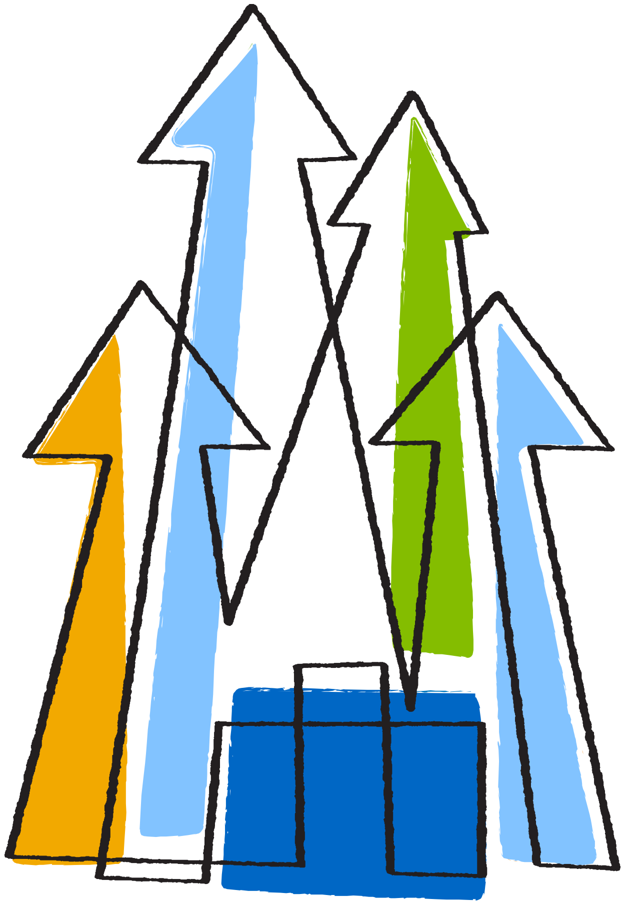
Go further, faster®

Deduplication and Incremental Accelleration in Bacula with NetApp Technologies

Peter Buschman
EMEA PS Consultant

September 25th, 2012

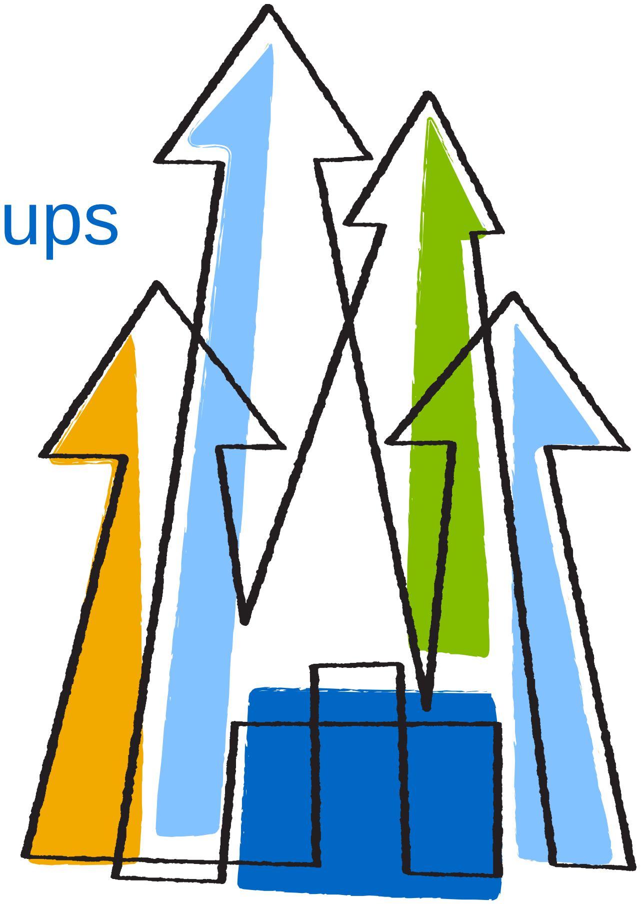


NetApp and Bacula Systems

- Bacula Systems became a NetApp Developer Partner in 2007
- Solutions developed using NetApp SDK and standardized protocols and interfaces
- Solutions that will be presented are not NetApp products
- Bacula Systems has the ability to open tickets with NetApp support should bugs be found in NetApp interfaces
- End-user support is available only from Bacula Systems

Go further, faster®

Accelerated Incremental Backups in High File Count NetApp Environments

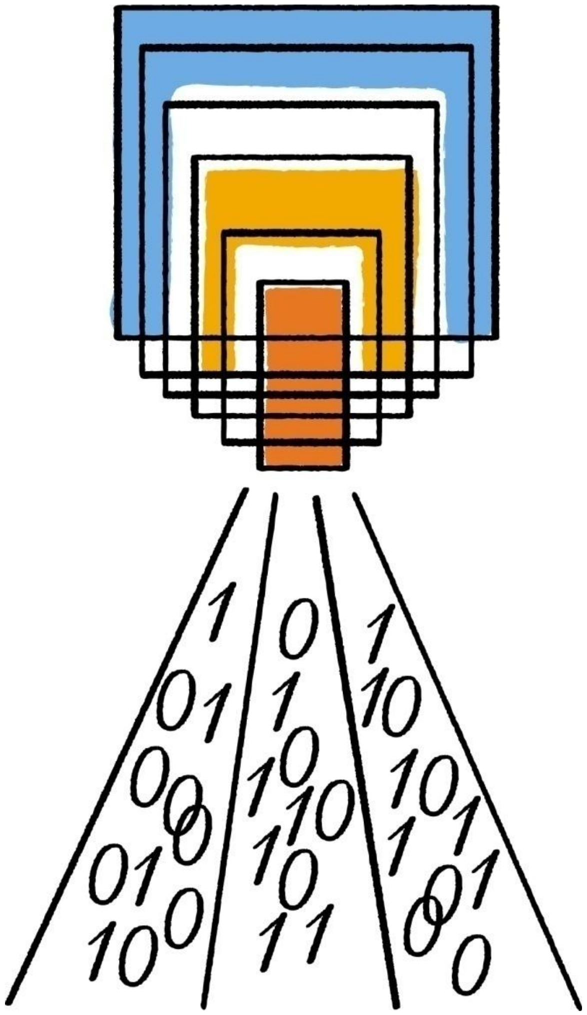


Peter Buschman

EMEA PS Consultant

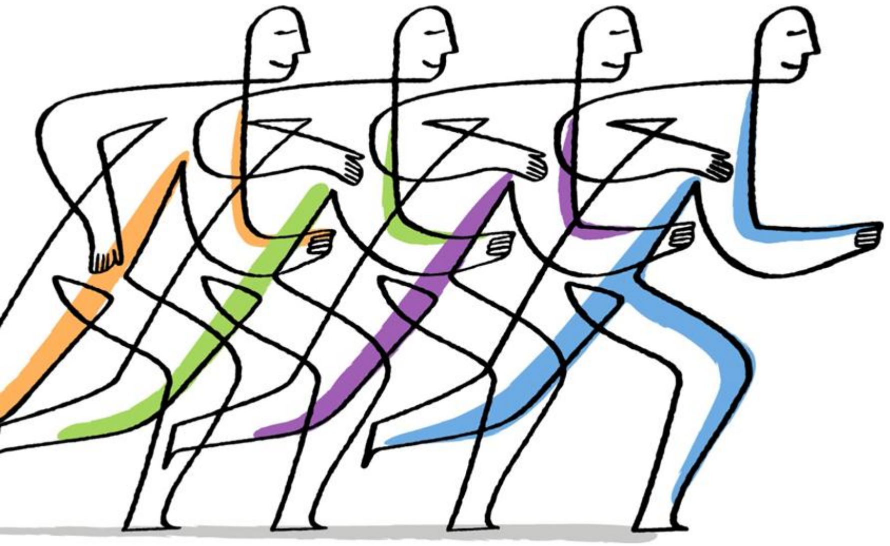
NetApp

The Problem with Lots of Files



- As the number of files increases, so does the time to complete the backup regardless of how much data has actually changed
- Problem is due to the time it takes the backup client to scan the filesystem in order to identify files changed since the previous backup
- The more files, the longer the scan takes
- On filesystems with 10s or 100s of millions of files, the scan phase can take hours
- NDMP Dump backups are particularly susceptible to this
- 24 hour backup times not uncommon even when only a few MB have actually changed
- Fileservers with large numbers of users and home directories are particularly likely to have this problem

Accellerating NetApp Backups



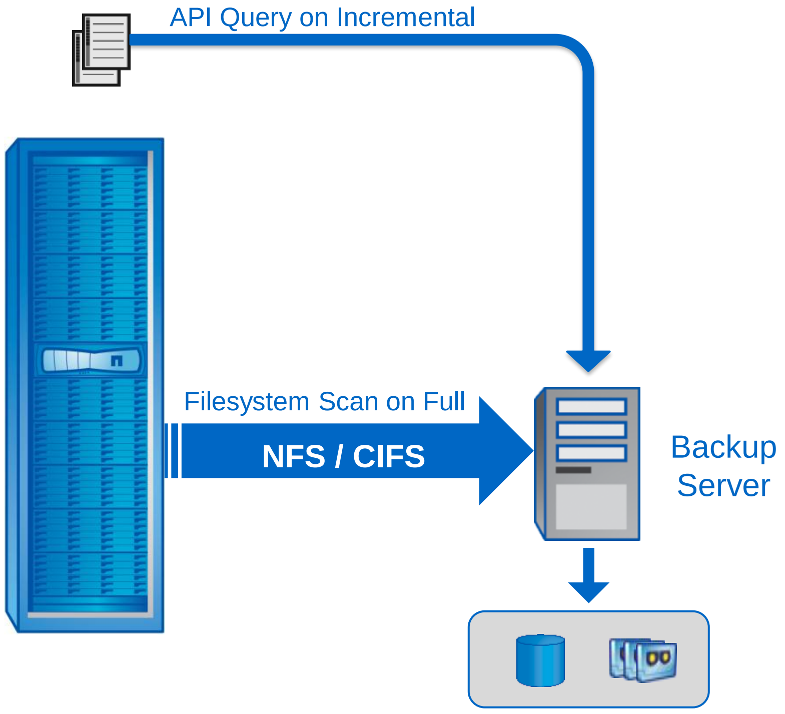
NDMP SMTAPE

- No filesystem scan phase
- Image backup of NetApp volumes
- Many times faster than NDMP Dump
- No single file restore
- Limited backward compatibility between Data ONTAP releases

Bacula Incremental Accellerator for NetApp

- Eliminates scan phase for incremental backups
- Maintains single file restore capability
- Data transfer via NFS / CIFS instead of NDMP

How Incremental Accellerator Works



Full Backup

- Snapshot taken when backup runs
- Normal filesystem scan is performed
- Not significantly different from a typical filesystem backup

Incremental Backup

- Relies on knowledge of the snapshot taken during the full backup
- Queries NetApp filer directly to get the list of files changed since the previous full
- Incremental speed-up dependent on how long the scan phase would have been

Subsequent Incrementals

- Reference snapshot taken during previous incremental
- Changed file list generated between incrementals same as between incremental and full.

Recommendations

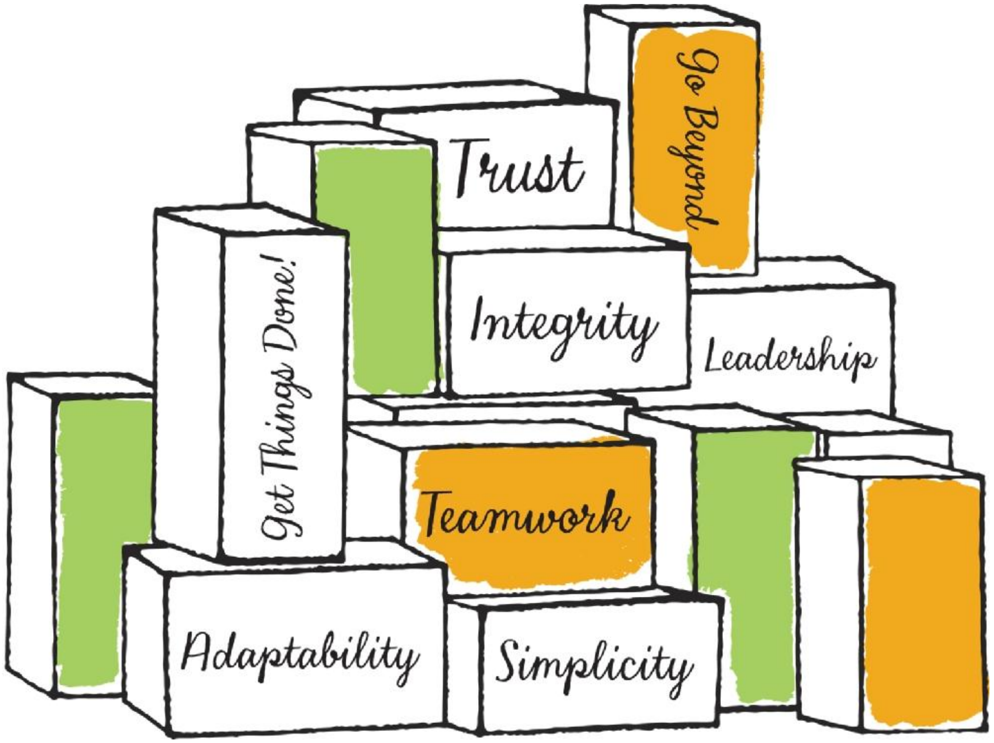
- Do not do incremental forever
  - Incremental Accellerator can miss some file change situations.
    - Moved directories are a problem
- Run accellerated incrementals during the week to meet backup windows and keep backups from running into the business day
- Run traditional full backups on the weekend or over holidays when there is time to accommodate a long filesystem scan
- Consider running regular NDMP SMTAPE backups for DR backups of whole volumes

Q&A

NetApp®

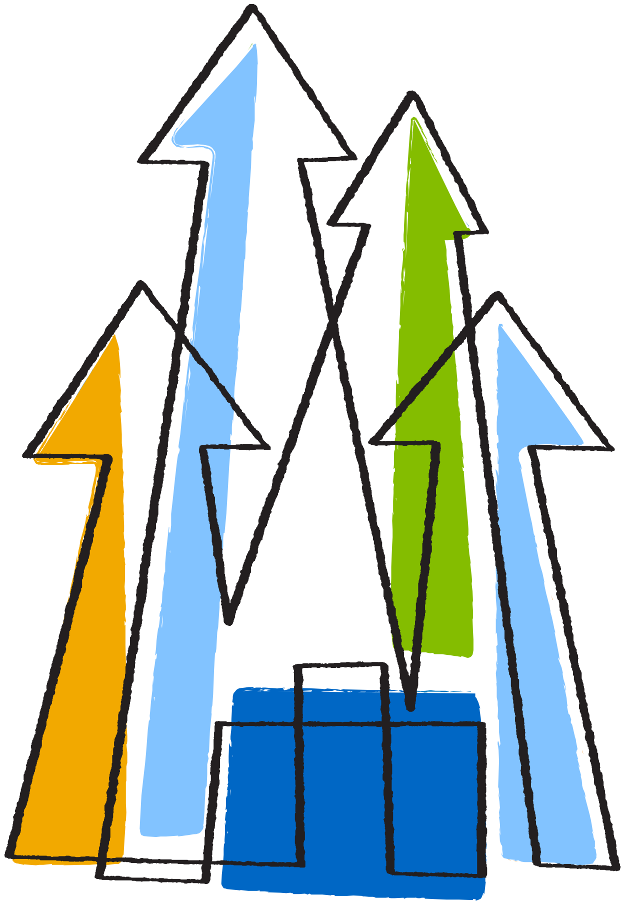
Thank you

peter.buschman@netapp.com
+49 151 527 555 24



Go further, faster®

NetApp Deduplication and Bacula Block Aligned Volumes



Peter Buschman

EMEA PS Consultant

NetApp

SNIA Definitions – Deduplication

data deduplication

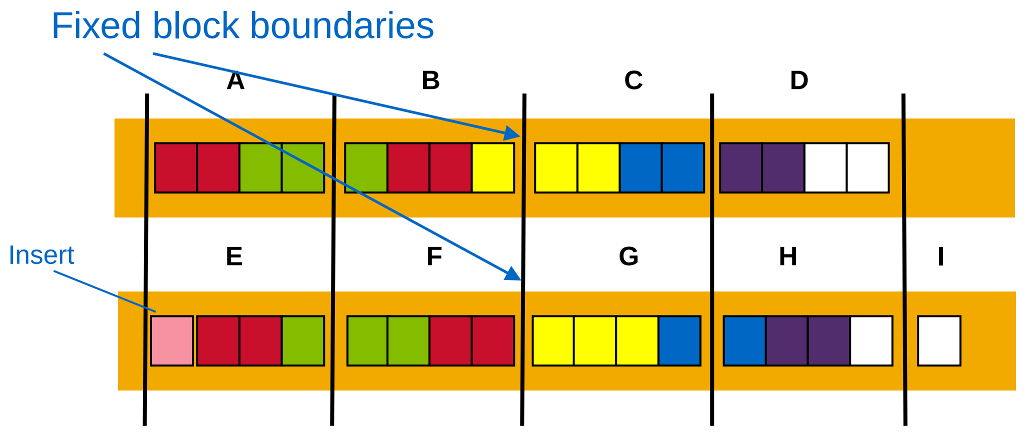
- The replacement of multiple copies of data - at variable levels of granularity - with references to a shared copy in order to save storage space and/or bandwidth

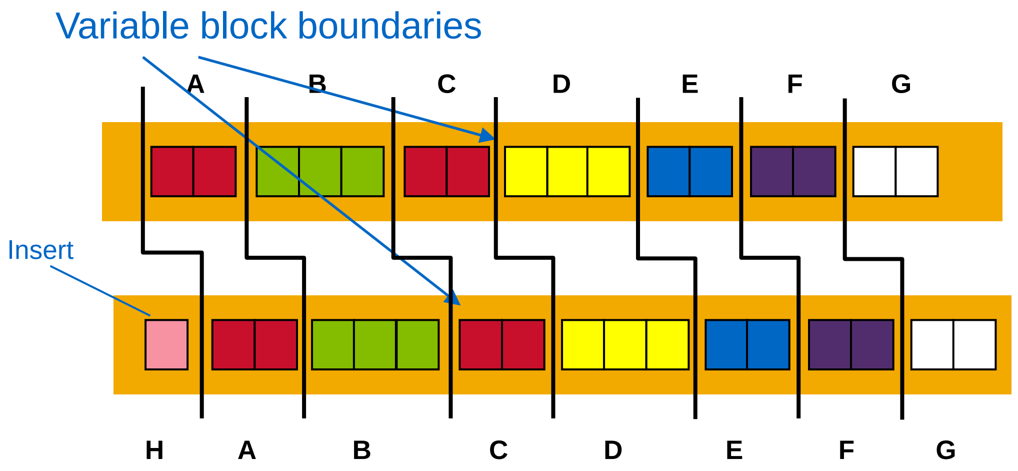
data deduplication ratio

- A space reduction ratio that only includes the effects of data deduplication
- Calculation of original Size / optimized size
  - (Optimized size includes deduplication and compression)
- Expressed as either
  - a ratio i.e. 20:1
  - a percentage i.e. 95% reduction

http://www.snia.org/education/dictionary/d/

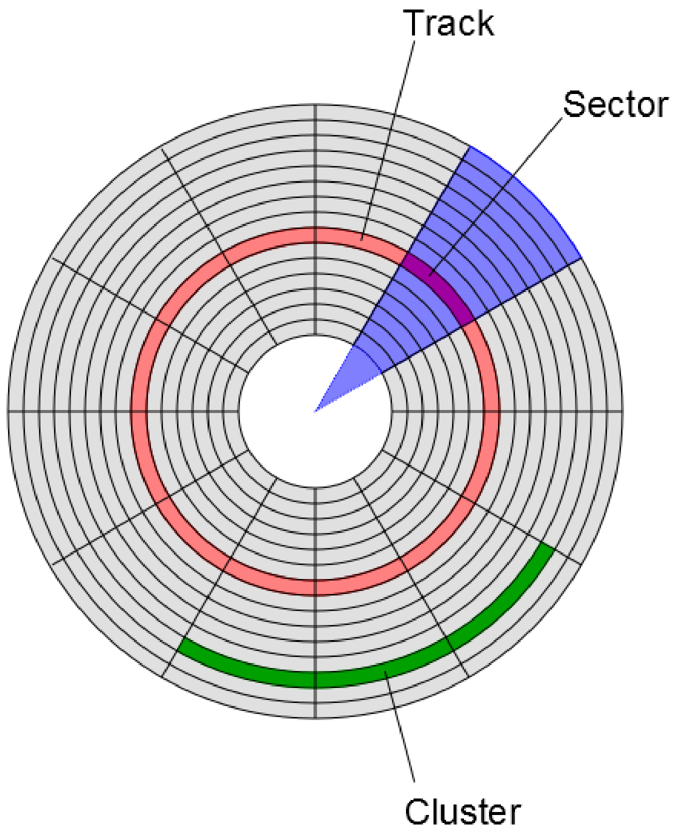
Deduplication Models





- Fixed Block
  - Data is scanned for duplicates on fixed boundaries from the beginning of the file.
  - Block boundaries are typically 4K but can be higher.
  - Generally quite fast at data "re-hydration" even as total data amount grows
  - But, insert 1 additional bit of data and all subsequent blocks change
- Variable Block
  - Data is scanned for duplicates on variable (rolling) boundaries from the beginning of the file. All possible boundaries are examined.
  - Can Identify identical "chunks" of data
  - Chunks can be megabytes, gigabytes, or even larger in size.
  - Can be slower at data re-hydration as total data amount grows.
  - But, insert 1 additional bit of data – all subsequent "chunks" remain the same

Where Fixed Blocks Come From



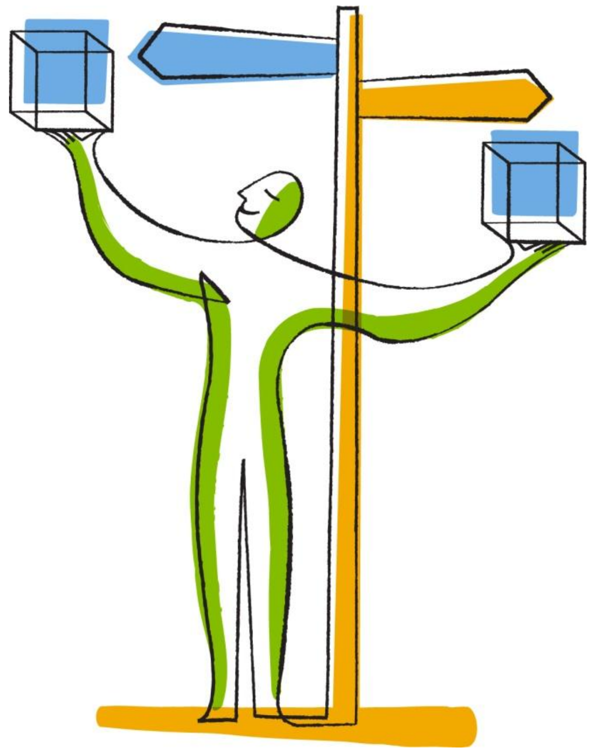
Disk Geometry

- Disks are divided into tracks and sectors
- A sector is the smallest unit of IO
  - Historically 512 Bytes
  - Increasingly 4K

Filesystem Design

- Group sectors into clusters for performance
  - Sequential sectors can be read or written with a single seek operation
- Cluster size can vary but is typically 4K
- When we refer to a filesystem‘s block size, we really mean the cluster size
- Files will consume a full block of space even if they are smaller than the block size.

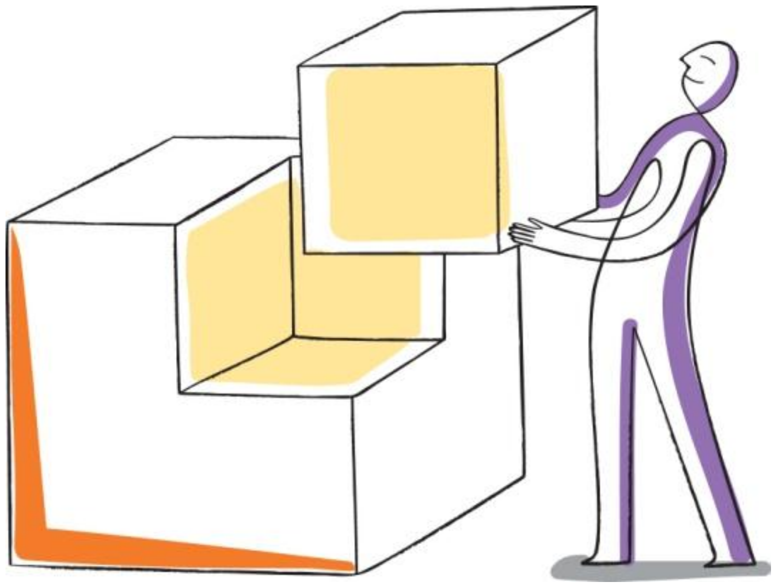
How Backup Software Stores Data



Traditional Backup

- Concepts date back to original Unix dump and tar utilities
- Still the core of most backup frameworks
- Full, Incremental, and Differential Backup levels
- Files are sent from client to backup server over a network
- Files that have changed between backups are re-sent to the backup server in their entirety
- *Originally designed for tape*
- Most backup frameworks have long since added support for disk filesystems as backup storage

Why Fixed Blocks Matter in Backup



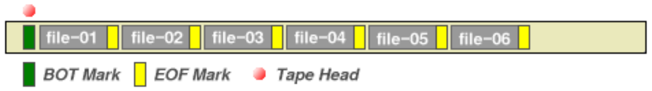
Data Organization

- Data is organized into files
- Files are allocated into blocks by the filesystem
- Changes affect only the blocks encompassing the change, not all blocks in the file
  - *Except for some Microsoft applications that re-write the entire file after a change*

Traditional Backup Software

- Identifies changed files by modification time
  - *Archive bit historically used in Windows but increasingly deprecated*
- Has no visibility into subfile changes at the block level.
- Modified files are transferred and stored in their entirety even if just a single block has changed.

Backup to Tape



Tape Media

- The original removable storage
- Sequential rather than random access
- Has a fixed record size similar to a filesystem's cluster size
- Files separated from each other by end-of-file (EOF) markers
- Historically cheaper than disk for long-term storage
- Most major backup frameworks had their backup formats designed for writing to tape
- Rather than write each file backed-up to the tape separately, backup software wraps files into "containers" that are written to the tape as a single file. *In the process, the block boundaries between files disappear*

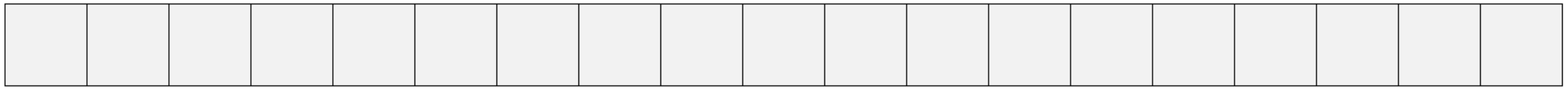
Backup to Disk

Disk Media

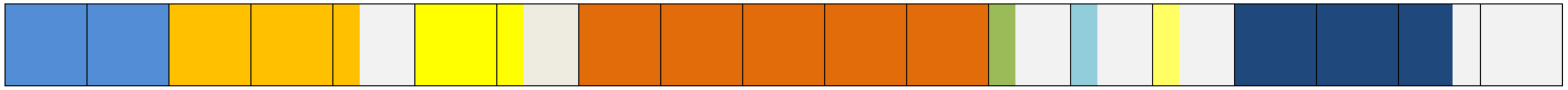
- Hard disks did not become cheap enough to use for backup until the late 1990s / early 2000s
- Backup frameworks did not have to deal with backing-up to disk before then. There was no demand for it
- When the market demanded this feature, vendors did the obvious (and easiest) thing
- They used the same backup format they used on tape and wrote it to a file on a filesystem
- To this day, most backup software treats backups on disk as if they were sequential tape media rather than files on a random access device
- *This is why deduplication of backups is so hard*

How Backup affects Block Boundaries

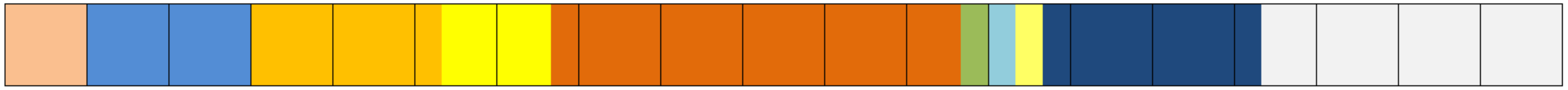
Empty Filesystem



Filesystem with Files



Backup to Traditional Tape or Disk Volume

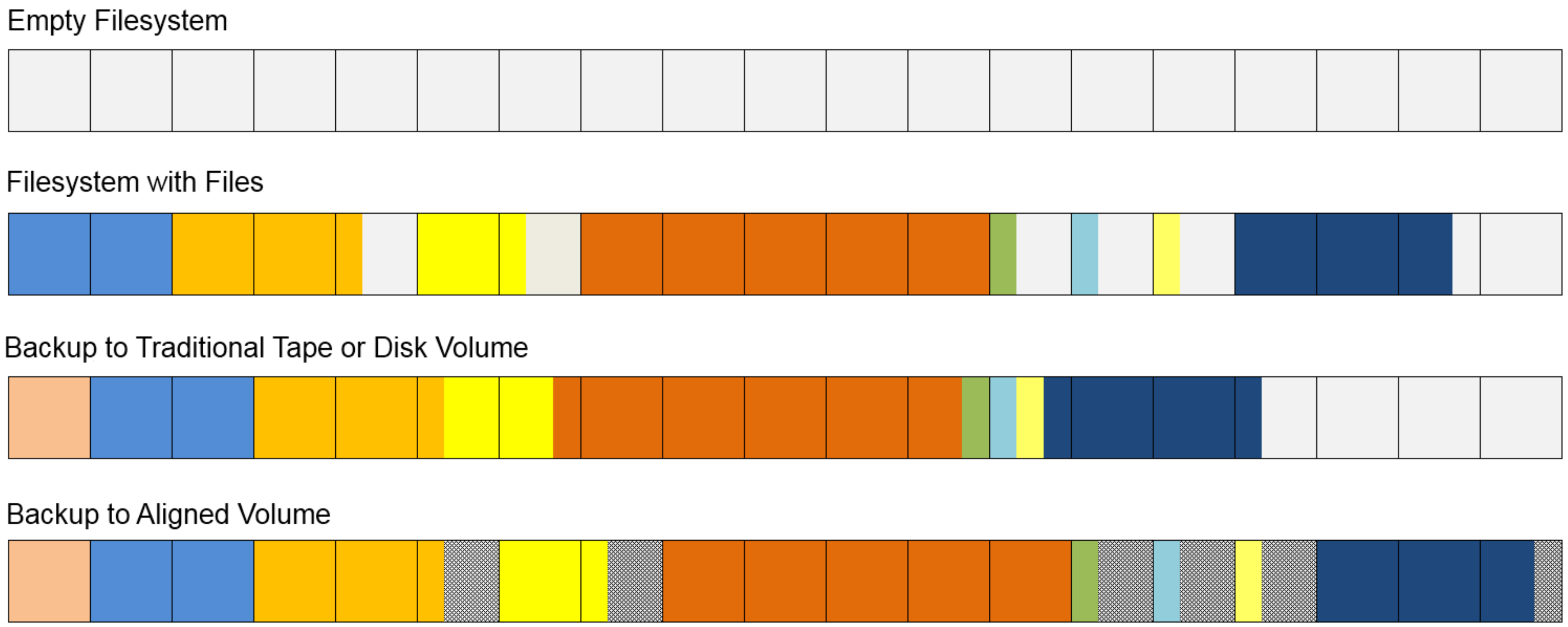


When backup software writes to tape, it only writes the data from the file. It doesn‘t care what the filesystem‘s block-size is. To avoid wasting space on tape, it stores one file after the other as efficiently as possible.

This is great for storing data on tape but it destroys the block boundaries between files. When backup software writes to a disk-based volume on a filesystem using the same format it uses for tape, the data is completely misaligned.

This is why backup data doesn‘t de-duplicate well using fixed-block deduplication. When blocks deduplicate at all, it is only a coincidence.

Bacula's Dedupe - Block Aligned Volumes



Bacula's answer to deduplication is quite simple. If the format it uses won't dedupe well, then change the format. With fixed-block deduplicating filesystems widely available, there is no need for Bacula to implement complex variable-block schemes in its own code. Instead, by simply preserving the block boundaries between files as part of its own volume format, it ensures that, when stored on a fixed-block deduplicating filesystem, the backup data will dedupe. Preserving block boundaries, however, means padding the data. When a file ends short of a block boundary, it will be padded with zeroes until it reaches the block boundary. These zeroes are then discarded during a restore.

The Consequences of Padding

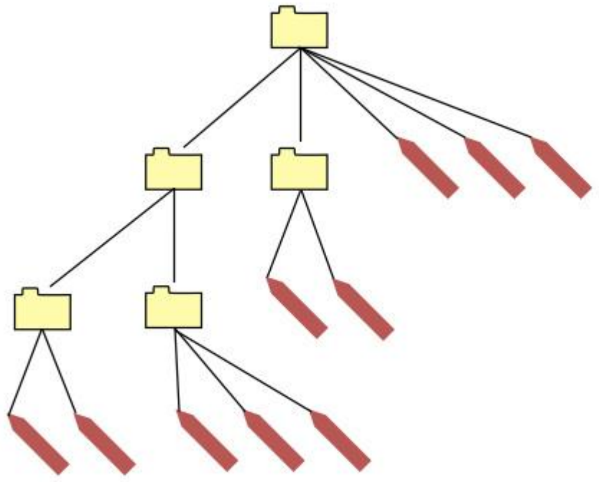
Disadvantages

- Padding with zeroes makes the backup bigger
- This is the opposite of what we are trying to achive
- Compensating for this requires having a filesystem that both compresses and deduplicates
- Such filesystems have not been around for very long
- Padding demands higher throughput on the back-end storage
- Network bandwidth is still consumed transferring whole files during backup and restore

Advantages

- Zero padding compresses to almost nothing
- Any remaining overhead is more than compensated for by the deduplication of data blocks
- With modern CPUs and storage controllers, it is easy to engineer for the additional throughput on the back-end storage
- Network bandwidth is cheap and renewable (the same bandwidth I used last night will be there tonight). Storage isn't (the space I consumed last night won't be available to me tonight)

Fixed Block Deduplicating Filesystems



Commercial

- NetApp Data ONTAP 8
  - Free deduplication and compression licenses
- ZFS
  - Ships with Solaris 10 and 11
  - Also used by Sun / Oracle storage appliances

Open Source

- ZFS
  - CDDL License (uncertain future)
- LessFS
  - Tunable block-size and compression algorithms

Many others but not yet suitable...

- OPENDEDUP
  - Lacks compression
- BTRFS
  - Has potential
  - Deduplication planned but still not implemented

Know Your Data

Deduplicates well

- Normal Files
  - Block alignment captures subfile changes
- NDMP SMTAPE backups
  - Native 4K block-aligned format

Does not deduplicate well

- Compressed Files
  - Small changes affect many blocks in the file
- Encrypted Files
  - Small changes affect all blocks in the file.
- NDMP Dump backups
  - Block boundaries removed during dump

Exception

- Duplicate Files
  - Identical files will dedupe against each other regardless of compression or encryption (even if they are backed-up from different systems.)

Recommendations

- Disable compression and encryption
  - Let the filesystem do the compression.
- If not using Data ONTAP for backup storage
  - Buy the best CPUs you can afford. Clock rate matters since your server will be both compressing and calculating hashes on incoming data
  - Install as much RAM as you can
  - Use a fast SSD for the checksum catalog
  - Use storage that is faster at sequential IO than your ingest rate (NetApp E-Series is good at this)
- If using a NetApp filer for backup storage
  - Remember to apply for your free deduplication (a-sis) and compression licenses. They do not ship automatically.
  - Pay attention to published size limits for dedupe-enabled volumes
  - Remember that deduplication on NetApp is post-process, not in-line You will not see the results immediately

Q&A

NetApp®

Thank you

peter.buschman@netapp.com
+49 151 527 555 24