09'Linux Plumbers Conference

Data de-duplication

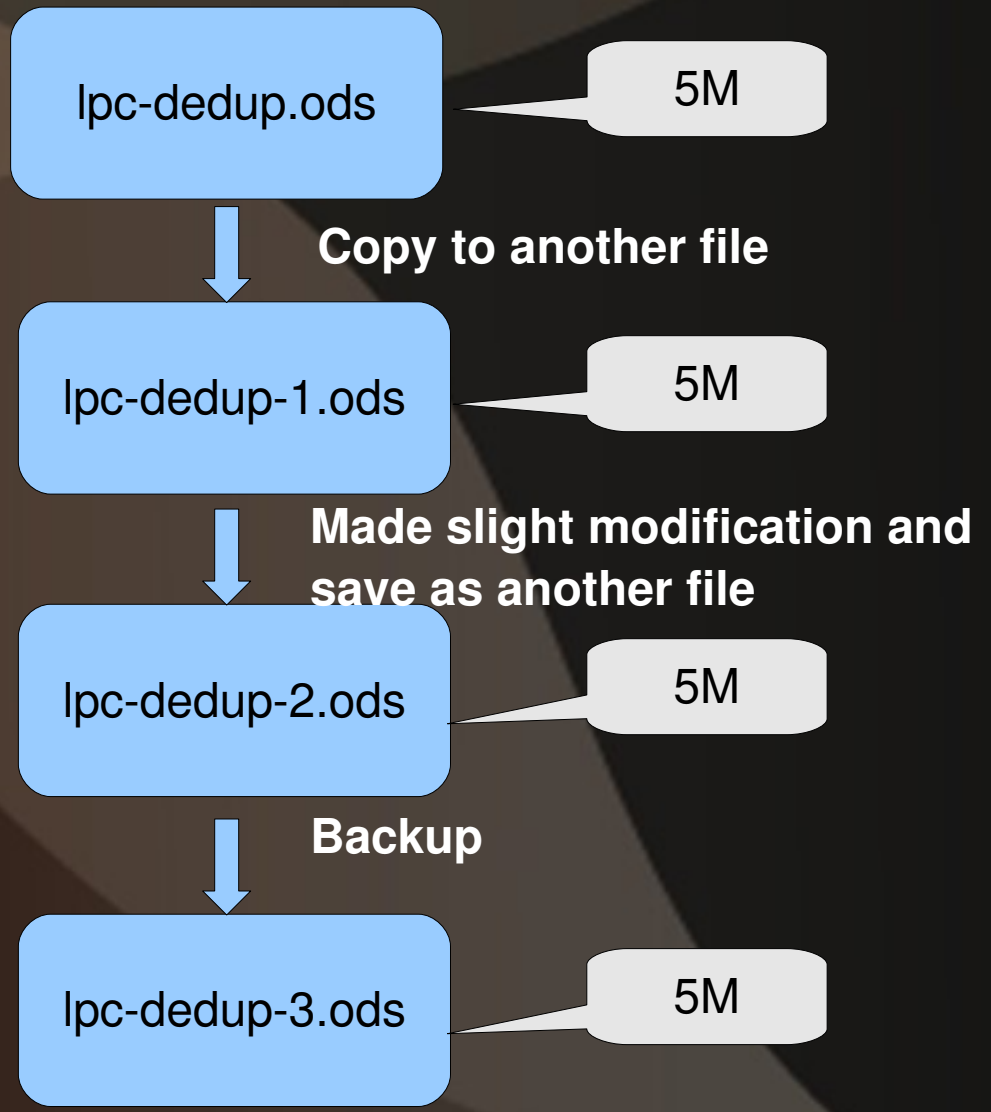
Mingming Cao
IBM Linux Technology Center
cmm@us.ibm.com
2009-09-25

Current storage challenges

- Our world is facing data explosion. Data is growing in a amazing rate
- Severe challenges just storing the data we have today, imagine how much more difficult and expensive storing six times more data tomorrow.
- Eliminating redundant data is very important!

Existing technology...

- Hard links/cow/file clone
- Compression
- All are done at file level. There are more room to save space from.
- Imagine this …



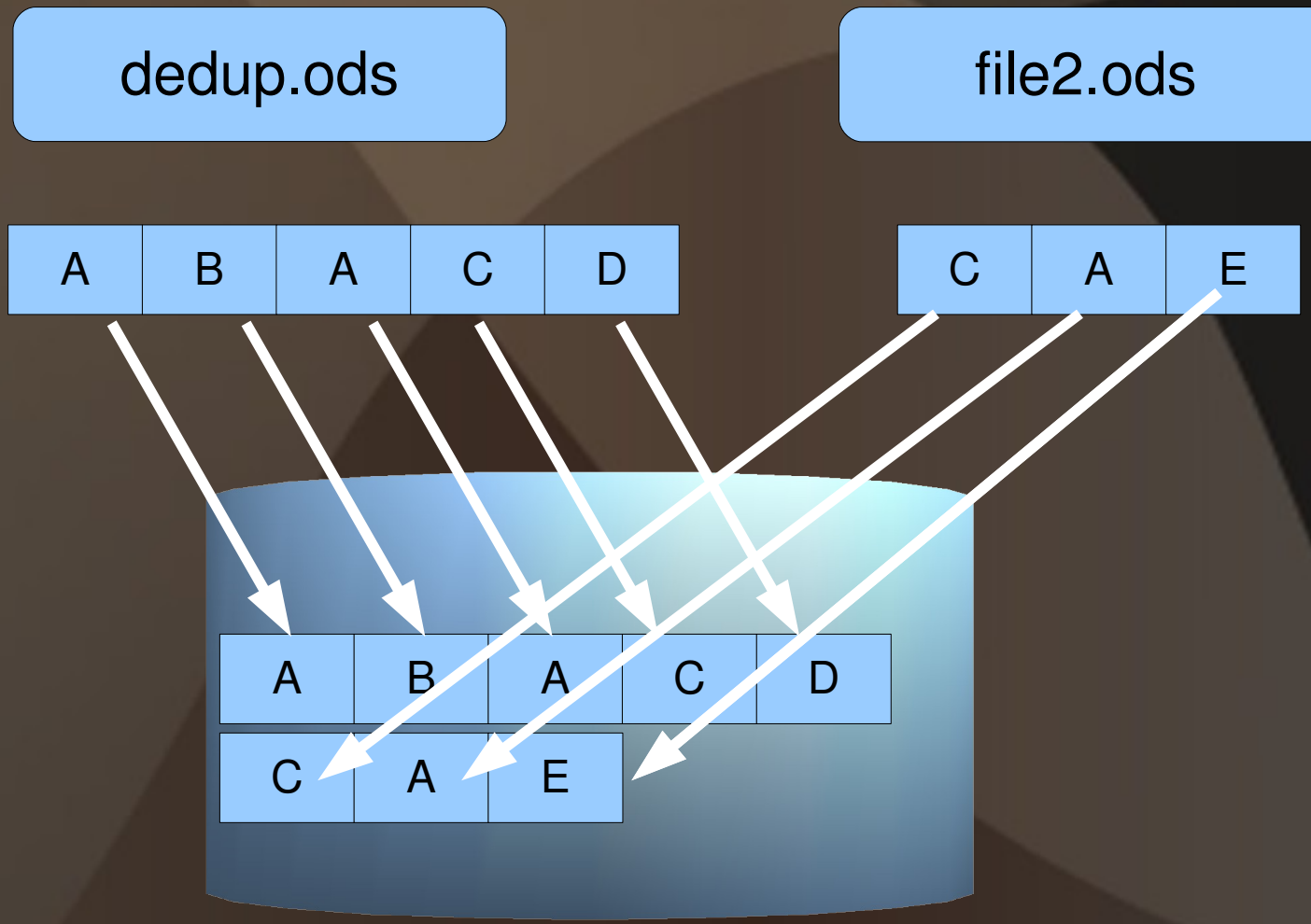
Definition

- Data de-duplication

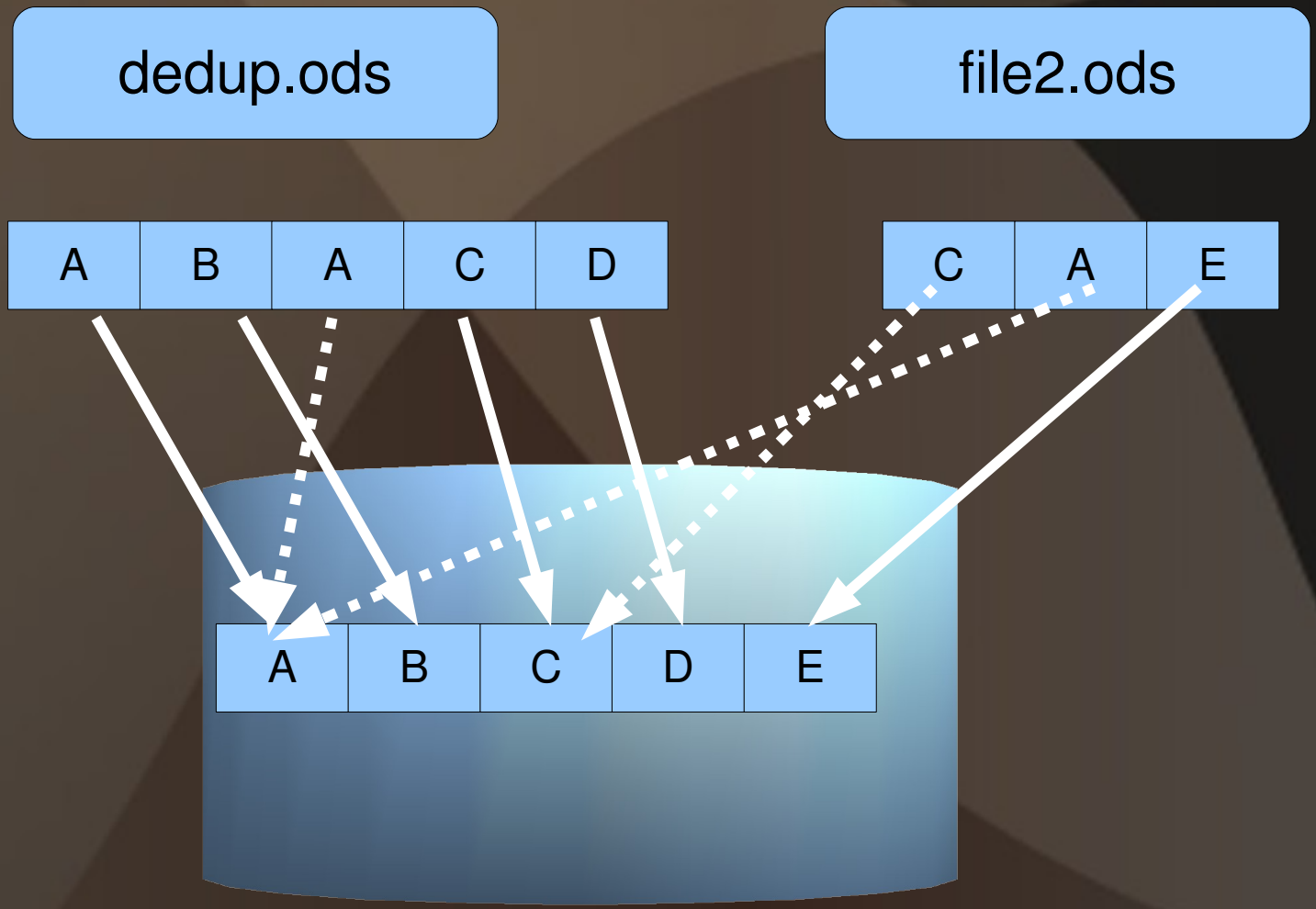
  A method of reducing storage needs by eliminating redundant data. Only one unique instance of the data is actually retained on storage media, such as disk or tape. Redundant data is replaced with a pointer to the unique data copy.

- Data de-duplication is extended compression, more efficient to remove the redundant data. Could be done at file level, sub-file(block) level and even bit level

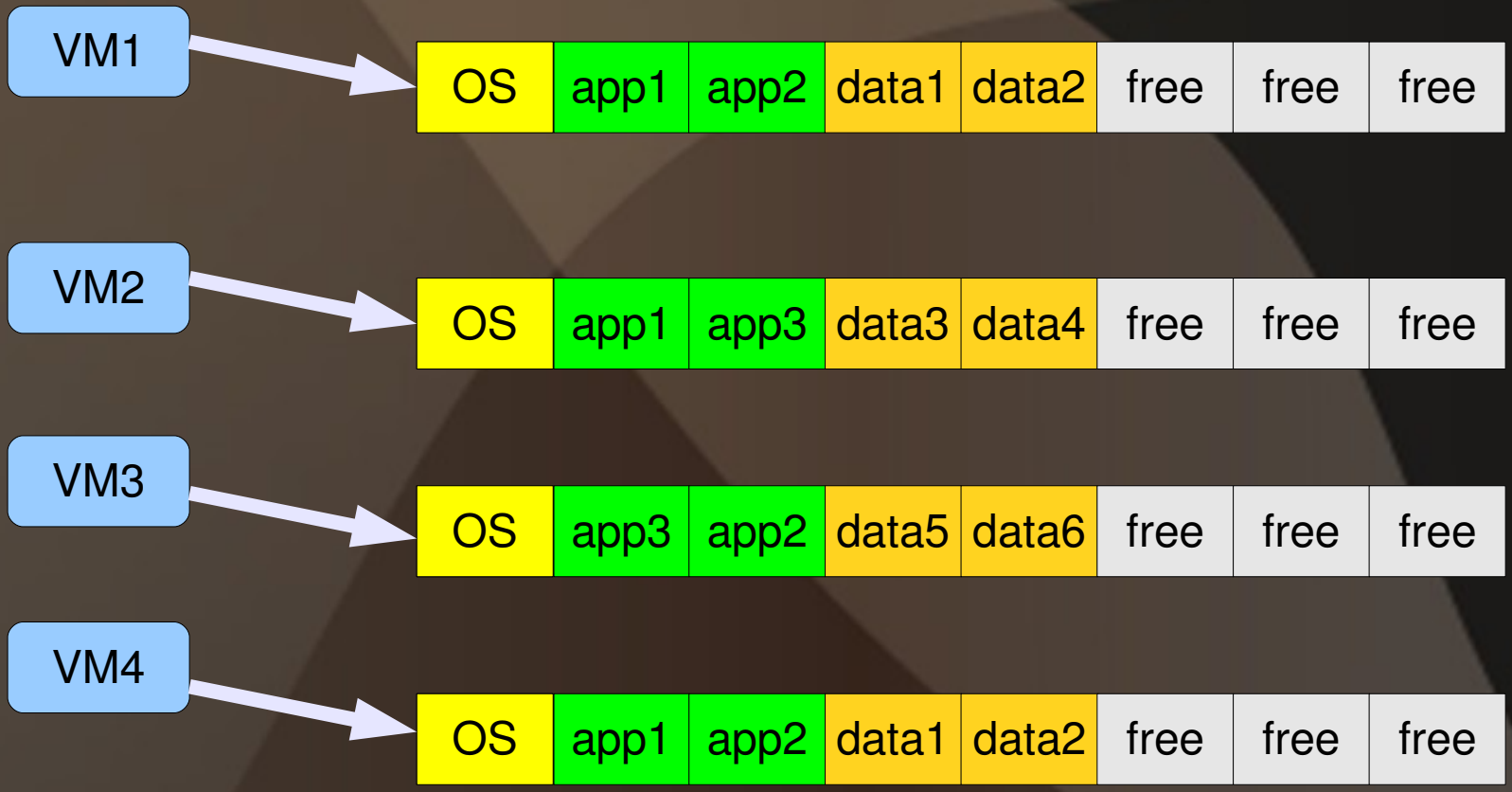
de-duplication before



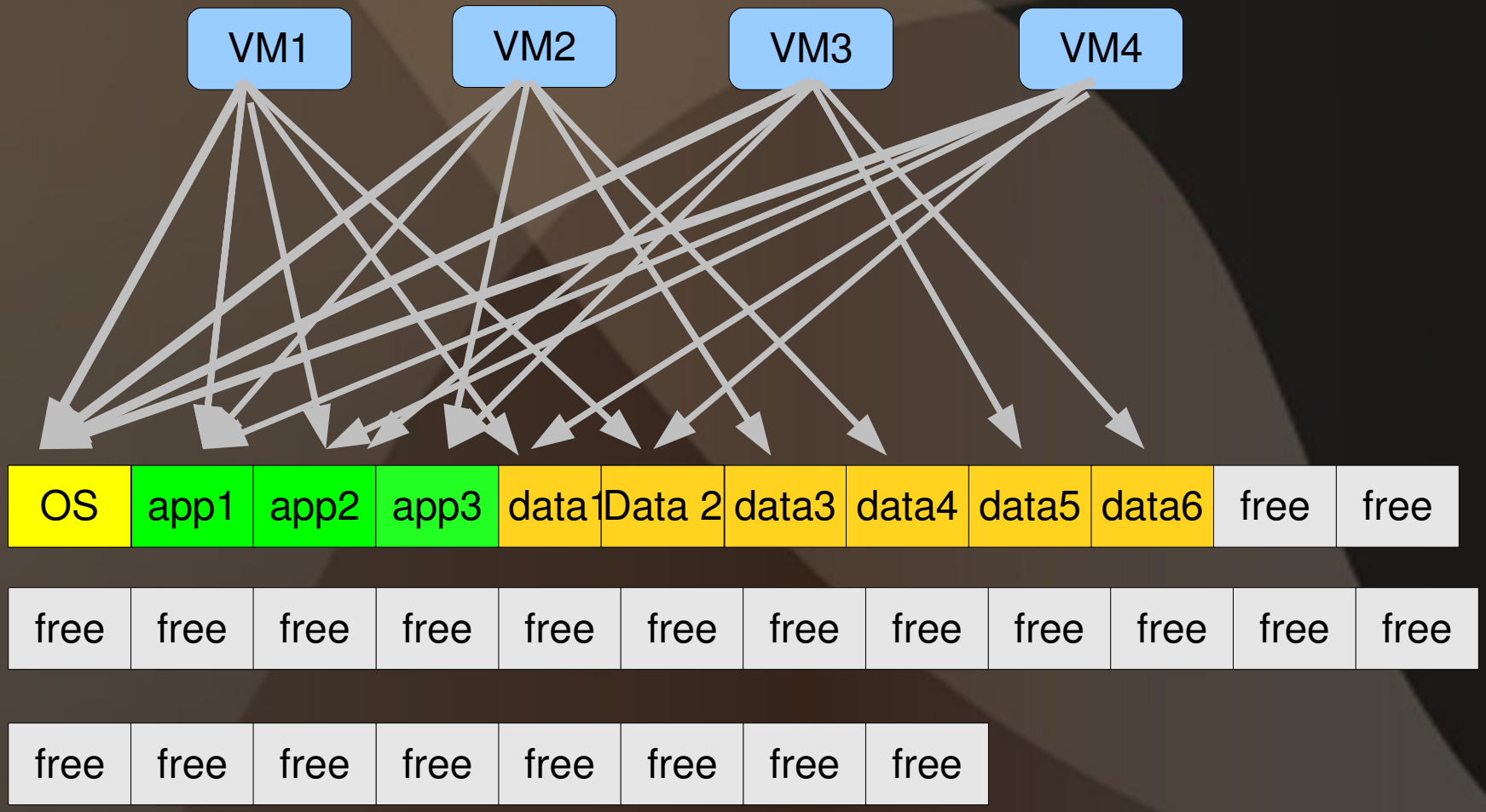
de-duplication after



de-duplication for VM



de-duplication for VM(after)



de-duplication benefit

- Two major savings
  - Storage footprint(6x healthcare, 3x VM, 20x backup)
  - Network bandwidth to transfer data across WAN
- Disks are cheep, but there is more than just space
  - Save more energy (power) and cooling
  - Disaster and recovery becomes manageable
  - Save resources to manage same amount of data
- Typical workload: Backup, Archives, Healthcare, Virtualization, NAS, remote office etc.

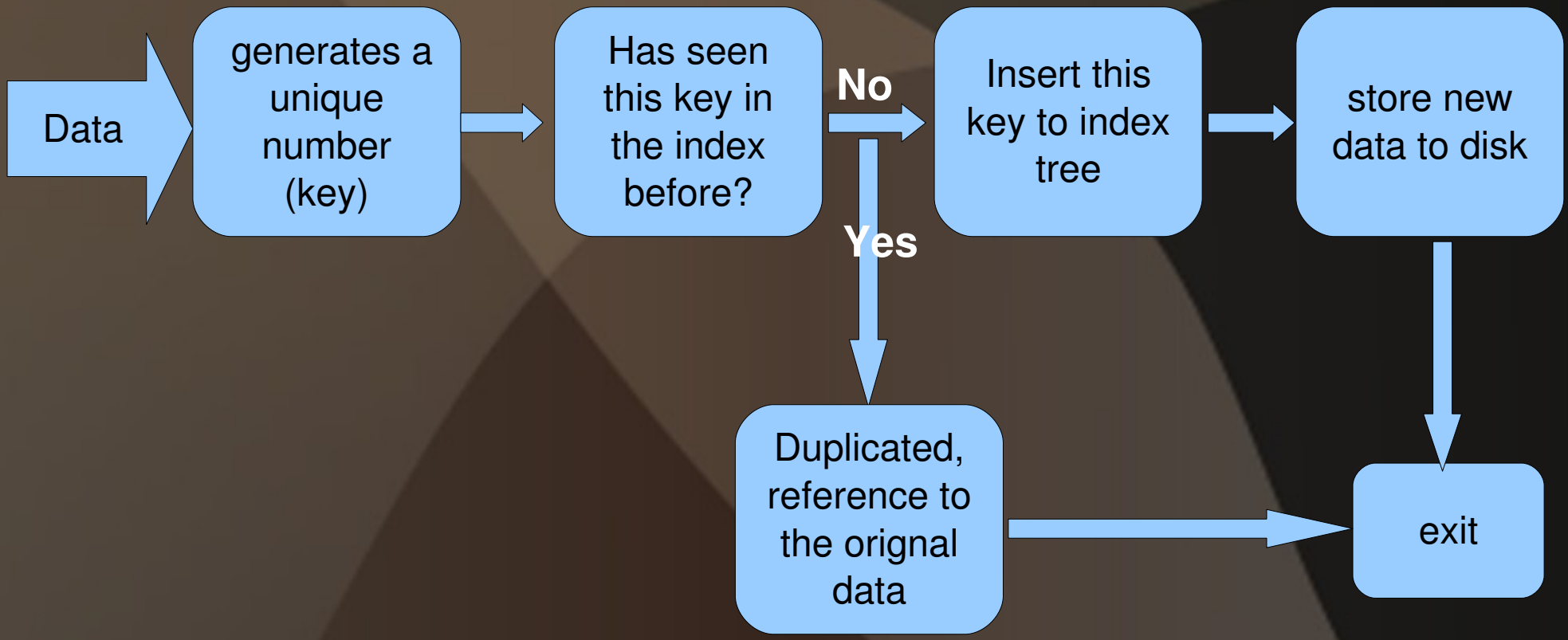
de-duplication concerns

- Large CPU and memory resources required for de-duplication processing
- Potentially more fragmented files/filesystem
- Potentially increase risk of lost data
- Might not work with encryption
- Hash collison still possible

de-duplication ratios

- Indicating how much reduction by de-duplication
  - Before 50TB, after 10TB, ratio is 5:1
- Ratio could various from 2:1 to 10:1, depends on
  - Type of data
  - Change rate of the data
  - Amount of redundant data
  - Type of backup performed (full, incremental or differential)
  - de-duplication methods

de-duplication process



Where: source vs target

method	where	advantages	disadvantages
Source	performed at the data source, before transfer to target location	Reduce network bandwidth;Awareness of data usage and format may allow more effective data dedup	Deduplication consumes CPU cycles on the file/ application server; May not dedup files across various sources
Target	performed at the target (e.g. by backup software or storage appliance)	Applies to any type of filesystem; No impact to data ingestion;possible for parellel data deduplication	Deduplication consumes CPU cycles on the target server or storage device

When: In-line vs Post process

method	when	advantages	disadvantages
In-line	deduplication occurs in the primary data path. No data is written to disk until the deduplication process is complete.	Immediate data reduction, uses the least disk space No post-processing	Perfornance concerns, high cput and memory cost;
Post process	Deduplication occurs on the secondary storage. Data were first store data on disk and then deduplicate	Easy to implement; No impact to data ingestion;possible for parellel data deduplication	Data being processed twice; Need extra space for dedup;Race with concurrent writes

In-line de-dup in btrfs: How to detect the redundancy?

- Make sense...Btrfs already create checksum for every fs block, and stored on disk. Re-use hash value for duplication key.
- To speed look up, need separate checksum index tree, indexed by checksums rather than logical offset
- Duplication screen could happen at data writeout time. After data get compressed, but before delayed allocation allocate space and flush-out data
- If hash collision occurs, do byte to byte compare to ensure no data lost

In-line de-dup in btrfs: How to lower the cost?

- Memory usage is the key to dedup performance
  - The dedup hash tree needs in memory. For 1TB fs needs 8G RAM for SHA256, or 4G RAM for MD5
  - Make dedup optional: filesystem mount option, or enable/disable dedup on file/subvolumes etc
- Fragmentation
  - Apply policies to defrag to group shared files close to each other
  - Reduce seek time: frequently and lately shared blocks are likely already pinged in memory
  - Might be less an issue with SSD

In-line dedup in btrfs: Keep the impact low

- Could have impact to running applications
- Gets some latency stats, enable/disable dedup if ingestion is high
- Could have a background scrub thread to do dedup on files that didn't get dedup in-line before writeout to disk
- Flag to each btrfs extent to indicating deduped or not, to avoid double dedup

User space de-duplication?

- User apps do the job instead of kernel. Could avoid ingestion.
- Could apply to any filesystem (ext4, btrfs, xfs etc)
- The checksum is maintained in userspace.
- Introduce VFS API to allow apps to poll whether chunk of data have been modified before merge
  - Could use inode ctime/mtime or inode version
  - Better, a new system call to tell a range of file(offset, size,transaction ID) has be changed since then

Summary

- Linux needs data de-duplication technology to able to control data explosion ...
- One size won't fit all … perhaps both?