CS 3204
Operating Systems

Lecture 37

Godmar Back

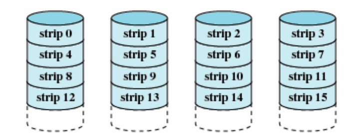
Announcements

- Project 4 due Wed, May 3, 11:59pm
- Read section 11.6 on RAID
- Skim 16.1-16.5

RAID – Redundant Arrays of Inexpensive Disks

- Idea born around 1988
- Original observation: it's cheaper to buy multiple, small disks than single large expensive disk (SLED)
  - SLEDs don't exist anymore, but multiple disks arranged as a single disk still useful
- Can reduce latency by writing/reading in parallel
- Can increase reliability by exploiting redundancy
- Several arrangements are known, 7 have "standard numbers"
- Can be implemented in hardware/software
- RAID array would appear as single physical volume to LVM

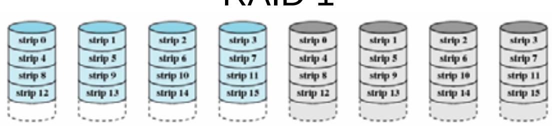
RAID 0



- RAID: Striping data across disk
- Advantage: If disk access go to different disk, can read/write in parallel, decrease in latency
- Disadvantage: Decreased reliability (MTTF(Array) = MTTF(Disk)/#disks

---

RAID 1



- RAID 1: Mirroring (all reads/writes go to both disks)
- Advantages:
  - Redundancy, Reliability – have backup of data
  - Better read performance than single disk – why?
  - About same write performance as single disk
- Disadvantage:
  - Inefficient storage use

Using XOR for Parity

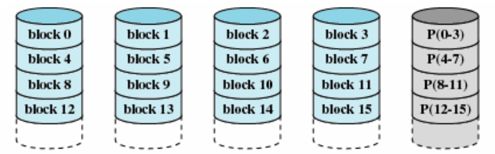
X Y Z W

- Recall:
  - X^X = 0
  - X^1 = !X
  - X^0 = X

XOR	0	1
0	0	1
1	1	0

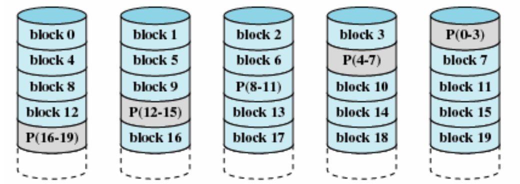
- Let's set: W=X^Y^Z
  - X^W=X^X^Y^Z=(X^X)^Y^Z=0^(Y^Z)=Y^Z
  - Y^X^W=Y^Y^Z=0^Z=Z
- Obtain: Z=X^Y^W (analogously for X, Y)

RAID 4



- RAID 4: Striping + Block-level parity
- Advantage: need only N+1 disks for N-disk capacity & 1 disk redundancy
- Disadvantage: small writes (less than one stripe) may require 2 reads & 2 writes
  - Read old data, read old parity, write new data, compute & write new parity
  - Parity disk can become bottleneck

RAID 5



- RAID 5: Striping + Block-level Distributed Parity
- Like RAID 4, but avoids parity disk bottleneck
- Get small read latency advantage
- Best large read & large write performance
- Only remaining disadvantage is small writes

Security & Protection

Security Requirements & Threats

- Requirement
  - Confidentiality
  - Integrity
  - Availability
  - Authenticity
- Threat
  - Interception
  - Modification
  - Interruption
  - Fabrication

The goal of a protection system is to ensure these requirements and protect against accidental or intentional misuse

Policy vs Mechanism

- First step in addressing security: separate the *what* should be done from the *how* it should be done part
- A protection system is the mechanism that enforces the security policy
- The security policy specifies what is allowed and what is not

Protection: AAA

- Core components of any protection mechanism
- Authentication
  - Verify that we really know who we are talking to
- Authorization
  - Check that user X is allowed to do Y
- Access enforcement
  - Ensure that authorization decision is respected
  - Hard: every system has holes
- Social vs technical enforcement

Authentication Methods

- Passwords
  - Weakest form, and most common
  - Subject to dictionary attacks
  - Passwords should not be stored in clear text, instead, use one-way hash function
- Badge or Keycard
  - Should not be (easily) forgeable
  - Problem: how to invalidate?
- Biometrics
  - Problem: ensure trusted path to device

Authorization

- Once user has been authenticated, need some kind of database to keep track of how they are allowed to do
- Simple model:
  - *Access Matrix*

Objects (e.g. files, resources)

Principals (e.g. users)

	File 1	TTY 2
User A	Can Read	Exclusive Access
User B	Can R/W	--

Representing Access Matrices

- Problem: access matrices can be huge
  - How to deal with them in a condensed way?
- Two choices:
- By row: Capabilities
  - What is principal X allowed to do?
- By column: Access Control Lists
  - Who has access to resource Y?

Access Control Lists

- General: store list of <user, set of privileges> for each object
- Example: files, for each file store who is allowed to access it (and how)
- Most filesystems support it.
- Groups can be used to compress the information:
  - Old-style Unix permissions rwxr-xr-x
- Q.: where in the filesystem would you store ACLs/permissions?

Capabilities

- General: store (capability) list of <object, set of privileges> for each user
- Typically used in systems that must be very secure
  - Default is empty capability list
- Capabilities also often function as names
  - Can access it if you know the name
  - Must make names unforgeable, or must have system monitors who holds what capabilities

Trusted Computing Base

- The part of the system that enforces access control decisions
  - Also protects authentication information
- Issues:
  - Bug in TCB, security is compromised
  - Need to keep it small and manageable
  - Usually: entire kernel is part of TCB (huge!)
    - Weakest link phenomenon