vms Software

VMS File System Update

- Andy Goldstein

September 26, 2016

vms Software

VMS File System Update

This information contains forward looking statements and is provided solely for your convenience. While the information herein is based on our current best estimates, such information is subject to change without notice.

```
; Bliss-32 10.1-416     Monday 21-AUG-1978 23:06:28     DBB3:[F11A.SRC]WITURN.B32;11                    Page 1
;
;       0001    MODULE WITURN (
;       0002                    LANGUAGE (BLISS32),
;       0003                    IDENT = 'A0007C'
;       0004                    ) =
;       0005    BEGIN
;       0006
;       0007    !
;       0008    ! Copyright (c) 1977
;       0009    ! Digital Equipment Corporation, Maynard, Massachusetts 01754
;       0010    !
;       0011    ! This software is furnished  under a license for use only on a single
;       0012    ! computer  system and  may be  copied only with  the inclusion of the
;       0013    ! above copyright notice.  This software, or any other copies thereof,
;       0014    ! may not be provided or  otherwise made available to any other person
;       0015    ! except for use on such system and to one who agrees to these license
;       0016    ! terms.  Title to and  ownership of the  software  shall at all times
;       0017    ! remain in DEC.
;       0018    !
;       0019    ! The information in this software is subject to change without notice
;       0020    ! and should  not be construed  as a commitment  by Digital  Equipment
;       0021    ! Corporation.
;       0022    !
;       0023    ! DEC assumes  no  responsibility  for  the use or  reliability of its
;       0024    ! software on equipment which is not supplied by DEC.
;       0025
;       0026    !++
;       0027    !
;       0028    ! FACILITY:  F11ACP Structure Level 1
;       0029    !
;       0030    ! ABSTRACT:
;       0031    !
;       0032    !       This module generates a window mapping t
;       0033    !       the supplied file header.
;       0034    !
;       0035    ! ENVIRONMENT:
;       0036    !
;       0037    !       STARLET operating system, includi
;       0038    !       and internal exec routines.
;       0039    !
;       0040    !--
;       0041    !
;       0042    !
;       0043    ! AUTHOR:  Andrew C. Goldstein, CREATIO
;       0044    !
;       0045    ! REVISION HISTORY:
;       0046    !
;       0047    !   Andrew C. Goldstein, 17-Mar-1977  16:
;       0048    !   X0002 - Add system interlock while tur
;       0049    !
;       0050    !   Andrew C. Goldstein, 26-Apr-1977  11:36
;       0051    !   X0003 - Move code to locked down PSECT
;       0052    !
;       0053    !   Andrew C. Goldstein, 21-Jul-1977  15:17
;       0054    !   X0004 - Add multi-header code
;       0055    !
```

7-Dec-1976

The file structure is older than you think

A big disk (back then)

Modern Storage Scale

- 200MB in 1976
- 8TB in 2016: 40,000x larger
  - Double every ~2.6 years, or
  - 0.38 bits/year growth
- The 32 bit LBN ran out of bits in 2009
- We can expect similar, if not faster, growth in the future
- Storage demand rises to meet capacity

Other Scale Issues

- Number of files on a volume
- Number of files in a directory
  - File names created by software vs humans
  - Non-random file name patterns
  - Square law delete performance
- Space allocation mechanisms
- Volume rebuild time…

Performance vs Safety

- Existing file system: “careful write”
  - Slow but safe
  - Space caching requires rebuild after crash
- Old Unix systems: “lazy write”
  - Fast, but indeterminate results after crash
  - fsck mandatory after crash
- Most present day systems: lazy write plus write-ahead logging

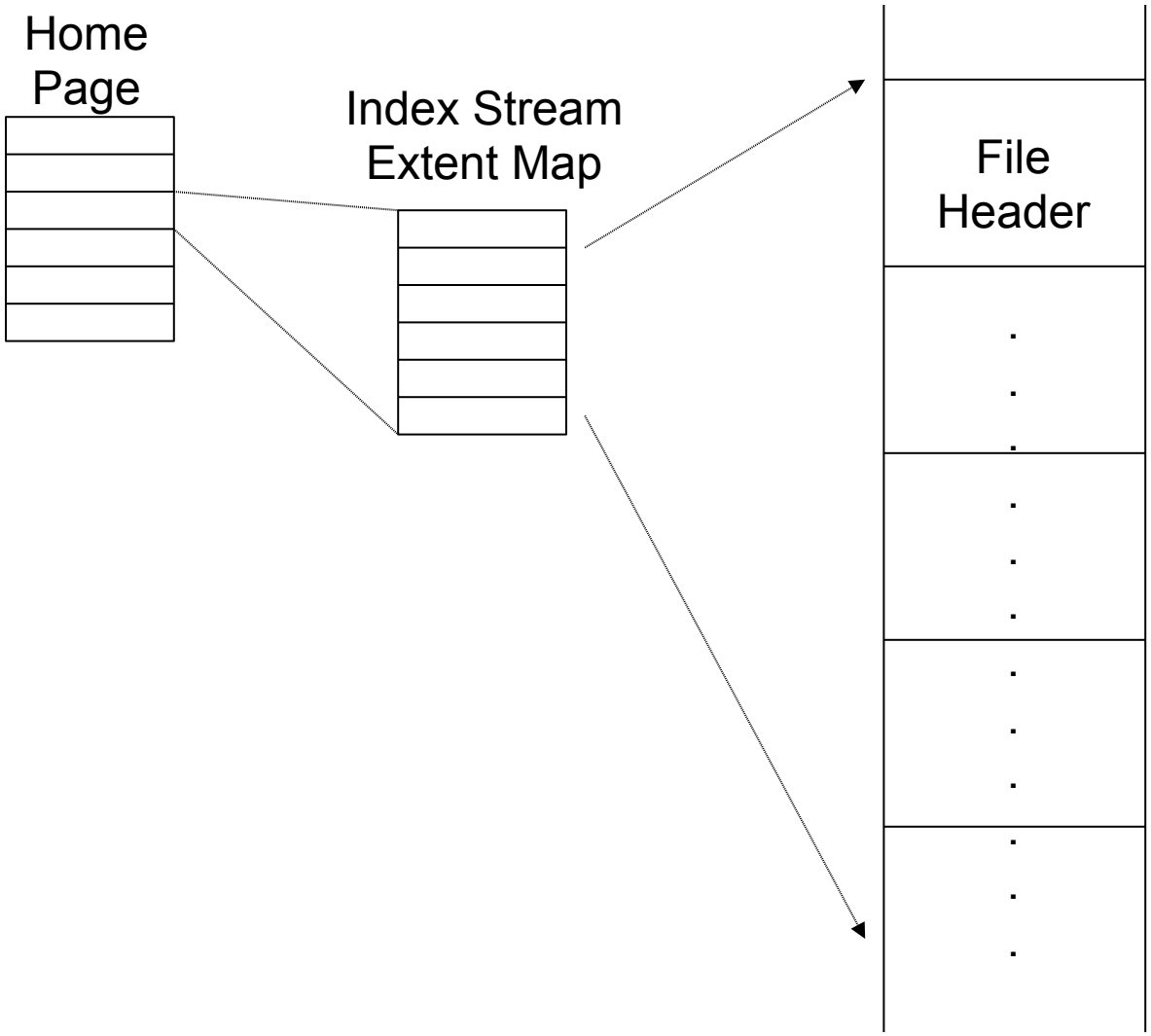
64 Bit LBN

- Promotion of disk size and LBN fields to 64 bits
- Clean up some overlays and other kluges
- Support in I/O exec and most drivers
- Disk geometry is going away
  - Home block placement by fixed search delta only
  - Geometry data is entirely, rather than mostly, fictional
- Rollout:
  - As much infrastructure as possible in V8.5
  - Complete in V9

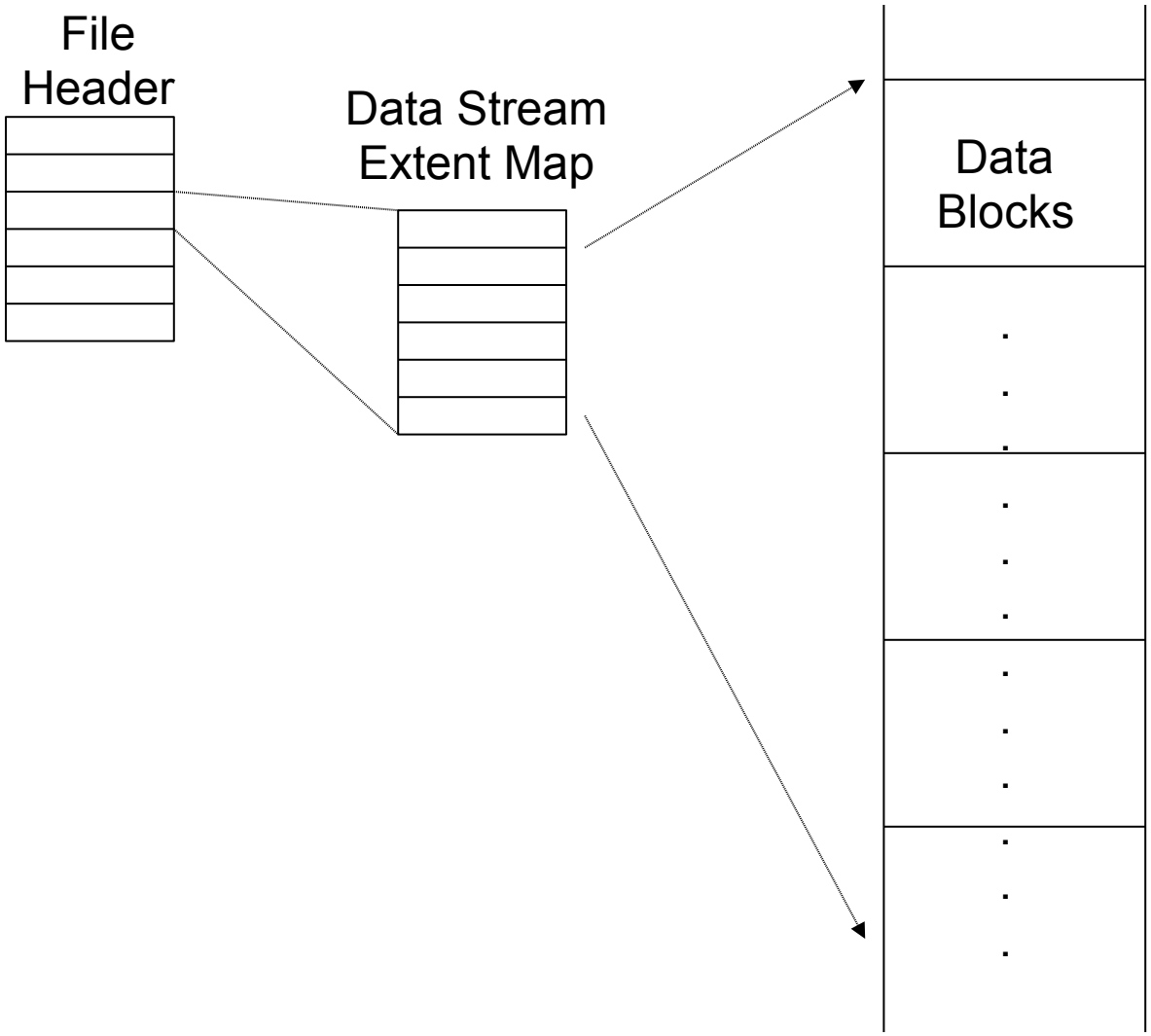
VAFS = VMS Advanced File System

- 64 bit architecture
- Write-ahead logging
- Conceptually compatible with existing file system
  - In-place update of metadata
  - File headers in an index file
  - Directories are files
- Fully compatible API
  - 99% of applications run without modification
- Coexists with existing file system

VAFS Volume Structure



VAFS File Structure

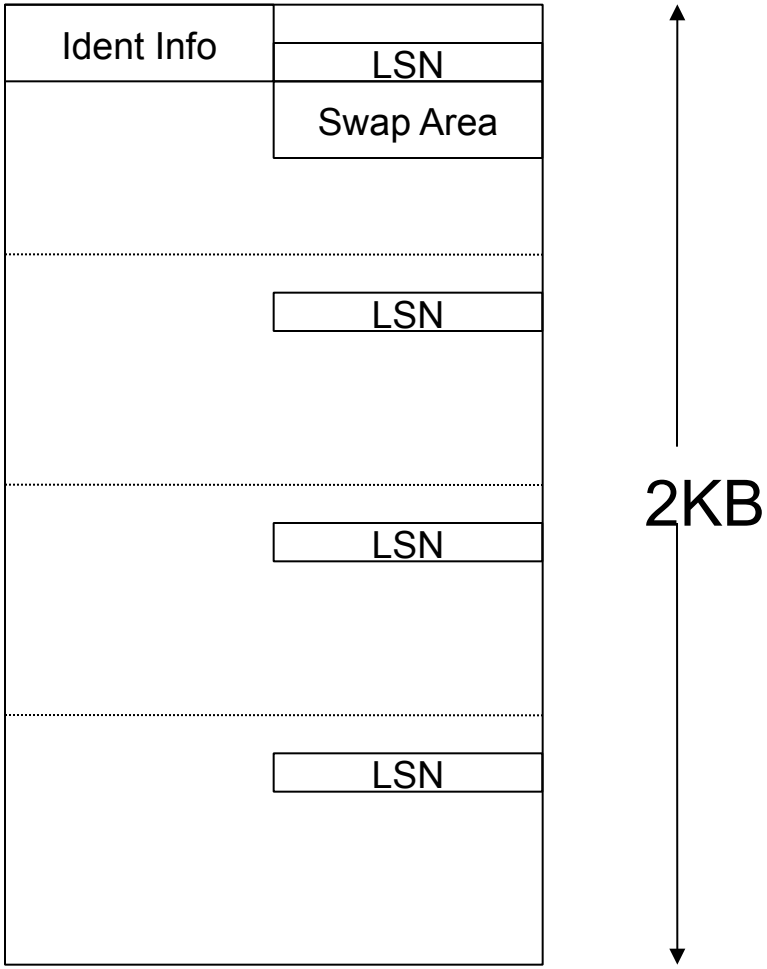


VAFS Internal Architecture

- Block signatures and check codes for high integrity
- Variable size index & directory blocks
- Generalized byte streams
- Extensible TLV structure for file metadata
- B-trees for directories and extent maps
- Write-ahead log and write-behind cache
- Transaction semantics

VAFS Architecture

Disk Page

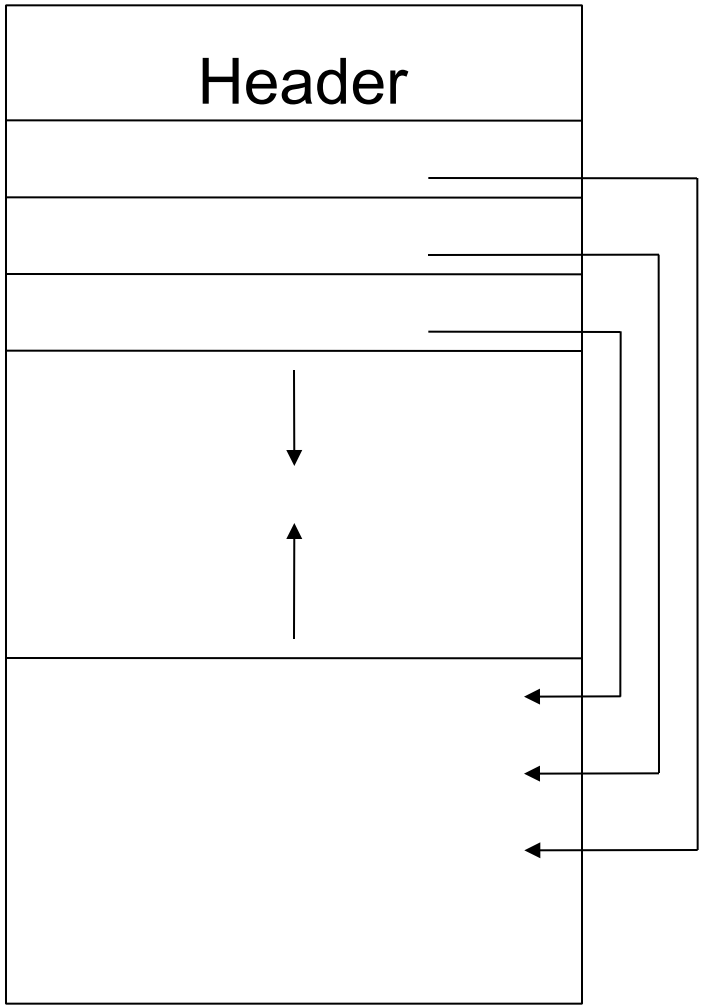


List Page

Attribute

Value

Pairs



Fixed

Size Key

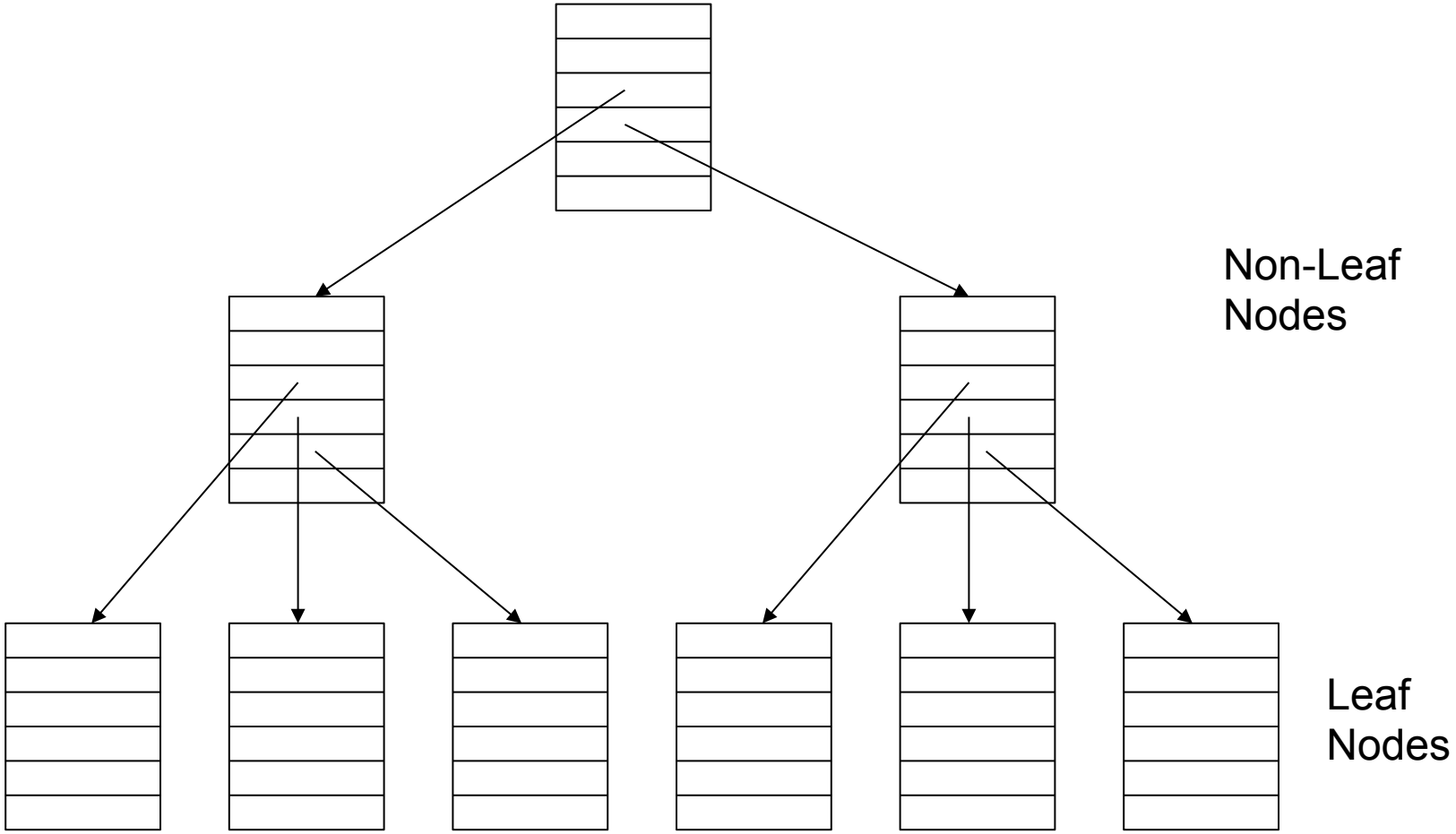
Prefix

Key

Remainder

& Value

Tree



Tree Nodes

- Leaf nodes: data content
  - Key = name of datum
  - Value = data
- Non-leaf nodes: index
  - Key = last key of page
  - Value = disk address of page
- Extent map nodes
  - Key = stream offset of extent end
  - Value = length & disk address of extent

Stream

- Just a sequence of bytes
- Direct stream
  - Stored as an atomic list page value
- Mapped stream
  - Stored in disk blocks
  - Extent map is a tree
  - Tree root is the list page value

Directory

- Special file type
- Directory content is a special file attribute, stored as a tree
- Directory entry
  - Key = file name, normalized Unicode + case flags
  - Value = file ID

Bitmap

- Used to allocate file IDs and free blocks
- Organized in page-size segments
- Extensible tree structure

Write-Ahead Log

- Physical log: updated pages written as is
  - Simple and fast
- Log is written with each transaction
- LSN and first/last flags identify a complete transaction
- Timer-driven log cleaning
- One log for each cluster node
- Recovery = copy logged pages to home locations

What you get

- 99% API / application compatibility with the existing file system
- Coexistence with the existing file system
- Traditional “shared everything” cluster operation
- Transaction semantics on file operations
- 2x or better create/delete performance
- 9 exabyte ($9 * 10^{18}$) volume size
- 4G files per volume
- Cleaner Unix compatibility
- Modern, maintainable code base

What you don’t get

- Volume sets
- Disk geometry and allocation placement
- Bad block handling
- Files are still limited to 1TB (RMS and its API)
- More than 4G files / volume (API)
- File management utilities (e.g., defraggers) need to be rewritten

Future Opportunities

- More lazy write operation, selected by application or user
- Allocate on write operation
- Small file data embedded in the header
- File size up to 9 exabytes (requires major API changes)
- 2^{48} files per volume (requires API change)
- Larger disk blocks

Rollout

- V8.5
  - 32 bit LBN only
  - No system disk support
  - Possibly lacking quotas and other non-critical features
- V9.0
  - 64 bit LBN
  - System disk (platforms TBD)
  - Feature complete

For more information, please contact us at:

RnD@vmssoftware.com

VMS Software, Inc. • 580 Main Street • Bolton MA 01740 • +1 978 451 0110