Operating Systems Concepts - Notes

1. Child Process - fork()

`fork()` is a system call used to create a new process by duplicating the current process.

- The new process is called the child process, and the original is the parent.
- Return values:
  * 0 : Child process
  * >0 : Parent process (returns PID of the child)
  * <0 : Error

Example:

```
  pid_t pid = fork();
  if (pid == 0) printf("Child process\n");
  else if (pid > 0) printf("Parent process\n");
```

2. Handling Common Signals

Signals notify a process of system-level events like interrupts.

- Common signals: SIGINT (Ctrl+C), SIGTERM, SIGKILL, SIGSEGV
- Use `signal()` or `sigaction()` to handle them.

Example:

```
  void handler(int sig) { printf("Caught signal %d\n", sig); }
  signal(SIGINT, handler);
```

3. Exploring Different Kernel Crashes

Kernel crashes may be caused by:

- Null pointer dereference
- Stack overflow
- Invalid memory access

Tools: dmesg, journalctl, kdump

Check logs in /var/log/, use kdump to capture crashes.

4. Time Complexity

Operating Systems Concepts - Notes

Measures how running time grows with input size.

- O(1): Constant
- O(log n): Binary Search
- O(n): Linear Search
- O(n log n): Merge Sort
- O(n^2): Bubble Sort

Helps ensure efficient algorithms.

5. Locking Mechanism - Mutex / Spinlock

Mutex:

- Puts waiting threads to sleep.
- Good for long waits.

Example:

```
  pthread_mutex_t lock;
  pthread_mutex_lock(&lock);
  pthread_mutex_unlock(&lock);
```

Spinlock:

- Spins (busy waits) on the CPU.
- Ideal for short critical sections.

Used in low-latency or kernel-space programming.