Tradeoffs in Protecting Storage:

A Meta-Data Comparison of Cryptographic, Backup/Versioning, Immutable/Tamper-Proof, and Redundant Storage Solutions

Joseph Tucek* Paul Stanton* Elizabeth Haubert Ragib Hasan*
Larry Brumbaugh William Yurcik*

StorageSS Research Group
National Center for Supercomputing Applications (NCSA)
Department of Computer Science
University of Illinois at Urbana-Champaign

IEEE COMPUTER SOCIETY

22nd IEEE - 13th NASA Goddard (MSST2005) Conference on
Mass Storage Systems and Technologies
April 11-14, 2005, Monterey, California USA

Motivation

- System break-ins
  - Attacks are increasingly sophisticated
  - Current payloads are “nice” – this may change
    - “Witty Worm”
- Insider attacks
  - Steal data
  - Cover-up unauthorized activity

More Motivation

- Legal regulations
  - HIPAA
  - Gramm-Leach-Bliley Act
  - Sarbanes-Oxley Act
  - SEC 17A-3 and 17A-4
  - California State Law SB 1386
- User experience
  - System unavailability is intolerable
  - Loss of data isn't either

Tutorial Plan

- Motivation
- **Overview**
- Survey of Protection
- Comparison
- Case Study: Tungsten at NCSA
- Conclusions

Overview - Metrics

- CIA
  - Confidentiality
  - Integrity
  - Availability
- Cost tradeoffs (finite budget)
  - Performance
  - Capital outlay
  - Management effort

Overview – C I A

- Confidentiality
  - Only authorized entities can read data
  - Provided by access control and encryption
- Integrity
  - Only authorized entities can modify data
  - provided by access control, tamper-proofing, immutability
- Availability
  - Security is comparatively easy
    - Unplug the box and bury it!
  - For C & I to be useful, data must be available

Overview – Cost Tradeoffs

- Performance
  - If it's too slow, it can't be used
- Capital outlay
  - Extra space, extra compute, special equipment
- Management effort
  - Imagine manually distributing encryption keys

Tutorial Plan

- Motivation
- Overview
- **Survey of Protection**
  - **Cryptography**
  - Immutability and Tamper-Proofing
  - Backup and Versioning
  - Redundancy
- Comparison
- Case Study: Tungsten at NCSA
- Conclusions

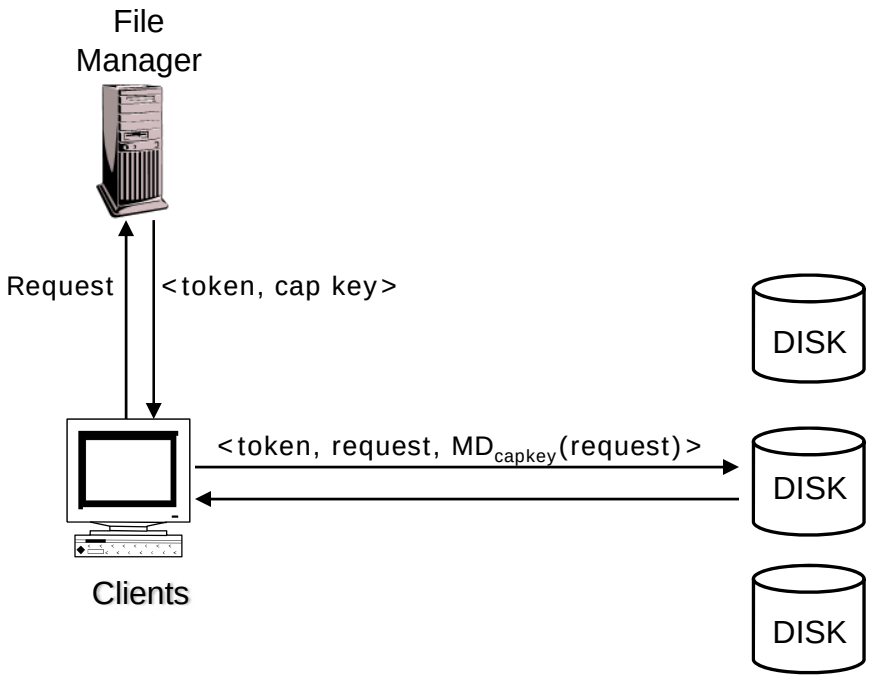
Cryptography - Overview

- Provides Confidentiality (some integrity)
- Emphasis on Key Management
  - Distribution
  - Revocation
  - Granularity

- **NASD** (CMU Parallel Data Lab)
- **SFS-RO** (NYU Secure Computer Systems & MIT Parallel/Distributed OS)
- **Plutus (**HP Labs)
- **SiRiUS** (Stanford Applied Crypto)

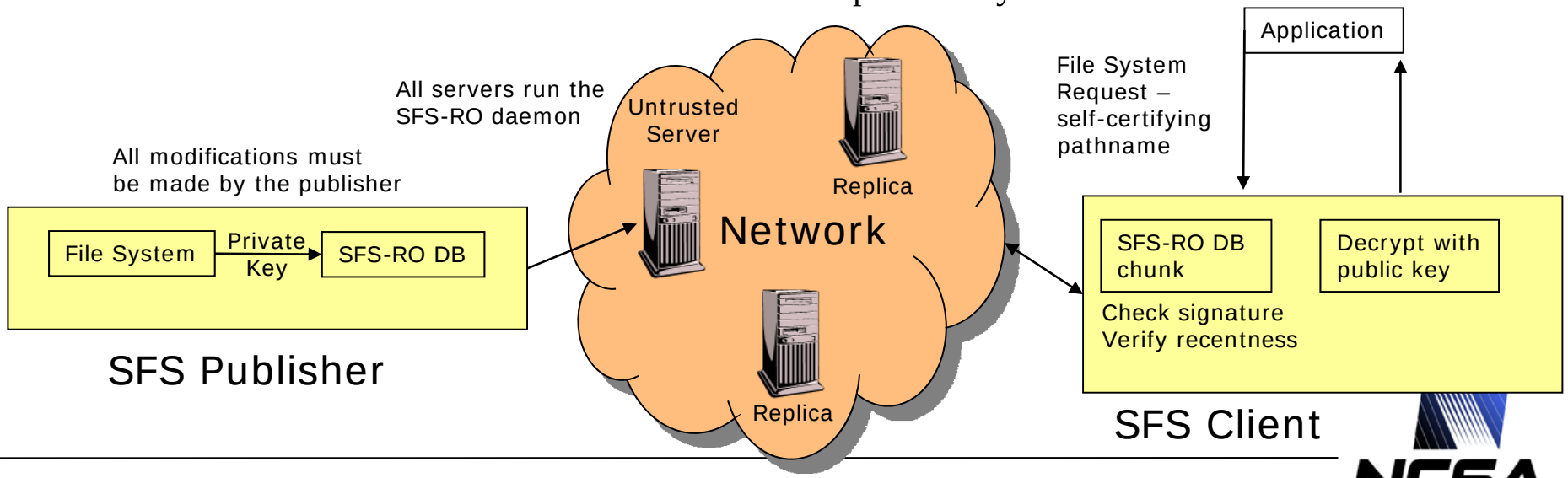
NASD: Network Attached Secure Disks

- Centralized file manager (FM)
- Request to FM results in a *capability object*
  - Token (access rights)
  - Capability key
- FM shares private key with intelligent disks
- User applies capability key to the request and accesses disk directly
- Disk uses secret key and token to verify the request digest
- Immediate revocation is possible with centralized server



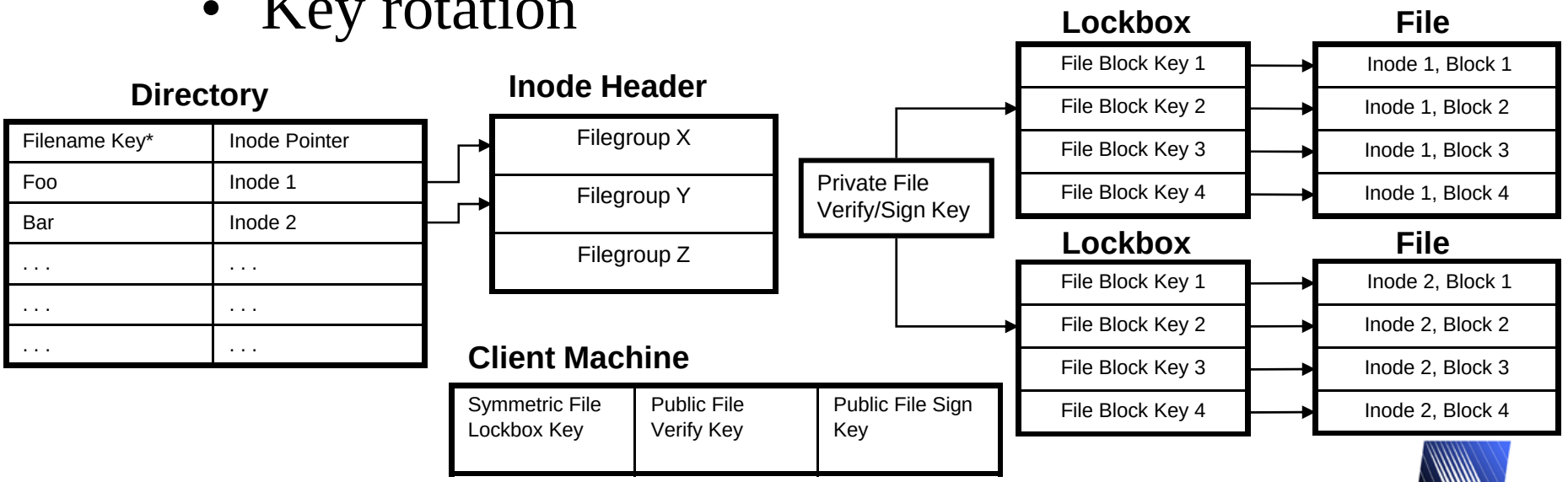
SFS-RO:
Fast and Secure Distributed Read-Only File System

- Encrypt file system contents as signed DB
- Replicate the DB on multiple servers
- Self-Certifying Pathnames
- Key revocation certificates
  - $\{path, location, public\ key\}_{private\ key}$



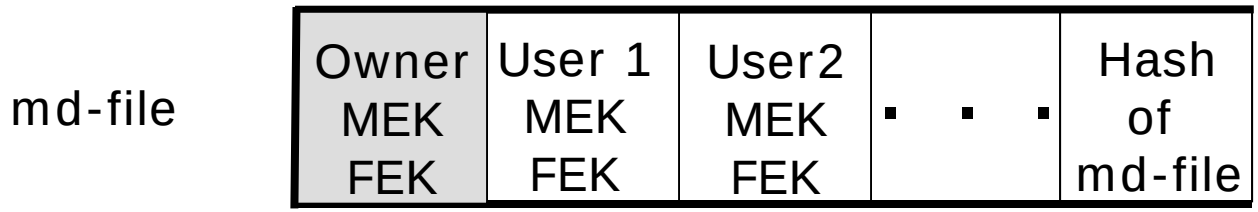
PLUTUS

- Lockbox mechanism for scalable key mgmt
- Manually distribute keys to clients
- File sharing via file groups
- Lazy revocation
- Key rotation



SiRiUS
Securing Remote Untrusted Storage

- Stop-gap security to legacy systems
  - NFS, CIFS, Yahoo, etc.
- Each user has an asymmetric Master Encryption Key
- Metadata file for each file
  - Master encryption for owner
  - File encryption/signing key stored for each user encrypted with MEK of user
  - Hash of contents signed with owner's MEK
- Revocation – simply remove the user's entry from the md-file



Key Management Comparison

System	Distribution	Revocation	Granularity	Duration
NASD	Trusted server	Immediate	Storage object	Session
SFS-RO	User managed	Immediate	File system	Permanent
PLUTUS	Key lockbox	Lazy	Lockbox, group, file, block	Permanent
SiRiUS	User managed	Immediate	Owner, file	Permanent

Key Management Comparison

System	Distribution	Revocation	Granularity	Duration
NASD	Trusted server	Immediate	Storage object	Session
SFS-RO	User managed	Immediate	File system	Permanent
PLUTUS	Key lockbox	Lazy	Lockbox, group, file, block	Permanent
SiRiUS	User managed	Immediate	Owner, file	Permanent

Tutorial Plan

- Motivation
- Overview
- **Survey of Protection**
  - Cryptography
  - **Immutability and Tamper-Proofing**
  - Backup and Versioning
  - Redundancy
- Comparison
- Case Study: Tungsten at NCSA
- Conclusions

Immutability

- **Immutability** means
  - To prevent modification
  - To thwart deletion
  - Brittle, but potentially strong prevention
- **Immutable** file systems allow
  - Appending
  - Writing new data

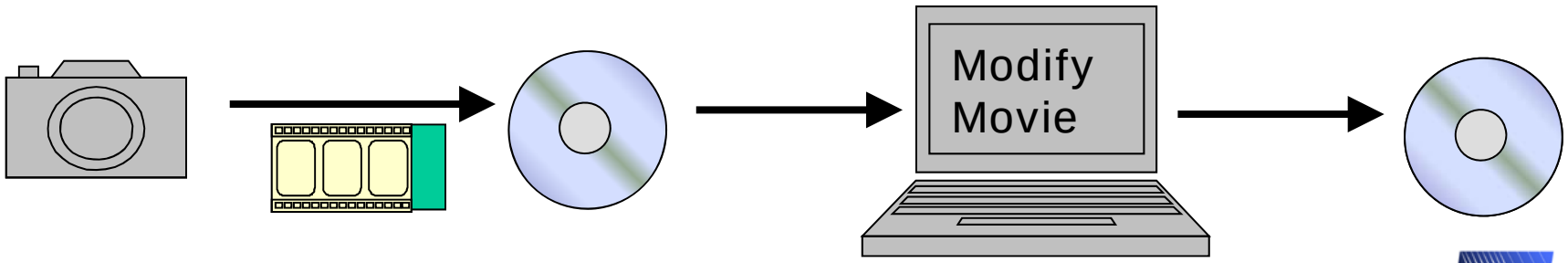
Examples of Immutable Systems

1. Physical WORM (Write Once Read Many)
   - CD-R, magneto-optical
     - Expensive, low capacity, slow
2. Embedded WORM
   - Write-once disk, tape, write-once SAN
     - Limited availability, current implementations not trustworthy
3. Software WORM
   - Write permission attributes, immutable attribute
     - Cheap, fast, easy, weak

Tamper-Proof

aka Tamper-Resistant or Tamper-Evident

- Demonstrate with high reliability that data has not changed improperly
- Not the same as confidentiality
- Not the same as immutability



Examples of Tamper-Proof Systems

1. **SFS-RO**
   - File names contain public keys
   - Blocks/inodes named by hash of content
   - Groups of handles hashed recursively
2. **PASIS**
   - Uses erasure codes, so data can be reconstructed with m of n fragments
   - Uses cross-checksums to identify corrupted data fragments
3. **OceanStore**
   - Restrict server capabilities
   - Erasure code fragmentation

Immutability vs. Tamper-Proof

- Immutability proves something hasn't changed
  - CAN NOT rewrite a CD-R
  - CAN make a new CD-R
- Tamper-Proof proves something is what you think it is
  - CAN NOT forge a signed log file
  - CAN erase a signed log file

Tutorial Plan

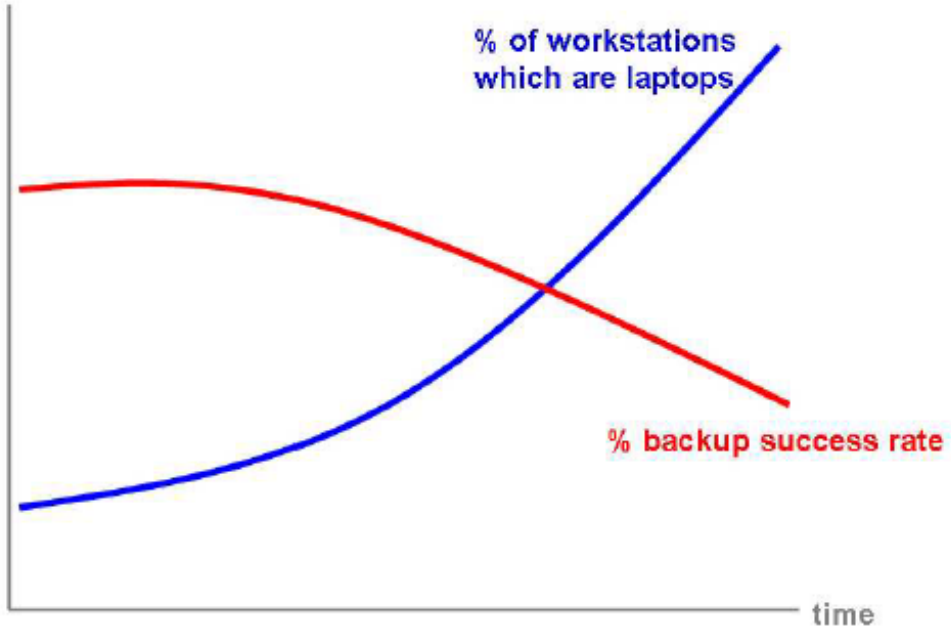
- Motivation
- Overview
- **Survey of Protection**
  - Cryptography
  - Immutability and Tamper-Proofing
  - **Backup and Versioning**
  - Redundancy
- Comparison
- Case Study: Tungsten at NCSA
- Conclusions

Backup and Versioning

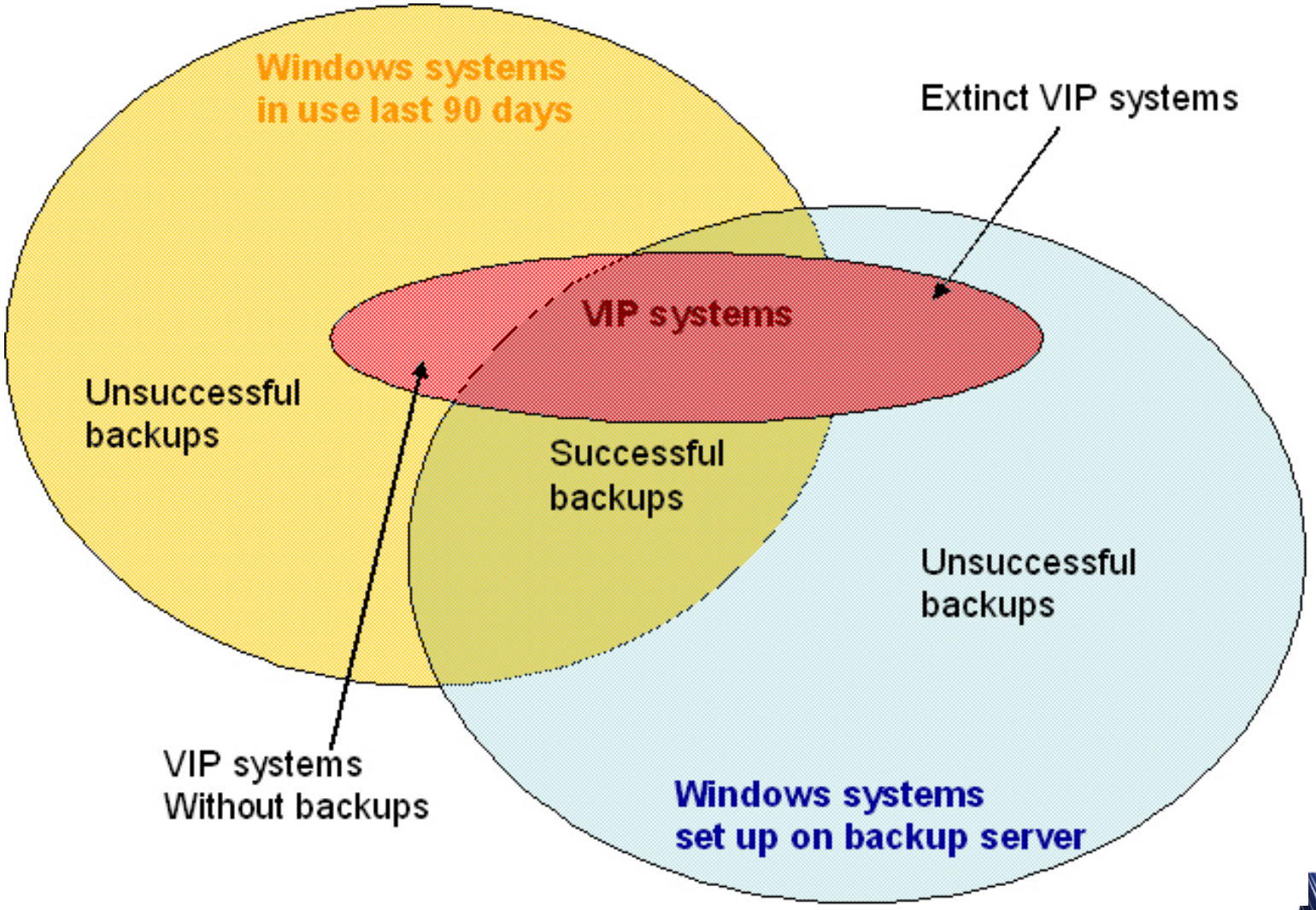
- ***Integrity & Availability***
  - Recovery from corruption **IF** event is known/detected
  - Can actually hurt confidentiality—all of those extra copies floating around
- **Difference is one of *degree* and *technique***
  - Degree in terms of "when"
    - Scheduled is typically backup (very often may be versioning)
    - Interrupt-driven is typically versioning (manual may be backup)
  - technique in terms of "what"
    - Full (typical building block for backup)
    - Differential
    - Incremental (typical for versioning)

Backup

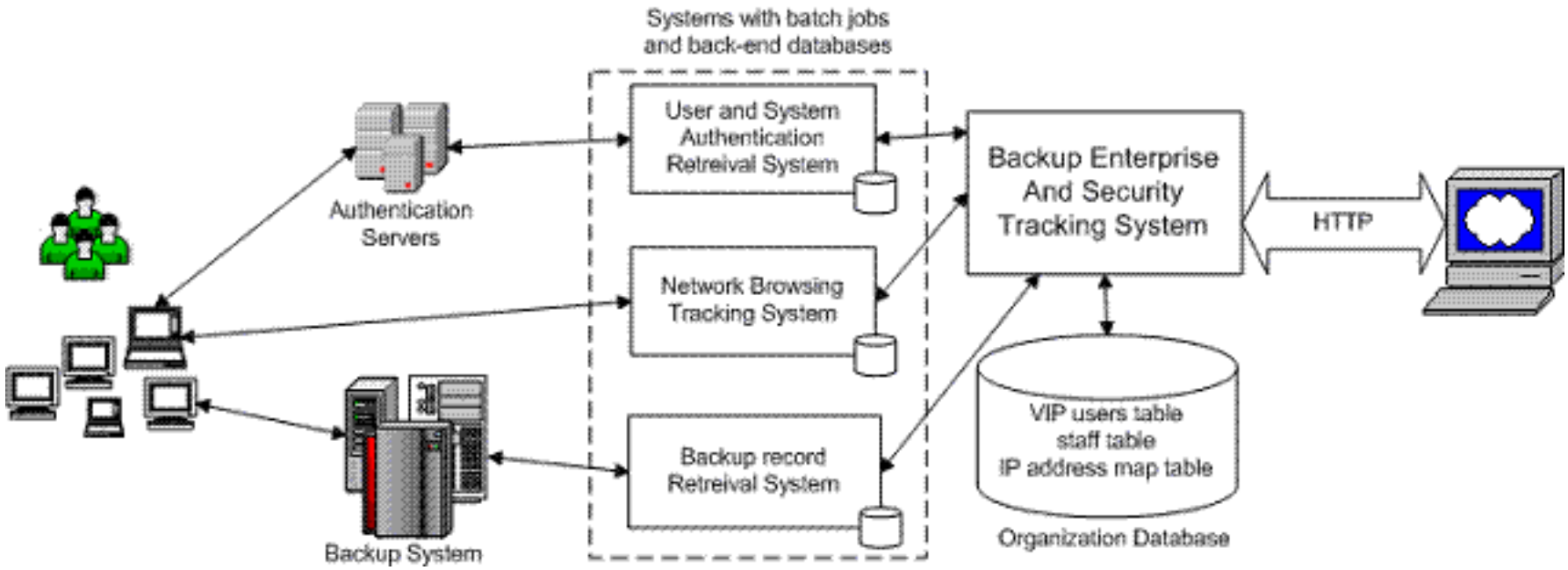
- Understood, now just management issues. Big management issues.
- Traditional backup (Amanda <-> BTS)
- New issues
  - Mobile hosts
  - Transient hosts
  - Restores



Prioritized Backups



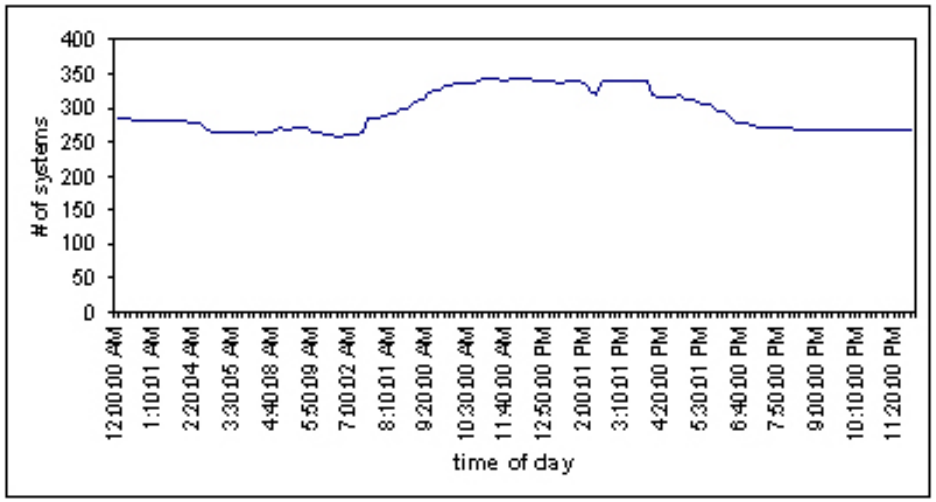
NCSA's Backup Tracking System*

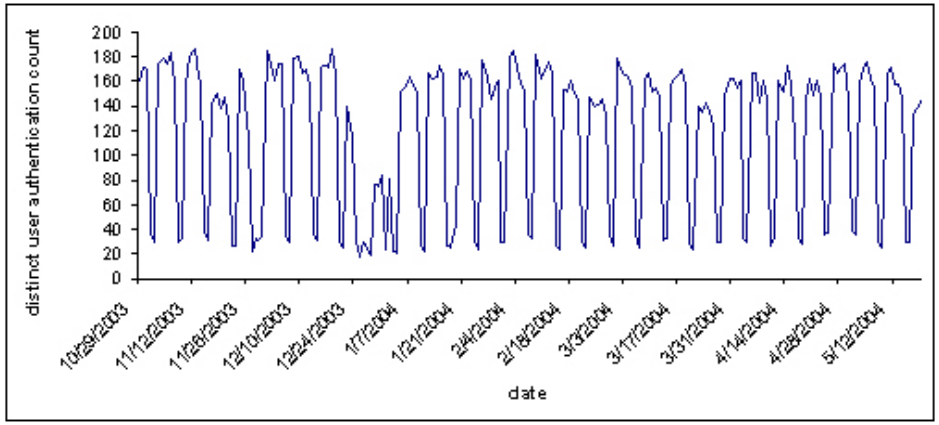


***** G. Pluta, L. Brumbaugh, W. Yurcik, and J. Tucek. "**Who Moved My Data? A Backup Tracking System for Dynamic Workstation Environments**," *Usenix LISA*, 2004.

Backup Data Useful for Other Purposes

Systems





Users

Versioning

- Versioning is *continuous* or *semi-continuous* backup
  - Elephant file system
    - Keep landmark versions (protect yourself)
  - S4
    - Keep everything (protect against others)
  - Recoverable File Service
    - Who did what to whom? (selective roll-back audit trail)

Feasibility of Full Versioning

- Average workstation has 200MB writes/day
  - 73 GB/year, < $300/year
    - What is the cost of lost data?
  - What level of compressability?

- Straw poll—who has .snapshot or OldFiles?

Tutorial Plan

- Motivation
- Overview
- **Survey of Protection**
  - Cryptography
  - Immutability and Tamper-Proofing
  - Backup and Versioning
  - **Redundancy**
- Comparison
- Case Study: Tungsten at NCSA
- Conclusions

Redundancy

- Space redundancy protects against
  - 1) Hardware failures
  - 2) Configuration failures
  - 3) Malicious attacks
- Usual technique is RAID
  - Somewhat well known…
  - See Peter Chen's "**RAID: High-Performance, Reliable Secondary Storage**"
  - Has some issues…

RAID Problems

- Only handles hardware failure
- Correlated failures are common
  - Same environment, same load, same disks…
  - Hot spare may not recover in time
  - RAID-6 type techniques are needed
  - Row-diagonal parity
- Some data more important (metadata)
  - D-GRAID
- Hard to manage
  - HP AutoRAID
  - Polus

Other redundancy techniques

- Erasure codes
  - A more complex “parity”
  - Possibly spread across sites
- Byzantine fault-tolerance
  - Don’t trust anybody
  - PASIS (from tamperproof)
- Secret Sharing
- Shortcomings
  - No protection against purposeful corruption
    - Nicely mirrored copies of tampered data
  - Very expensive

Tutorial Plan

- Motivation
- Overview
- Survey of Protection
- **Comparison**
- Case Study: Tungsten at NCSA
- Conclusions

Comparison

Technique	Confidentiality	Integrity	Availability	Cost
Encryption	High	Medium	Negative	CPU
Secret Sharing	High	High	High	CPU, Latency, Space
Tamper-Proof	None	High	None	CPU
Immutability	None	High	High	Latency, Space
Backup	None	Medium	Medium	Bandwidth, Space
Versioning	None	Medium	High	Space
RAID	None	Low	Medium	Space

Comparison

Technique	Confidentiality	Integrity	Availability	Cost
Encryption	High	Medium	Negative	CPU
Secret Sharing	High	High	High	CPU, Latency, Space
Tamper-Proof	None	High	None	CPU
Immutability	None	High	High	Latency, Space
Backup	None	Medium	Medium	Bandwidth, Space
Versioning	None	Medium	High	Space
RAID	None	Low	Medium	Space

Confidentiality is limited to cryptography

Comparison

Technique	Confidentiality	Integrity	Availability	Cost
Encryption	High	Medium	Negative	CPU
Secret Sharing	High	High	High	CPU, Latency, Space
Tamper-Proof	None	High	None	CPU
Immutability	None	High	High	Latency, Space
Backup	None	Medium	Medium	Bandwidth, Space
Versioning	None	Medium	High	Space
RAID	None	Low	Medium	Space

Availability costs in space

Comparison

Technique	Confidentiality	Integrity	Availability	Cost
Encryption	High	Medium	Negative	CPU
Secret Sharing	High	High	High	CPU, Latency, Space
Tamper-Proof	None	High	None	CPU
Immutability	None	High	High	Latency, Space
Backup	None	Medium	Medium	Bandwidth, Space
Versioning	None	Medium	High	Space
RAID	None	Low	Medium	Space

Hardware cost is mostly CPU and space

Comparison

Technique	Confidentiality	Integrity	Availability	Cost
Encryption	High	Medium	Negative	CPU
Secret Sharing	High	High	High	CPU, Latency, Space
Tamper-Proof	None	High	None	CPU
Immutability	None	High	High	Latency, Space
Backup	None	Medium	Medium	Bandwidth, Space
Versioning	None	Medium	High	Space
RAID	None	Low	Medium	Space

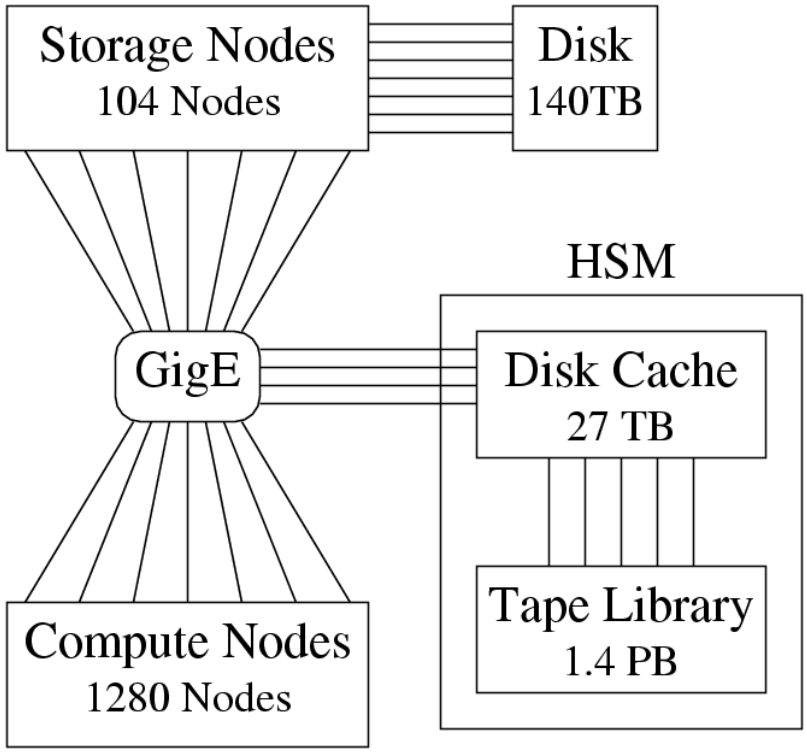
Each technique is best at different things

Tutorial Plan

- Motivation
- Overview
- Survey of Protection
- Comparison
- **Case Study: Tungsten at NCSA**
- Conclusions

Case Study

- Storage at NCSA
  - Tungsten
    - # 10 on the Top 500
    - 104 storage nodes
    - 140 TB of disk
  - Hierarchical Storage Manager
    - 27 TB disk cache
    - 1.4 PB tape
- Big systems



Specific System Characteristics

- High Performance Work Space (Scratch)
  - 11.1 GB/sec
  - Ephemeral—purge after 14 days
- Mass Storage
  - Write heavy
    - .3:1 ratio of read:write
      - Does anybody even look at it all?
  - Growing (fast)
    - Between 2-20 TB a week

Securing High-Performance Workspace

1. Confidentiality
   - Cryptography is possible
     - Software AES at 50MB/sec
     - Hardware >200MB/Sec
     - Encrypt on-wire loads compute nodes
       - But they're waiting for I/O anyway…
2. Integrity
   - Immutability is impossible
   - Tamper-proof possible, desirable?
3. Availability already sufficient
   - Not time critical

Securing Mass Storage

1. Immutability
   - We sort of already do this (formalize?)
2. Availability
   - No lost files, yet
     - Approaching media limits
   - Tape performance issues
3. Encryption trivial
   - Lower performance requirements

Tutorial Plan

- Motivation
- Overview
- Survey of Protection
- Comparison
- Case Study: Tungsten at NCSA
- **Conclusions**
  1. **Storage Protection > Cryptography**
  2. **No Panacea**
  3. **Mission Possible!**
  4. **Challenges in the Road Ahead**

1. Storage Protection > Cryptography

- Storage security is about more than secrets
  - Secret data isn't useful if it:
    - Is tampered with
    - Is deleted
    - Fails to meet performance goals
    - Costs too much
    - Becomes unavailable

2. No Panacea

Different techniques excel at different things:

Encryption → Confidentiality

Versioning → Integrity

Redundancy → Availability

3. Mission Possible!

It is possible to secure even large, high performance storage systems:

- One must be careful of the design
- Current commercial systems are a bit short—wait a few years

4. Challenges in the Road Ahead

Usability

- Storage is complex, many faults induced by management complexity

Unification with clusters

- Clusters becoming more ubiquitous
- Cluster files systems are (somewhat) feature poor

Leveraging unique properties

- HPC and MSS are different

The End.

Questions?

<http://www.ncassr.org/projects/storage-sec/>