

## *Survivable Storage*

- ◆ This part of the discussion of survivable storage is based on the CMU paper below [Wylie-2001].
- ◆ “Selecting the Right Data Distribution Scheme for a Survivable Storage System”,
  - Jay J. Wylie, Mehmet Bakkaloglu, Vijay Pandurangan, Michael W. Bigrigg, Semih Oguz, Ken Tew, Cory Williams, Gregory R. Ganger, Pradeep K. Khosla
  - May 2001
  - CMU-CS-01-120

## *Survivable Storage*

- ◆ Design based on mature technologies from decentralized storage systems
- ◆ Key issues is the selection of the data distribution scheme
  - specific algorithms for data encoding and partitioning
  - set of values for its parameters
- ◆ Algorithms are based on
  - encryption
  - replication
  - striping
  - erasure-resilient coding
  - secret sharing
  - combinations of the above

## *Survivable Storage*

- ◆ Each algorithm has tunable parameters
- ◆ Results are schemes using different levels of
  - performance
    - » e.g. throughput
  - availability
    - » probability that data is accessible
  - security
    - » effort required to compromise confidentiality and integrity of stored data

## *Survivable Storage*

- ◆ Example
  - replication results in high availability but at high cost with respect to network bandwidth and storage
  - secret sharing provides security at lower storage and bandwidth cost
    - » disadvantage includes higher CPU utilization
    - » what happens if the number of shares increases?

## Survivable Storage

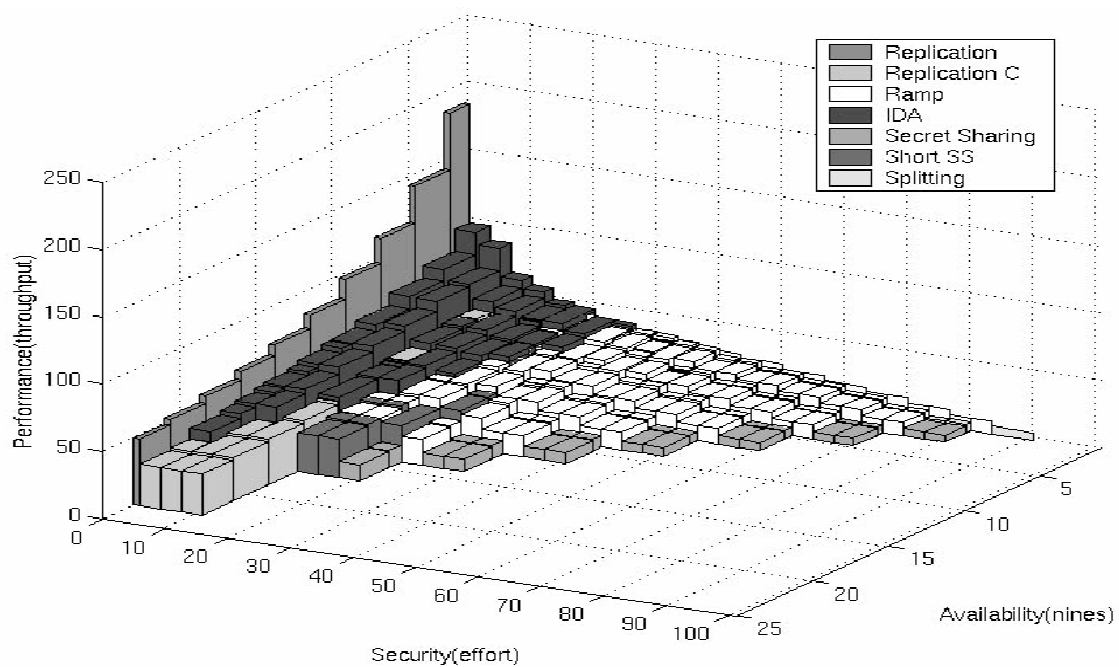
- ◆ There is no snake oil!
  - life is a compromise
- ◆ Paper enumerates on
  - possible data distribution schemes
    - » <algorithm, parameters>
  - modeling the consequences of the schemes
  - identification of best approach for given requirements

5

CS448/548 Sequence 17

## Survivable Storage

[Wylie-2001] figure 1



6

CS448/548 Sequence 17

# Survivable Storage

## ◆ Assumptions

- no individual service, node, or user can be fully trusted
- view compromised entities as common rather than the exception
- encode and distribute data across independent storage nodes
- if confidentiality is required
  - » unencoded data should not be stored on single node

7

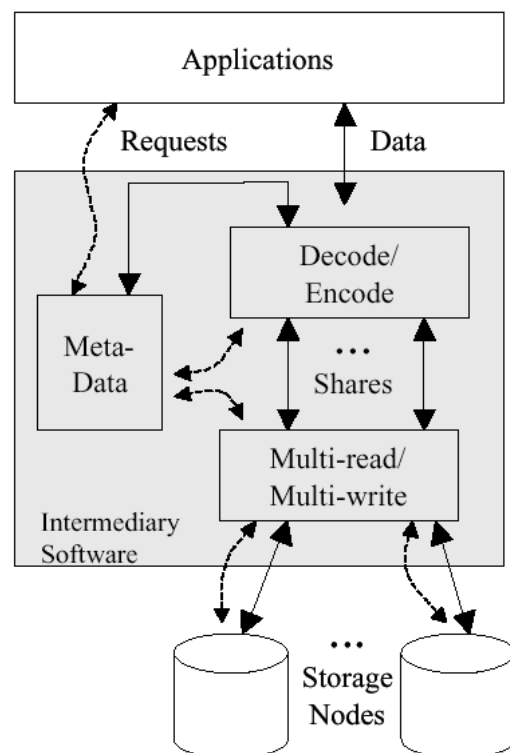
CS448/548 Sequence 17

# Survivable Storage

[Wylie-2001] figure 2

## ◆ Generic decentralized storage architecture

- solid lines trace data path
- dashed lines trace metadata path



8

CS448/548 Sequence 17

## Survivable Storage

- ◆ Threshold algorithms
  - encryption, replication, striping, erasure resilient coding, information dispersal, secret sharing
  - 3 parameters  $(p, m, n)$ 
    - »  $n$ : data is encoded into  $n$  shares
    - »  $m$ : any  $m$  shares can reconstruct the data
    - »  $p$ : less than  $p$  shares reveal no information about the encoded data

## Survivable Storage

- ◆ Replication  $(1, 1, n)$ 
  - $n$  replicas are stored
  - any single replica provides the entire data ( $m=1$ )
  - each replica reveals information about the data, in this case, about *all* the data ( $p=1$ )

## Survivable Storage

- ◆ Striping  $(1, n, n)$ 
  - large block of data is partitioned into  $n$  equally sized blocks
  - need all  $n$  sub-blocks to retrieve the data
  - each sub-block reveals some information
- ◆ Splitting  $(n, n, n)$ 
  - $n-1$  sub-blocks contain random values, 1 value is EXOR of the  $n-1$  values and the original value
  - all  $n$  sub-blocks are needed to extract the data, hence  $p=m=n$

## Survivable Storage

- ◆ Secret sharing  $(m, m, n)$ 
  - need  $m$  components to reassemble the data
  - possible implementation
    - » interpolation points on a polynomial in a finite field
    - » secret value together with  $m-1$  random values determines the encoding polynomial of order  $m-1$

## Survivable Storage

- [Ganger-2001]

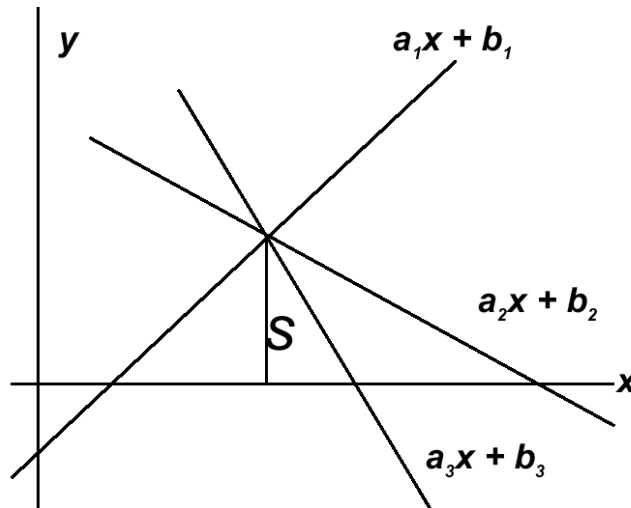


Figure 1. Blakley's secret sharing scheme with  $m=2$ ,  $n=3$ , and original data  $S$ .

13

CS448/548 Sequence 17

## Survivable Storage

- ◆ Ramp scheme  $(p, m, n)$ 
  - can be implemented using polynomial-based math
  - $p-1$  random values and  $m-(p-1)$  secret values
  - for  $p=1$  this is equivalent to information dispersal
  - for  $p=m$  this is equivalent to secret sharing

14

CS448/548 Sequence 17

## *Survivable Storage*

- ◆ In general
  - given  $N$  storage nodes there are  $N^3$  different options to consider
  - considerations
    - » availability
    - » confidentiality
    - » CPU cost
    - » storage requirements
      - implies network bandwidth

## *Survivable Storage*

- ◆ Encryption
  - common approach to protection of confidential data
  - Symmetric key encryption
    - » single parameter, i.e. key length
  - Hybrid data distribution algorithms
    - » combining replication with encryption
    - » two security issues:
      - how well is key protected
      - how difficult is crypto-analysis



## *Survivable Storage*

- ◆ Encryption cont.
  - Short Secret Sharing
    - » encrypt original with random key
    - » store key using secret sharing
    - » store encrypted data using information dispersal
    - » parameters are  $m$ ,  $n$ , and  $k$  (key length)
  - Compression
    - » can be applied to data before applying other algorithms
    - » reduce size of data

## *Survivable Storage*

- ◆ Encryption cont.
  - Cryptographic algorithms
    - » e.g. MD5, SHA-2
    - » can be applied before encoding and used to verify data integrity
    - » store signature with data or separate

# Availability

## ◆ Evaluating Availability

- Typical assumptions
  - » independence of failure, i.e. uncorrelated failures
- example  $(p,m,n)$  threshold scheme
  - » this is basically an  $m$ -of- $n$  system in fault-tolerance
    - but, be aware of the implications of the fault model,
      - e.g. benign vs. malicious
  - » let  $f_{node}$  be the probability that a node has failed or is unavailable
  - » the “read” availability of the stored information is

$$Availability_{read} = \sum_{i=0}^{n-m} \binom{n}{i} (f_{node})^i (1 - f_{node})^{n-i}$$

CS448/548 Sequence 17

# Availability

- example  $(p,m,n)$  threshold scheme
  - » write operations are more complicated
  - » system could require  $m$  to  $n$  nodes to operate correctly to write
  - » if  $n$  nodes are required  $\Rightarrow$  poor availability
  - » assume  $N > n$  storage nodes
  - » now write operation is finished if  $n$  shares have been written
    - this is essentially an  $n$ -of- $N$  system
  - » if an  $m$ -of- $N$  system is assumed, recovery would be more complicated
    - paper assumes this approach

$$Availability_{write} = \sum_{i=0}^{N-m} \binom{N}{i} (f_{node})^i (1 - f_{node})^{N-i}$$

CS448/548 Sequence 17

# Availability

- availability discussion
  - » requirements for availability are high, 0.9999...x
  - » how realistic is the assumption that failures are uncorrelated?
    - e.g. DoS attack, what are the consequences w.r.t. availability?
    - availability measures are meaningless (!)
    - availability measure are useful (!)

# Security

- ◆ Evaluating Security
  - How does one measure security?
    - » no proven metric available to date
  - One approach: reuse mathematics of fault tolerance
    - » how many nodes must be compromised to bypass confidentiality or integrity?
    - » e.g. use fault-model approach, use PRA
    - » potential problems:
      - need probabilities of being compromised
      - difficult or impossible to estimate such probabilities
      - attacks are often very correlated
        - problem with independence of faults assumption
      - how does one include concepts like encryption?
        - e.g. node is taken over, but data is still safe

## Security

- Level of Effort
  - » paper uses “effort (E) required for an active foe to compromise the security of the system”
  - » example (n,n,n)
    - threshold is n
    - assume that no encryption used
    - two ways to break system
      - break authentication
      - break in all n systems
    - effort to break confidentiality

$$E_{Conf} = \min [E_{Auth}, (n \times E_{BreakIn})]$$

## Security

- » example (n,n,n)
  - this time assume encryption is used
  - several ways to break the system
    - attempt to break the code after getting the data
    - get data and steal the key
    - trick the authentication system
  - effort to break confidentiality

$$\begin{aligned} E_{Conf} = & \min [E_{Auth}, (n \times E_{BreakIn}) \\ & + \min [E_{Cryptanalysis}, E_{StealKey}] \\ & + \min [E_{IdentifyShares}, E_{StealNames}]] \end{aligned}$$

## Security

- paper uses effort units on the security axis
  - » normalized to (0 to 100)
- $E_{breakIn}$  is assumed constant for all nodes
  - » how realistic is this?
  - » given the assumption, what constitutes a homogeneous and heterogeneous systems?
    - this is a bit different to the terms used in computer architecture

## Security

- Security definition
  - » combined term to address
    - availability
    - confidentiality
    - integrity
  - » paper considers confidentiality only
    - availability is on separate axis
    - integrity can be dealt with quite effectively and predictably

## *Performance*

### ◆ Evaluating Performance

- Model considers
  - » CPU time for encoding and decoding
  - » network bandwidth
  - » storage node response time

## *Performance*

- CPU Time
  - » encoding and decoding uses CPU extensively
  - » different schemes have different cost, differing by orders of magnitude
  - » Figure from paper considers 32kB block for  $(p,m,n)$  threshold scheme
    - $n < 26$ , different  $m$  are considered

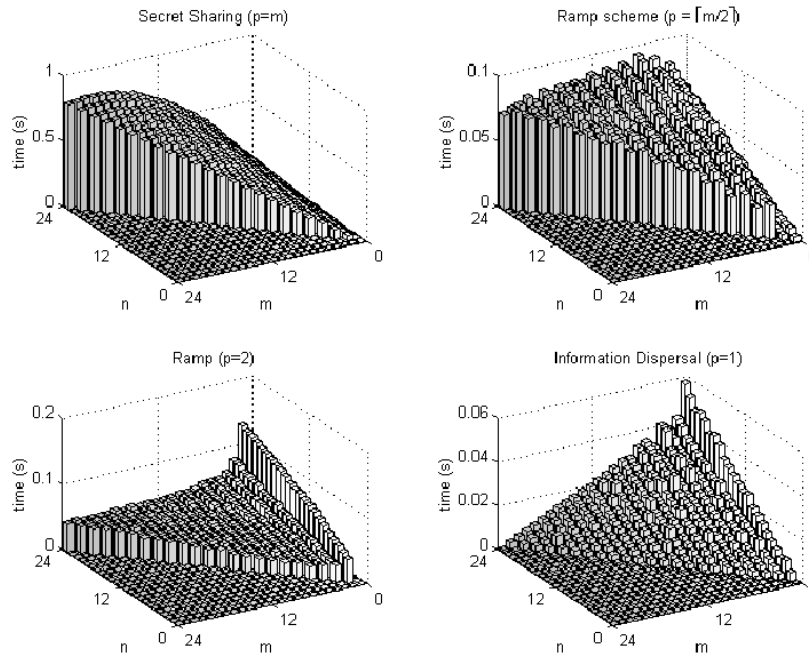


Figure 3: Measured encode times for 32 KB blocks on a 600 MHz Pentium III. All  $m$  and  $n$  combinations up to  $n = 25$  are shown for four threshold schemes: secret sharing ( $p=m$ ), ramp scheme with  $p = \lceil m/2 \rceil$ , ramp scheme with  $p = 2$ , and information dispersal ( $p = 1$ ).

## Performance

- Network Bandwidth
  - » bandwidth depends on
    - read-write ratio,
    - values for  $p, m$ , and  $n$
  - » size of each share is equal to original size divided by  $m - (p - 1)$ 
    - example (1,1,4) scheme
      - 4-fold data redundancy
    - example (1,2,4) scheme
      - now only 2-fold redundancy, i.e. data redundancy is half
  - » assumptions
    - write: all  $n$  shares must be updated
    - read: only  $m$  shares are requested

# Performance

- Network Bandwidth
  - » single bottleneck link determines the aggregate bandwidth
  - » Where is the bottleneck?
    - LAN or dial-up: client NIC
    - WAN: link at edge router through which client interacts

# Performance

- Storage Node Latency
  - » two issues to consider
    - getting data at storage node
    - moving data over the network
  - » client only sees the combined response time latency
    - modeling should consider both delays separately



# Performance

- Overall performance model
  - » for write sum:
    - CPU time +
    - storage node latency to write one share to one server +
    - $n$  times the network transition time per share
  - » for read:
    - same in reverse, however only  $m$  shares are needed