HDFS RAID

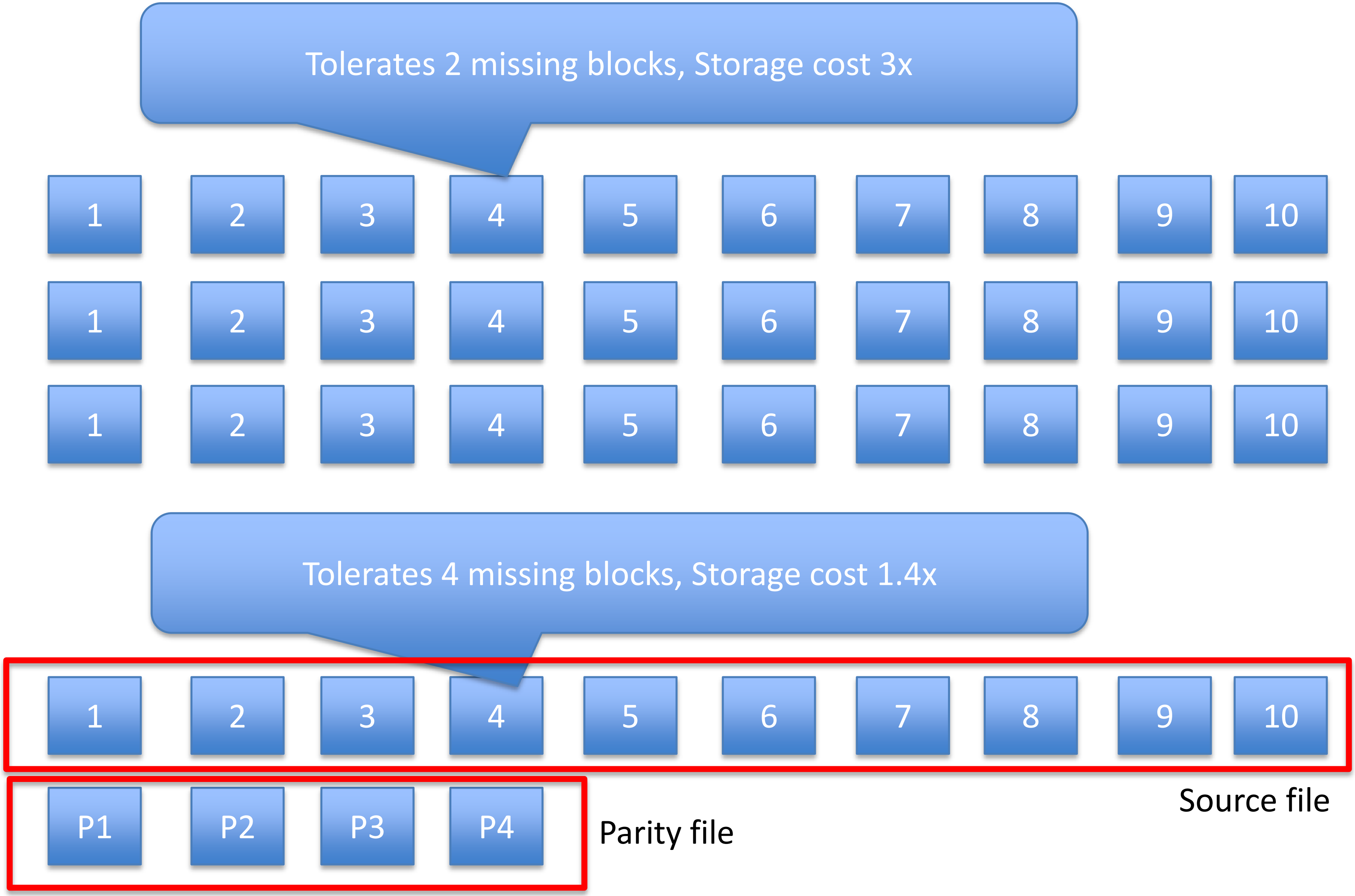
DhrubaBorthakur (dhruba@fb.com)
Rodrigo Schmidt (rschmidt@fb.com)
RamkumarVadali (rvadali@fb.com)
Scott Chen (schen@fb.com)
Patrick Kling (pkling@fb.com)

Agenda

- What is RAID
- RAID at Facebook
- Anatomy of RAID
- How to Deploy
- Questions

What Is RAID

- Contrib project in MAPREDUCE
- Default HDFS replication is 3
  - Too much at PetaByte scale
- RAID helps save space in HDFS
  - Reduce replication of “source” data
  - Data safety using “parity” data



Tolerates 2 missing blocks, Storage cost 3x

Tolerates 4 missing blocks, Storage cost 1.4x

Source file

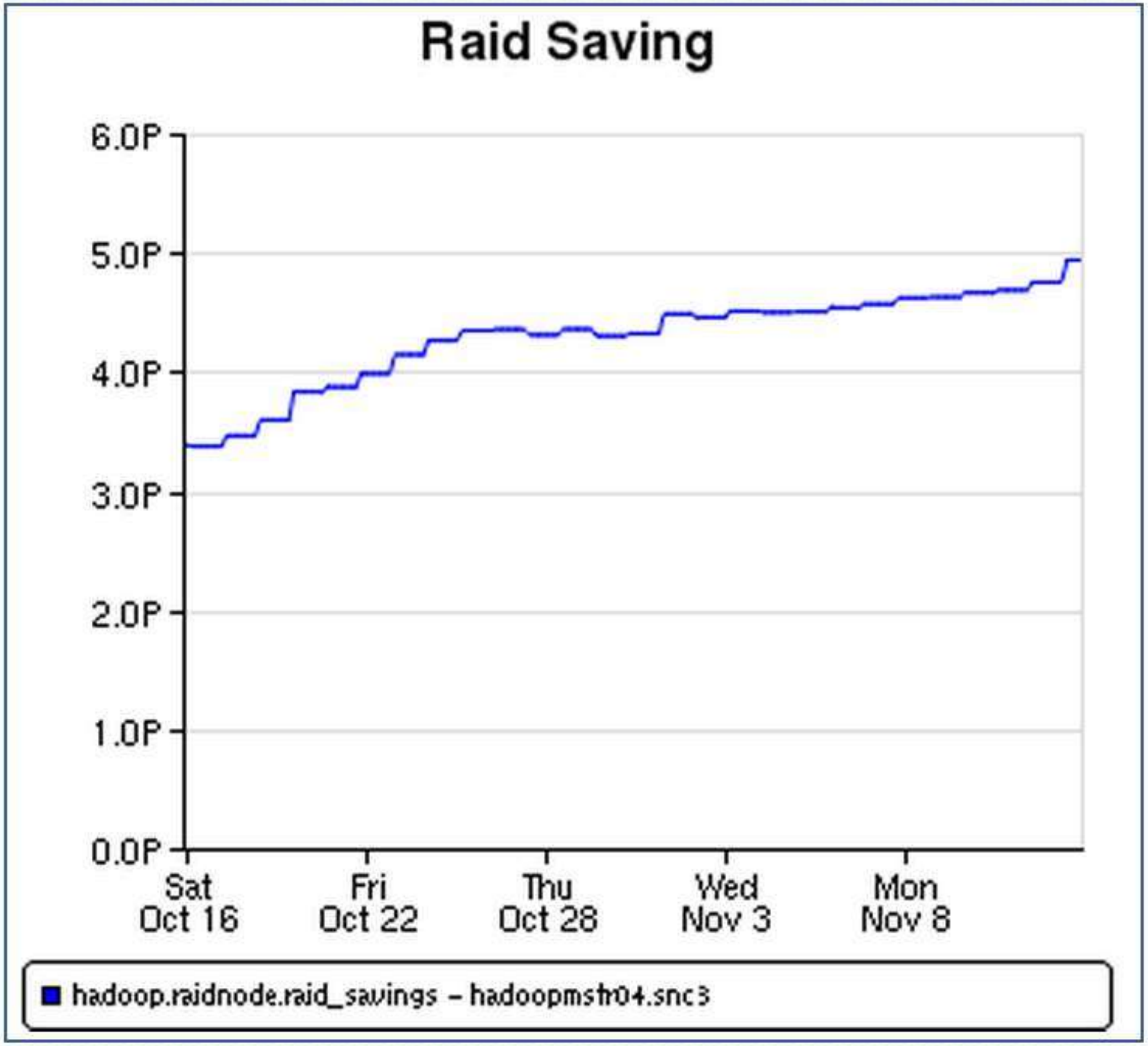
Parity file

Reed-Solomon Erasure Codes

RAID at Facebook

- Reduces disk usage in the warehouse
- Currently saving about 5PB with XOR RAID
- Gradual deployment
  - Started with few tables
  - Now used with all tables
  - Reed Solomon RAID under way

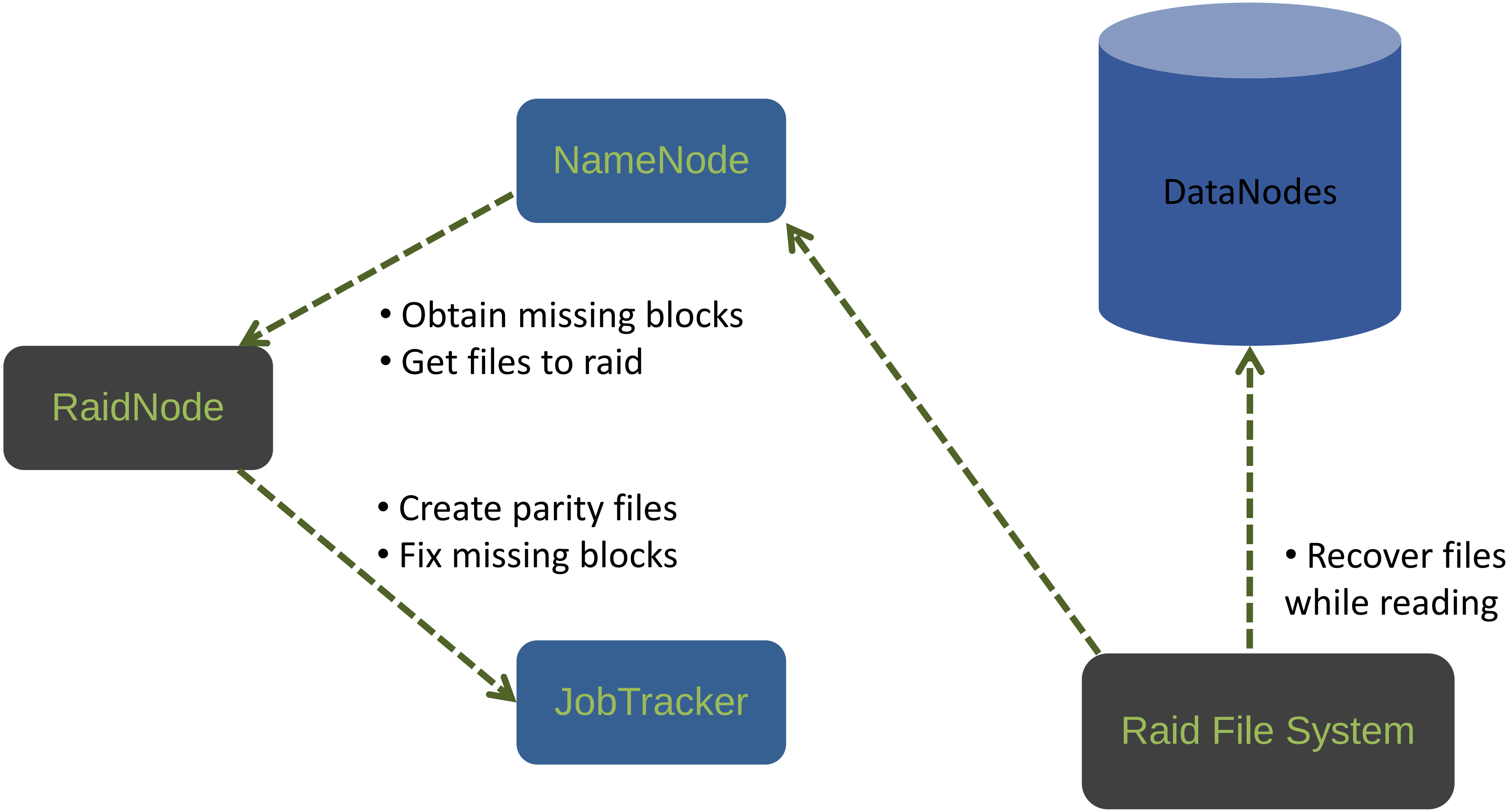
Saving 5PB at Facebook



Anatomy of RAID

- Server-side:
  - RaidNode
    - BlockFixer
  - Block placement policy
- Client-side:
  - DistributedRaidFileSystem
  - Raid Shell

Anatomy of RAID



- Obtain missing blocks
- Get files to raid

- Create parity files
- Fix missing blocks

- Recover files while reading

RaidNode

- Daemon that scans filesystem
  - Policy file used to provide file patterns
  - Generate parity files
    - Single thread
    - Map-Reduce job
  - Reduces replication of source file
- One thread to purge outdated parity files
  - If the source gets deleted
- One thread to HAR parity files
  - To reduce inode count

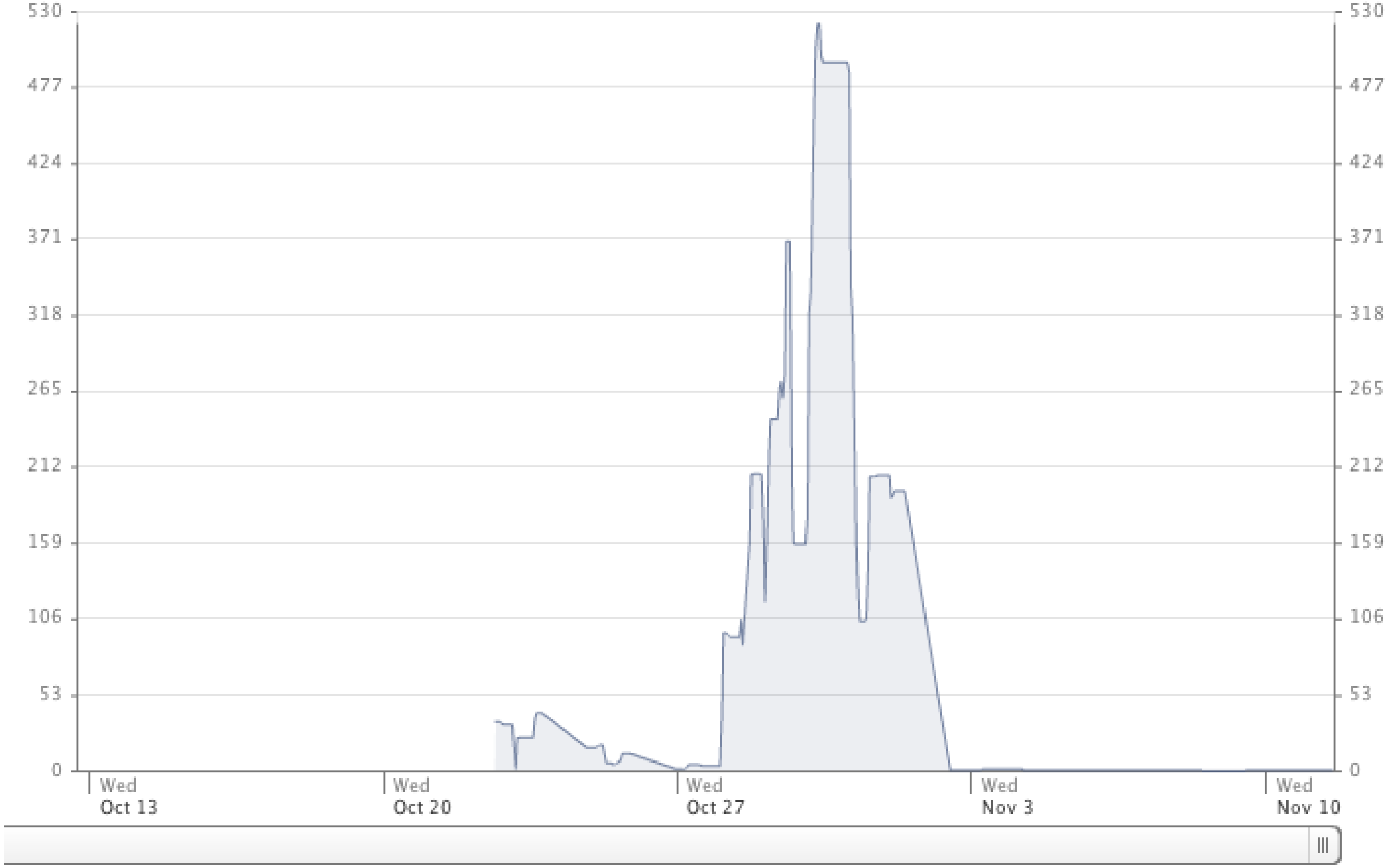
Block Fixer

- Reconstructs missing/corrupt blocks
- Retrieves a list of corrupt files from NameNode
- Source blocks are reconstructed by “decoding”
- Parity blocks are reconstructed by “encoding”

Block Fixer

- Bonus: Parity HARs
  - One HAR block => multiple parity blocks
  - Reconstructs all necessary blocks

Block Fixer Stats



Erasure Code

- ErasureCode
  - abstraction for erasure code implementations

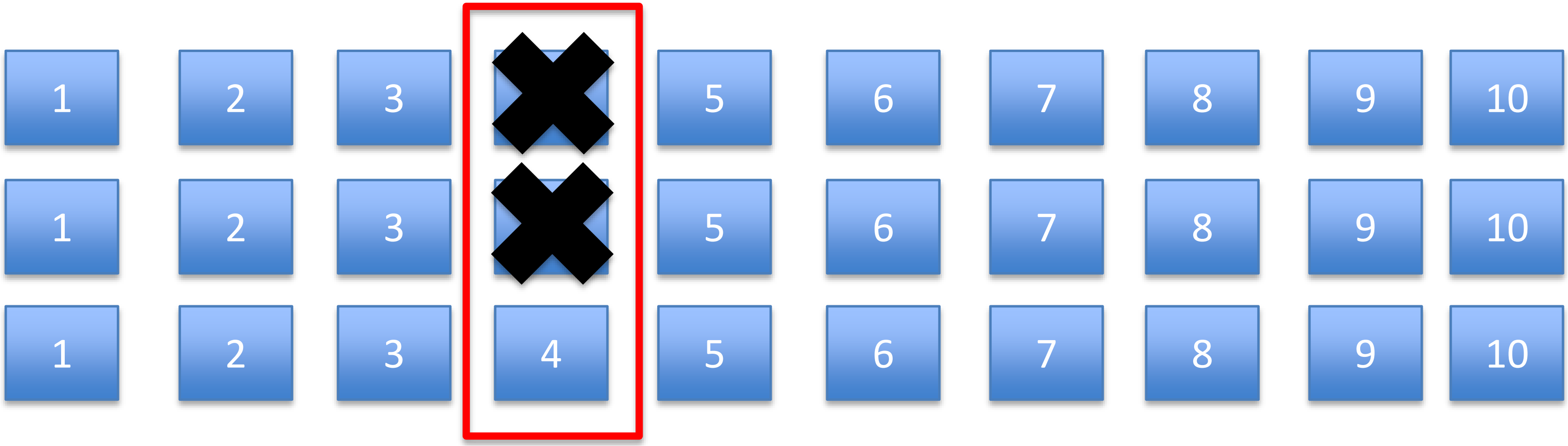
```
  public void encode(int[] message, int[] parity);

  public void decode(int[] data,
int[] erasedLocations,
int[] erasedValues);
```

- Current implementations
  - XOR Code
  - Reed Solomon Code
- Encoder/Decoder – uses ErasureCode to integrate with RAID framework

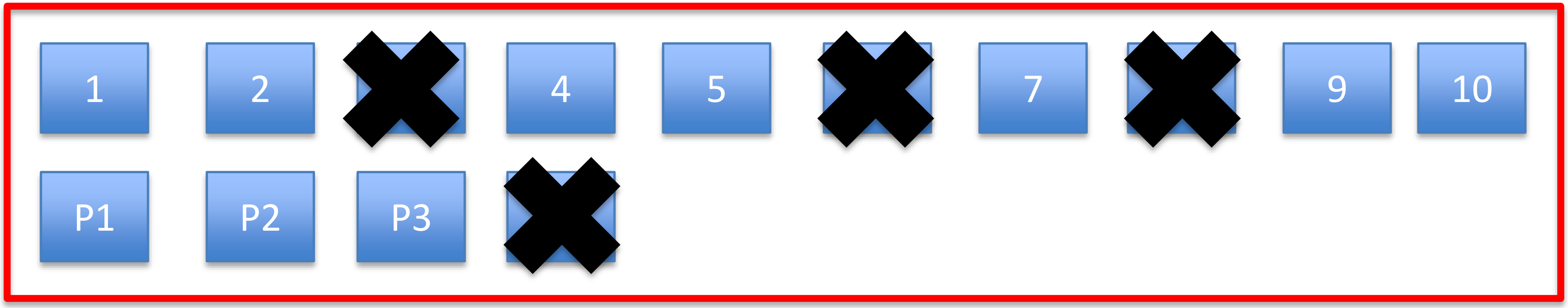
Block Placement

Replication = 3, Tolerates any 2 errors



Dependent Blocks

Replication = 1, Parity Length = 4, Tolerates any 4 errors



Dependent Blocks

Block Placement

- Raid introduces new dependency between blocks in source and parity files
- Default block placement is bad for RAID
  - Source/Parity blocks can be on a single node/rack
  - Parity blocks could co-locate with source blocks
- Raid Block Policy
  - Source files: After RAIDing, disperse blocks
  - Parity files: Control placement of parity blocks to avoid source blocks and other parity blocks

DistributedRaidFileSystem

- A filter file system implementation
- Allows clients to read “corrupt” source files
  - Catches BlockMissingException, ChecksumException
  - Recreates missing blocks on the fly by using parity
- Does not fix the missing blocks
  - Only allows the reads to succeed

RaidShell

- Administrator tool
- Recover blocks
  - Reconstruct missing blocks
  - Send reconstructed block to a data node
- Raid FSCK
  - Report corrupt files that cannot be fixed by raid
- Handy tool as a last resort to fix blocks

Deployment

- Single configuration file “raid.xml”
  - Specifies file patterns to RAID
- In HDFS config file
  - Specify raid.xml location
  - Specify location of parity files (default: /raid)
  - Specify FileSystem, BlockPlacementPolicy
- Starting RaidNode
  - start-raidnode.sh, stop-raidnode.sh
- http://wiki.apache.org/hadoop/HDFS-RAID

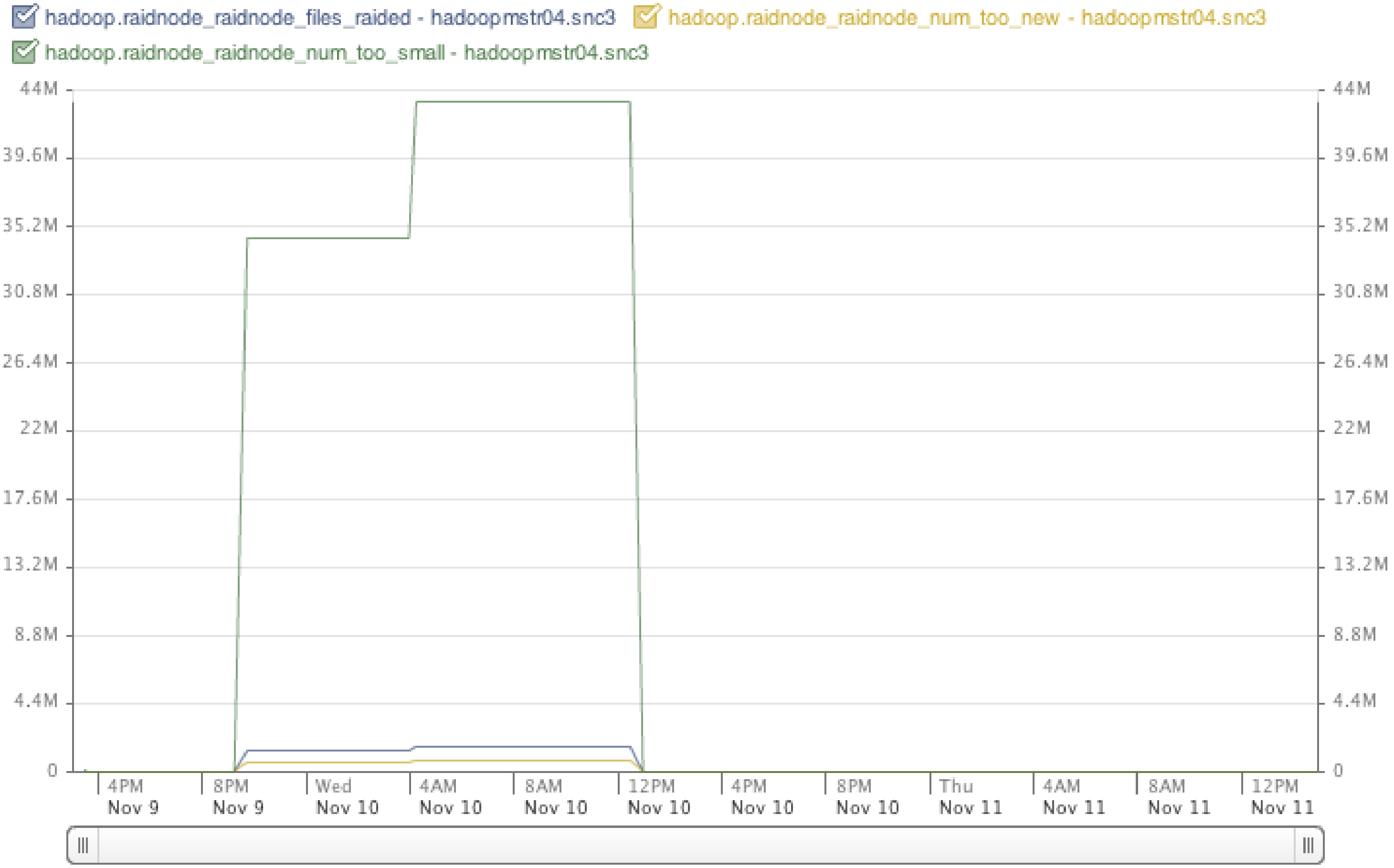
Questions?

http://wiki.apache.org/hadoop/HDFS-RAID

Limitations

- RAID needs file with 3 or more blocks
  - Otherwise parity blocks negate space saving
  - Need to HAR small source files
- Replication of 1 reduces locality for MR jobs
  - Replication of 2 is not too bad
- Its very difficult to manage block placement of Parity HAR blocks

File Stats


☑ hadoop.raidnode_raidnode_files_raided - hadoopmstr04.snc3
☑ hadoop.raidnode_raidnode_num_too_new - hadoopmstr04.snc3
☑ hadoop.raidnode_raidnode_num_too_small - hadoopmstr04.snc3
44M
39.6M
35.2M
30.8M
26.4M
22M
17.6M
13.2M
8.8M
4.4M
0
4PM Nov 9
8PM Nov 9
Wed Nov 10
4AM Nov 10
8AM Nov 10
12PM Nov 10
4PM Nov 10
8PM Nov 10
Thu Nov 11
4AM Nov 11
8AM Nov 11
12PM Nov 11