HITACHI
Inspire the Next

A Flexible Device Name Assignment Method

LinuxCon Japan 2010/09/28

Systems Development Laboratory
Linux Technology Center

Nao Nishijima

Overview

HITACHI
Inspire the Next

My goal is to solve device name mismatch

This presentation will show the following.

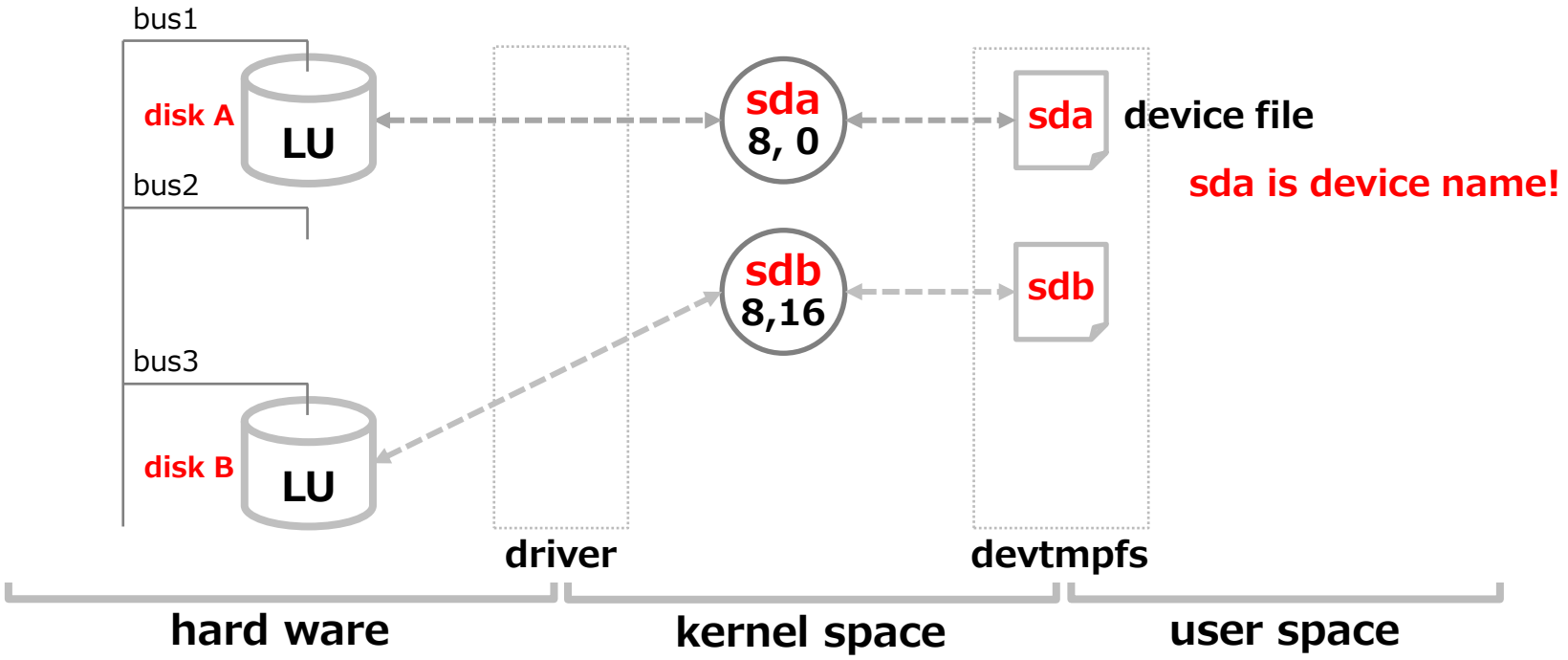
- What is device name
  The first part describes disks ,device names, device files and their relationship.
- Device name mismatch
  What causes a device name mismatch, and How to solve device name mismatch by udev.
- Issue of udev's solution
  What are some problems on persistent device naming by udev.
- A Flexible Device Name Assignment
  Proposal: Device renaming

What is device name

A device name identifies a disk

In Unix-like operating system, a device file(or device node) is an interface for a device that appears in /dev. Userspace applications can use these device files to access the device (e.g. disk).

Kernel identified device by a major number and a minor number. kernel assigned a name to disk(e.g sda). the name is used by device file name.



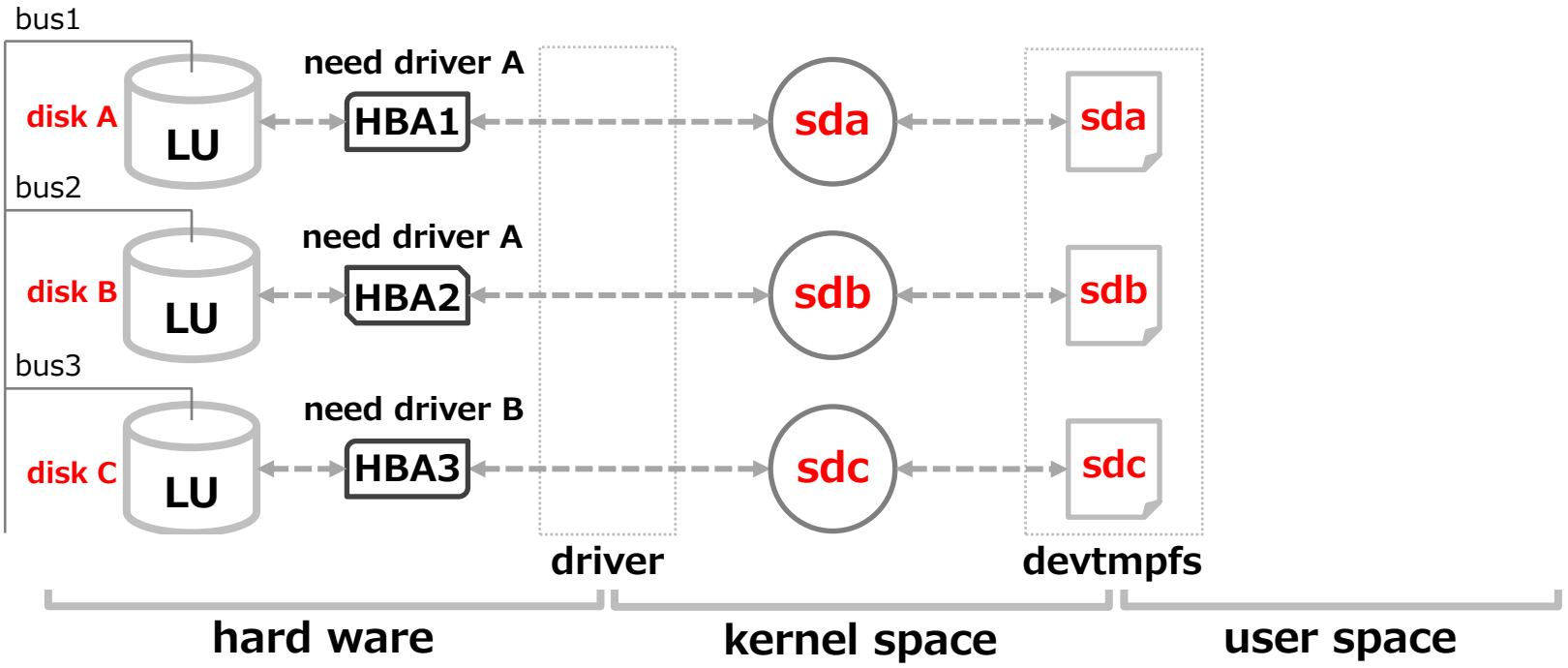
How to assign disk with device name

HITACHI
Inspire the Next

Order of device recognizing

Device names(e.g. sda, sdb...) are assigned by the order of driver loading, and device recognizing (usually from small bus number).

This case shows that driver A is loaded before driver B, thus disk A and B are recognized before disk C!



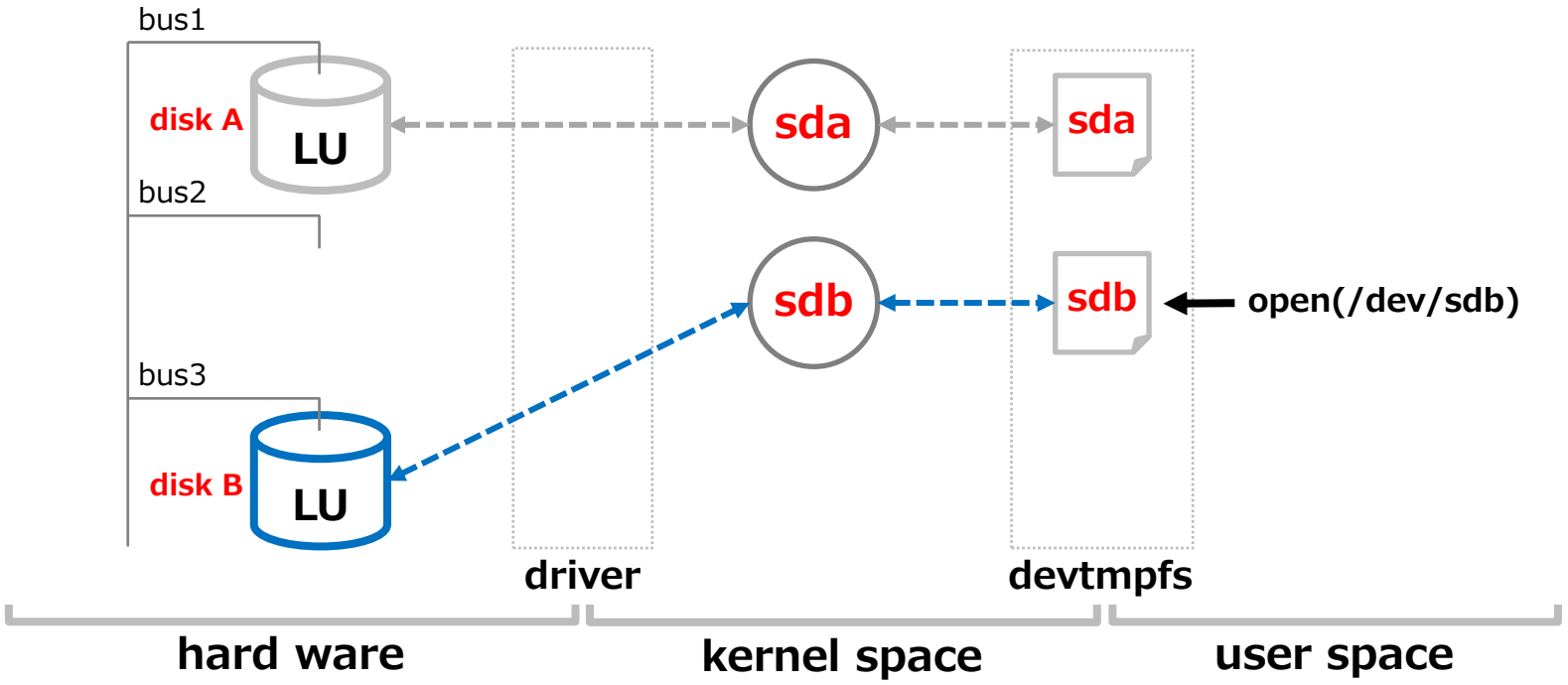
Device name mismatch from hot-plug

HITACHI
Inspire the Next

Case1: Changing the order of device list

There are three cases of device name mismatch. The first case is “changing order of device list”.

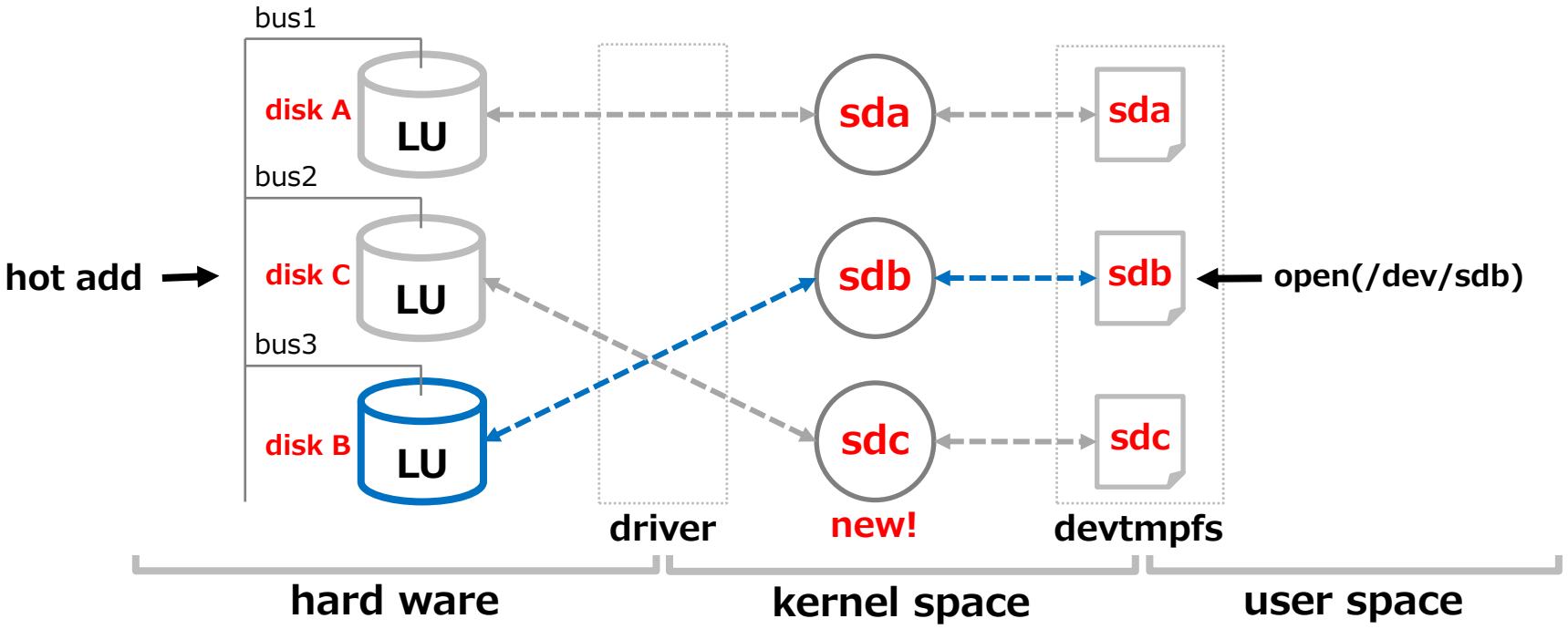
- This server connected 2 disks. bus1 access disk A, bus3 access disk B. An application open /dev/sdb for access disk B, to write and read.



Device Hot add

Case1: Changing order of device list

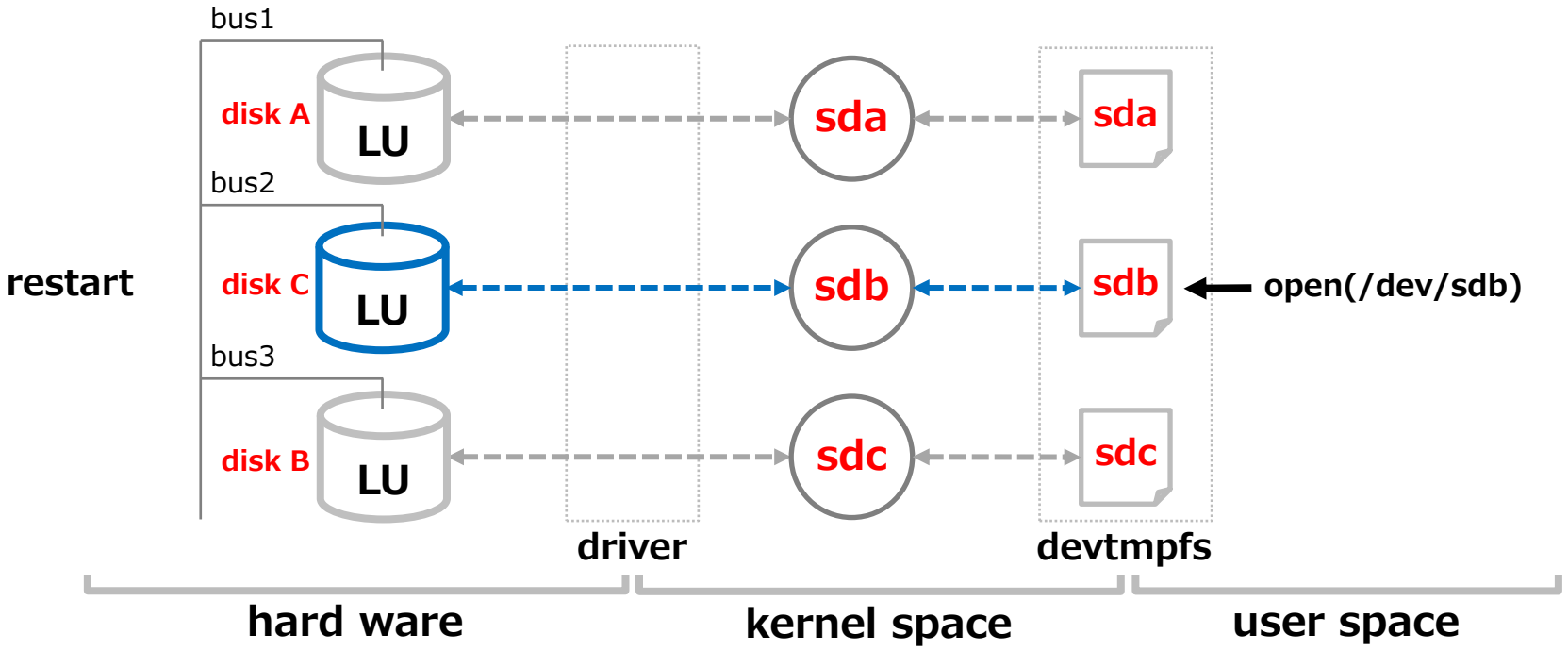
- Disk C was connected bus2 by hot-add (or cold-add).
- Kernel assigned sdc to disk C (hot-plug action)
- udev makes /dev/sdc (hot-plug action)
- And then system reboots...



HITACHI
Inspire the Next

Case1: Changing order of device list

- After boot up, the application opens /dev/sdb for reading data which it had written. But now, since /dev/sdb points disk C, it can't read the data.
- Device file has been changed because of the different order of devices.

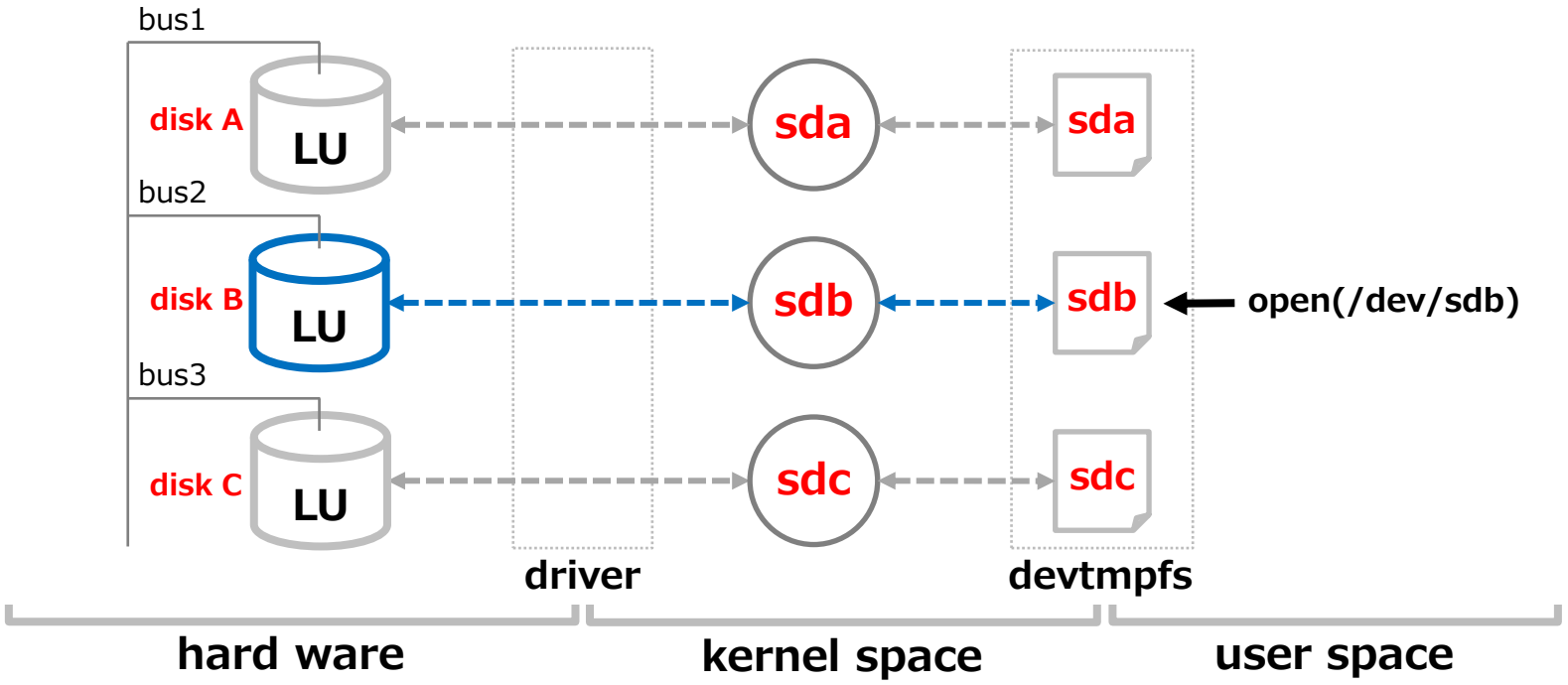


HITACHI
Inspire the Next

Case2: Device breakdown

Second case of device name mismatch is “Device breakdown”.

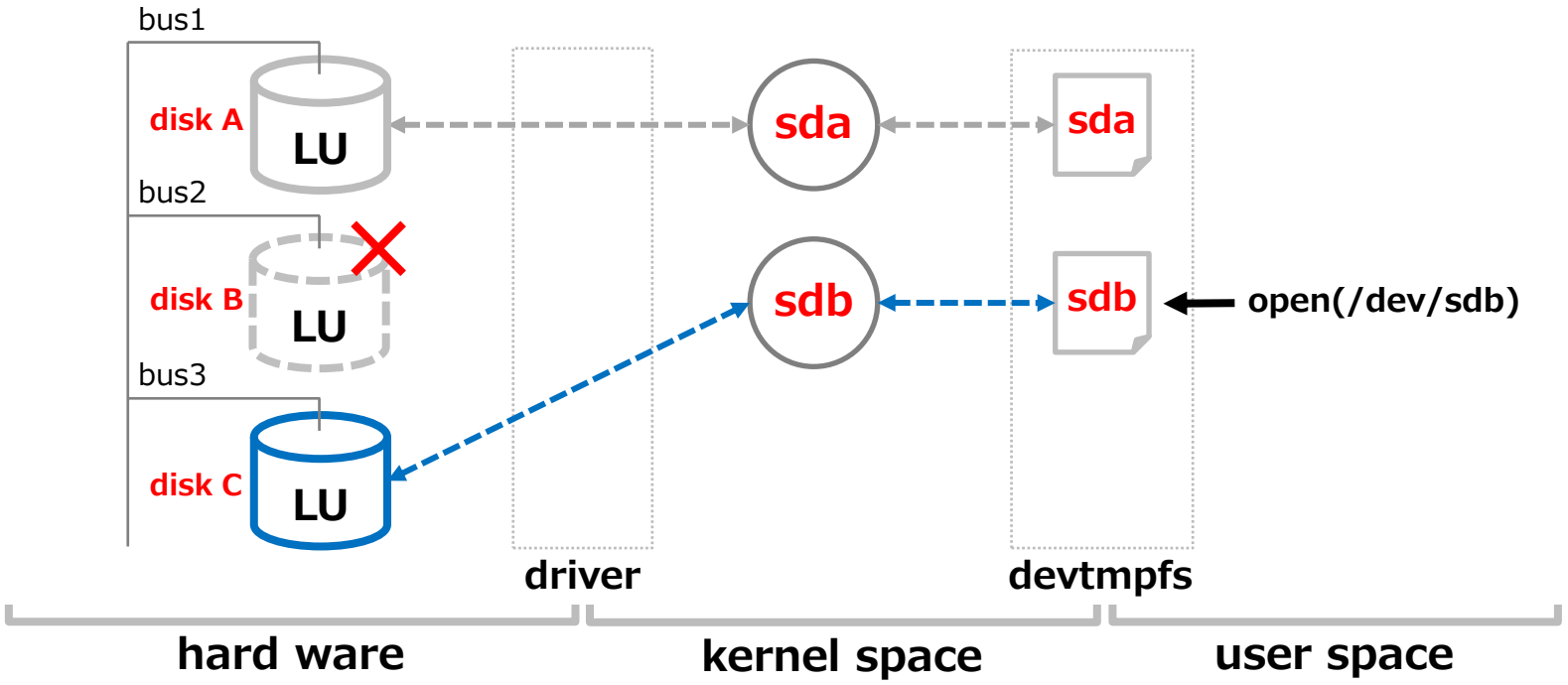
- Here, disk A, B, and C are connected to bus1, 2, 3 respectively.
- And an application opens disk B via /dev/sdb, for its work.



HITACHI
Inspire the Next

Case2: Device breakdown

- While the system rebooting, disk B has been broken down.
- So Linux assigns /dev/sdb to disk C
- The application opens /dev/sdb again, but it can't read its data, because it's disk C.

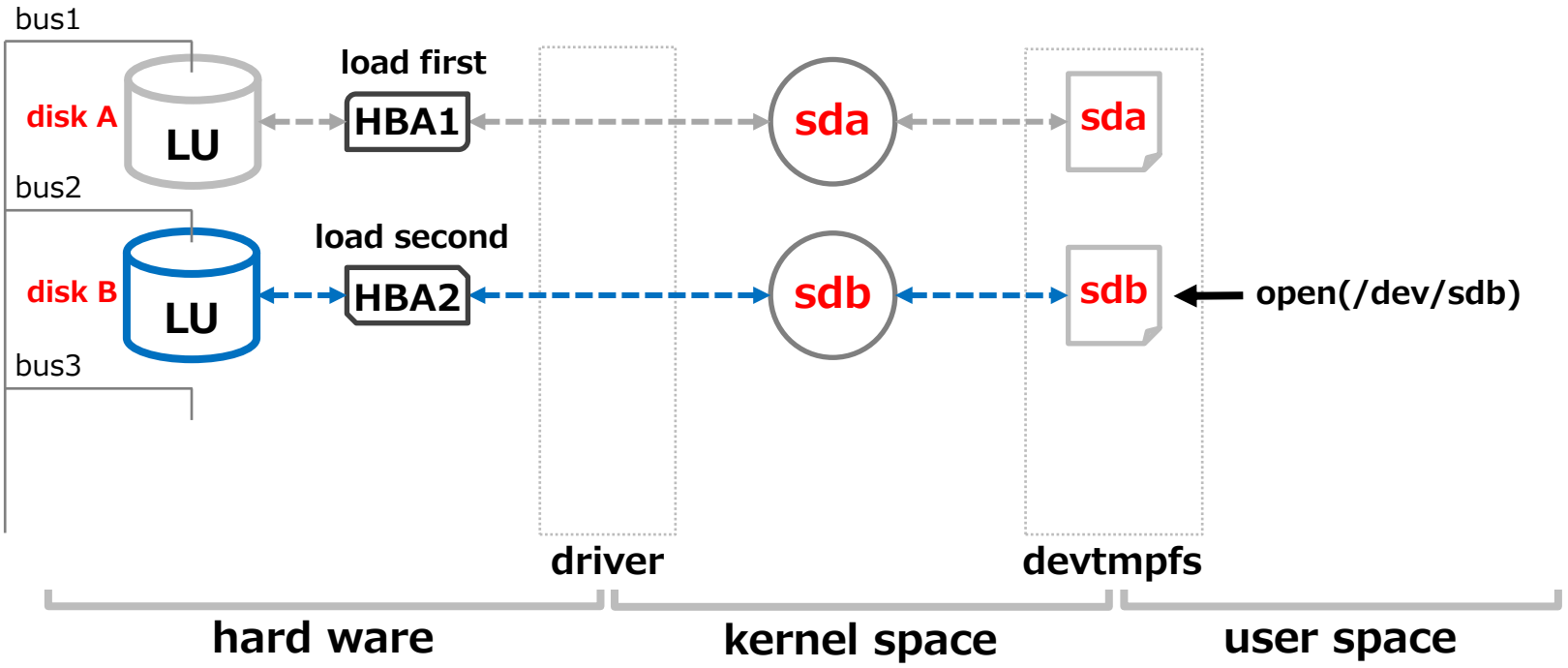


HITACHI
Inspire the Next

Case3: Changing the order of driver

Third case of device name mismatch is “Changing order of driver”.

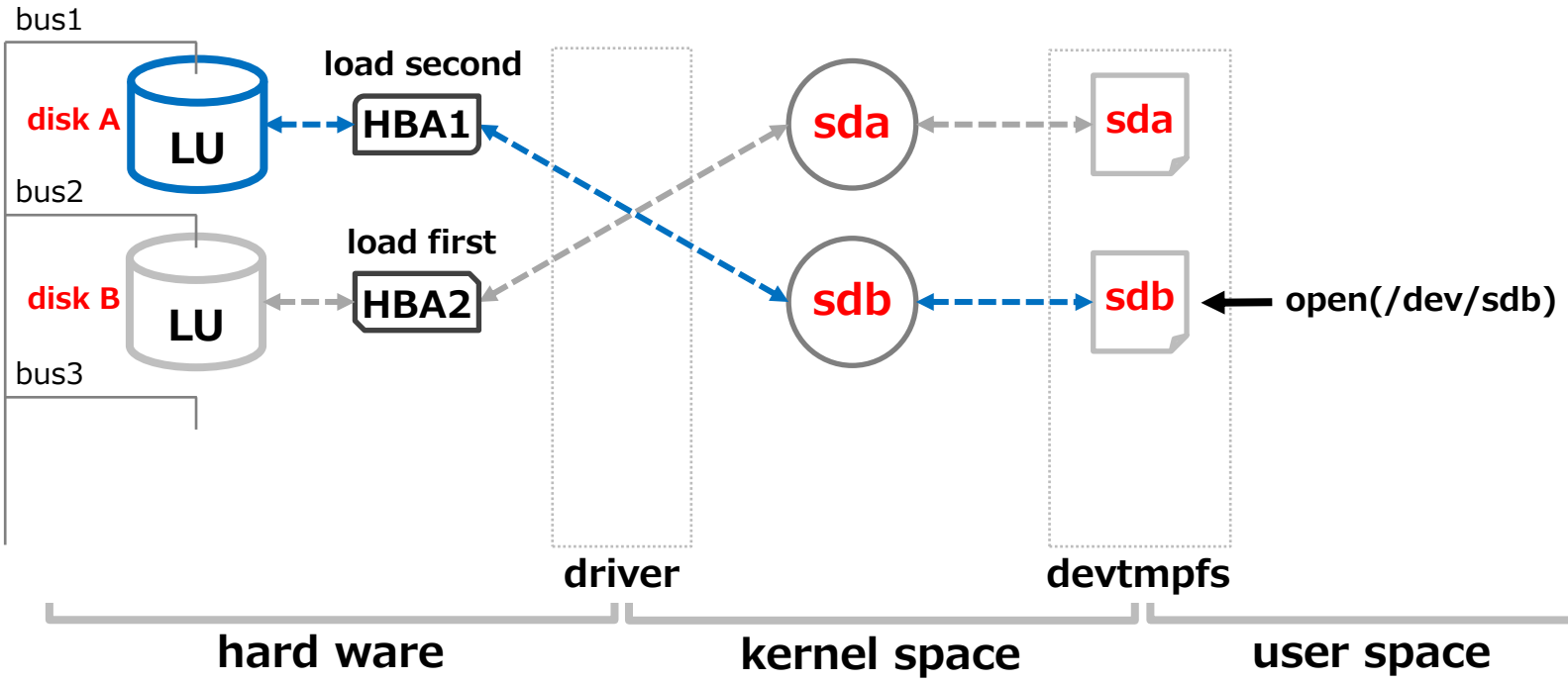
- HBA1 needs driver A and HBA2 needs driver B.
- When the driver A has been loaded before driver B, disk A is assigned to sda, and disk B is assigned to sdb.



HITACHI
Inspire the Next

Case3: Changing the order of driver

- However, if only driver A was updated and the driver order has been changed, driver B has been loaded before driver A.
- And disk B is assigned to sda, and disk A is assigned to sdb!



What happens if device name mismatch?

HITACHI
Inspire the Next

There are issues comes from using inconsistent device names

- Unexpected disk access
  An application supposed to write disk B, but writes disk C.
- If you use device file name to a configuration file, device mismatch occurs. Thus, redundancy will be broken
  - E.g.
    - multipath
    - software-raid

Why device name mismatch?

HITACHI
Inspire the Next

Root causes of device name mismatch

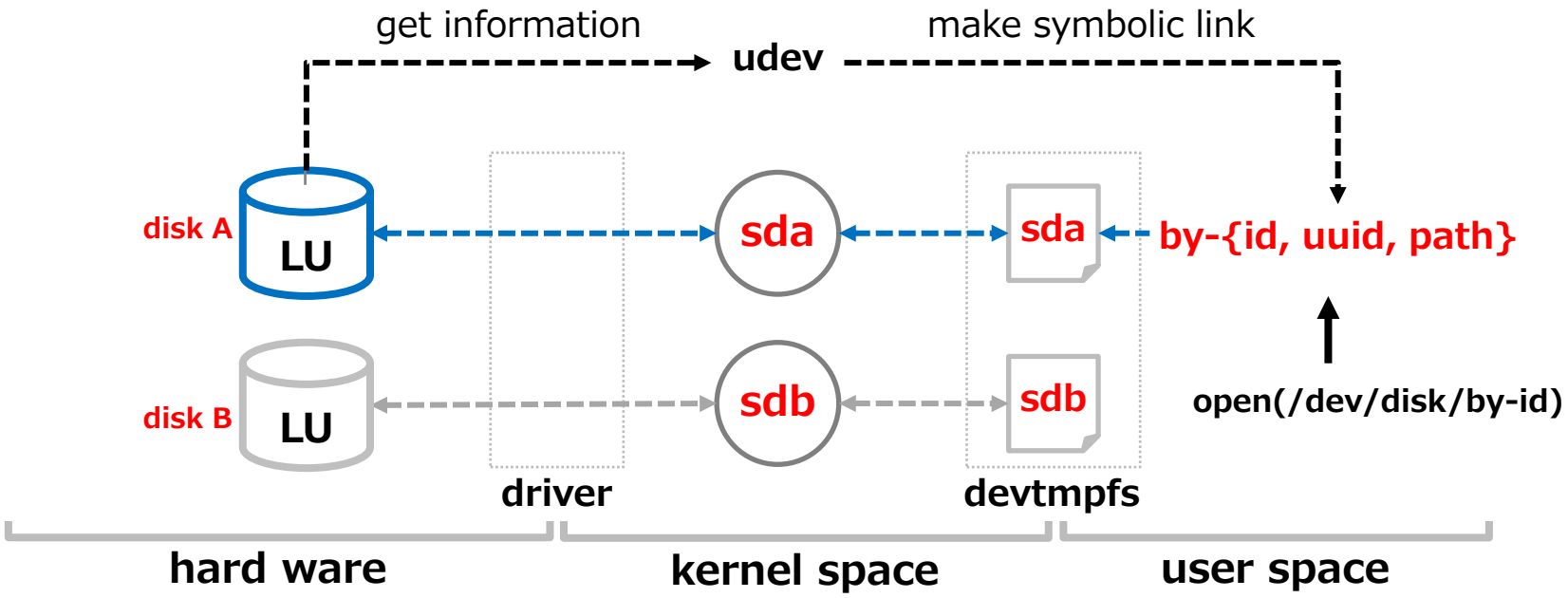
- Device configuration changing which changes bus number
  - E.g.
    - Hot-plug
    - Device Breakdown
- System configuration changing which changes driver loading order
  - E.g.
    - Driver update
    - Changing modprobe.conf

How to avoid device name mismatch

HITACHI
Inspire the Next

Persistent device name by udev

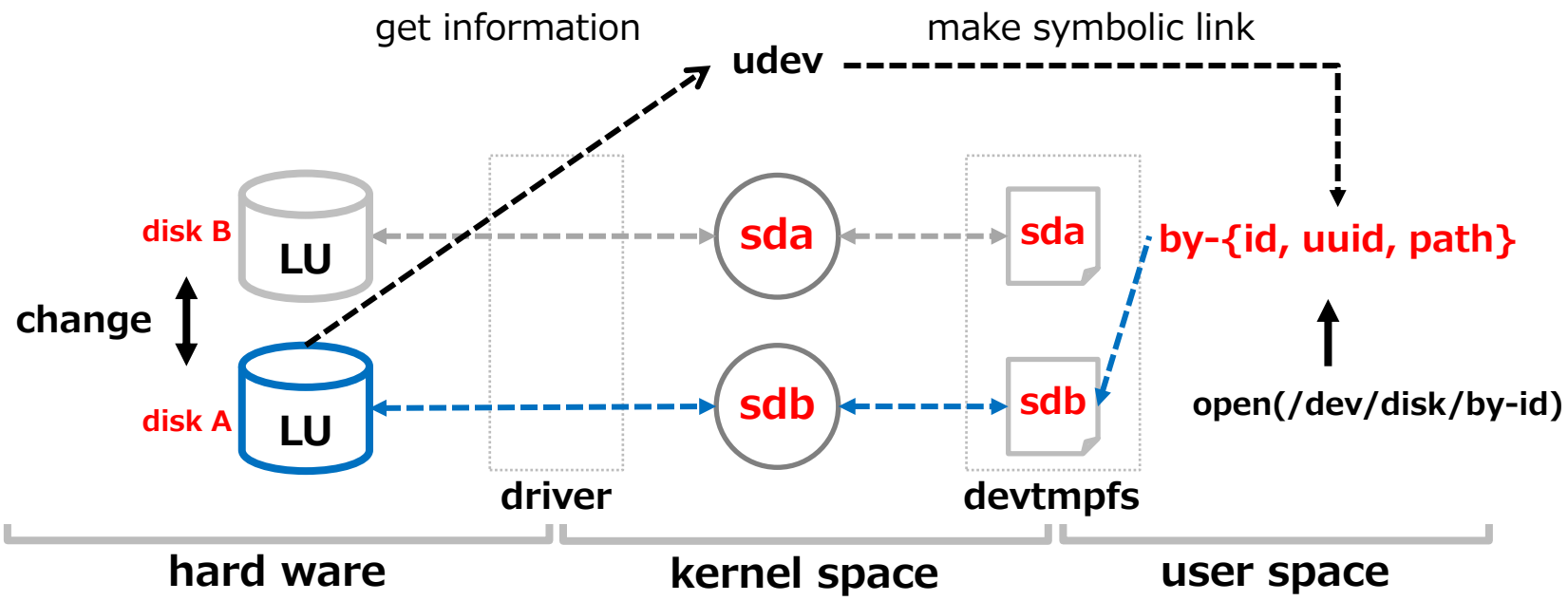
- Udev makes persistent symbolic links(by-id, by-uuid, by-path) pointing each device based on device information
- Applications access the device via these symbolic links.



Always point same device

Replace disk A and disk B

- Since udev directly read the information from device itself, those symbolic links can always point same device



How to use by-{id, uuid, path}

HITACHI
Inspire the Next

What is by-{id, uuid, path}

- by-id creates a unique name depending on the hardware serial number.
  - If disk itself has been changed (recover from backup), this doesn't work
- by-uuid creates a unique name depending on the Universally Unique Identifier. It is a mechanism to give each filesystem a unique identifier.
  - If user uses raw device, there is no uuid on the disk.
- by-path creates a unique name depending on the physical path.
  - If device moves to other slot, this doesn't work.

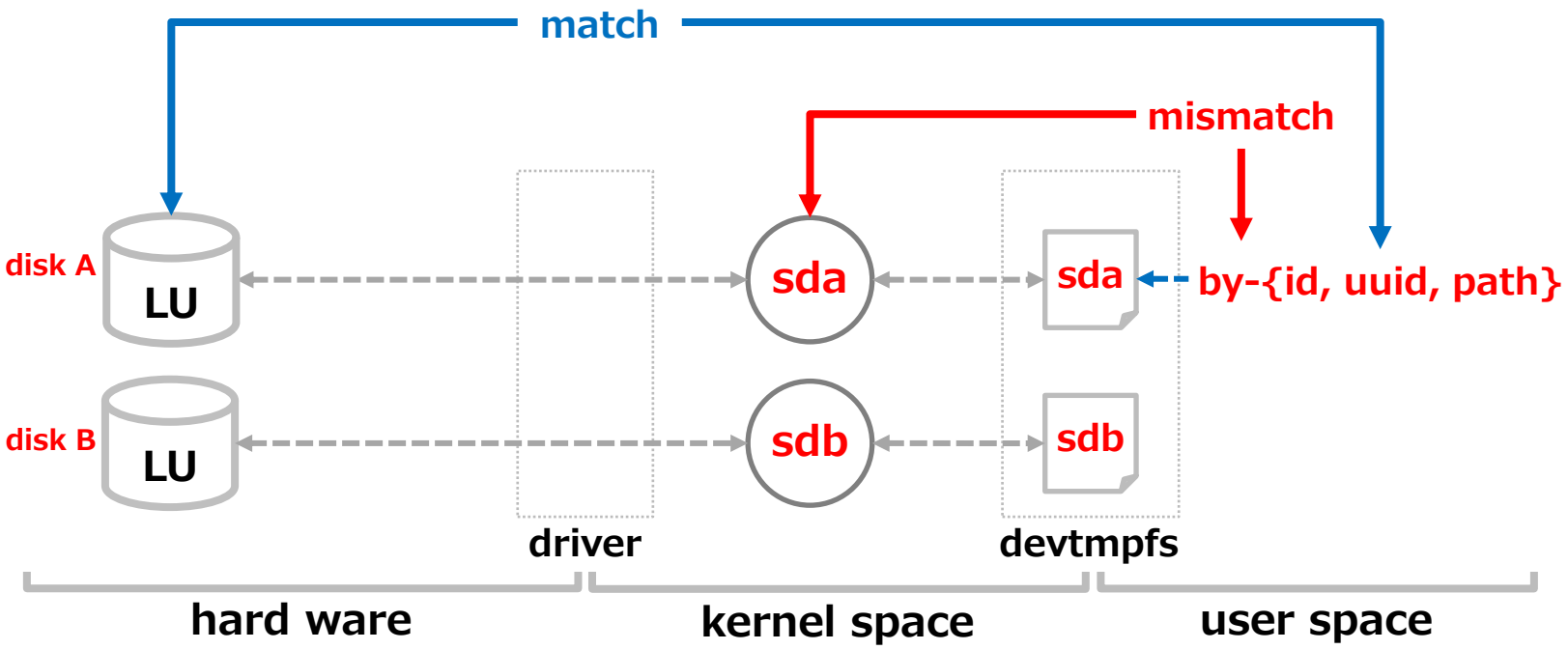
There is no best way, depends on the situation.

Is this perfect solution?

HITACHI
Inspire the Next

Issue of persistent device naming by udev

- Udev solves mismatch between **device file** and **physical disk**.
- But the persistent name mismatches kernel's **device name**.
- I'd like to show you what issues will happen on this situation.



Issue 1: Which disk is "*by-uuid*"?

HITACHI
Inspire the Next

/proc information

- /proc/partitions, /proc/diskstat gives you device names
- User have to run "ls –l /dev/disk/by-*" or "udevadm" for finding corresponding persistent name

which disk is "77c2a61-d50f-4710-84d8-880c24b244a5"?

```
$ cat /proc/partitions
   major   minor     #blocks   name

   8        0     488386584   sda
   8        1        194560   sda1
   8        2       7812096   sda2
   8        3      19531776   sda3
   8        4      19530752   sda4
   8        5      19531776   sda5
   8        6     421784576   sda6
```

Issue 2: Kernel log shows device name

HITACHI
Inspire the Next

Kernel log

- dmesg outputs **device name** instead of persistent name
- A user do not know which disk sda, because a user identify a disk by a persistent device name.

```
$ dmesg
…
sda:
  sda1 sda2 sda3 sda4 sda5 sda6
sd 0:0:0:0: [sda] Attached SCSI disk
dracut: Mounted root filesystem /dev/sda3
EXT4-fs (sda3): INFO: recovery required on readonly filesystem
EXT4-fs (sda3): write access will be enabled during recovery
EXT4-fs (sda3): recovery complete
EXT4-fs (sda3): mounted filesystem with ordered data mode
dracut: Mounted root filesystem /dev/sda3
…
```

Issue 3: Wrong output

HITACHI
Inspire the Next

Commands not work

- Some system commands don't accept symbolic link
  (e.g. df, smartctl, sgpio, grub-isntall, iostat)
- Those commands just expect sdX device name or check input by /proc information

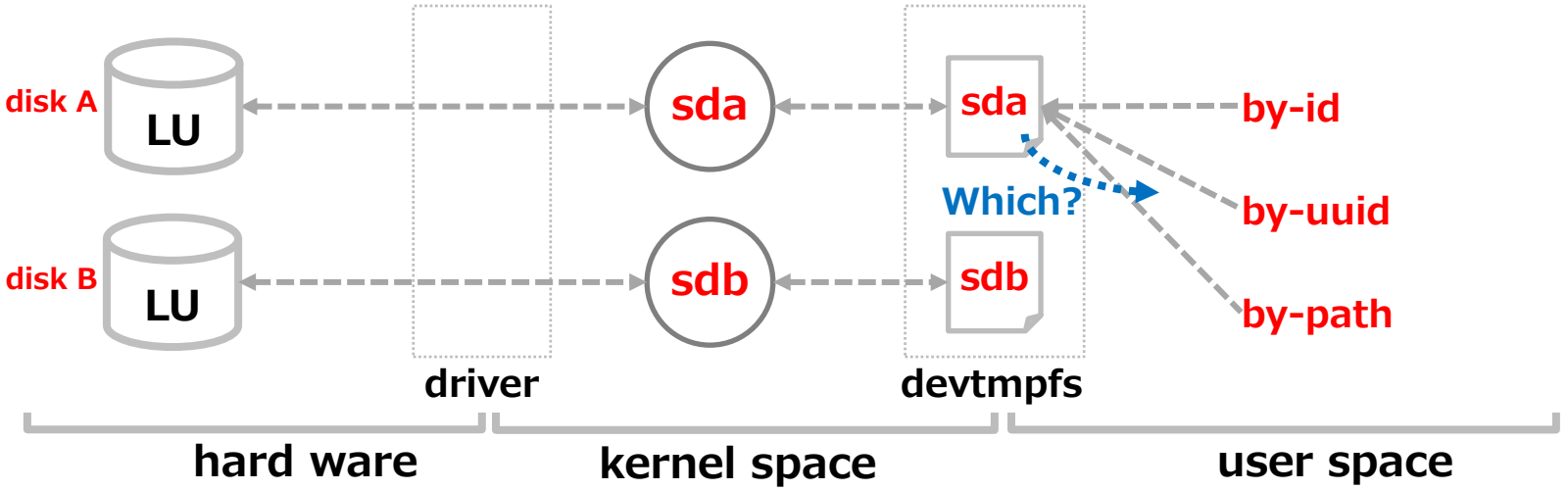
```
$ ls -l /dev/disk/by-uuid/8d06a843-9482-47a6-9209-e39c8b284c18
/dev/disk/by-uuid/8d06a843-9482-47a6-9209-e39c8b284c18 -> ../../sda1

$ df /dev/sda1
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1               188403     57449    121226  33% /boot

$ df /dev/disk/by-uuid/8d06a843-9482-47a6-9209-e39c8b284c18
Filesystem           1K-blocks      Used Available Use% Mounted on
-                      3051548       240   3051308   1% /dev   <-- wrong
```

Issue 4: Which name will user need?

HITACHI
Inspire the Next

Undecided symbolic link

- Several symbolic links point a device file.
- Can not determine a symbolic link from a device.



HITACHI
Inspire the Next

Interface name can be renamed

Network interface has a special interface for renaming by udev or ifcfg

ifcfg-eth0:

```
DEVICE=eth0
BOOTPROTO=dhcp
HWADDR=00:24:E8:2D:BC:83
ONBOOT=yes
…
```

udev rules:

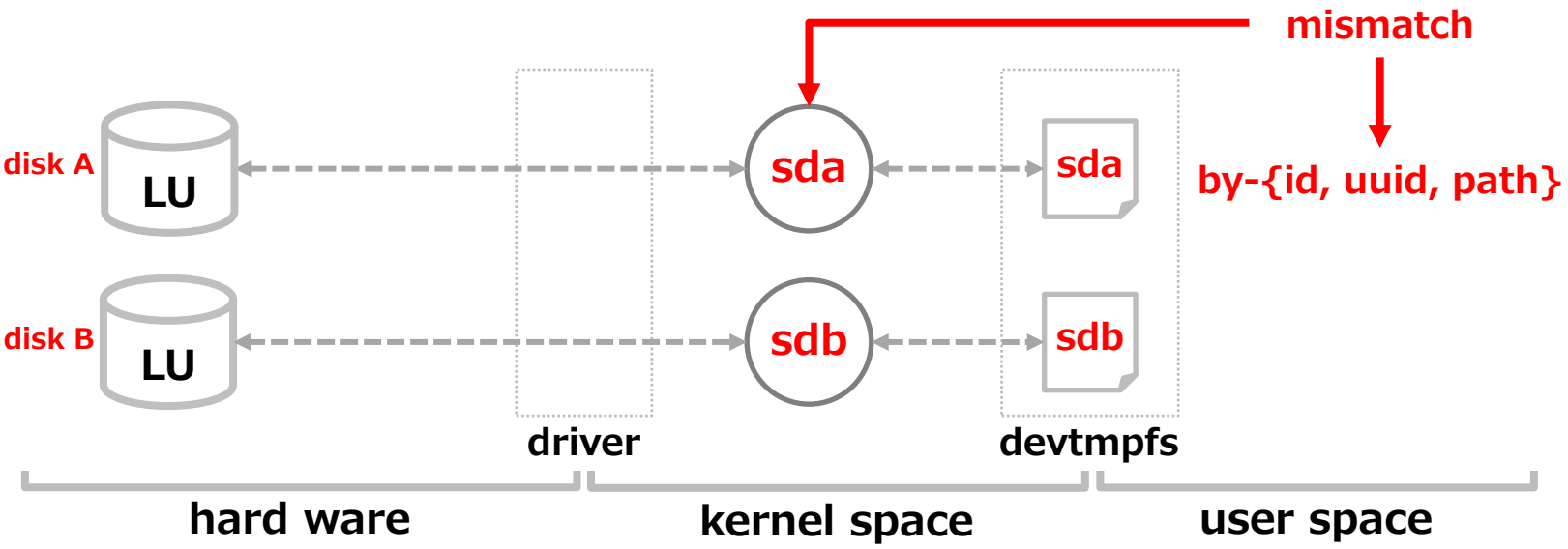
```
KERNEL==“eth*”, SYSFS{address}==“00:24:E8:2D:BC:83”, NAME=“eth0”
```

We need another solution

HITACHI
Inspire the Next

Symbolic link is not the perfect solution

- Issues come from the mismatch of device name itself and symbolic link
- Network works very very fine by changing interface name itself
- Why we can't change device name itself too?



Proposal: Device Renaming

HITACHI
Inspire the Next

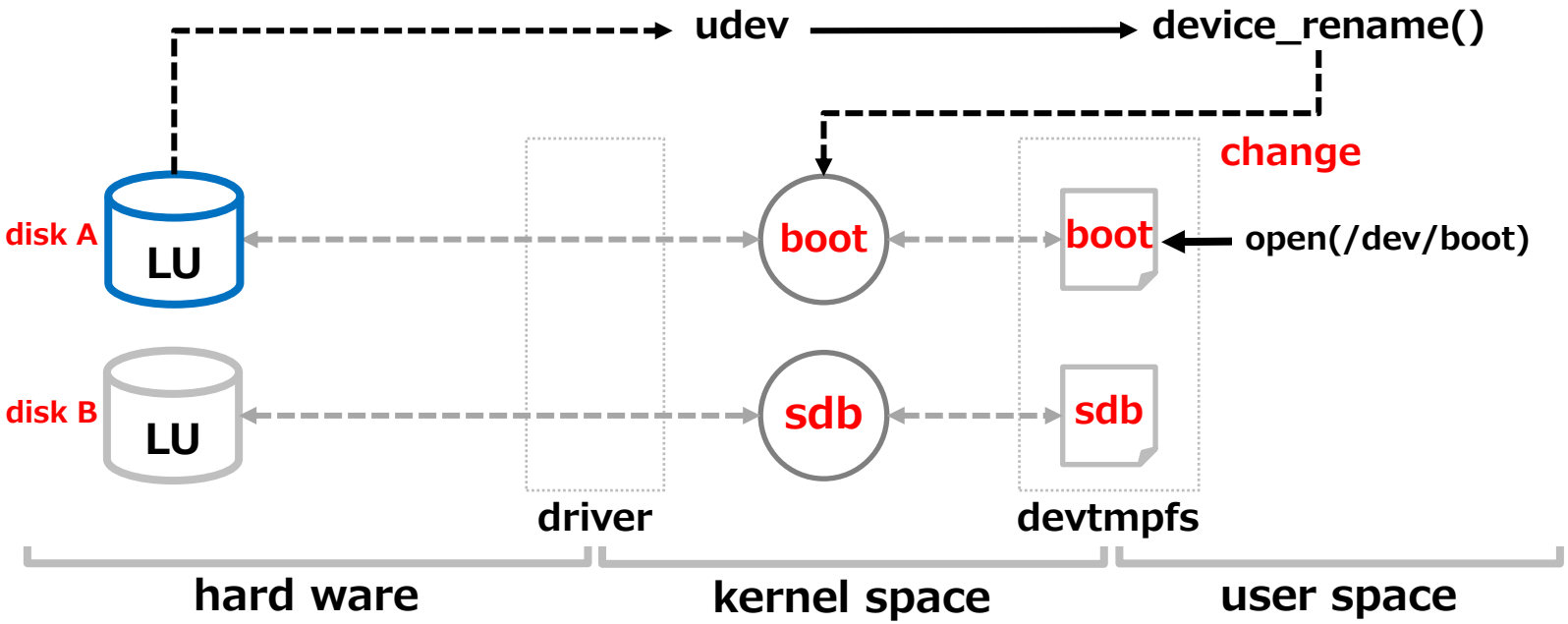
Introduce device rename interface

If we have an kernel interface for device renaming, combining it with udev persistent device naming can solve those issues.

For example, this sets “boot” *by-uuid’s* device.

udev rules:

```
KERNEL==“sd*[!0-9]”, PROGRAM=“device_rename”, BYUUID=“by-uuid”, DEVNAME=“boot”
```



Items to be solved

HITACHI
Inspire the Next

We still have some items to be solved

- When should device name be changed?
  Where can i change device name on boot sequence? After udev daemon service started? Sorry, I don't know how that will affect other subsystem by change device name. Please let me know what will happen.
- To what name should device name be changed
  - What name is "good"?
  - Some system command options accept only sd*, hd*.

Conclusion

HITACHI
Inspire the Next

Summary

- Device name are assigned by the order of device recognizing
- Cases of device name mismatch
  - Changes bus number, driver loading order
- Symbolic link is not the perfect solution
  - The persistent name mismatches kernel's device name
- Network interface can be renamed

Future plans

- There are two issue
  - Persistent device naming → **This issue can be solved by udev**
  - Device name and Symbolic link mismatch
- How can we solve device name and symbolic link mismatch?
  - Make interface to change device name
    - When change device name?
- We would like to discuss on LKML and make device renaming patch.

Trademarks

HITACHI
Inspire the Next

- Linux is a trademark of Linus Torvalds in the United States, other countries, or both.
- Other company, product, or service names may be trademarks or service marks of others.

HITACHI
Inspire the Next

END

A Flexible Device Name Assignment Method

LinuxCon Japan 2010/09/28

Systems Development Laboratory
Linux Technology Center

Nao Nishijima

HITACHI

Inspire the Next