Tactics for Availability (1)

主讲教师：王灿
Email: wcan@zju.edu.cn
TA: 李奇平 liqiping1991@gmail.com
Course FTP: ftp://sa:sa@10.214.51.13

Availability

- Availability = reliability + recovery
  - Availability is about minimizing system failure time by mitigating faults
- Failure VS fault:
  - A “system failure” is observable by the system’s user.
  - A system fault may cause a “system failure” or it might be masked.
- **α = MTBF / (MTBF + MTTR)**

System Availability Requirements

Table 5.1. System Availability Requirements

Availability	Downtime/90 Days	Downtime/Year
99.0%	21 hours, 36 minutes	3 days, 15.6 hours
99.9%	2 hours, 10 minutes	8 hours, 0 minutes, 46 seconds
99.99%	12 minutes, 58 seconds	52 minutes, 34 seconds
99.999%	1 minute, 18 seconds	5 minutes, 15 seconds
99.9999%	8 seconds	32 seconds

P81 Table 5.1

Availability General Scenario (1)

- Stimulus---a fault of following classes:
  - Omission
  - Crash
  - Incorrect timing
  - Incorrect response
- Source of stimulus
  - Internal/external: people, hardware, software, physical infrastructure…

Availability General Scenario (2)

- Response: reactions to a system fault
  - Detect and isolate the fault
    - Log and notify
  - Recover from the fault
    - Disable the faulty source
    - Fix the fault and contain damage or
    - Operate in a degraded mode at meantime
- Response measure
  - Availability percentage (e.g., 99.999%)
  - Time to detect and repair the fault, time in degraded mode

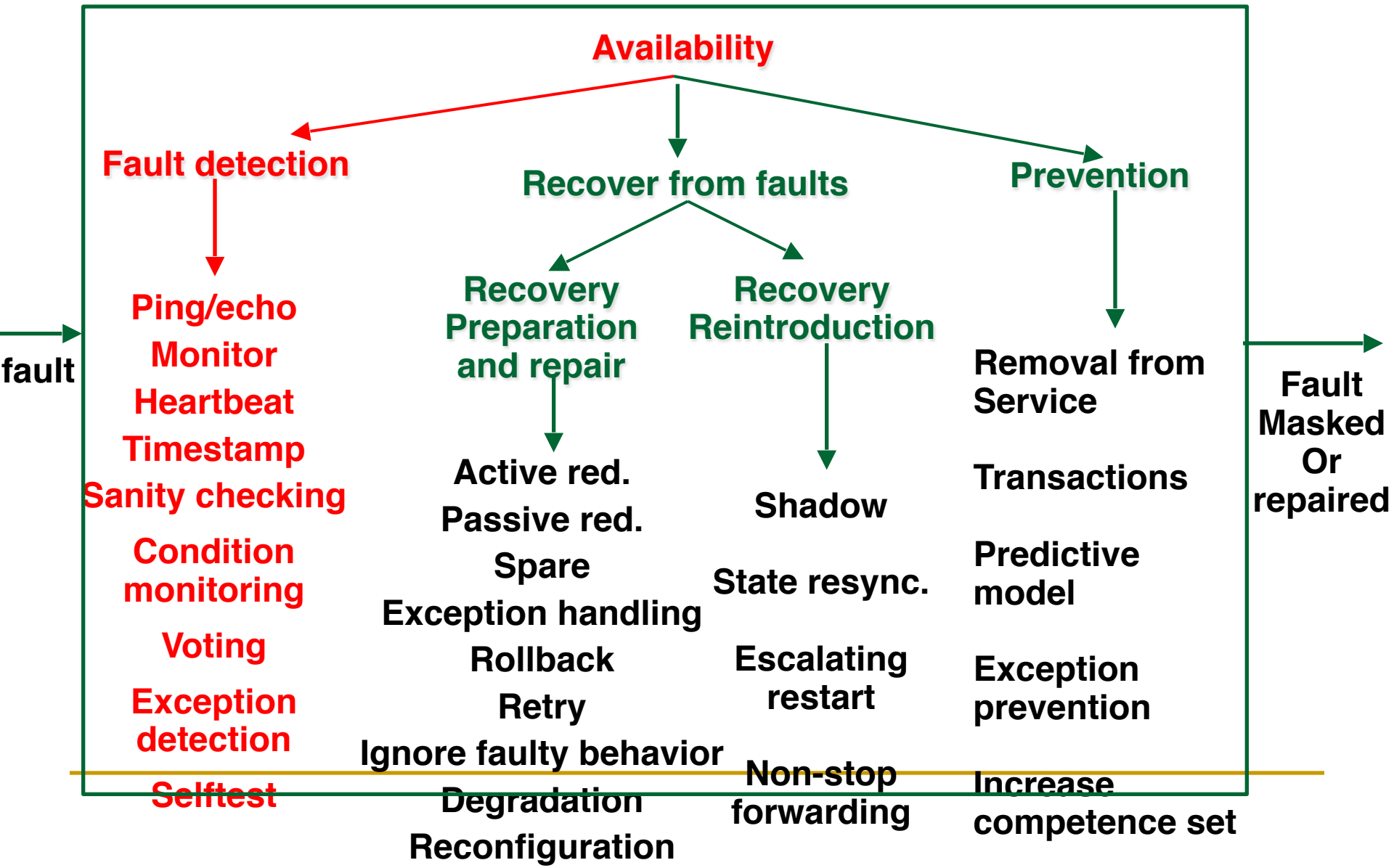
Availability General Scenario (3)

- Artifact
  - System's processors, communication channels, persistent storage, processes
- Environment
  - Normal operation, startup, shutdown, repair mode, degraded operation, overloaded operation

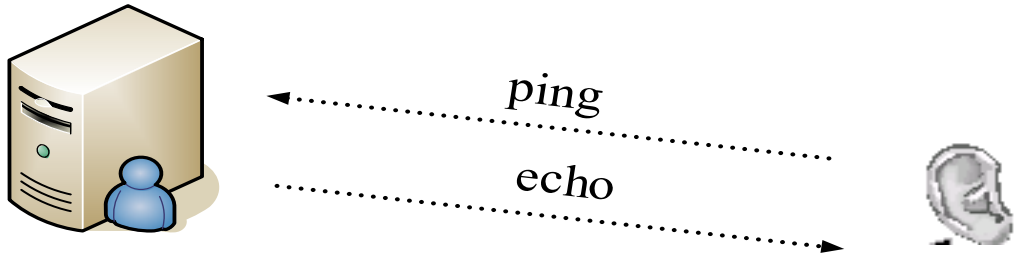
Availability Tactics

- How to build a highly available system
  - Keep faults from becoming failures
  - Limit the effects of a fault and make repair
- Approaches to maintain availability include:
  - Monitoring or detecting faults
  - Recovering from failure
  - Preventing faults

Availability Tactics Hierarchy

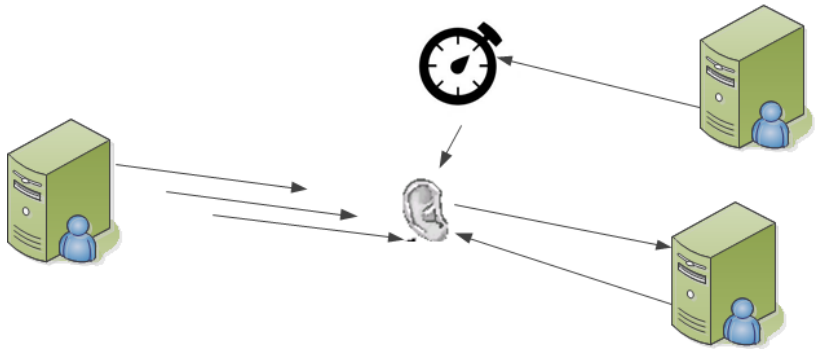


Ping/Echo



- Ping/echo determines the reachability and the round-trip delay
  - Sometimes also check whether the pinged component is alive and responding correctly
- Often sent by a system monitor
- Ping/echo requires a time threshold to be set

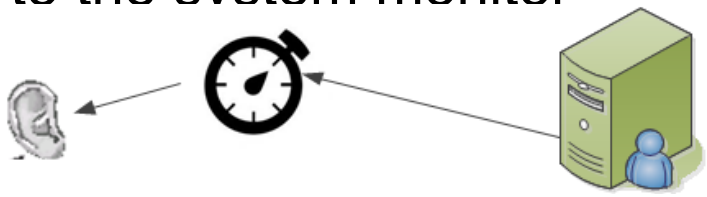
Monitor, Watchdog and Heartbeat (1)



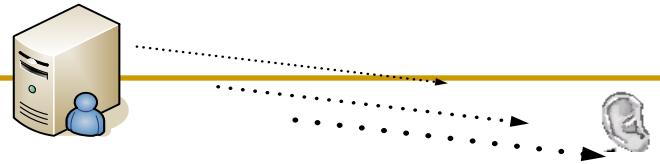
- A system monitor watches the running state of other system components: processors, processes, I/O, memory, detect failure or congestion in the network or other shared resources
  - E.g. Process monitor in Oracle
  - Detect failure or congestion in the network or other shared resources

Monitor, Watchdog and Heartbeat (2)

- Watchdog: the detection mechanism is implemented using a counter or timer that is periodically reset by the process being monitored ("petting the watchdog")
  - The expiration of the timer indicates to the system monitor of a fault occurrence in the process



- Heartbeat: a periodic message exchange between a system monitor and a process being monitored
  - Reducing overhead by piggybacking heartbeat messages on to other control messages being exchanged between the process being monitored and the distributed system controller.



Time Stamp

- ***Time stamp*** detects incorrect sequences of events, primarily in distributed message-passing systems
  - Assigning a sequence number or local clock to the event immediately after it occurs

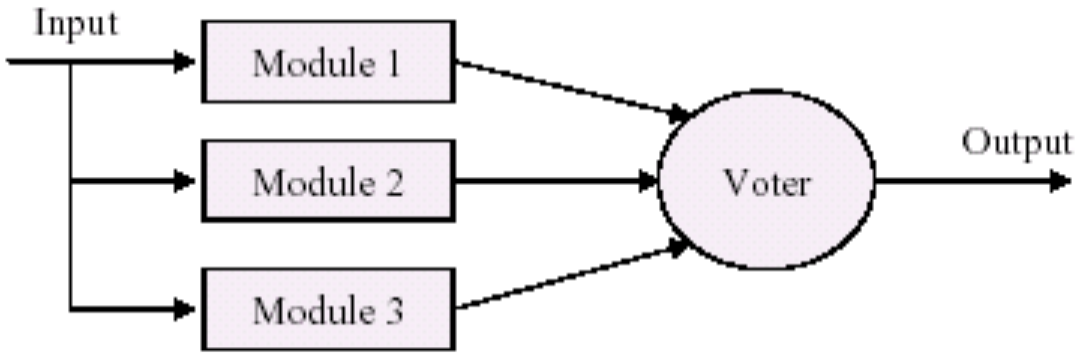
Sanity Checking & Condition Monitoring

- ***Sanity checking*** checks the validity or reasonableness of specific operations or outputs of a component
- ***Condition monitoring*** involves checking conditions in a process or device, or validating assumptions made during the design
  - E.g. checksums

Voting (1)

- Processes running on redundant processors each take the equivalent input, compute and report a simple output to a “voter”

TMR



  - Majority rules
  - Preferred component

Voting (2)

- Replication: multiple copies of identical components
  - Effective in protecting against random failures of hardware, but this cannot protect against design or implementation errors
- Functional redundancy
  - Components must always give the same output given the same input
  - But they are diversely designed and diversely implemented: separate processors, separate implementation teams, … dissimilar platforms

Voting (3)

- *Analytic redundancy*: Functional redundancy + diversity among the components' inputs and outputs
  - E.g. avionics programs use multiple ways to compute aircraft altitude: barometric pressure, the radar altimeter, and geometrically using the straight-line distance and look-down angle of a point ahead on the ground.

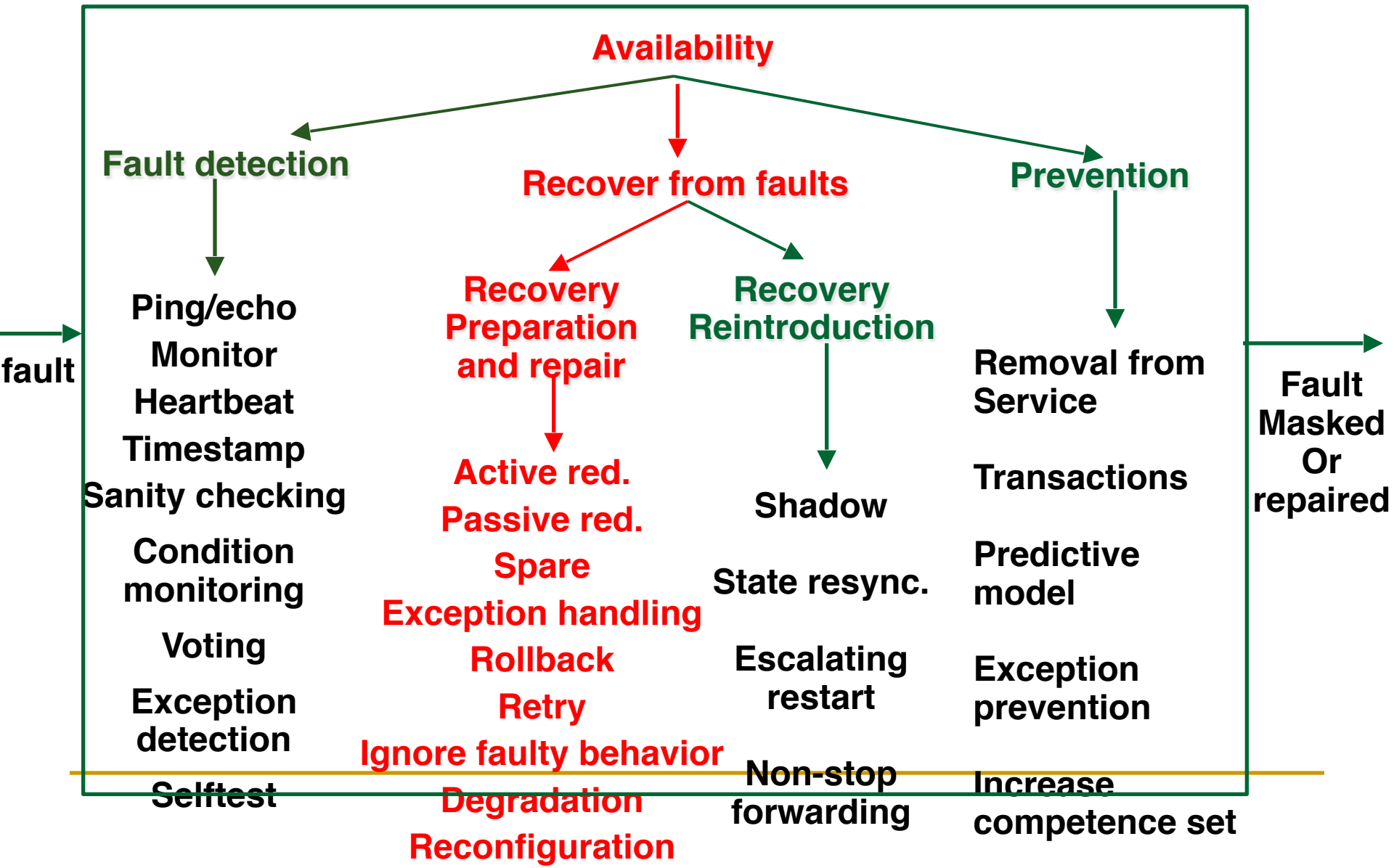
Exception Detection

- System exception
  - System raises an exception when it detects a fault (divide by zero, bus and address faults, il-legal program instructions)
- Parameter fence
  - A special data pattern placed immediately after any variable-length parameters of an object to detect memory overwriting
- Parameter typing
  - TLV, Strong typing
- Timeout
  - Exceptions raised when time constraints fail

Self-test

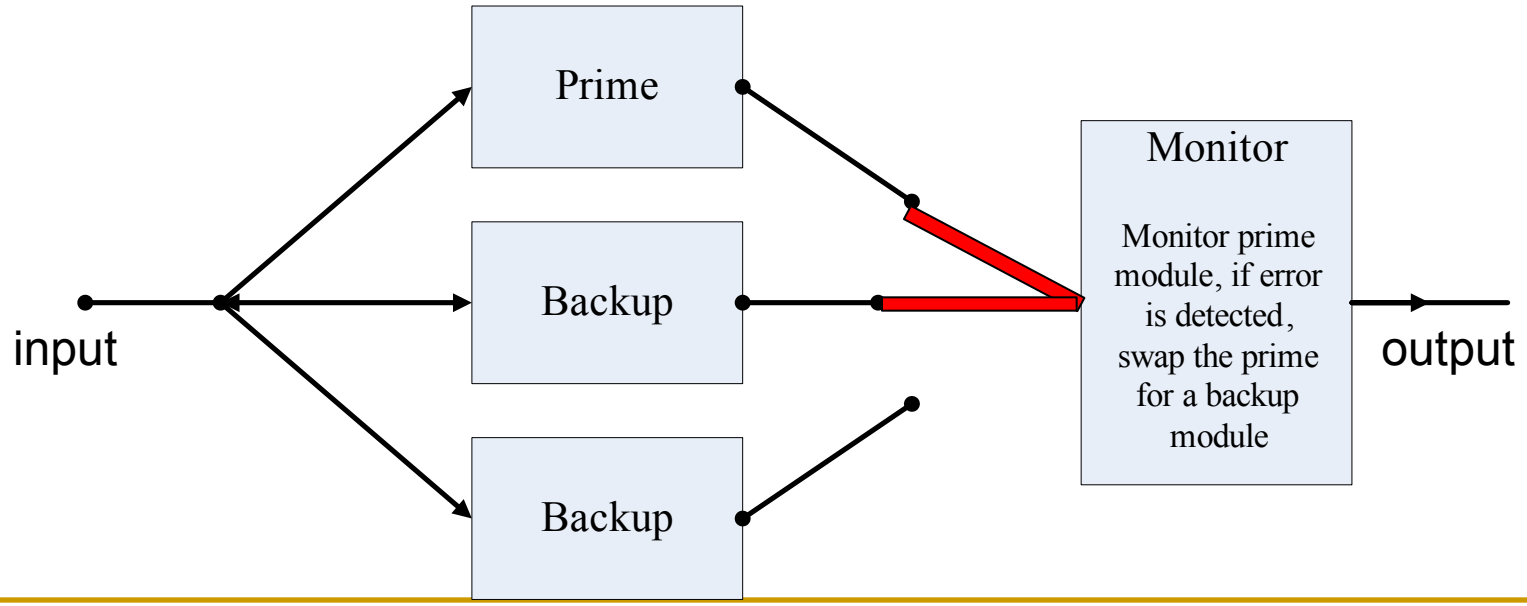
- Components run procedures to test themselves for correct operation
  - Can be initiated by the component itself or invoked from time to time by a system monitor

Availability Tactics Hierarchy



Active Redundancy (Hot Spare) (1)

- Redundant components synchronized at start and respond to events in parallel, usually the first component to return is accepted as the correct answer.

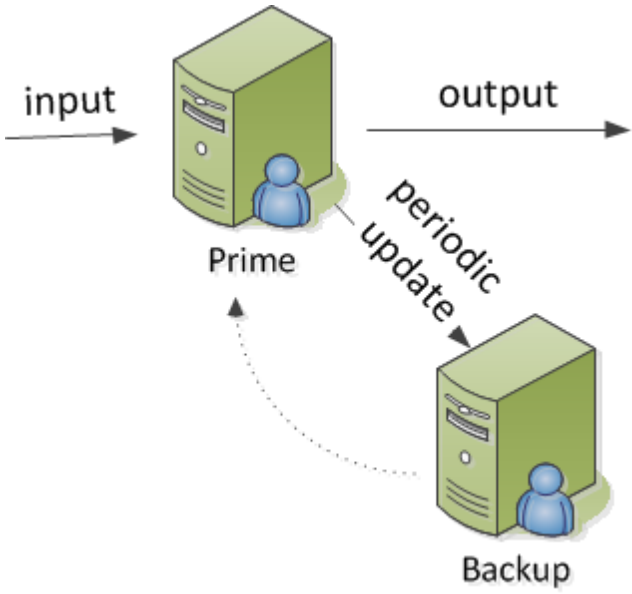


Active Redundancy (2)

- Synchronization
  - Ensuring that all messages to any redundant component are sent to all redundant components.
- Downtime is switching time
  - Usually only milliseconds (switching to another component).
- E.g. highly available LANs, DB

Passive Redundancy (Warm Spare)

- One component responds to events and informs the standbys of state updates.
- Synchronization is the responsibility of the primary component
  - May use atomic broadcasts to the secondaries to guarantee synchronization.
- Upon failure the system must:
  - Ensure that the backup is sufficiently fresh (checkpoint + log).
  - The standby components take over from the primary.



Spare (Cold Spare)

- A standby spare computing platform is configured to replace many different failed components.
- The spare must be rebooted to the appropriate software configuration and have its state initialized when a failure occurs.
- Synchronization and taking over: checkpoint + logging
  - A checkpoint is an event that flushes the modified data from the buffer cache to the disk, resulting in a consistent snapshot.

Exception Handling

- Information return from an captured exception depends on development environment
  - Error codes
  - Exception classes
    - Containing helpful information such as the name of the exception thrown, the origin of the exception, and the cause of the exception thrown etc.
- Exception information can be used to mask the fault by correcting the cause of the exception and retrying the operation

Rollback

- In case of failure, ***rollback*** reverts the system to a previous known good state, referred to as the "rollback line"
  - A copy of a previous good state can be saved by making a ***checkpoint*** of the system
  - Usually combines with active or passive redundancy
    - After a rollback, a standby version of the failed component is promoted to active status

Retry

- In networks and in server farms where failures are common and usually transient, the ***retry*** tactic will retry the failed operation which usually lead to success
  - A limit on the number of retries shall also be placed

Ignore Faulty Behavior

- Ignoring messages sent from a particular source when we determine that those messages are spurious
  - E.g. ignoring the message from a specific source in an DOS attack

Degradation

- Maintaining the most critical system functions in the presence of component failures, dropping less critical functions

Reconfiguration

- Recovering from component failures by reassigning responsibilities to the (potentially restricted) resources left functioning, while maintaining as much functionality as possible

Reading Assignment

- Read Chapter 5 of the textbook.