Parul® University
Vadodara, Gujarat

NAAC GRADE A++

Information and Communication Technology

Deadlocks & Secure Resource Management

Prof. Khushal Bhoyar
Department of CSE (Cyber Security)
PIET, Parul University

5.1 Deadlock Characterization and Conditions

Definition: Deadlock is a system state where two or more processes are unable to proceed because each is waiting for a resource held by another.

Four Necessary Conditions (all must exist simultaneously):
Mutual Exclusion: Resources are non-sharable; only one process can use a resource at a time.
Hold and Wait: A process holds at least one resource while waiting to acquire additional resources held by other processes.
No Preemption: Resources cannot be forcibly taken away from a process; they must be released voluntarily.
Circular Wait: A circular chain of two or more processes exists, where each process waits for a resource held by the next process in the chain.

-cont

Resource Allocation Graph (RAG):
A model used to represent processes, resources, and their relationships (requests and assignments).
The presence of a **cycle** in the graph indicates a potential deadlock.
For single resource instances, a cycle **guarantees** a deadlock.
For multiple resource instances, a cycle indicates a **possibility** but not a certainty of deadlock.

Deadlock Management:
Detection: Algorithms periodically examine the system state (e.g., using the RAG or wait-for graphs) to identify deadlocks.
Recovery: Strategies to restore system operation after detection, primarily through:
Process Termination: Terminating one or more deadlocked processes.
Resource Preemption: Preempting (taking away) resources from one or more processes and allocating them to others.

5.2 Deadlock Prevention and Avoidance

Deadlock Prevention:
Aims to **eliminate at least one** of the four necessary conditions.
Eliminate Mutual Exclusion: Not possible for all resources, but can be applied to resources that are inherently sharable (e.g., read-only files).
Prevent Hold and Wait:
Require processes to request **all** resources they will need **before** execution begins (**static allocation**).
Force a process to **release all** its currently held resources before it can request any new ones.
Allow Preemption: If a process requests a resource that is not available, its currently held resources may be preempted.
Prevent Circular Wait:
Impose a **total ordering** (a numbering) on all resource types.
Require processes to request resources **only in increasing order** of these numbers, preventing circular waits.

-cont

Deadlock Avoidance:

- Uses **advance knowledge** of future resource requests (maximum needs) to make allocation decisions.
- The system must guarantee that it will always remain in a **safe state** (a state where there exists a sequence for all processes to finish without deadlock).
- **Banker's Algorithm:** A classic avoidance algorithm that simulates potential resource allocation to determine if the resulting state would be safe before granting a request.

-cont

Trade-offs:

- Prevention methods are generally **more restrictive**, leading to lower device utilization and reduced system throughput.
- Avoidance is less restrictive than prevention but requires knowledge of future requests and involves runtime overhead.
- Both involve trade-offs between **resource utilization, system throughput, and implementation complexity**.

5.3 Security Risks in Resource Management

Key Risks and Attacks:

- **Denial of Service (DoS) Attacks:** Malicious attempts to **exhaust** critical system resources (memory, CPU, network connections) to deny service to legitimate users.
- **Starvation:** A process **indefinitely waits** for a resource due to unfair allocation policies or malicious resource monopolization by another actor.
- **Resource Exhaustion Attacks:** Exploit system limits by making **excessive allocation requests** or by **failing to release resources** (e.g., memory leaks, file descriptor leaks).

-cont

Secure Resource Allocation Policies:

- **Quotas:** Set limits on resource consumption (CPU time, memory, disk space) per user or process to prevent any single entity from monopolizing resources.
- **Aging:** Gradually increase the priority of processes that have been waiting a long time for a resource to **prevent starvation**.
- **Fair-Share Scheduling:** Allocate resources (especially CPU) equitably among users or groups to ensure **equitable distribution**.
- **Resource Reservation:** Guarantee a **minimum allocation** of resources for critical processes to ensure they can run even under attack or heavy load.

Monitoring and Auditing:

- Continuously **track resource usage patterns** (who is using what, and how much).
- Used to **detect anomalies** that may indicate an attack (like a sudden spike in connections) or a system malfunction.
- Enables **proactive response** (like throttling or blocking a suspicious user) before severe service degradation occurs.

1. https://www.druva.com/glossary/what-is-a-disaster-recovery-plan-definition-and-related-faqs

2. https://www.konverge.co.in/virtualization-in-cloud-computing-need-types-and-importance/

3. https://www.crowdstrike.com/cybersecurity-101/cloud-security/cloud-application-security/

Parul® University

NAAC GRADE A++

https://paruluniversity.ac.in/