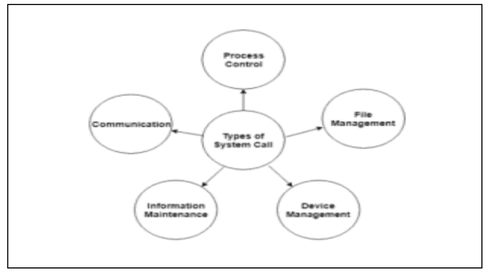
# Modifying CMP408 Application with Spy System Calls

**Ethan Hastie**

**CMP408**

## Introduction

The separation of the OS into two main modes, the user and kernel mode, provides a higher level of stability and security by allowing the kernel mode full access to resources while user space possesses limited access (IONOS, 2020). System calls are the only method to allow user space programs access to resources. Common categories of system calls include those shown below (Pal, 2019):




Valuable data can be captured from these functions if they are hijacked, such as files opened or network data, or their behaviour can be changed entirely, without alerting the user. If used maliciously, it can act as a rootkit (GoldenOak, 2020) burying itself in the target system and masking its presence.

The main aim of this project is to demonstrate the use of spy system calls within the CMP408 demo application. To accomplish the aim, the following objectives must be met:

- Create the malicious system call using an LKM to hijack the connect system call and log network data such as IP addresses and port numbers.
- Using the generated log data, filter the data and send this to a database hosted in AWS.

## Methodology

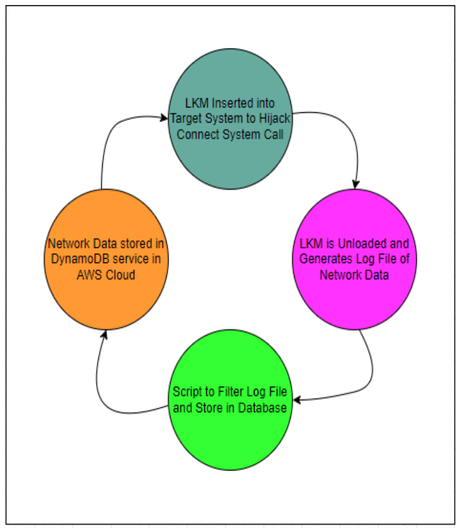
The CMP408 application was modified with two extra LED's that would turn on and off when data was sent to AWS. To modify the connect() system call, an LKM was created and inserted into the Raspberry Pi that would store the base address of the system call table. This would be used to obtain the addresses of the system calls held within the system call table and store the original addresses. Once the system call addresses were obtained, the original addresses held within the system call table can be overwritten to point to the new code whenever that function is invoked. When the connect() system call is invoked it will point to a fake system call programmed within the LKM that extracts network information such as IP addresses and port numbers (Patel, 2008), allowing it to spy on network connections to the CMP408 application and log it to a file, as well as run the original connect() system call.

```
long this_will_be_fake_connect_call(*aargs) {

        # modified code here

        return real_connect_call(*aargs)  # optional

}
```

Within the modified system call, it calls two other system calls: getsockname() and getpeername(). Both perform the same function except the former returns the address of the source socket while the latter obtains the address of the peer socket. It also obtains the UID that invoked the connect() system call.

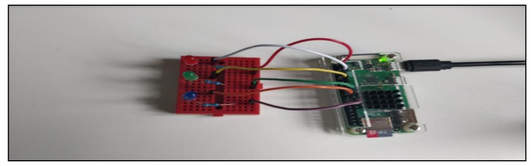
Once the LKM is unloaded from the Raspberry Pi, it generates the log files with the network information and resets the system call table to its previous state. A script is then applied to the log file that extracts the data and sends it to the DynamoDB service run from AWS cloud using Boto3, which is the AWS SDK for Python used to interact with AWS services. When the script is invoked, the LED's will indicate when data is successfully sent to the database. Using the AWS management console, data such as port numbers, IP addresses, system call types and UID's of the invoked system calls can be viewed, which can allow it to monitor for the collected system calls (Patel, 2008) and in this case spy on the targeted system.

On the right, the diagram demonstrates the full process involved with the LKM to generate the log file and upload the data to AWS.




## Project Highlights

All components of the project are well addressed concerning the hardware, software and cloud elements. The most important element of the project was the software component as the LKM was required to create the modified system call. DynamoDB was an effective choice to store the data in, allowing for future scaling and efficient management. The setup was also efficient and allowed data to be sent to it seamlessly from the Raspberry Pi device. The LED's connected to the device allow for clear indication of AWS connection.



## Future Work

The project can be improved to record the correct network data. A wider collection of system calls can be recorded including those dealing with files. Data sent from the Pi to AWS can be programmed to upload logs on a periodic basis and the LKM can be modified to record timestamps data.

## References

Patel, B., 2008. *User Activities Monitoring System Using LKM.* California: California State University. [Online] Available at https://csu-csus.esploro.exlibrisgroup.com/esploro/outputs/graduate/User-activities-monitoring-system-using-LKM/99257830873801671

GoldenOak, 2020. *Linux Kernel Module Rootkit — Syscall Table Hijacking.* [Online] Available at: https://infosecwriteups.com/linux-kernel-module-rootkit-syscall-table-hijacking-8f1bc0bd099c

IONOS, 2020. *System calls: What are system calls and why are they necessary?.* [Online] Available at: https://www.ionos.com/digitalguide/server/know-how/what-are-system-calls/

Pal, T., 2019. *System Calls in Operating Systems - Simple Explanation.* [Online] Available at: https://technobyte.org/system-calls-in-operating-systems-simple-explanation/