Evolving File Model

Gordon Waidhofer

CTO

Traakan, Inc.

gww@traakan.com

File Model

- When I say “file” what jumps to your mind is the “model”
- Different minds, different models
- Present day evidence of divergence and confusion
- Happy news! The time is ripe for convergence

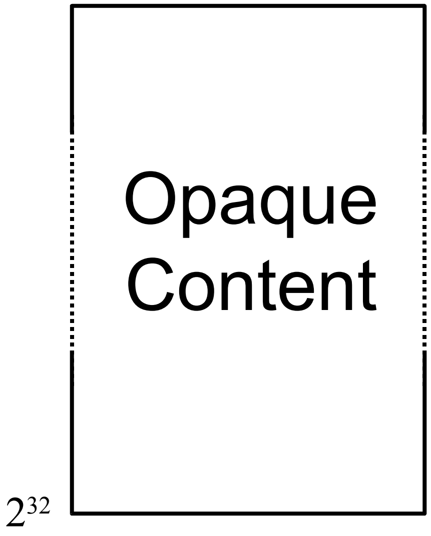
File Model

- The model is the “melody”
- The infrastructure is the “orchestra”
  - API (open, close, read, write, seek, stat)
  - File systems (UFS, FFS, XFS, EXT3, NTFS, etc)
  - File utilities (cp, mv, rm, chmod, setfacl, etc)
  - Archivers (tar, zip, cpio)
  - File networking (ftp, scp, nfs, http, email)
- When the model (melody) changes, the infrastructure (orchestra) must adjust

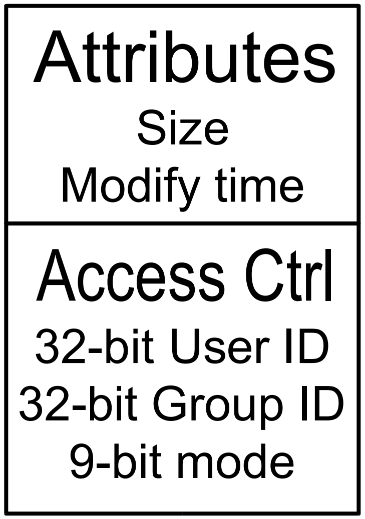
Influences on Evolution

- Driving
  - Security and Regulatory
  - Data Management and Protection
  - Fancy stuff – multimedia, databases
  - Keeping up with the Jones
- Restraining
  - Divergence – APIs, implementation constraints
  - Lagging Interchange – tar, nfs
  - Reluctance of application developers – avoid hassle by avoiding features

File Model Circa 1994

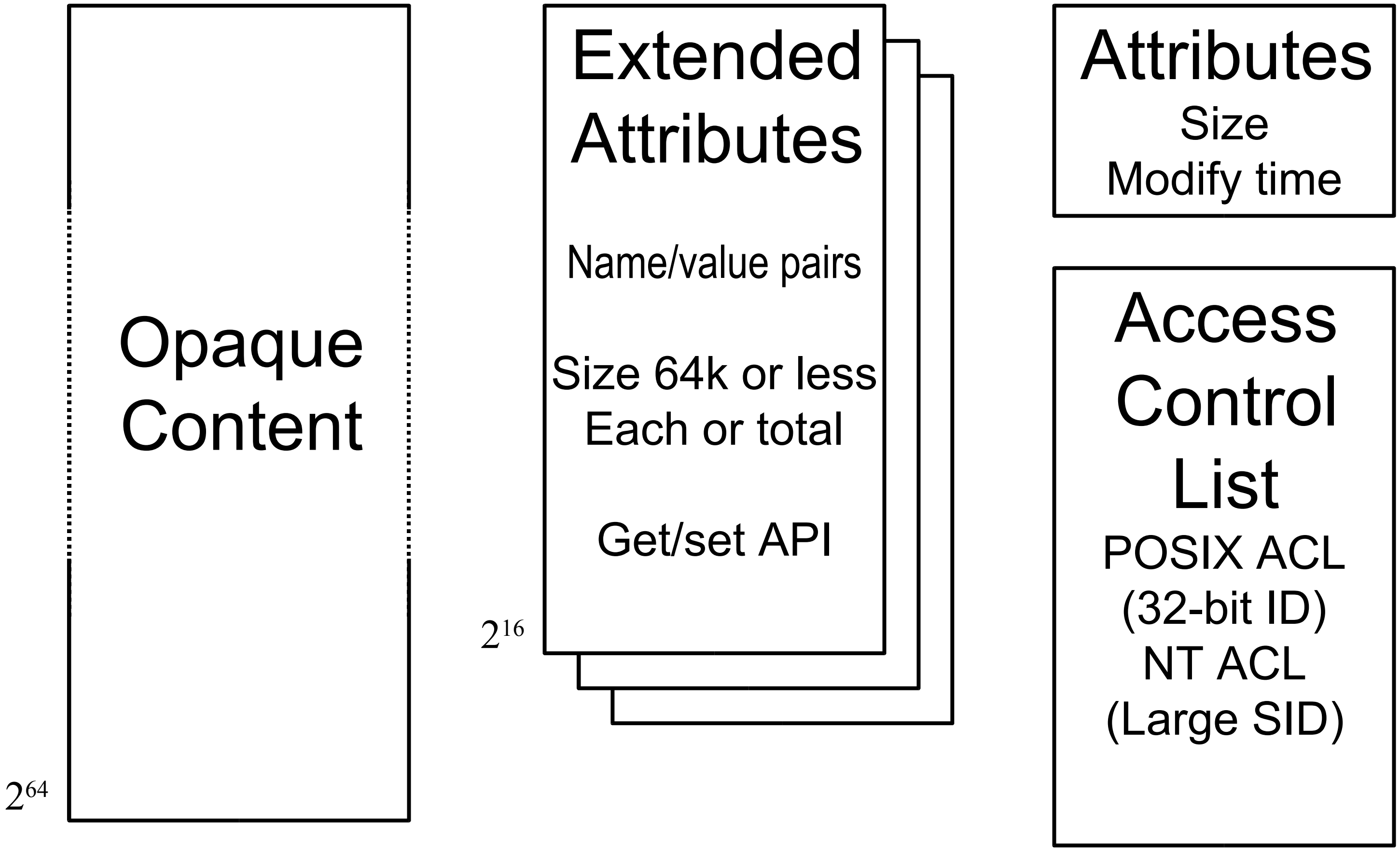


The underlying operating system and file system don’t care what these bits are. The application decides if its an email message, document, image, or whatever.



The underlying operating system and file system do care what these bits are. The application can not be arbitrary about them.

File Model Circa 2004



Opaque Content

2^{64}

Extended Attributes

Name/value pairs

Size 64k or less
Each or total

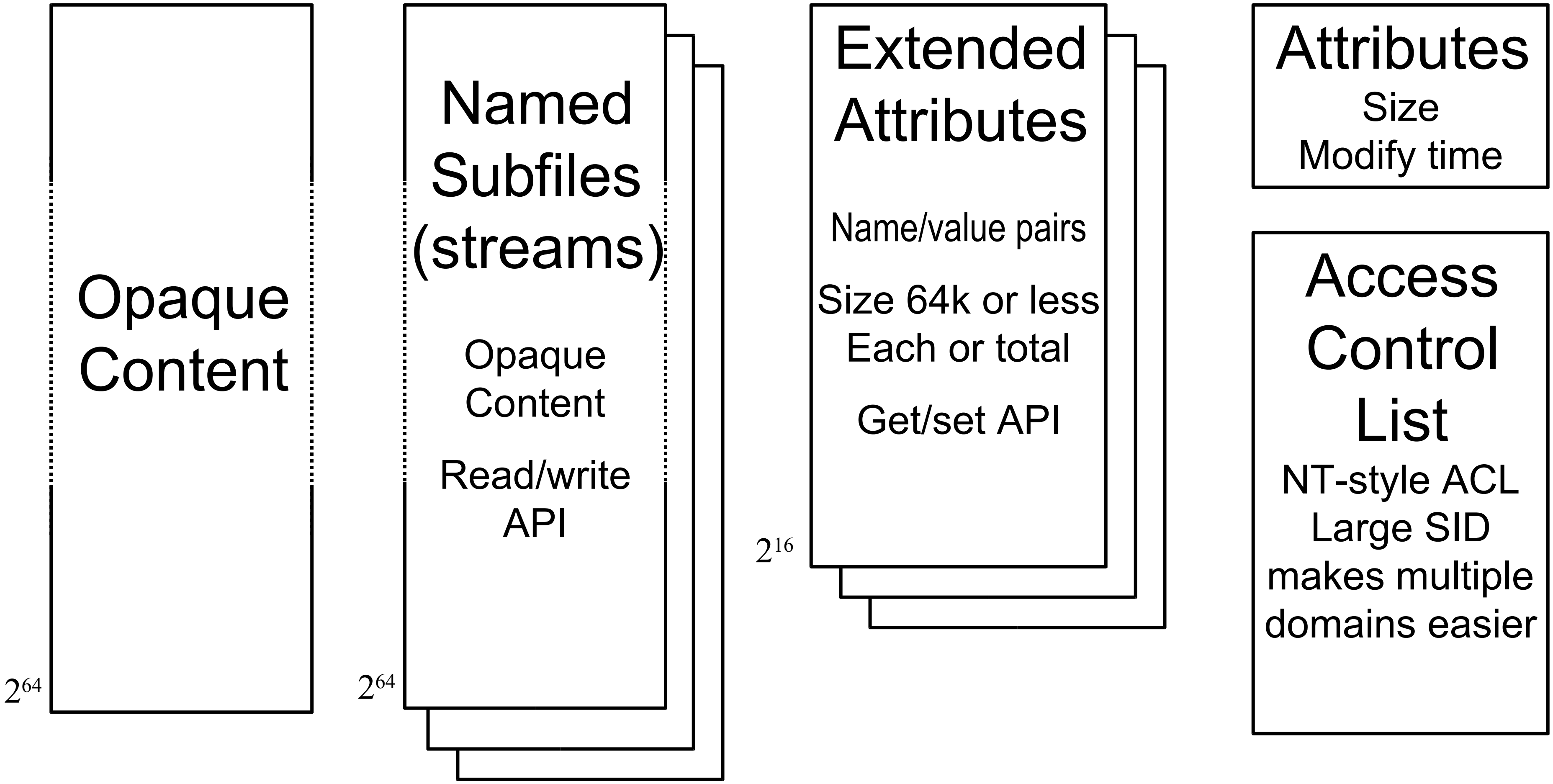
Get/set API

2^{16}

Attributes
Size
Modify time

Access Control List
POSIX ACL
(32-bit ID)
NT ACL
(Large SID)

File Model Circa 2014



Opaque Content

2^{64}

Named Subfiles (streams)

Opaque Content

Read/write API

2^{64}

Extended Attributes

Name/value pairs

Size 64k or less Each or total

Get/set API

2^{16}

Attributes

Size

Modify time

Access Control List

NT-style ACL

Large SID makes multiple domains easier

What’s What

- Opaque content
  - Never meaningful to file infrastructure
- Attribute
  - Describes file (size, modification time)
  - Meaningful to both file infrastructure and application

What’s What

- ACL – Access Control List
  - Fine control of permissions for users or groups
  - Tricky part is user/group identification
  - POSIX ACL – 3-bit permissions (rwx), 32-bit identities, single matching “allow” entry
  - NT ACL – 12-bit permissions, fully qualified identities (SIDs), cumulative matching allow/deny entries
  - NFSv4 – NT-style ACL, identities are user@domain, lots of POSIX/NFSv4 mapping implemented

What's What

- Extended Attribute (EA)
  - OS/2 HPFS contemplated Extended Attributes to enable extensible file system
  - Textually named small data structure (name/value)
  - API: Get/set (no seek or append)
  - May or may not be meaningful to system (system/user)
  - Used on some systems to store ACLs
  - Often considered poor man's subfile

What’s What

- Named subfile (stream)
  - Content is opaque to file system
  - Ordinary file names, arbitrary size
  - API: Read/write, seek, append
  - Example: Macintosh forks – resource and data
  - Example: Database with multiple indexes (by name, by address, by account)
  - Example: Video with different soundtracks

What’s What

- Confusion about EAs and subfiles
  - A name and some bits
    - So they superficially look the same
    - They are not!
  - Extended Attribute or subfile?
    - Icons
    - Summaries
    - Thumbnails
  - Crux of correctable divergence
    - We’ll explore this

Lightweight Survey of File Systems

File Systems

- Let’s look at some file systems
  - How prepared are they for evolution, new features?
  - How are they doing things?
  - What are the capacities and other implied constraints?
  - Any nifty ideas?
- Could be examples of how others might prepare their file systems for evolution

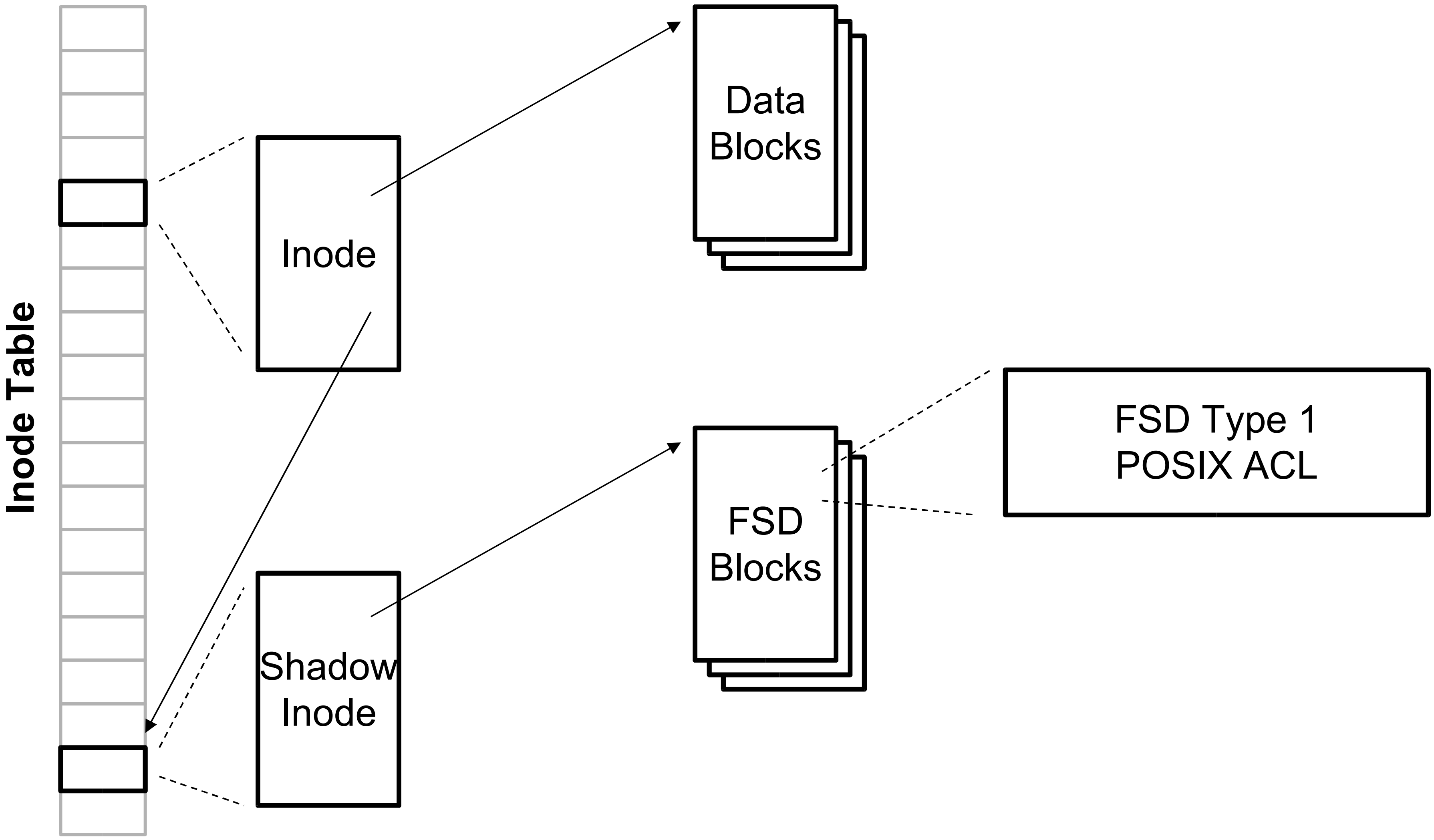
Solaris Files

- POSIX ACLs
  - Acl(2)
  - Single ACL, inherit (default) bit (ala NT)
  - Stored as FSD under shadow inode
- Extended Attributes
  - FSD (File System Data?), variable length item under shadow inode, name is small integer
  - No user EAs, no get/set API
  - Quite extensible

Solaris Files

- Named subfiles (streams)
  - Openat(2), attropen(3), read()/write()
  - Great mechanism, unfortunate name
  - Names and semantics readily pair with NFSv4 OPENATTR
  - Addition of tar type ‘E’ records
  - Addition of -@ to common utilities

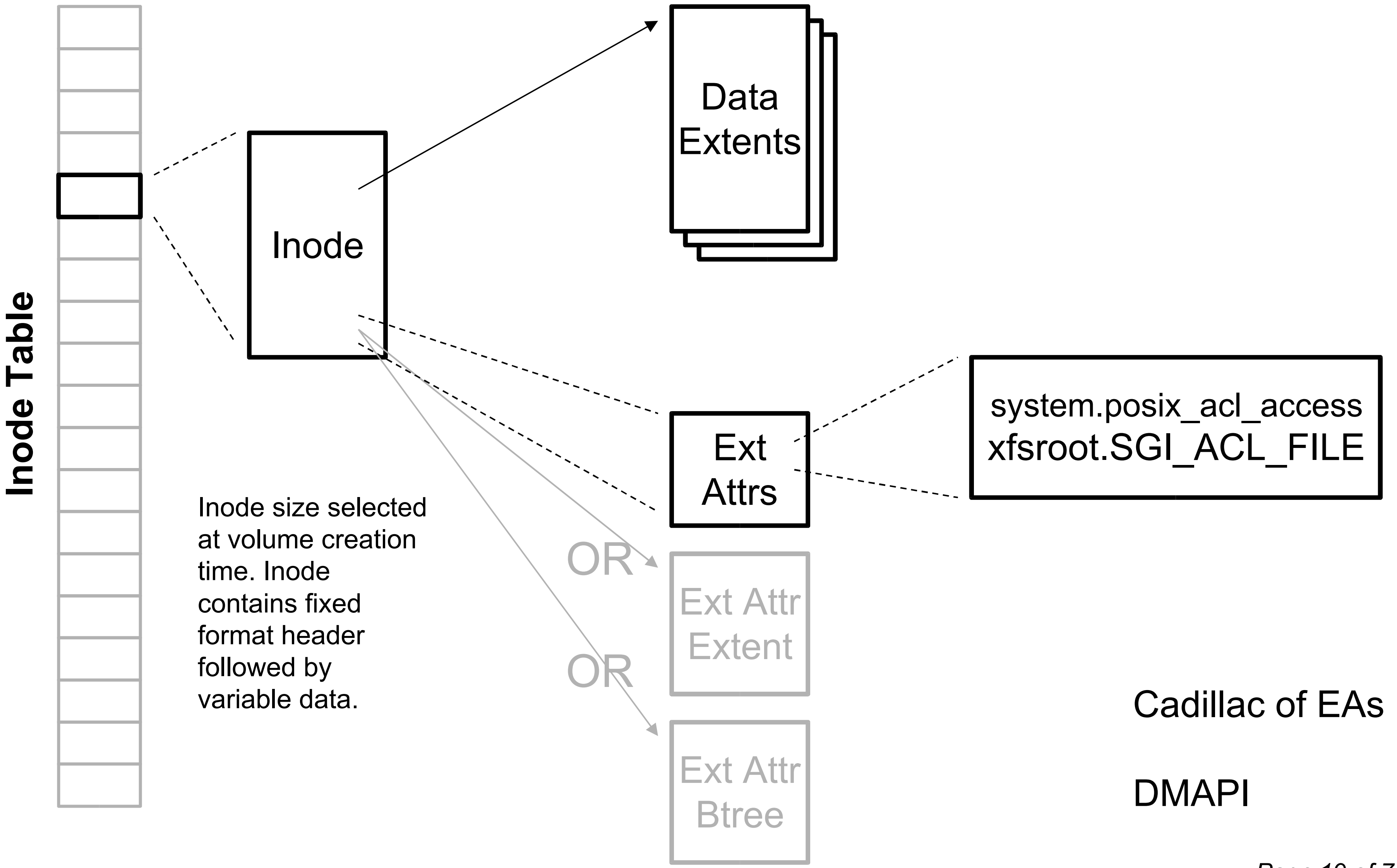
Solaris UFS


Inode Table
Inode
Data Blocks
Shadow Inode
FSD Blocks
FSD Type 1
POSIX ACL

Linux Files

- POSIX ACLs
  - acl_get_file(3), acl_set_file(3)
  - Separate access ACL and default ACL
  - Stored as EA “system.posix_acl_access”
  - Accessible through EA interfaces
- Extended attributes
  - Getxattr(2), setxattr(2)
  - Names are simple strings
  - User EAs allowed, name must start “user.”
- No named subfiles (streams)

Linux XFS

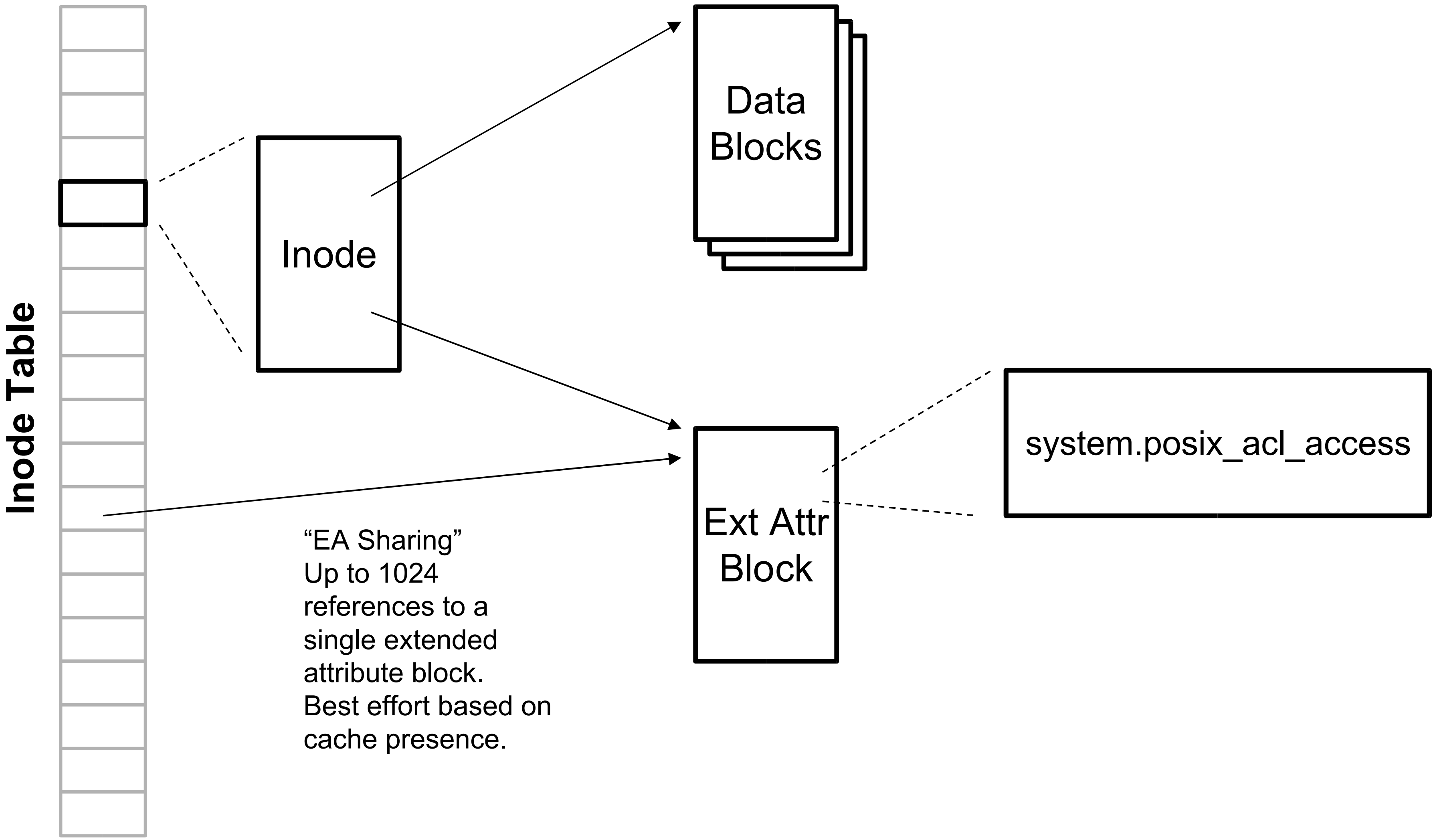


Inode size selected at volume creation time. Inode contains fixed format header followed by variable data.

Cadillac of EAs

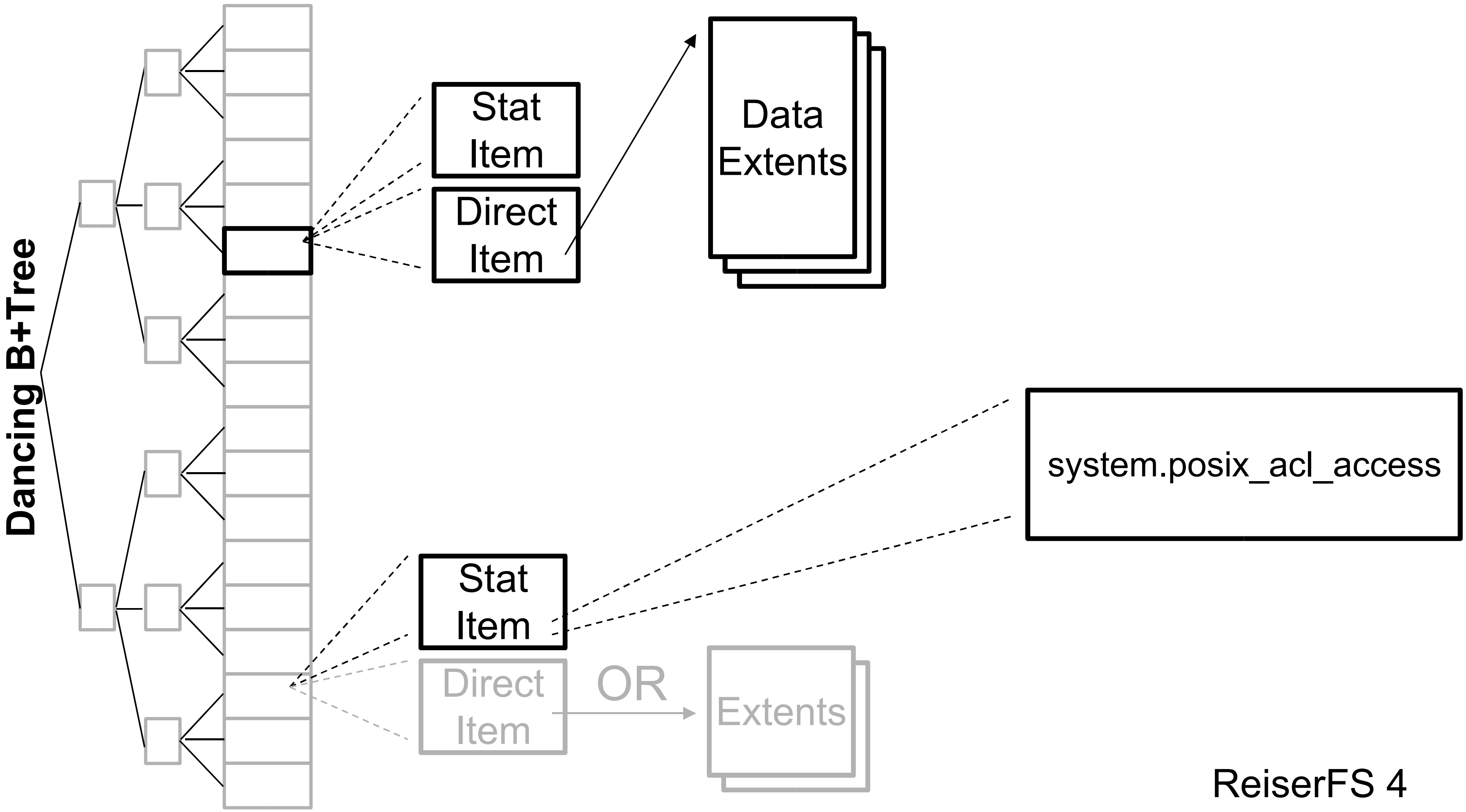
DMAPI

Linux EXT3



"EA Sharing"
Up to 1024 references to a single extended attribute block.
Best effort based on cache presence.

Linux ReiserFS



ReiserFS 4
Plugins

/.reiserfs_priv/xattr/*INUM.GEN*/system.posix_acl_access

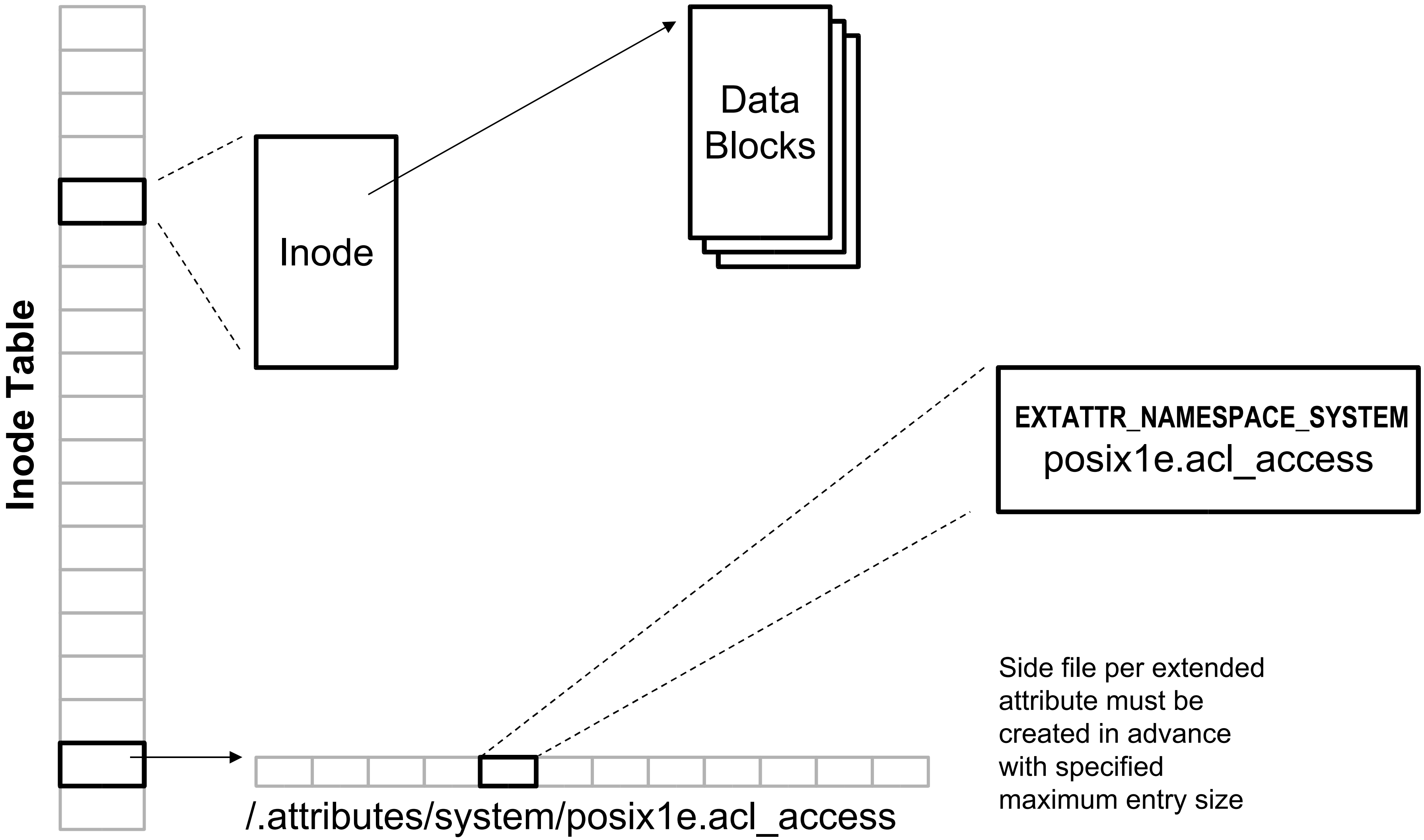
Linux Files

- Linux getxattr()/setxattr() splendid example of how to think about extended attributes
  - ACLs are manipulated through getxattr()/setxattr()
  - Internal get/set_posix_acl() gone in 2.6
  - All file systems translate between internal and API data structures
  - XFS has alias for ACLs, two names, same result
  - Extended Attributes are requests with operands (ala ioctl) rather than explicit bits under a name

BSD Files

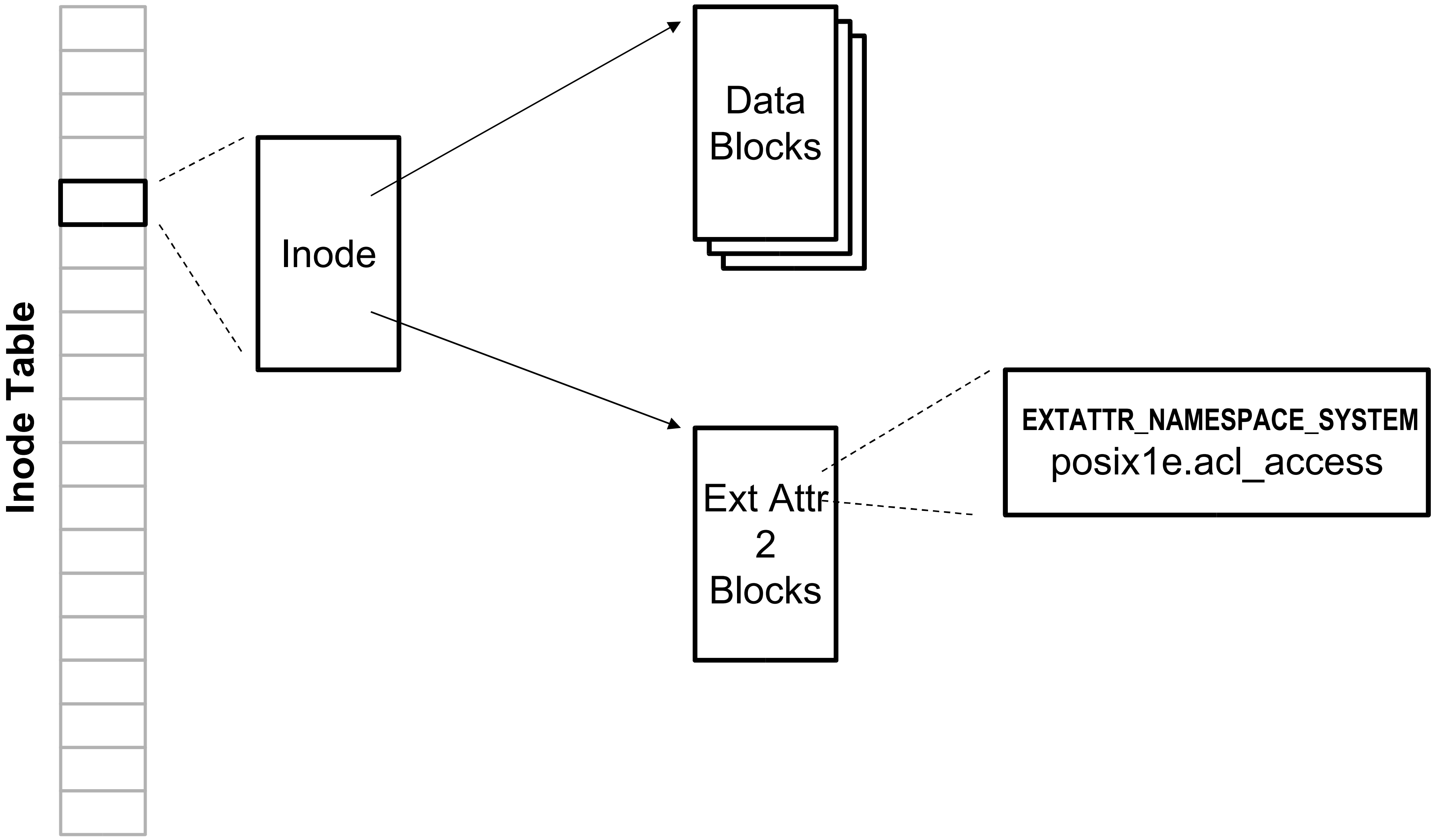
- POSIX ACLs
  - acl_get_file(3), acl_set_file(3)
  - Separate access ACL and default ACL
  - Stored as EXTATTR_NAMESPACE_SYSTEM "posix1e.acl_access"
  - Accessible through EA interfaces
- Extended attributes
  - Extattr_get_file(2), extattr_set_file(2)
  - Names are namespace number and string
  - User EAs, EXTATTR_NAMESPACE_USER
- No named subfiles (streams)

BSD UFS1



Side file per extended attribute must be created in advance with specified maximum entry size

BSD UFS2


Inode Table
Inode
Data Blocks
Ext Attr 2 Blocks
EXTATTR_NAMESPACE_SYSTEM
posix1e.acl_access

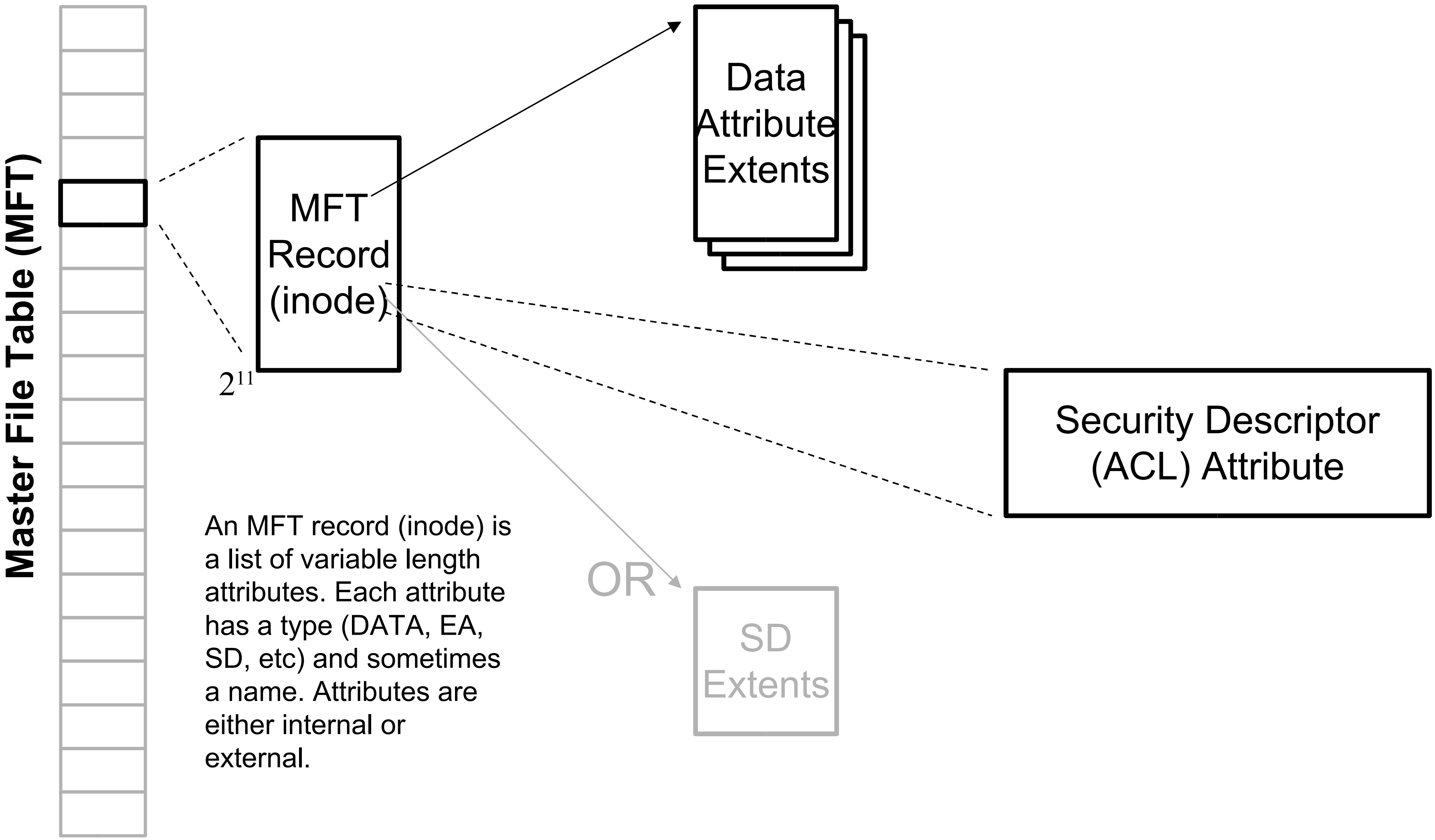
NTFS Files

- NTFS ACLs
  - GetFileSecurity()/SetFileSecurity()
  - Consistently used for more than files
  - Lots of discrete permissions, inheritance bits
  - Stored as an MFTR attribute (see diagram)
- Extended attributes
  - MFTR attributes, type, variable length, named or unnamed
  - OS/2 legacy EA, each an MFTR named attribute

NTFS Files

- Named subfiles (streams)
  - One unnamed DATA attribute
  - ADS – Alternate Data Stream, named DATA attributes
  - ♣SummaryInformation – Summary tab in Properties dialogue
  - ?Q30lsldxJoudresxAaaqpcawXc – Image thumbnail (and a UUID generated name)
  - Scarlet letter – uneven support, used by exploits
  - Please notice that subfiles are being used for the sorts of things folks would believe are Eas
  - Indications ADS will be used much more in future

Windows NTFS

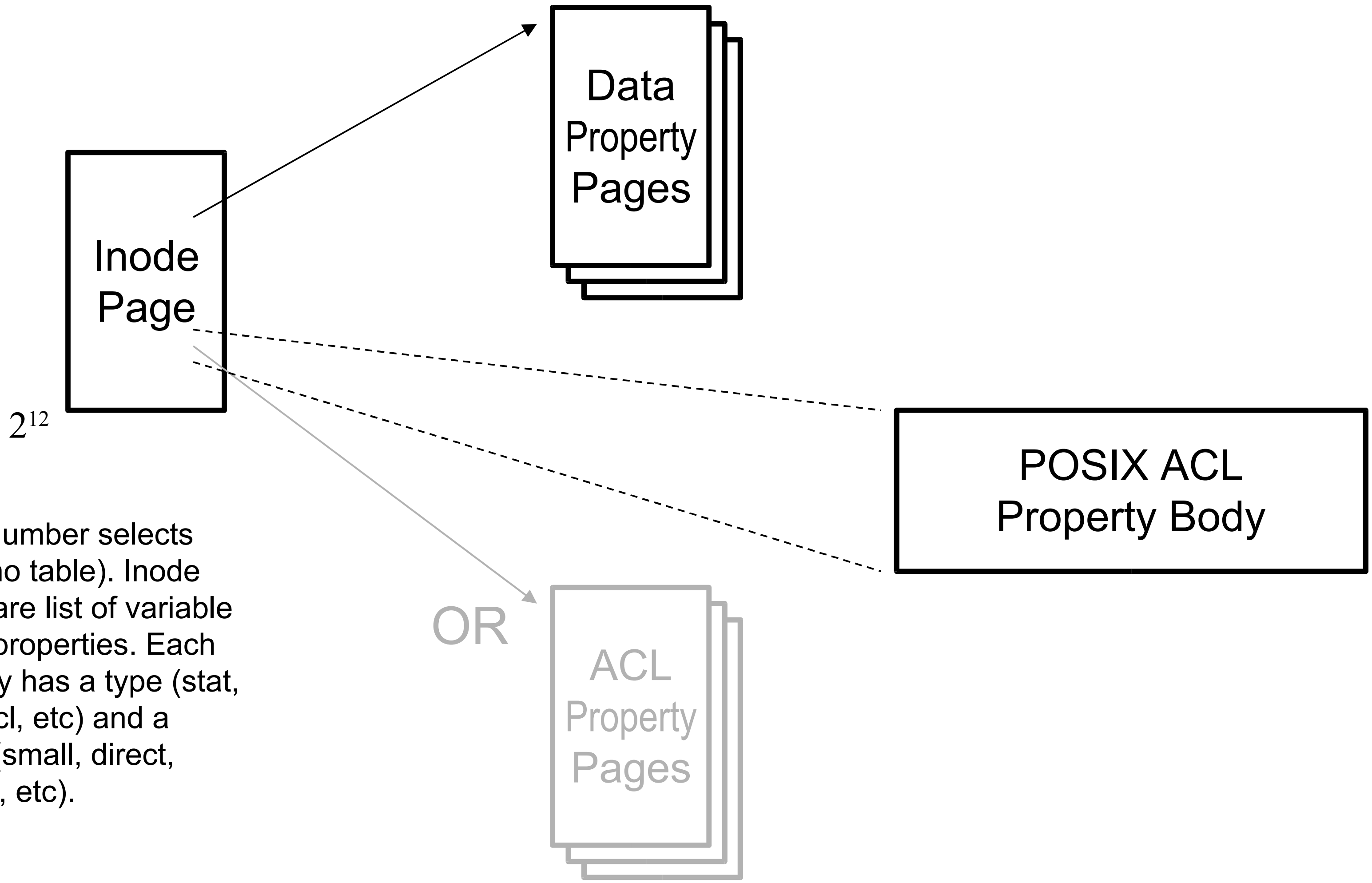


An MFT record (inode) is a list of variable length attributes. Each attribute has a type (DATA, EA, SD, etc) and sometimes a name. Attributes are either internal or external.

Traakan OTFS

- APIs of host system
- Third generation
  - Tired of fighting “brittle” data structures
- Inode page
  - Addressed by inode number (no table)
  - List of variable length properties
  - Each property has a type (standard attr, file data, dir data, posix acl, etc) and a shape (small, direct, indirectN)

Traakan OTFS



Inode number selects page (no table). Inode pages are list of variable length properties. Each property has a type (stat, data, acl, etc) and a shape (small, direct, indirect, etc).

Others

- Tru64
  - POSIX ACLs
  - Has extended attributes, calls them properties
- AIX
  - Spirit of POSIX ACLs
  - Three types of ACE (specify, allow, deny)
  - Three additional permission bits (c, d, i)
- HP/UX
  - Spirit of POSIX ACLs
  - Identities are user within group
  - Stat() is a view

I wish I'd had time

- TrustedBSD
- Macintosh classic, Mac OS 10
- IRIX
- WinFS? – NTFS successor
- VxFS
- NetApp WAFL
- EMC DART
- More….

Extensibility

- Unnamed extension attributes
  - Solaris UFS, OTFS, NTFS
- Named extension attributes
  - Linux, BSD, OS/2
  - Early influences of OS/2 HPFS, XFS, TrustedBSD
- No form of extension attribute was ever intended to be subfiles
  - That was an accident

Filesystems Aren’t Everything

Interchange

- Suppose I want to move a pile of files from system A to system B
  - Suppose further they have ACLs, subfiles, and other wizzy attributes
- System B has to be up to it
  - Underlying file system must implement the same file model
- How its moved counts, too
  - Inexcusable to get a different result because of how the pile was moved: network or archive

Interchange

- So, let’s have one big tent called “Interchange”
  - Utilities (cp, et al)
  - Archivers (tar)
  - File network protocols (NFS, CIFS)
- Expect and require consistent results
- Let’s have a look at how Interchange technologies are prepared for evolution

Interchange

- PAX – portable archive interchange
  - Tar successor
  - Type ‘x’ file extension header, content is name/value pairs, rules and reservations for names (security.*, realtime.*)
  - Opengroup release doesn’t support ACLs, Extended Attributes, or Subfiles
- Gnu tar
  - Doesn’t support ACLs, Extended Attributes, or Subfiles

Interchange

- Solaris tar
  - Type ‘X’, origin of pax ‘x’
  - Type ‘A’, ACL, checksum followed by text representation
  - Type ‘E’, extended attribute (named subfile), pair of records (header+data)

Interchange

- Star (ess-star) – terrific
  - Follows pax spec, very current, enjoys adoption
  - Runs on Linux, Solaris, BSDs, etc…
  - ‘x’ SCHILY.acl.access and SCHILY.acl.default, values are slight extension of POSIX standard text representation to include numeric UID/GID
  - ‘x’ SCHILY.xattr.*name* for extended attributes
  - Other conforming extensions defined
  - No apparent provision for subfiles
  - Recognizes but does not implement other extensions (Solaris A, E)

Interchange

- FreeBSD tar (libarchive)
  - Supports ACLs, uses star ‘x’ names and conventions
  - Supports reading Solaris ‘A’ records (ACLs)
  - No extended attributes or subfiles
- UDF – Universal Data Format (CD/DVD)
  - Contemplates extended attributes and named subfiles (streams)

Interchange

- File Access Protocols
  - NFSv4 – rich attributes, NT-style ACLs, quite extensible, named “attributes”
  - SMB/CIFS – driving a lot of evolution
- File Transfer Protocols
  - Email, FTP, HTTP
  - No real influence of evolution on these protocols found

Other Stuff

- Security model evolving too
  - Capabilities, privileges, etc
- Extra flags on BSD and Linux
  - Append-only, immutable
- Async I/O to regular files
  - Databases managers like it
- Kqueue (FreeBSD, NetBSD)
  - Notifications of updates, supports SMB, refreshing open file manager windows
- Unicode names

Seeds of Divergence

Seeds of Divergence

- Nathan Scott (SGI XFS on Linux)
  - [Extended Attributes] are intended for use to augment the metadata associated with an inode, rather than the more exotic uses that the "non-data fork" is designed for in some file systems.
- RFC3530 – NFSv4
  - Named attributes are meant … to associate application specific data with a regular file or directory.

Seeds of divergence

- Too easy to read intent into similarity or sameness of names
  - But there is no intent, similarity is an accident
  - Look past the names, study semantics
  - Perfectly human to not do the homework and get suckered by the names – still embarrassing
- Names should be corrected when we see that confusion is likely
  - Take human factors seriously

Seeds of Divergence

- Extended Attributes are not blobs of data, rather they are about enhancing file system functionality
- NFSv4 Named Attributes are blobs
- The superficial similarity is in the name, but they aren’t the same animal and are easily confused.
- THERE IS NO SUCH THING AS AN OPAQUE ATTRIBUTE

Seeds of divergence

- Named Extended Attributes vary
  - Names not just similar, but are the same (oh my)
  - Solaris has “Extended Attributes” (tar type ‘E’)
  - Linux/BSD have “Extended Attributes” (tar type ‘x’)
  - Other than the name, they bear little resemblance
  - Solaris implement NFSv4 Named Attributes with Solaris Extended Attributes
  - If NFSv4 Named Attributes are implemented with Linux/BSD-type Extended Attributes, things would be quite wonky

Seeds of Divergence

- NFSv4 Named Attributes are subfiles
  - When first contemplated, it wasn’t clear what direction they would take….now, it is….
  - Solaris – Fsattr(5) smells like subfiles (read/write/seek), except for the names that are clearly motivated by NFSv4
  - NetApp – Implements NFSv4 Named Attributes as CIFS Streams (subfiles)
  - Hummingbird on NT – NFSv4 Named Attributes likely to be ADS (Alternate Data Stream)

Seeds of Divergence

- Linux, FreeBSD (et al), OS/2
  - Name/value get/set Extended Attributes
  - Small in general, really small in some implementations
  - Quite inadequate given Solaris/NetApp/NT precedent (capacities)
  - Can’t reliably map read/write to get/set
  - GOOD NEWS! NFSv4 implementations not using EAs for NFSv4 Named Attributes

Convergence Is At Hand

Convergence

- Subfiles
  - Read/write
  - Arbitrary size
  - Unchecked
  - Opaque
  - Informal
  - Textual names
  - Stored literally
  - Tar ‘E’ records
- Extended Attrs
  - Get/set
  - Smallish
  - Checked at set()
  - Meaningful
  - Formal
  - Named or not
  - Internal translation
  - Tar ‘x’ records

Convergence

- NFSv4 Named Attributes
  - Recognize that it is a subfile by virtue of read/write interface
  - Rename OPENATTR to OPENSUBFILES or OPENADS or OPENSTREAMS or anything but ATTR
    - Not a small matter, remember human factors
  - Solaris APIs, too
    - How about “sf” instead of “at”, opensf(), sfopen(), renamesf(), etc.
    - Reserve “Extended Attribute” for getxattr()/setxattr()

Convergence

- Adopt Linux-style Extended Attributes
  - These are the only things called “extended attribute”
  - getxattr()/setxattr()
  - Easily done at library level
  - Adopt as interface for manipulating ACLs, too
  - Strawman: star archiver without ifdefs
  - Solaris, BSD, and others – this means you

Convergence

- Adopt Solaris-style subfiles
  - Right after they fix the name (perhaps opensf() rather than openat(), et al)
    - Solaris can maintain both API sets in interim
  - Do not use Extended Attributes to implement NFSv4 Named Attrs – Linux, BSD, this means you
  - Interim implementation ala ReiserFS example
    - /.subfile/*INUM.GEN*/ for the opensf() directory
    - Readily matches Solaris/NetApp/NT NFSv4 semantics
    - Construct library-level of opensf() APIs, kernel level later
  - Update utilities – -@, see Solaris fsattr(5) list

Convergence

- Interchange (star, NFSv4) looks good
  - Let's get behind star(1), leading charge, BSD following
  - Tar 'x' records for new file system attributes
  - NFSv4 minor versioning for new file system attributes, opaque encapsulation
  - Tar 'E' records for subfiles (needed in star)
  - NFSv4 named *thingy* for subfiles
  - Need to make sure tar 'E' records make sense native on NT

Convergence

- Rigorous Definition of Extended (Named) Attributes
  - Official API name and star (pax) name (IANA?)
  - API data structures (operands)
  - Canonical text definition – star (pax) "value" part
  - XDR definition
  - XML definition – browsers, email, php, etc
  - NFSv4 attribute number (IANA?)
  - Documentation, of course

What it All Means

What it all means

What you should remember tomorrow

- The file model is evolving
- The file model is diverging
- There is posturing for evolution
  - Solaris FSD, OTFS Properties, Linux/BSD EAs, NTFS MFTR Attributes, ReiserFS plugins, star, UDF, NFSv4
- Current issues (ACLs) are rather simple
- Future issues are much more difficult
- And, above all, we can converge now

What it all means

What you should remember tomorrow

If convergence sounds expensive,

consider what divergence costs.

Predictions

Predictions

Primary Attributes

- Primary attributes (stat) must evolve
  - 32-bit UID/GID insufficient
  - Count of subfiles, EAs, etc
  - Legacy “views” will be available

Predictions

Access Control List (ACLs)

- ACLs much like NFSv4/NT
  - Will continue to evolve
  - ADD_DATA and ADD_FILE permissions need to be distinct because of subfiles
  - Don’t know what to make of EA permissions
  - Expect NTFS to improve subfile access control
  - Gating factor is user/group identity on UNIX-type systems

Predictions

Extended Attributes are Named Views

- Extended attribute get/set name/value API is pleasant, flexible, and extensible
  - Textually named views
  - Not direct access to data structure.
  - Ordinary (primary) attributes also
  - Legacy attribute views available “stat99”
  - Precedent: many NT interfaces have “level”
  - Maybe: operand will be XDR encoded
  - The day will come when we just say “attribute” and not “extended”

Predictions

Named subfiles (streams)

- A subfile will have its own set of attributes
  - Solaris model really looks good
  - Easy case for size and timestamps
  - Elaborate (ACL) could be hard in general
    - NTFS doesn't look well prepared for it, but it looks like they'll have to do something
  - UDF forbids elaborate stream (subfile) attributes
  - Near term restraint advisable

Predictions

Interchange – File Transfer

- Email attachment methods will need to adapt for subfiles
  - How does Mac do this?
- FTP and HTTP – Unchanged
  - Main data will be transferred, no subfiles
  - HTTP queries and redirects awkward to replace
- Attributes probably don’t need to be conveyed
  - But maybe integrity labels.

Predictions

Interchange – File Access

- NFSv4 is right idea
  - OPEN-NAMED-THINGY will be for subfiles
  - New attributes need formality, minor version
  - Perhaps XDR opaque encapsulation of new attributes so they can be hopped over
- SMB/CIFS
  - Hard to know or influence what will happen to SMB/CIFS or its successors

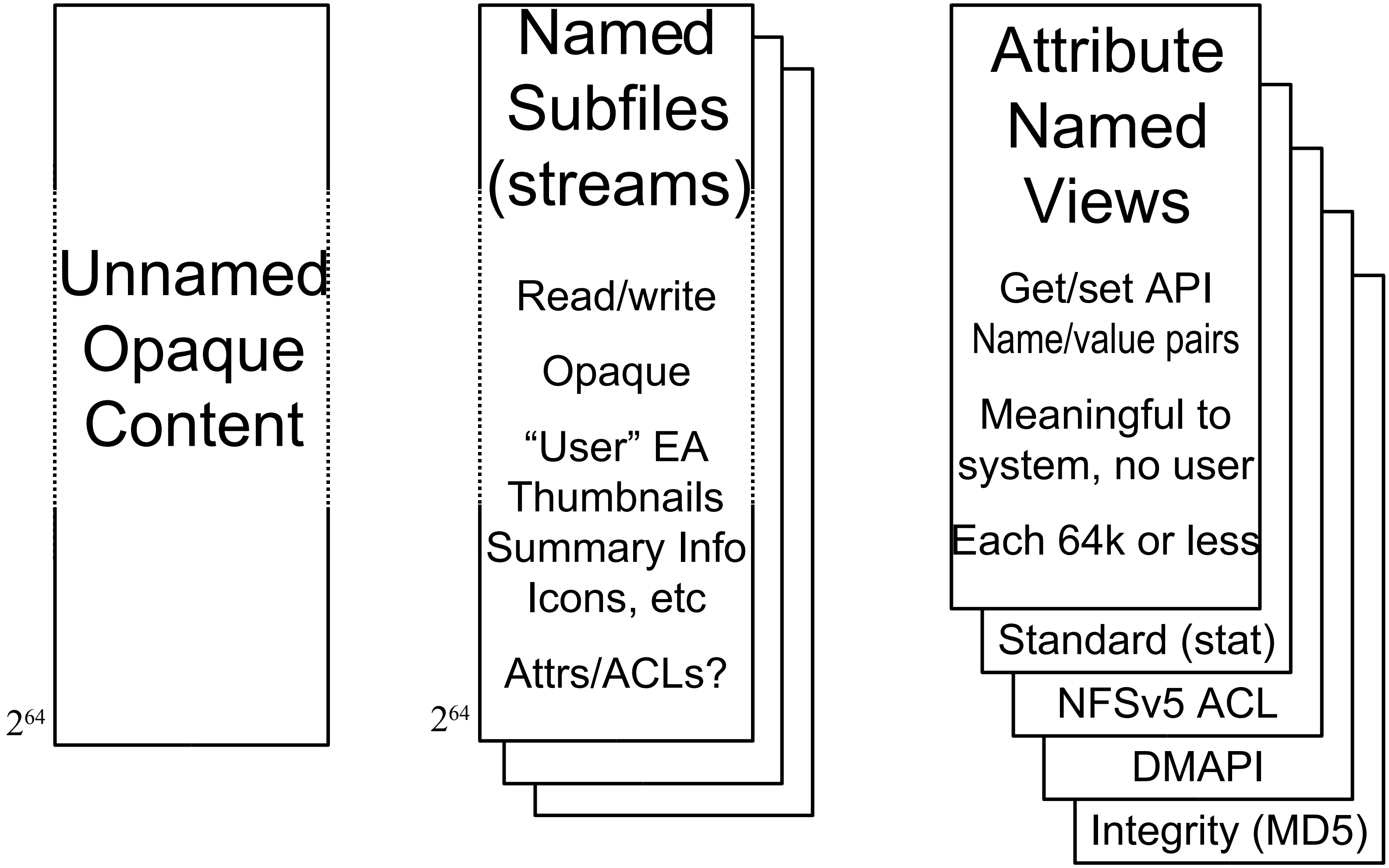
Predictions

Interchange – File Transfer and Access

- WebNFSv4 wins
  - As requirements to handle new file features emerge over the next decade it is unlikely FTP and HTTP will be revised
  - WebNFSv4 will be sitting there good to go
  - Likely to become preferred Internet file transfer because attributes and subfiles are handled
  - Expect folks to get tired of transferring tarballs to capture subfiles and extended attributes.
  - Simply, WebNFSv4 is poised as the path of least resistance.

Predictions

File Model Circa 2014



Unnamed Opaque Content

2^{64}

Named Subfiles (streams)

Read/write

Opaque

“User” EA
Thumbnails
Summary Info
Icons, etc

Attrs/ACLs?

2^{64}

Attribute Named Views

Get/set API
Name/value pairs

Meaningful to system, no user

Each 64k or less

Standard (stat)

NFSv5 ACL

DMAPI

Integrity (MD5)

Encryption
Compression
Secure level
HSM
Replication
Etc….

Discussion

References

POSIX Access Control List (1003.1e draft 17)

- http://wt.xpilot.org/publications/posix.1e/
- http://wt.xpilot.org/publications/posix.1e/download.html

Linux POSIX ACLs and Extended Attributes

- http://acl.bestbits.at/
- http://acl.bestbits.at/man/man.shtml
- http://lwn.net/2000/1026/a/extended-attributes.php3
- http://www.suse.de/~agruen/acl/linux-acls/online/ Terrific paper
- http://www.osnews.com/story.php?news_id=69 jfs, xfs, reiserfs comparisson

FreeBSD ACLs and Extended Attributes

- http://www.trustedbsd.com/components.html
- http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/fs-acl.html
- http://www.freebsd.org/cgi/man.cgi?query=extattr_get_fd&sektion=2&apropos=0&manpath=FreeBSD+5.2-current

Solaris ACLs and Extended Attributes

- Acl(2) http://docs.sun.com/db/doc/817-3938/6mjgf9052?a=view
- Fsattr(5) http://docs.sun.com/db/doc/817-3946/6mjgmt4m0?a=view

References

TRU64 Properties and ACLs

- Proplist(4) http://h30097.www3.hp.com/docs/base_doc/DOCUMENTATION/V50_HTML/MAN/MAN4/0200____.HTM
- Acl(4) http://h30097.www3.hp.com/docs/base_doc/DOCUMENTATION/V50_HTML/MAN/MAN4/0083____.HTM

HPUX ACLs, stat(2) view

- http://www.informatik.uni-frankfurt.de/doc/man/hpux/acl.5.html

AIX ACLs, DCE ACLs

- http://nscp.upenn.edu/aix4.3html/aixuser/usrosdev/access_control_list.htm
- http://www.dsps.net/ACL.html DCE file acl c,i,d

White paper on wrapping ACLs into a common view (good, quick read)

- http://www.engenio.com/pdf/TAS/acladminwp.pdf

OS/2 Extended Attributes

- http://www.naspa.com/PDF/96/T9607014.pdf

NTFS

- http://patriot.net/~carvdawg/docs/dark_side.html
- http://www.winnetmag.com/Articles/Print.cfm?ArticleID=15900
- http://linux-ntfs.sourceforge.net/ntfs/index.html
- http://www.ntfs.com/ntfs_basics.htm

References

Traakan OTFS

- http://www.traakan.com/support.html

PAX (Portable Archive Interchange), updated tar format

- PAX http://www.opengroup.org/onlinepubs/009695399/utilities/pax.html
- STAR http://www.fokus.fraunhofer.de/research/cc/glone/employees/joerg.schilling/private/star.html
- GNU tar http://www.gnu.org/software/tar/tar.html

UDF – Universal Data Format

- ECMA-167 http://www.ecma-international.org/publications/standards/Ecma-167.htm
- OSTA UDF http://www.osta.org/specs/pdf/udf250.pdf
- Rockridge extensions to ISO-9660ftp://ftp.ymi.com/pub/rockridge/

NFSv4

- http://www.nfsv4.org
- http://www.ietf.org/rfc/rfc2624.txt