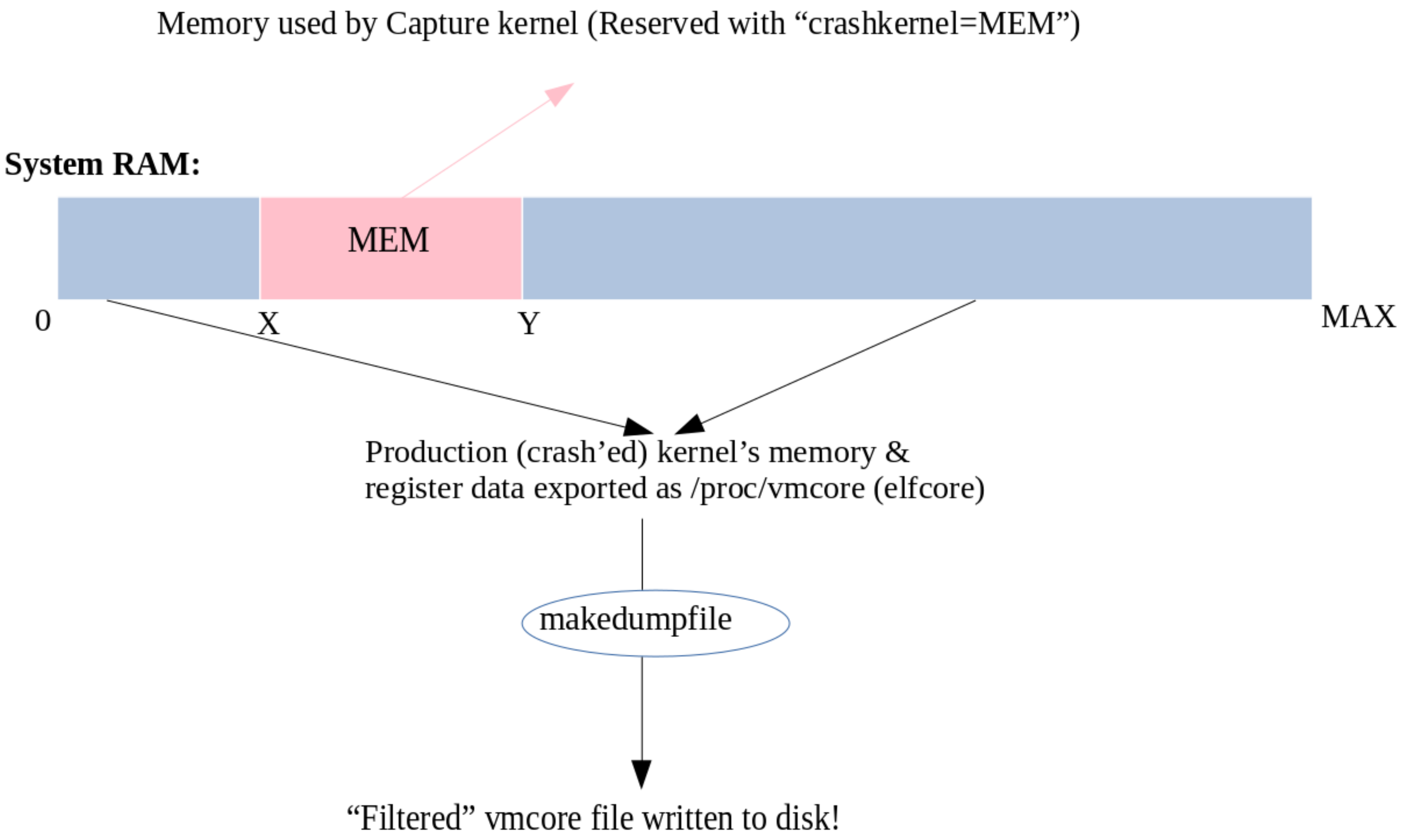
LINUX PLUMBERS CONFERENCE | Vienna, Austria
Sept. 18-20, 2024

Agenda

- Overview of kdump
- Advantages and inherent issues with kdump
- A brief introduction to fadump
- Advantages and concerns with fadump
- Concerns mitigated so far
- How fadump fares now
- One last concern
- What it takes to enable fadump support

Overview of kdump

- First Crash Dump solution accepted in mainline.
- Relies on kexec – a kernel to kernel bootloader.


Memory used by Capture kernel (Reserved with “crashkernel=MEM”)
System RAM:
MEM
0
X
Y
MAX
Production (crash’ed) kernel’s memory &
register data exported as /proc/vmcore (elfcore)
makedumpfile
“Filtered” vmcore file written to disk!

Advantages with kdump

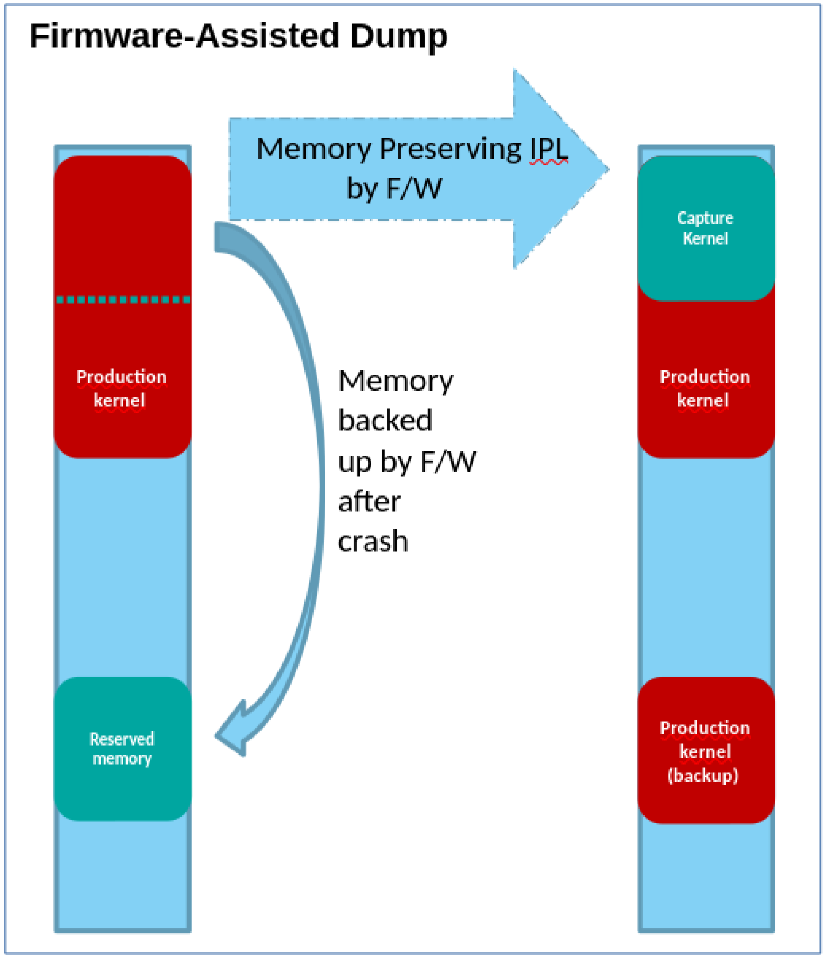
- Special initrd and cmdline to reduce capture kernel memory footprint.
- Flexibility to choose dump target device.
- Scope in userspace to filter the vmcore before offloading to disk.

Inherent issues with kdump

- Dependent on crashed kernel to kexec into kdump kernel.
- Devices are in inconsistent state.
- Prone to driver initialization failures in capture kernel.
- Buggy driver code can result in failure to offload vmcore to the dump target.
  - every new device needs to support soft reset.
  - driver needs to know how to do soft reset.
- Brief service lapse to refresh elfcorehdr after cpu/memory hot add/remove operations.

Firmware-assisted dump (fadump)

- Crash Dump solution accepted in mainline in kernel 3.4
- Kernel registers with f/w for fadump
  - on crash, a hook in kernel crash path informs f/w about the kernel crash.
  - f/w quiesces CPUs (except crashing CPU) and saves register state.
  - f/w backs up memory regions requested.
  - f/w flags off a memory preserved boot.
  - f/w notifies that the boot is after crash.
  - kernel preserves context and exports /proc/vmcore file.
  - fadump reuses kdump flow from here:
    - filtering vmcore
    - offloading to disk
    - analyzing the vmcore with gdb/crash/drgn


Firmware-Assisted Dump
Memory Preserving IPL by F/W
Production kernel
Reserved memory
Memory backed up by F/W after crash
Capture Kernel
Production kernel
Production kernel (backup)

Advantages with fadump

- Flexibility to choose dump target device (kdump).
- Scope in userspace to filter the vmcore before offloading to disk (kdump).
- Memory preserved by f/w.
- Boots like regular kernel (reset).
  - loaded with a fresh copy of the kernel.
  - PCI and I/O devices are fully reset.

Concerns with fadump

- Does not have special initrd for capture kernel boot and no existing provision to pass additional parameters
  - as capture kernel boots via the regular boot loader just like production kernel.
  - kexec loads the special initrd and cmdline for kdump.
- Capture kernel for fadump has relatively higher memory footprint.
- Brief service lapse to update elfcorehdr after memory hot add/remove operations.

Does not have special initrd for capture kernel

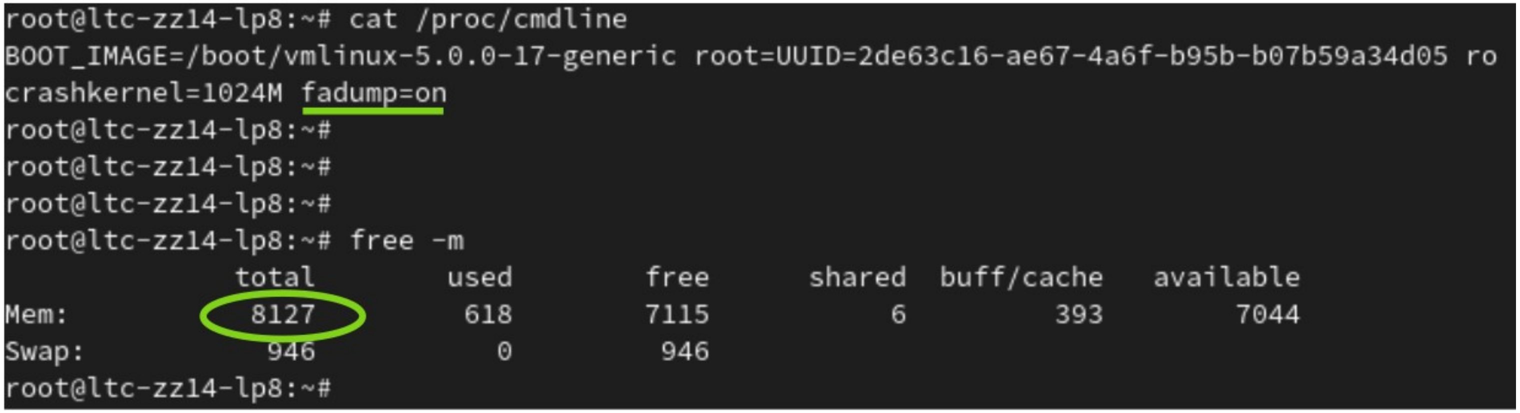
- Uses the same initrd used for production kernel boot
  - initrd built for production kernel is not ideal for fadump capture kernel.
- A special out-of-tree dracut module to pack initrd for capture kernel
  - fadump initrd is embedded into the production kernel initrd.
  - unpacked only while booting fadump capture kernel.
- Using special initrd scripts for fadump capture kernel ensures
  - no interference of fadump optimizations in production kernel boot.
  - no overhead in fadump capture kernel.

No existing provision to pass additional parameters

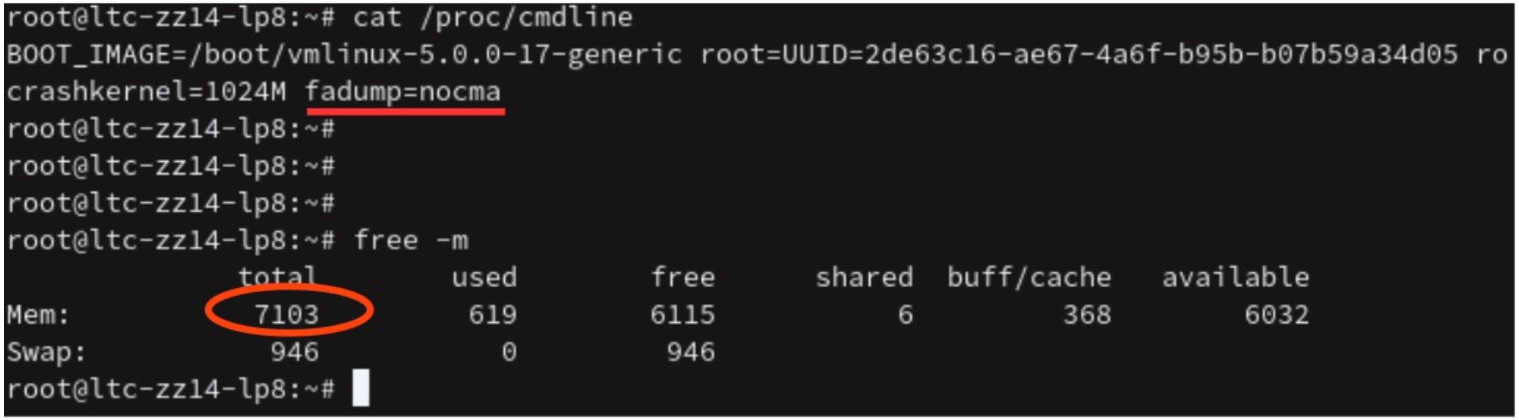
- As fadump relies on regular boot loader
  - passing additional parameters can be tricky, unlike kdump.
  - leverage firmware’s memory preserving boot feature.
  - locate a memory region and pass arguments via this region between the kernels.
  - https://lore.kernel.org/all/20240509115755.519982-1-hbathini@linux.ibm.com/
- This helps minimize the memory footprint of fadump capture kernel.
- Also, allows disabling unnecessary/troublesome features/components/drivers.

Relatively larger memory reservation requirement

- Use CMA for memory reservation
  - this makes the memory reserved for fadump available for userpages
  - So, except for some metadata, all memory reserved for fadump is now available via CMA.
  - assumes vmcore is filtered for only kernel pages (default).



```
root@ltc-zz14-lp8:~# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinux-5.0.0-17-generic root=UUID=2de63c16-ae67-4a6f-b95b-b07b59a34d05 ro
crashkernel=1024M fadump=on
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~# free -m
              total        used        free      shared  buff/cache   available
Mem:           8127         618        7115           6         393        7044
Swap:           946           0         946
root@ltc-zz14-lp8:~#
```



```
root@ltc-zz14-lp8:~# cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinux-5.0.0-17-generic root=UUID=2de63c16-ae67-4a6f-b95b-b07b59a34d05 ro
crashkernel=1024M fadump=nocma
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~#
root@ltc-zz14-lp8:~# free -m
              total        used        free      shared  buff/cache   available
Mem:           7103         619        6115           6         368        6032
Swap:           946           0         946
root@ltc-zz14-lp8:~#
```

https://lore.kernel.org/all/153475298147.22527.9680437074324546897.stgit@jupiter.in.ibm.com/

Service lapse after memory hot add/remove operations

- On Memory hot add/remove operations
  - elfcorehdr used to describe the crash’ed system (/proc/vmcore) needs update.
  - elfcorehdr is updated by re-registering.
- Instead, create the elfcorehdr in capture kernel boot
  - by snooping through the memblock list during early boot in capture kernel.
  - https://lore.kernel.org/all/20240422195932.1583833-1-sourabhjain@linux.ibm.com/
  - eliminates the need to reload service after memory hot add/remove operations.
  - with this change, fadump is **always ready** to capture a kernel dump.

How fadump fares now

Concern	Resolution
Does not have special initrd for capture kernel boot	- Special initrd for capture kernel built into production kernel initrd - This special initrd is activated only if a f/w variable indicates fadump is active
No existing provision to pass additional parameters	- A dedicated memory region for passing additonal parameters - Production kernel sets up this region - Capture kernel reads from this region during early boot and updates cmdline
Relatively larger memory reservation requirement	- Except for metadata, CMA is used for memory reservation - This makes the memory available for production kernel use - Effectively almost all memory is available for production kernel use
Service lapse after memory hot add/remove operations	- elfcorehdr generation delayed till capture kernel boots - Eliminates the need to re-register on memory hot add/remove operations - Snoops memblock list in capture kernel to generate elfcorehdr - fadump is always ready to serve a crash with this change

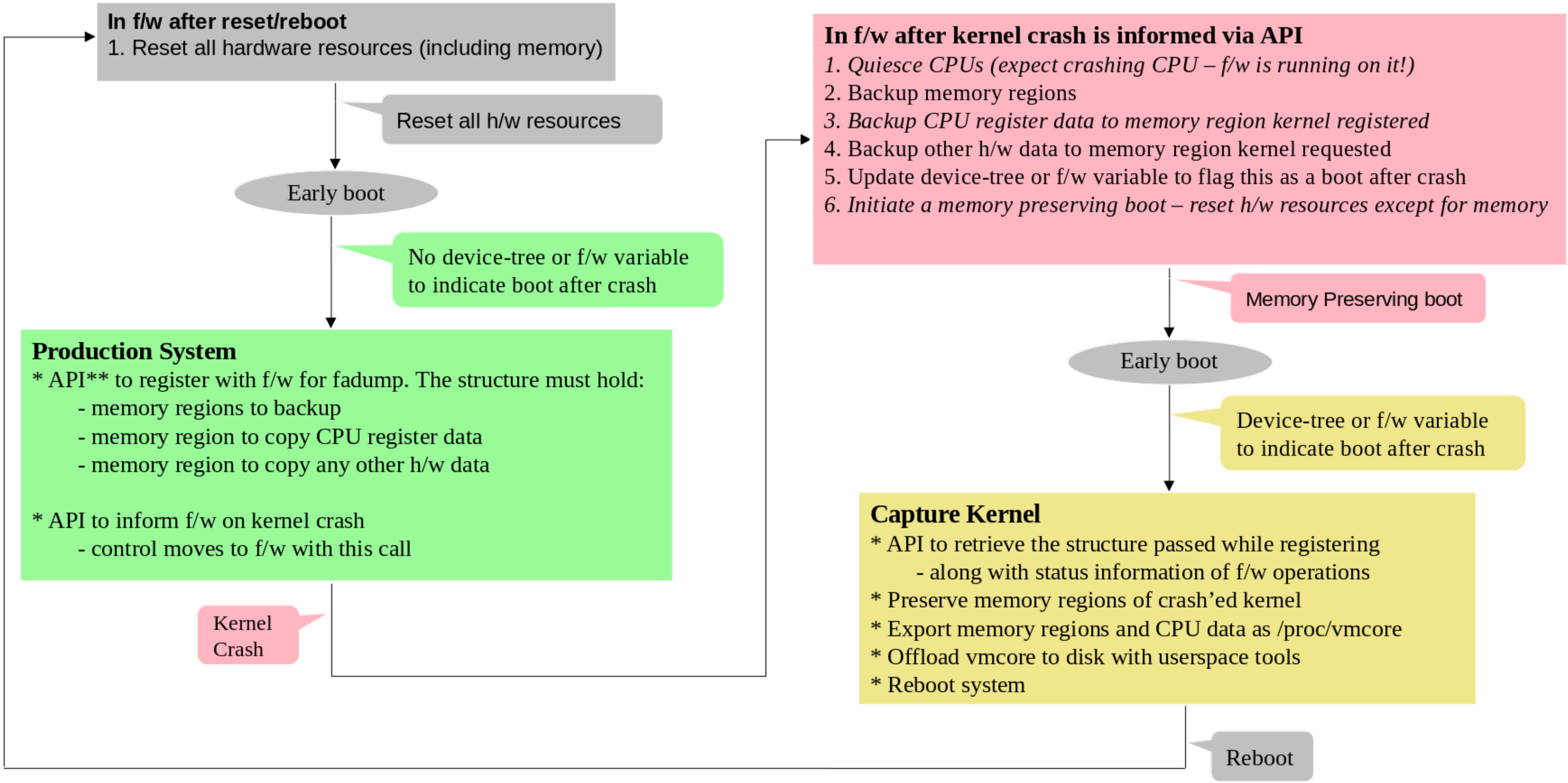
One last concern

- What is the right memory size to reserve for capture kernel?
  - both kdump and fadump face this challenge.
  - memory requirement for capture kernel is a moving target
  - it depends on
    - build options used
    - features enabled
    - devices attached
    - services used
  - approach..
    - reserve fixed memory for any system configuration.
    - reclaim memory in capture kernel on-demand.
    - the idea is to build capability in capture kernel to free up non-kernel memory during early boot.
    - assumes vmcore is filtered for kernel pages only (default).
  - the key reason to solve the memory reservation problem is to **simplify** fadump configuration in **deployments**.

Advantages with fadump

- Flexibility to choose dump target device (kdump).
- Scope in userspace to filter the vmcore before offloading to disk (kdump).
- Memory preserved by f/w.
- Boots like regular kernel (reset)
  - loaded with a fresh copy of the kernel.
  - PCI and I/O devices are fully reset.
- Special initrd for fadump
  - ensures no overhead of production kernel configurations.
- Passing additional parameters
  - helps reduce memory footprint and disable troublesome components.
- All system memory available for production kernel use
  - with CMA reservation for fadump.
- Always ready crash recovery support
  - with elfcorehdr prepared in capture kernel.
- … and fixed memory reservation …
  - by reclaiming memory in capture kernel

What it takes to support fadump



In f/w after reset/reboot
1. Reset all hardware resources (including memory)

Reset all h/w resources

Early boot

No device-tree or f/w variable to indicate boot after crash

Production System
* API** to register with f/w for fadump. The structure must hold:
- memory regions to backup
- memory region to copy CPU register data
- memory region to copy any other h/w data

* API to inform f/w on kernel crash
- control moves to f/w with this call

Kernel Crash

In f/w after kernel crash is informed via API
1. Quiesce CPUs (expect crashing CPU – f/w is running on it!)
2. Backup memory regions
3. Backup CPU register data to memory region kernel registered
4. Backup other h/w data to memory region kernel requested
5. Update device-tree or f/w variable to flag this as a boot after crash
6. Initiate a memory preserving boot – reset h/w resources except for memory

Memory Preserving boot

Early boot

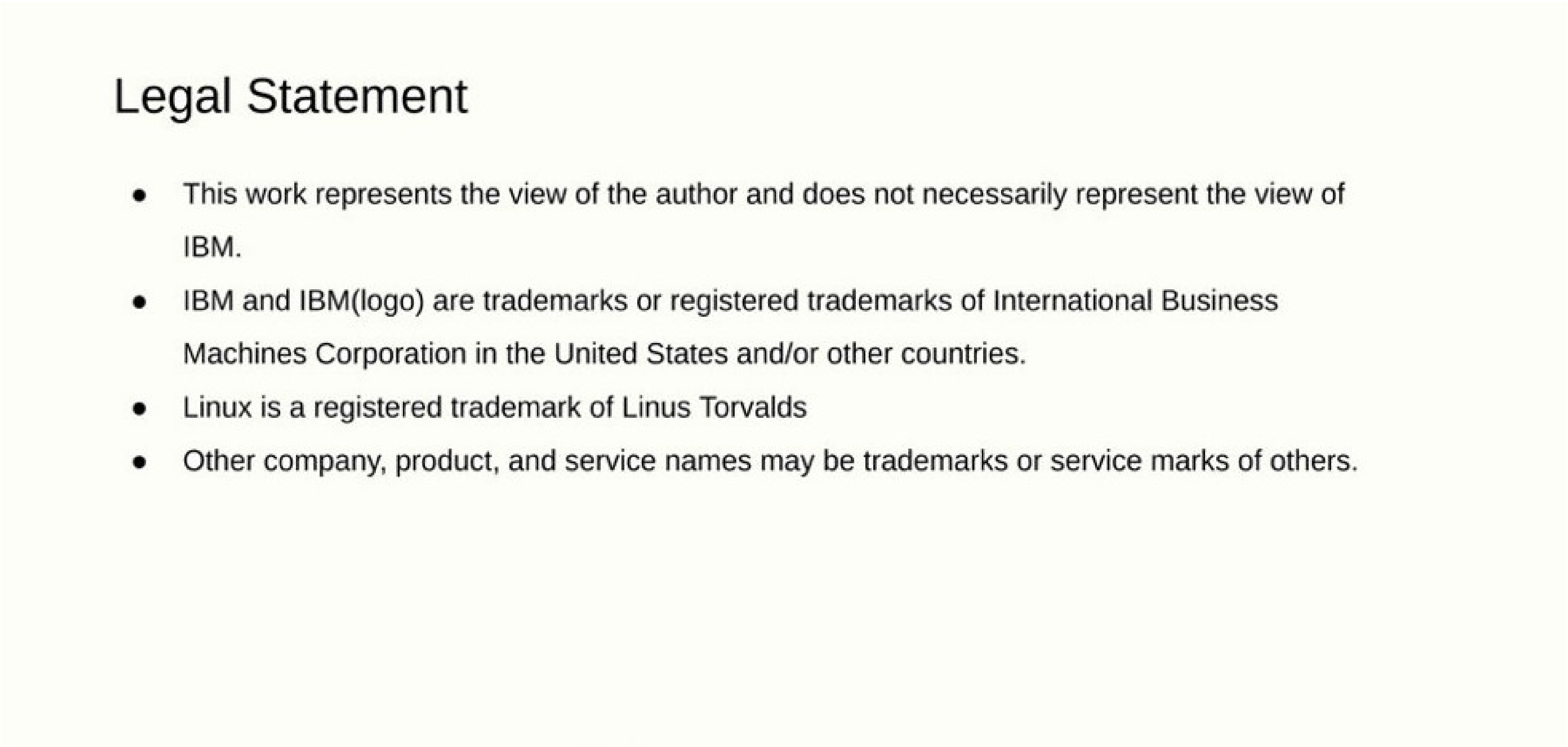
Device-tree or f/w variable to indicate boot after crash

Capture Kernel
* API to retrieve the structure passed while registering
- along with status information of f/w operations

* Preserve memory regions of crash'ed kernel
* Export memory regions and CPU data as /proc/vmcore
* Offload vmcore to disk with userspace tools
* Reboot system

Reboot

** API mentioned above is to refer to the calling interface from kernel to firmware.



Legal Statement

- This work represents the view of the author and does not necessarily represent the view of IBM.
- IBM and IBM(logo) are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries.
- Linux is a registered trademark of Linus Torvalds
- Other company, product, and service names may be trademarks or service marks of others.

Thanks!