**Deadline:** 1 November 2024, 9:50 AM

**Problem Statement:** Analyzing SHA-256 Implementation in Kernel Space, User Space, and Through System Calls

**Objective:**  
You are required to implement and analyze the SHA-256 cryptographic hash function in three different environments: kernel space, user space, and via system calls. The goal is to understand the differences in implementation, execution, and performance between these environments. You are expected to document every step of the process in detail, from setup and development to testing and analysis.

**Problem:**  
Kernel Space Implementation:  
- Implement the SHA-256 hashing algorithm within the Linux kernel (as a kernel module).  
- Use appropriate kernel cryptography libraries (e.g., crypto/hash.h) for the implementation.  
- Test your kernel module by hashing a string or file and outputting the result to the kernel log (via dmesg).  
User Space Implementation:  
- Implement the same SHA-256 hashing algorithm in user space using a standard user-space cryptography library (e.g., OpenSSL or cryptlib).  
- Create a user-space application that reads input from a file or string, hashes the data using SHA-256, and outputs the result.

System Call-Based Implementation:  
- Investigate and implement a system call that hashes data using SHA-256 by calling cryptographic functionality provided by the kernel.  
- Compare the use of this system call to both the kernel space and user space implementations in terms of ease of use, performance, and security.

**Requirements:**  
Documentation:  
Document every step of your development process in detail, including:  
- Setting up the development environment for kernel module development and user-space applications.  
- Writing the code for each implementation, with explanations of how the SHA-256 algorithm is integrated.  
- How you test each implementation (e.g., sample input, output, testing tools, and validation).  
- Challenges faced during development and how they were overcome.  
- Tools and libraries used for both kernel and user-space implementations.

**Performance Analysis:**  
- Run performance benchmarks comparing the three implementations (kernel space, user space, and system call).  
- Document the time taken for the hash computation on identical inputs.  
- Discuss the results and identify any bottlenecks or performance optimizations.

**Security Considerations:**  
- Provide a detailed discussion of security implications of running SHA-256 in each environment (kernel space vs. user space).  
- How does the choice of environment affect the security of the hashing operation?   
- Consider the risk of vulnerabilities, access control, and privilege escalation.

**Deliverables:**  
- Source code for all three implementations (kernel space, user space, and system call).  
- Detailed documentation covering every aspect of your work.  
- A report analyzing the performance and security aspects of each implementation, including tables or graphs where necessary.

**Assessment Criteria:**  
Correctness and completeness of the SHA-256 implementations.  
Quality of the documentation (clarity, thoroughness, and detail).  
Quality of the performance and security analysis.  
Depth of understanding demonstrated in addressing the problem and overcoming challenges.  
This task will give you hands-on experience in both kernel and user-space programming, as well as in working with cryptographic functions in a low-level system environment.