

Network Management

Khula Molapo

20240001

Week: 3

1.

A smart agriculture farm uses IoT sensors to monitor soil moisture, temperature, humidity, and water tank levels across large farming areas. Each sensor device is connected to a network and configured as an SNMP agent. A central monitoring server works as the SNMP manager and continuously collects performance data from these devices. The manager sends SNMP GET requests to retrieve information such as device status, battery levels, and environmental readings. When abnormal values occur, such as extremely low soil moisture or high temperature, the SNMP agents send TRAP messages to alert the manager instantly. The network administrator uses monitoring software to view device health, uptime, and network traffic. Polling intervals are optimized to avoid excessive traffic while maintaining accurate monitoring. Access control is implemented to prevent unauthorized users from reading or modifying device data. Through SNMP monitoring, farmers receive real-time insights, allowing them to automate irrigation systems and prevent crop damage while improving efficiency and reliability.

2.

SNMPv3 is an improved version of the Simple Network Management Protocol that focuses strongly on security. Earlier versions used simple community strings, which were easy to intercept. SNMPv3 introduces authentication, encryption, and message integrity to protect network management data. Authentication ensures only authorized users and devices communicate with the SNMP manager. Encryption protects sensitive information such as configurations and performance statistics from being read during transmission. Message integrity checks prevent attackers from modifying packets while they are being sent across the network. SNMPv3 also supports role-based access control, allowing administrators to define permissions for different users, improving overall network security.

3.

Example Simulation Code:

```
#include <iostream>
#include <string>
using namespace std;

class SNMPAgent {
public:
    string deviceName;
```

```

int cpuUsage;

SNMPAgent(string name, int cpu) {
    deviceName = name;
    cpuUsage = cpu;
}

void sendResponse() {
    cout << "SNMP Response from " << deviceName
        << " | CPU Usage: " << cpuUsage << "%" << endl;
}
};

int main() {
    SNMPAgent router("Router-01", 35);
    cout << "SNMP Manager requesting device status..." << endl;
    router.sendResponse();
    return 0;
}

```

Reflection: While creating a basic SNMP simulation in Visual Studio Code using C++, I learned how SNMP communication works conceptually between a manager and an agent. The program simulated how a manager sends a request and how an agent responds with device information such as CPU usage. Writing classes helped represent network devices logically. Testing the program also showed how monitoring systems collect device performance data automatically. Although real SNMP implementations require networking libraries, building a simplified simulation helped me understand the core communication flow first. Overall, the exercise improved my understanding of network monitoring logic and object-oriented programming.

4.

An SNMP architecture diagram shows how network devices communicate with a central monitoring system. The diagram includes an SNMP manager, multiple SNMP agents, and the network connecting them. The manager sends requests to devices and collects performance information. SNMP agents provide status data when requested. Communication flows through GET, SET, and TRAP messages. GET retrieves information, SET changes configurations, and TRAP sends automatic alerts when problems occur. The diagram may also include a Management Information Base (MIB), which defines how device data is structured. By visualizing the architecture, administrators understand monitoring processes, communication paths, security points, and potential bottlenecks.

