Problema 1: Diseño de red segura

Escenario

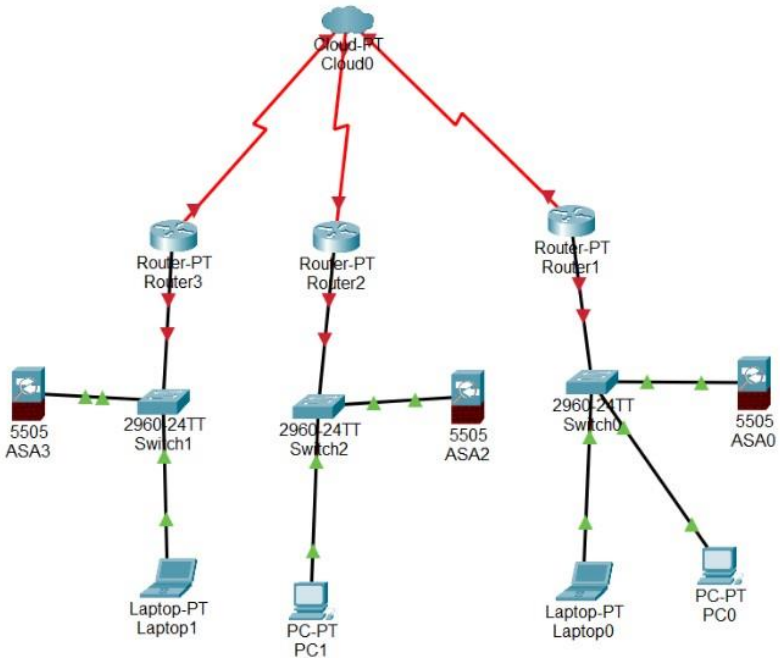
Una empresa necesita diseñar una red segura que conecte tres sucursales ubicadas en diferentes ciudades utilizando tecnología WAN y LAN. La empresa maneja datos confidenciales y requiere que la comunicación entre sucursales sea cifrada.

Preguntas

1. ¿Qué tipo de tecnología de WAN utilizarías para conectar las sucursales y por qué?
   Para conectar las sucursales ubicadas en diferentes ciudades, optaríamos por MPLS (Multiprotocol Label Switching). MPLS proporciona un alto nivel de seguridad y calidad de servicio (QoS), lo que es crucial para una empresa que maneja datos confidenciales y requiere comunicación cifrada entre sucursales. Además, MPLS permite la creación de redes privadas virtuales (VPN) fácilmente configurables, lo que garantiza la privacidad y seguridad de los datos transmitidos.

2. Describe cómo implementarías el cifrado en la red. ¿Qué tipos de claves y protocolos utilizarías?
   Para asegurar una comunicación cifrada de extremo a extremo, utilizaríamos el protocolo IPsec (Internet Protocol Security). IPsec proporciona autenticación, integridad de datos y confidencialidad mediante la encapsulación de paquetes IP en un túnel seguro. Utilizaría claves precompartidas para la autenticación y el cifrado, junto con el algoritmo AES (Advanced Encryption Standard) para garantizar una alta seguridad.

3. Dibuja una topología de red que incluya dispositivos como routers, switches, y firewalls. Explica la función de cada dispositivo en tu diseño. Puedes utilizar PacketTracer



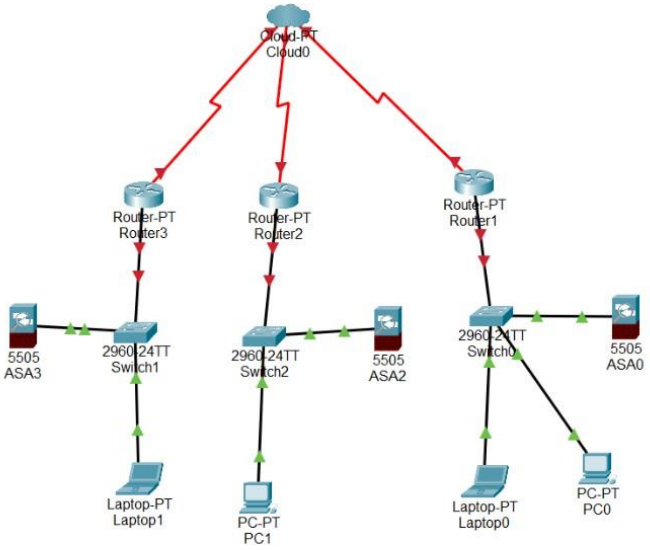
4. ¿Cómo garantizarías la integridad y autenticidad de los datos transmitidos entre las sucursales? Detalla el uso de checksums o CRC.

Para garantizar la integridad y autenticidad de los datos transmitidos entre las sucursales, implementaría checksums o CRC (Cyclic Redundancy Check) en cada router. Estos mecanismos de verificación de integridad permiten detectar cualquier alteración en los datos durante la transmisión y aseguran que los datos recibidos sean los mismos que los enviados.

Parte 1: Diseño de topología de red

Preguntas

1. Dibuja una topología de red para este escenario que incluya los dispositivos de red necesarios en cada sucursal.



2. Explica cómo cada dispositivo contribuye a la seguridad y eficiencia de la red.

    - **Router**: Actúa como el punto de conexión a la red WAN/MPLS, gestionando el tráfico entrante y saliente hacia y desde otras sucursales. Su función principal es enrutar los paquetes de datos entre las redes LAN de las sucursales y la red WAN, proporcionando conectividad segura y eficiente.
    - **Switch**: Conecta los dispositivos de la red local de cada sucursal, permitiendo la comunicación entre ellos a través de la conmutación de paquetes. Los switches segmentan el tráfico de la red, mejorando la eficiencia al limitar la difusión de datos únicamente a los dispositivos destinatarios. Además, algunos switches pueden implementar funciones de seguridad como VLANs para aislar el tráfico y prevenir accesos no autorizados.
    - **Firewall**: Protege la red local de cada sucursal contra accesos no autorizados desde Internet y aplica políticas de seguridad. Actúa como una barrera entre la red interna y externa, filtrando el tráfico según reglas predefinidas para permitir o denegar el acceso a determinados servicios o recursos. Los firewalls son fundamentales para garantizar la seguridad de la red al bloquear amenazas externas y prevenir ataques maliciosos.

Parte 2: Configuración de VPN con Python

Utilizando la biblioteca paramiko de Python, escribe un script que configure una VPN en los routers de cada sucursal. Supondremos que los routers son dispositivos Cisco y que el script debe configurar automáticamente la VPN utilizando IPsec.

```
# Código Python para configurar la VPN en los routers de cada sucursal
import paramiko

# Función para conectar al router
def connect_to_router(hostname, username, password):
    client = paramiko.SSHClient()
    client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    client.connect(hostname, username=username, password=password)
    return client

# Función para configurar la VPN
def configure_ipsec_vpn(client, peer_ip, local_network, remote_network):
    commands = [
        'crypto isakmp policy 10',
        'encr aes 256',
        'authentication pre-share',
        'group 5',
        'crypto isakmp key mysharedsecret address ' + peer_ip,
        'crypto ipsec transform-set myset esp-aes 256 esp-sha-hmac',
        'crypto map mymap 10 ipsec-isakmp',
        'set peer ' + peer_ip,
        'set transform-set myset',
        'match address 100',
        'access-list 100 permit ip ' + local_network + ' ' + remote_network,
        'interface g0/0',
        'crypto map mymap',
        'end'
    ]
    for command in commands:
        stdin, stdout, stderr = client.exec_command(command)
        print(stdout.read().decode())
    client.close()

# Ejemplo de uso
hostname = '192.168.1.1'
username = 'admin'
password = 'password'
client = connect_to_router(hostname, username, password)
configure_ipsec_vpn(client, '192.168.2.1', '192.168.1.0 255.255.255.0',
'192.168.3.0 255.255.255.0')
```

Preguntas adicionales

3. **¿Cómo implementarías el cifrado de extremo a extremo además de la VPN? Considera el uso de claves públicas y privadas.**
   Además de la VPN, implementaría el cifrado de extremo a extremo utilizando claves públicas y privadas. Cada sucursal tendría un par de claves única. Cuando un dispositivo de una sucursal envía datos a otra sucursal, cifraría los datos con la clave pública de la

sucursal receptora. Así, solo la sucursal receptora, con su clave privada correspondiente, podría descifrar los datos. Esto proporciona una capa adicional de seguridad sobre la VPN.

4. **Proporciona un esquema para implementar un sistema robusto de logs y monitoreo de la red utilizando herramientas modernas de gestión de red. ¿Cómo podría Python automatizar la recopilación y análisis de logs?**
   Para implementar un sistema de logs y monitoreo de la red, utilizaría herramientas como ELK Stack (Elasticsearch, Logstash, Kibana) o Splunk. Estas herramientas permiten recopilar, almacenar, visualizar y analizar logs de manera eficiente. Python podría automatizar la recopilación y análisis de logs mediante el uso de bibliotecas como `elasticsearch`, `logstash`, `kibana`, `splunk-sdk`, entre otras, para interactuar con estas herramientas y realizar tareas específicas de monitoreo y análisis de logs de forma programática.

Integrantes:

- Huanca Hampuero Lila Zaray
- Gavino Isidro Michael Richard
- Manosalva Peralta Yojan Alexander