Universitat Oberta de Catalunya

Grado de Ingeniería Informática
Trabajo de Fin de Grado (01/2024)

Implementación y administración de redes seguras basadas en perfiles de acceso

Autor: Daniel Pérez Torres
Consultor: Amadeu Albós Raya
Área: Redes de Computadores

Contenido

1. Contexto y Justificación.

2. Situación esperada.

3. Estudio de la solución.

- 802.1x
- MAB

4. Diseño de los productos.

- Red basada en perfiles de acceso.
- Herramienta de gestión centralizada.

5. Implementación de los productos.

- Red basada en perfiles de acceso.
- Herramienta de gestión centralizada.

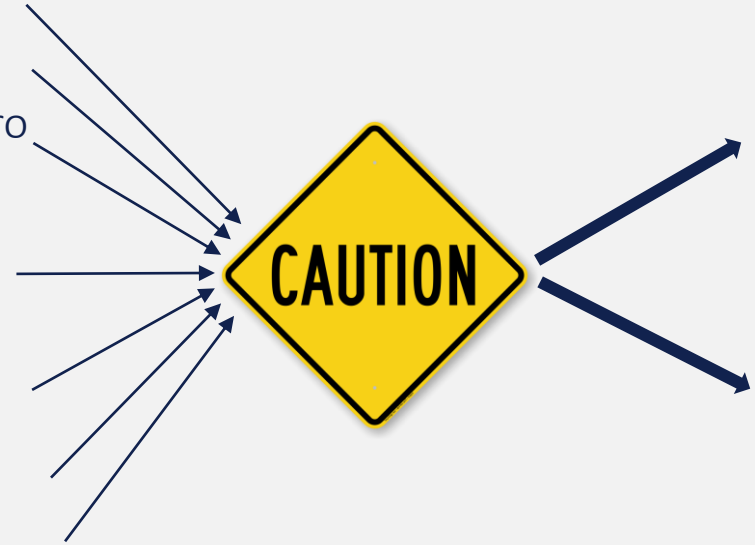
6. Resultados.

7. Conclusiones.

1. Contexto y Justificación

Necesidades de conectividad en redes corporativas

- Wifis para Clientes
- BYOD *(Bring Your Own Device)*
- Movilidad de empleados dentro de la organización
- Servicios publicados en la Internet
- Acceso de diferentes tipos de usuario
- Acceso de diferentes tipos de dispositivos
- Multitud de aplicaciones y sistemas operativos

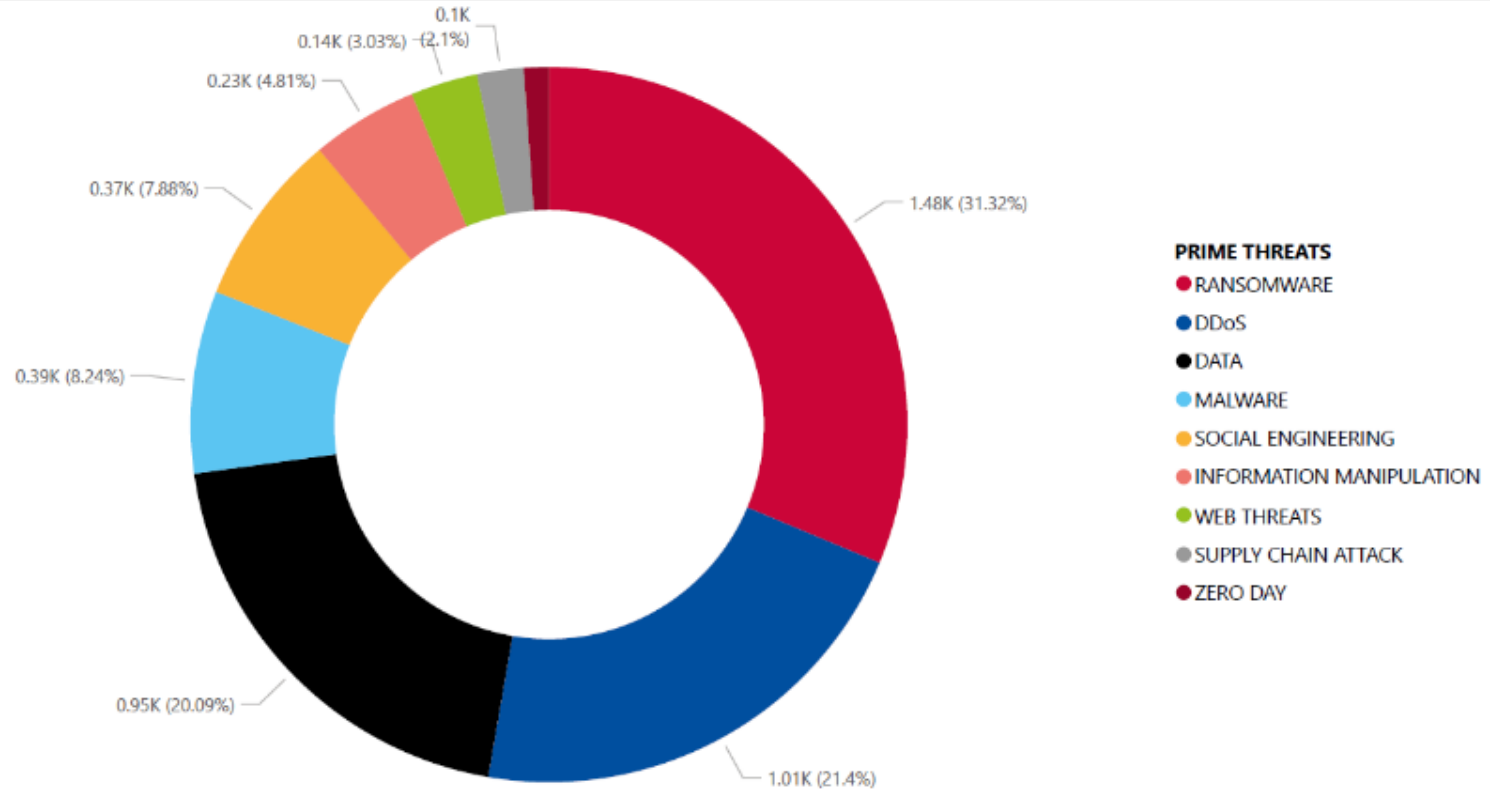


Pérdida de control sobre los activos de la red

Exposición permanente a amenazas en Internet

1. Contexto y Justificación

Amenazas para la continuidad del negocio



Activo más preciado:

Información corporativa

Consecuencia en caso de éxito:

Pérdida de continuidad de negocio

Vector de entrada más habitual:

Usuarios finales

1. Contexto y Justificación

¿Cómo protegernos?

- Capa 2: Segmentación (VLANs), control de puertos…
- Capa 3: Firewalls, IDS, IPS…
- Capa 4: TLS, SSL…
- Capa 7: WAF, proxy inverso, EDR…
- Concienciación de usuarios

¿Soluciona el problema?

No en su totalidad

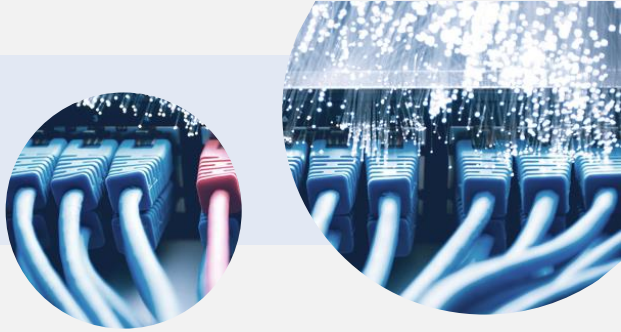
¿Qué necesitamos?

Visibilidad, control de usuarios y dispositivos, control de acceso, control de permisos, automatización…

Infraestructura de red basada en perfiles de acceso (NAC)

Herramienta de gestión centralizada

2. Situación esperada



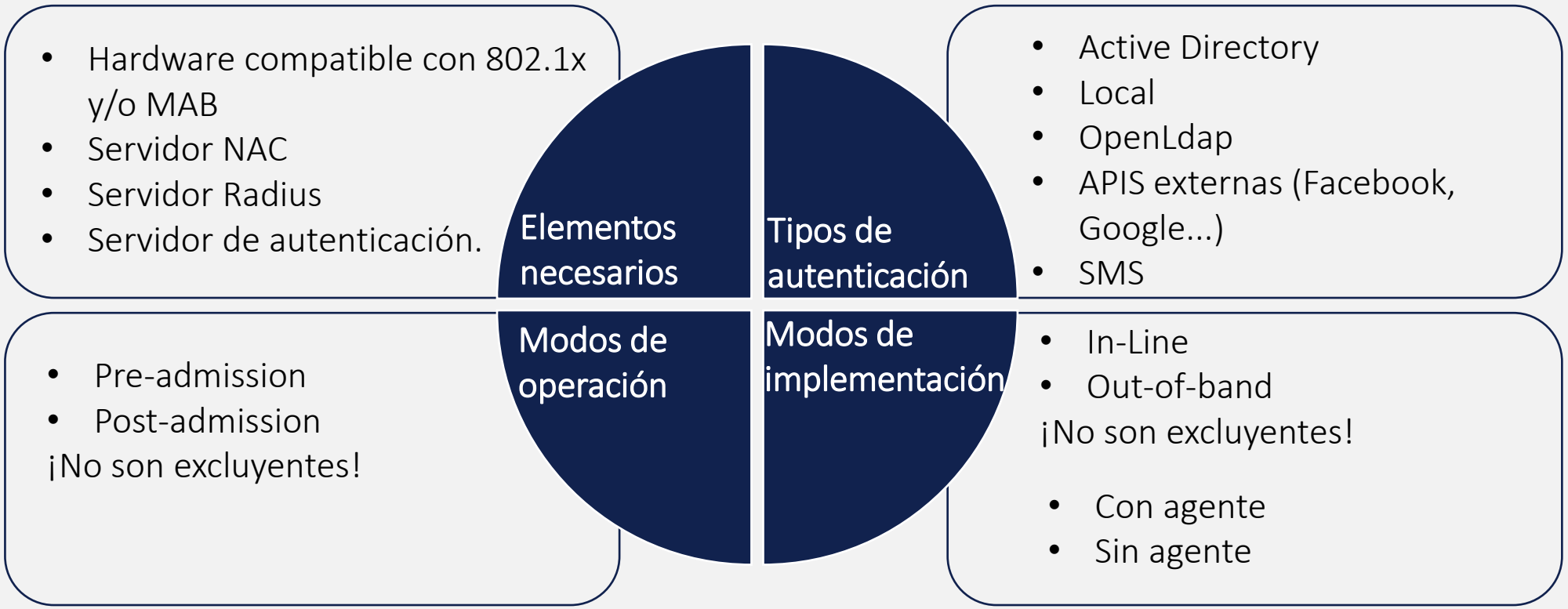
- Wifis para Clientes
- BYOD *(Bring Your Own Device)*
- Movilidad de empleados dentro de la organización
- Servicios publicados en la Internet
- Acceso de diferentes tipos de usuario
- Acceso de diferentes tipos de dispositivos
- Multitud de aplicaciones y sistemas operativos

NAC

- ✓ Detectar automáticamente el tipo de usuario y aplicarle un perfil de acceso con permisos adecuados a sus necesidades.
- ✓ Analizar el dispositivo que intenta conectar a la red y decidir si es seguro o no, impidiendo su conexión en caso de que no lo sean.
- ✓ Habilitar el acceso a la red únicamente a usuarios autenticados.
- ✓ Otorgar los mismos permisos a cada tipo de usuario sin importar el equipo o toma de red desde la que conecten.
- ✓ Otorgar los mismos permisos a cada tipo de usuario sin importar el equipo o toma de red desde la que conecten.
- ✓ Visibilidad y control sobre todos los activos conectados a la red.

3. Estudio de la solución

Características de NAC



- Hardware compatible con 802.1x y/o MAB
- Servidor NAC
- Servidor Radius
- Servidor de autenticación.

- Active Directory
- Local
- OpenLdap
- APIS externas (Facebook, Google...)
- SMS

- Pre-admission
- Post-admission

¡No son excluyentes!

- In-Line
- Out-of-band

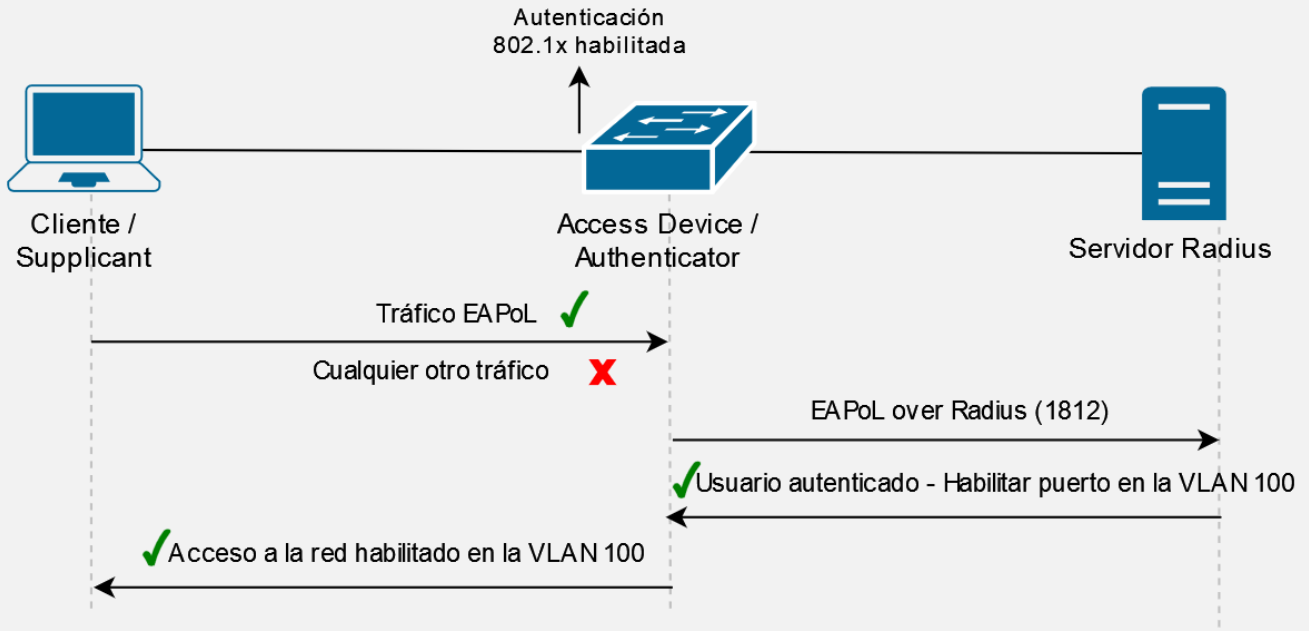
¡No son excluyentes!

- Con agente
- Sin agente

3. Estudio de la solución

Autenticación en capa 2: 802.1x

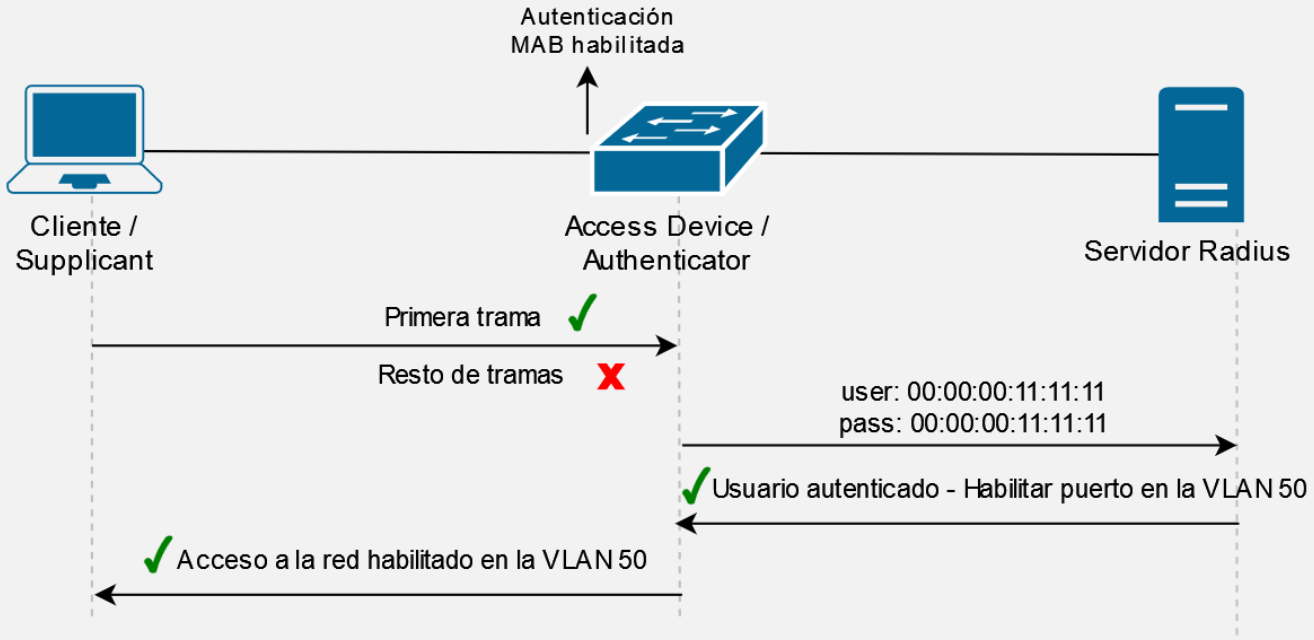
- ✓ Función: Forzar la autenticación de usuarios para acceder a la red y asignar los parámetros de conectividad recibidos desde el servidor Radius.
- Roles: Cliente *(Supplicant),* Access Device (*Authenticator),* Radius Server
- Protocolo: EAP (Con sus variantes EAPoL y EAP over Radius).
- Modos de autenticación: EAP-TLS, EAP-PEAP…



3. Estudio de la solución

Autenticación en capa 2: MAB (*Mac Authentication Bypass)*

- ✓ Función: Permitir la autenticación en la red de dispositivos mediante su dirección MAC. (normalmente se aplica sobre aquellos incompatibles con 802.1x).
- Modos: *Single-host, Multi-domain, Multi-Authentication, Multi-host.*
- ¡¡¡ Cuidado !!! ¡¡ Una MAC se puede falsear con facilidad !! Aplicar sólo a dispositivos con accesos muy restringidos



Fase de diseño

1. Diseño de una red segura basada en perfiles de acceso

2. Diseño de una herramienta de gestión centralizada

4. Diseño de los productos

1.- Diseño de una red segura basada en perfiles de acceso

Estrategia de diseño

1. Diseño lógico de la red
2. Características y elección del servidor NAC
3. Diseño de una estrategia de autenticación
4. Diseño de los permisos para los perfiles de acceso

4. Diseño de los productos

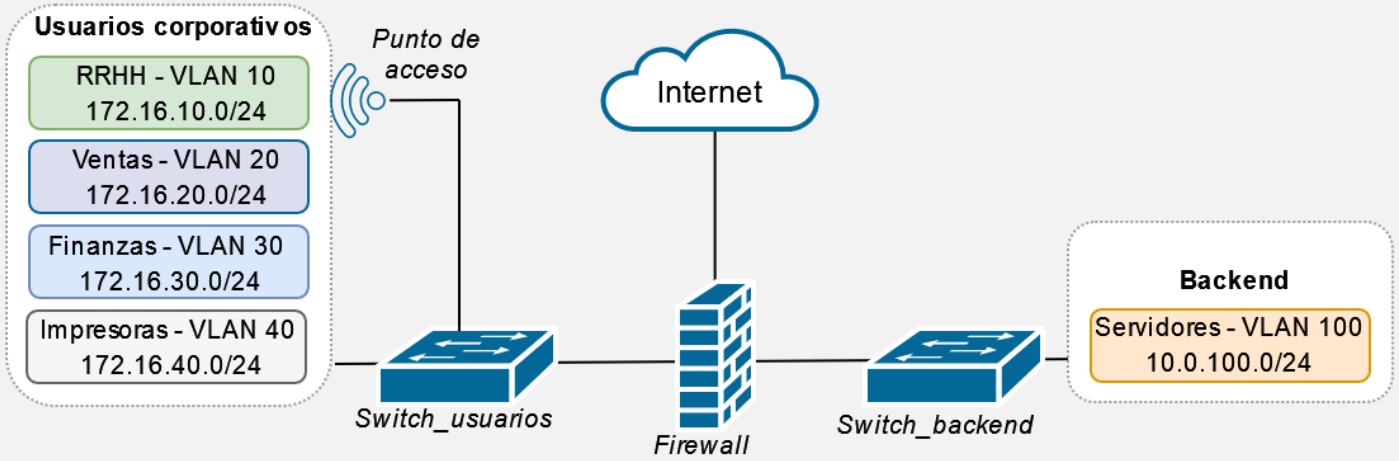
1.- Diseño de una red segura basada en perfiles de acceso

Diseño lógico de la red

✓ Objetivo: Simular las características mas habituales de una red corporativa en producción.

- Identificación de requisitos 1

✓ Compuesta por diferentes departamentos, cada uno de ellos en un segmento de red diferente.
✓ Se permite la movilidad de los empleados dentro de la organización.
✓ Dispone de Wifi para empleados.
✓ Dispone de un Firewall.
✓ Conectividad con Internet.
✓ Red de servidores independiente y protegida por un firewall.



4. Diseño de los productos

1.- Diseño de una red segura basada en perfiles de acceso

Identificación de requisitos de NAC

2 Servidor NAC

- Modo de operación **pre-admission**
- Implementación **out-of-band**
- **Sin agente**
- Protocolo de autenticación **EAP-PEAP.**
- Protocolos **802.1x y MAB** para acceso cableado.
- Protocolo **WPA2-Enterprise** para acceso inalámbrico.
- Permitir validación basada en **AD**

3 Estrategia de Autenticación

- **Switch**:

1.- 802.1x
2.- MAB

- **Punto de acceso:**

1.- WPA2/Enterprise

- **Servidor NAC:**

Validar los intentos de conexión contra el AD y asignar un determinado perfil en base al grupo al que pertenezca.
- RRHH: VLAN 10
- Ventas: VLAN 20
- Finanzas: VLAN30
- MAC 00:00:00:11:11:11: VLAN 40

4 Permisos de perfiles de acceso

Asignar diferentes permisos en capa 3 para los segmentos de red de cada departamento, ajustando los permisos a sus necesidades.

Solución ideal: **PacketFence + pfSense**

4. Diseño de los productos

2.- Diseño de una herramienta de gestión centralizada

Estrategia de diseño

1. Selección de tipo de aplicación y lenguaje
2. Análisis de requisitos funcionales
3. Diseño de la arquitectura web
4. Diseño de diagrama de actividades
5. Actores y casos de uso

4. Diseño de los productos

2.- Diseño de una herramienta de gestión centralizada

Diseño de la aplicación

1 Tipo de aplicación y lenguaje

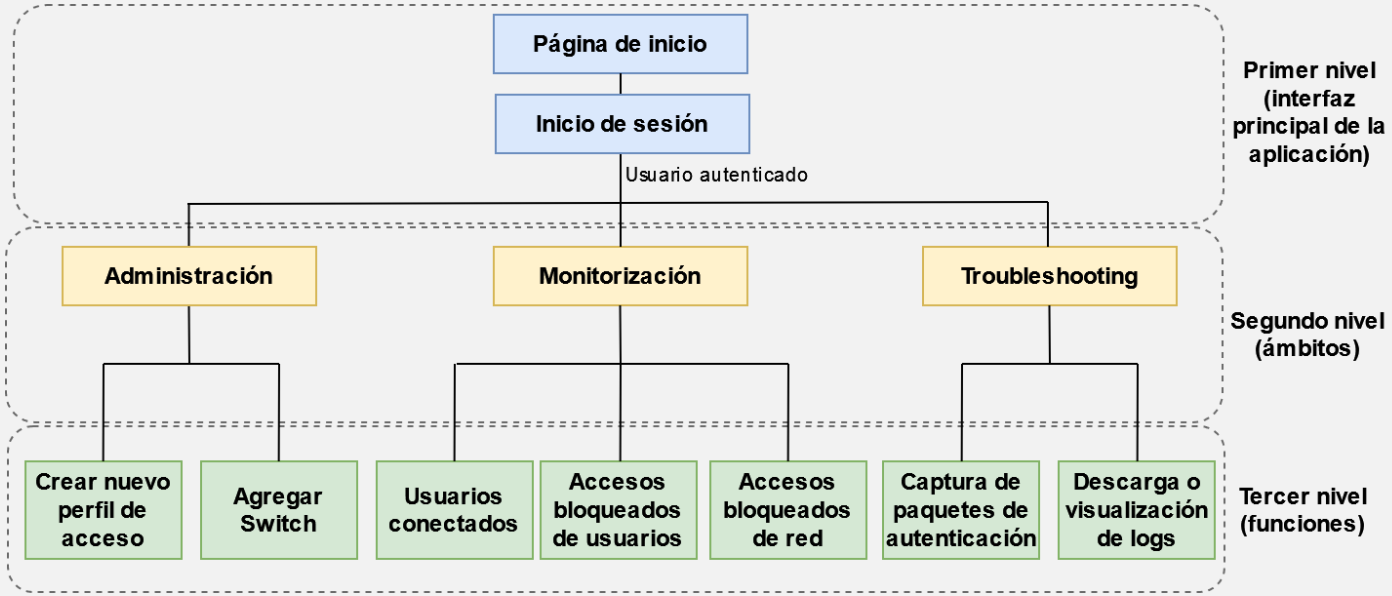
- Tipo de aplicación: **Web**
- Lenguajes: **HTML y Phyton**
- Plataforma ideal: **Flask**

2 Requisitos funcionales

- Autenticación de usuario
- Crear perfil de acceso
- Agregar un nuevo Switch
- Ver usuarios conectados
- Ver usuarios con acceso denegado
- Ver accesos de red bloqueados
- Captura de paquetes
- Visualización de logs

3 Arquitectura web (modelado)

- Nivel 1: Página de validación y menú principal
- Nivel 2: Menús de funciones
- Nivel 3: Funciones

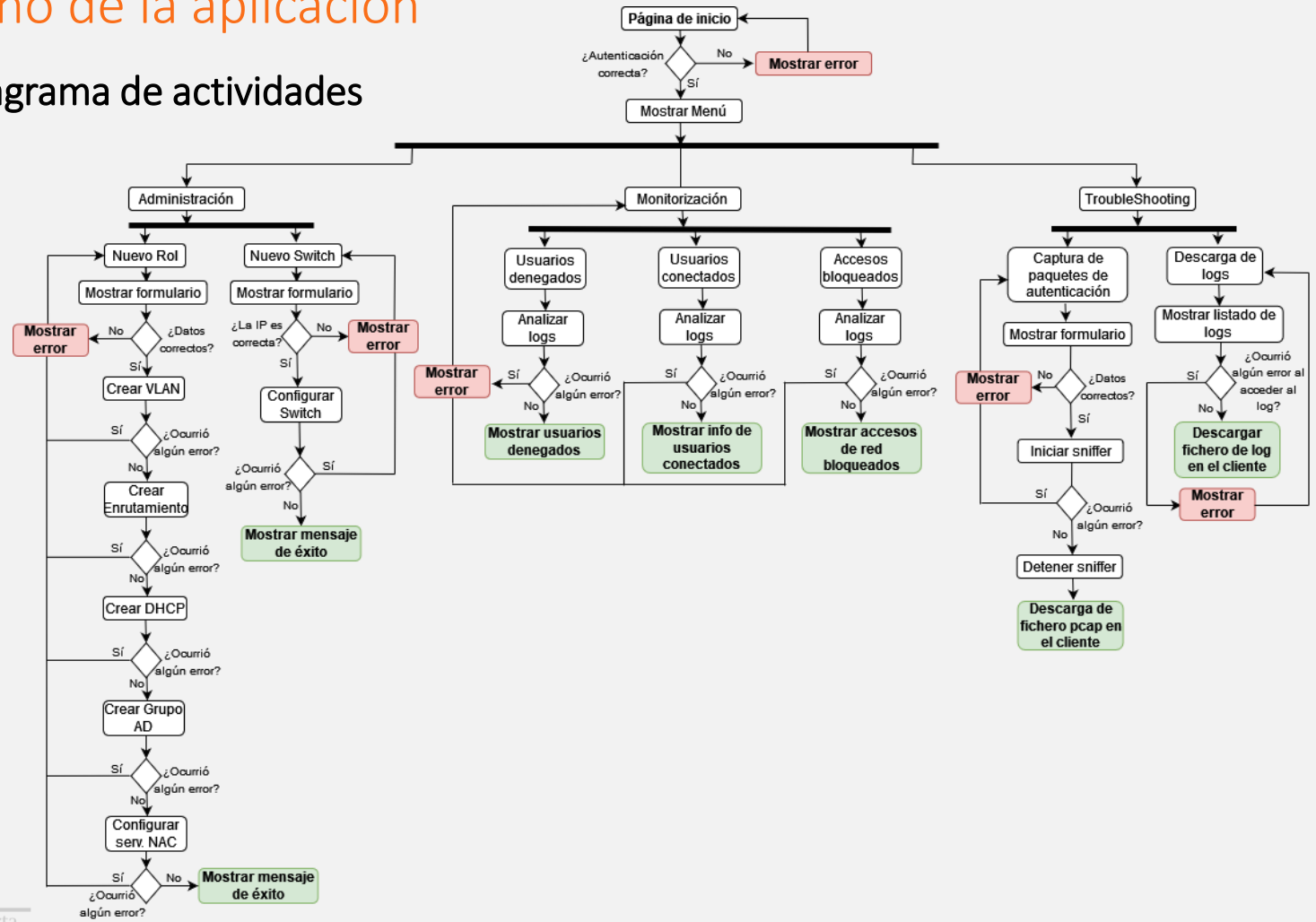


4. Diseño de los productos

2.- Diseño de una herramienta de gestión centralizada

Diseño de la aplicación

4 Diagrama de actividades



5 Casos de uso

- Autenticación de usuario
- Crear perfil de acceso
- Agregar un nuevo Switch
- Ver usuarios conectados
- Ver usuarios con acceso denegado
- Ver accesos de red bloqueados
- Captura de paquetes
- Visualización de logs

Actores

- Administradores/as de red

Fase de Implementación

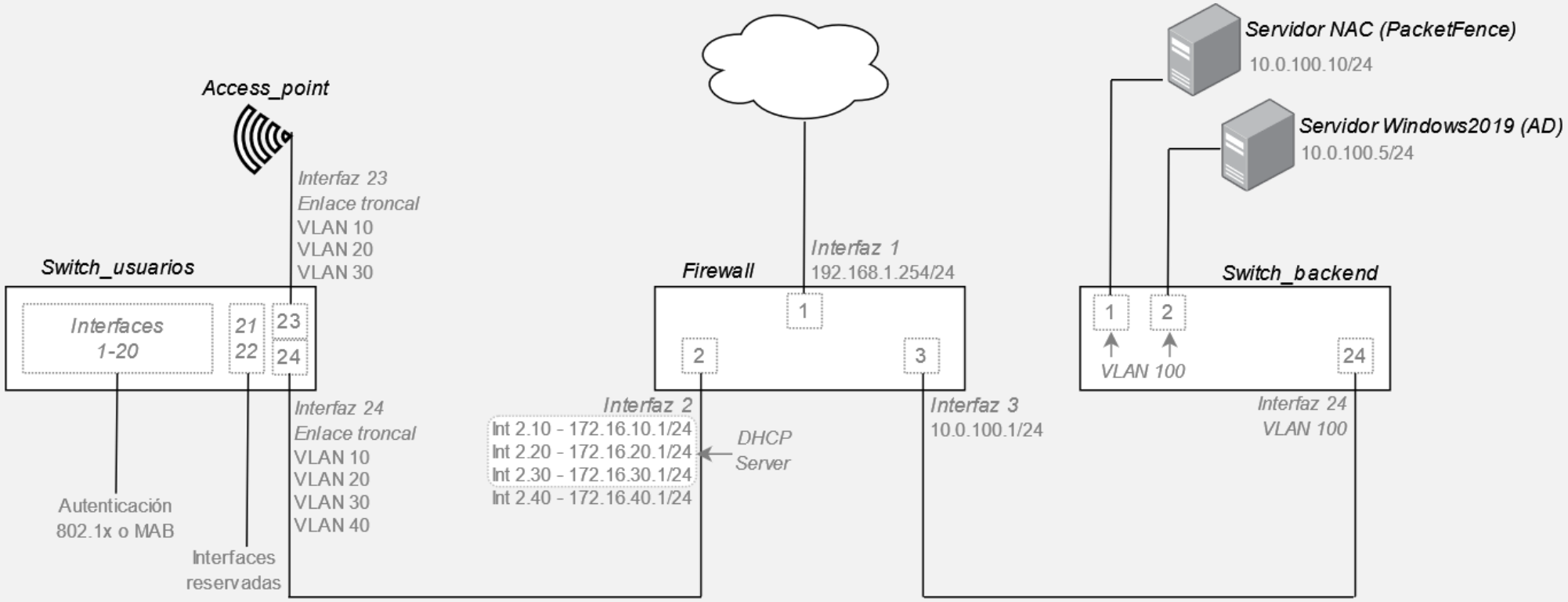
1. Implementación de una red segura basada en perfiles de acceso

2. Implementación de una herramienta de gestión centralizada

5. Implementación

1.- Implementación de una red segura basada en perfiles de acceso

Paso 1: Crear la red - Esquema de direccionamiento



5. Implementación

1.- Implementación de una red segura basada en perfiles de acceso

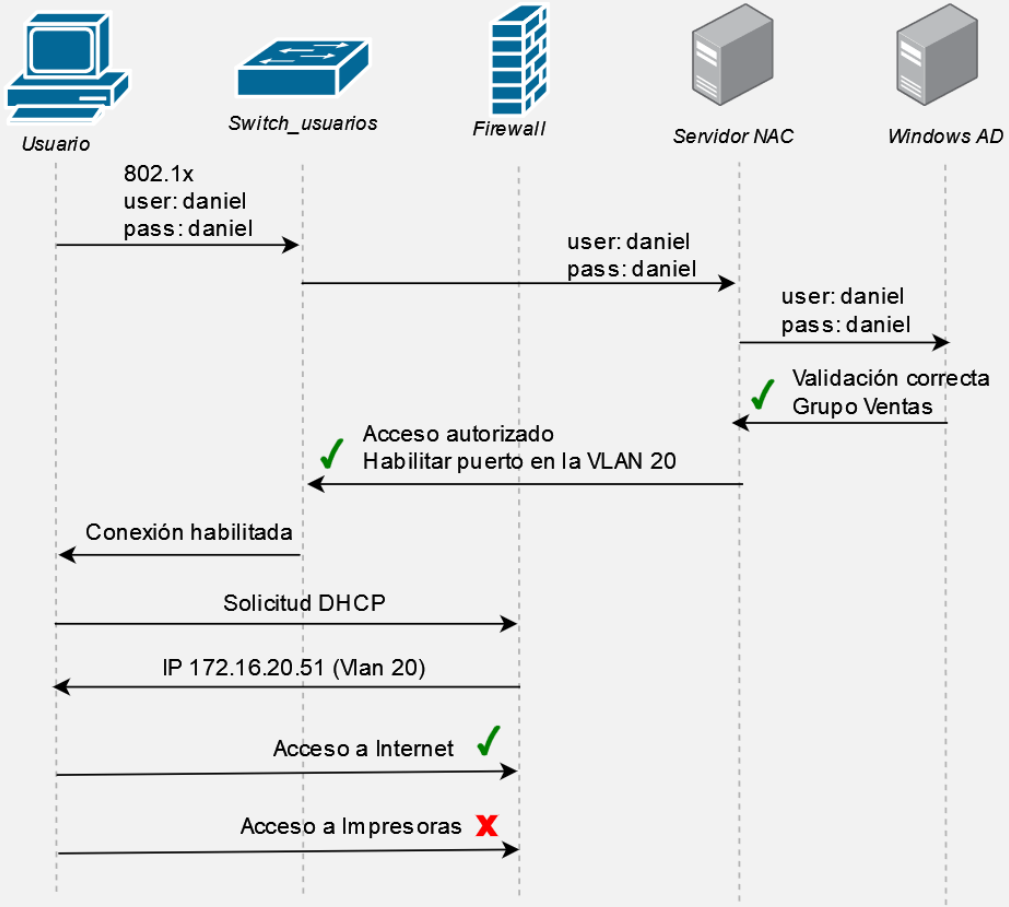
Paso 2: Configuración (relativa a NAC) de dispositivos

1. **Configuración del Firewall (pfSense)** — Políticas en capa 3 para cada perfil de acceso. Servidor DHCP para cada perfil de acceso.
2. **Configuración del Switch** — Crear VLAN para cada departamento. Habilitar la autenticación 802.1x y MAB en cada interfaz. Definir la conexión con el servidor Radius de autenticación.
3. **Configuración del punto de acceso** — Habilitar WPA2 Enterprise como método de autenticación. Definir la conexión con el servidor Radius.
4. **Configuración de Active Directory** — Crear un grupo para cada departamento. Crear usuarios y agregarlos a los grupos.
5. **Configuración de servidor NAC (PacketFence)** — Integrar el Switch y punto de acceso como fuentes de autenticación.
   Integrar el Directorio activo como servidor de autenticación.
   Crear perfiles de conectividad basados en 802.1x y MAB.
   Crear un rol para cada perfil de acceso.
   Crear reglas de autenticación para cada rol.

5. Implementación

1.- Implementación de una red segura basada en perfiles de acceso

Resultado de la implementación (Ejemplo para perfil de acceso de Ventas)



5. Implementación

1.- Implementación de una herramienta de gestión centralizada

Paso 1: Preparación del entorno y estructura de la aplicación

- IDE: *PyCharm Community Edition*

- Framework: **Flask**
- pip install flask
- nacAPP.py

```
@app.route('/', methods=["GET", "POST"]) # Creación de la ruta “/”
def inicioSesion(): # Función que se ejecutará cuando se acceda a la ruta
   nacApp.check_access = "False"
     if request.method == "POST":
        if request.form.get("uname") == "admin" and request.form.get("psw") == "admin":
               nacApp.check_access = str("True")
               return redirect("/main")
        else:
           messagebox.showinfo("Error","Usuario y/o contraseña incorrecta.")
     return render_template('login.html')
```

Servidor Web (Flask)

- **"/"** → Página de autenticación de usuario
- **"main"** → Menú principal
  - **"admin"** → Menú de funciones de administración
    - **"new-switch"** → Función para agregar un nuevo Switch
    - **"new-rol"** → Función para agregar un nuevo rol
  - **"monitor"** → Menú de funciones de monitorización
    - **"connectedUsers"** → Función para visualizar usuarios conectados
    - **"deniedUsers"** → Función para visualizar usuarios denegados
    - **"blocked"** → Función para visualizar accesos de red denegados
  - **"tshoot"** → Menú de funciones de troubleshooting
    - **"sniffer"** → Función para capturar paquetes de autenticación en el servidor NAC
    - **"logs"** → Función para descargar logs de los diferentes elementos de NAC

5. Implementación

1.- Implementación de una herramienta de gestión centralizada

Paso 2: Interfaz gráfica de menús y funciones

- Función: Mostrar menús, formularios de configuración y resultados al usuario
- Código: **HTML**


```
<body>
  <form action="{{ url_for('inicioSesion')}}" method="post">
    <h1>Administración NAC de TFGDanielPerez</h1>
    <div class="formcontainer">
    <hr/>
    <div class="container">
      <label for="uname"><strong>Nombre de usuario</strong></label>
      <input type="text" placeholder="Usuario" name="uname" required>
      <label for="psw"><strong>Contraseña</strong></label>
      <input type="password" placeholder="Contraseña" name="psw" required>
      </div>
    </div>
    <button type="submit"><b>Iniciar sesión</b></button>
  </form>
 </body>
```

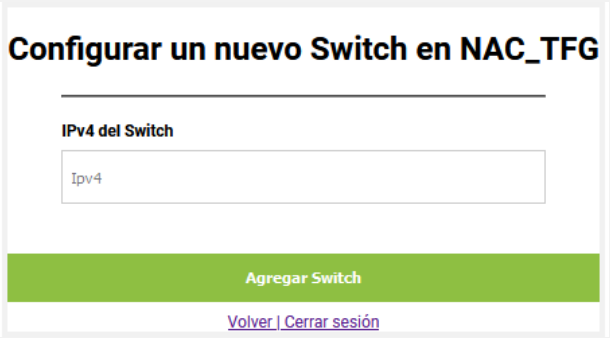
Ruta “/” → fichero login.html

5. Implementación

1.- Implementación de una herramienta de gestión centralizada

Paso 3: Desarrollo de Scripts

- Función: Ejecutar las funciones de la aplicación
- Código: **Phyton**



Formulario de la opción “Crear un nuevo Switch”

→ add_new_switch.py

```
def new_switch(ip_add):
try:
        switch = paramiko.SSHClient()
        switch.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        switch.connect("ip_add", port=22, username="admin", password=(“pass"))
        switch_channel = switch.invoke_shell()
        # CREAR VLANS
        time.sleep(0.5)
        switch_channel.send('set vlan create 10\n’)
…
……
……
```

```
def add_switch_to_nac(ip_add):
try:
        nac = paramiko.SSHClient()
        nac.set_missing_host_key_policy(paramiko.AutoAddPolicy())
        nac.connect("10.0.100.10", port=22, username="root", password=(“pass"))
        nac_channel = nac.invoke_shell()
        #AGREGAR SWITCH AL FICHERO DE CONFIGURACIÓN
        nac.exec_command("printf
                    '\n["+ip_add+"']\nVentasVlan=20\nFinanzasVlan=30\nImpresorasVlan=40"
                    "\ngroup=default\nisolationVlan=1\nradiusSecret=Passw0rd\nRRHHVlan=10"
                    "\ndescription=Nuevo_Switch\nregistrationVlan=1\ncliPwd=Passw0rd"
                     "\nclieEnablePwd=Passw0rd\ncliTransport=SSH\ncliUser=admin\n' >>
/usr/local/pf/conf/switches.conf")
        nac.exec_command("/etc/init.d/freeradius restart")
…
……
……
```

5. Implementación

1.- Implementación de una herramienta de gestión centralizada

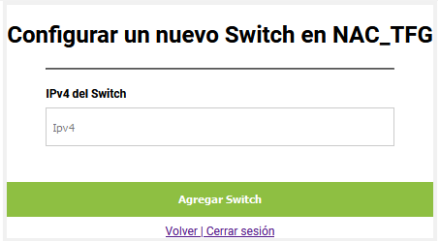
Resultado obtenido – Ejemplo de ejecución de la función “Agregar un nuevo Switch”

Paso 1

El usuario accede a la aplicación, se valida correctamente y accede a la ruta “**admin/new-switch**”, para agregar un nuevo Switch al entorno NAC.

Paso 2

La solicitud es recibida por **nacApp.py**, que lee el código asociado a la ruta solicitada. En este caso, carga el fichero new_switch.html, que contiene un formulario.



Paso3

El usuario introduce una IP válida y pulsa sobre el botón “Agregar Switch”.

Paso 4

nacAPP.py captura la IP y llama al script **add_new_switch.py**, que ejecutará las funciones necesarias para agregar un nuevo Switch. En este caso, configurar el Switch e integrarlo en el servidor NAC.

Paso 5

El script devuelve el resultado obtenido y es mostrado al usuario haciendo uso de la plantilla **new_switch_result.html**.

6. Resultados

Criterio	Red basada en perfiles de acceso	Herramienta de gestión
¿ Se han logrado todos los objetivos iniciales?	80%	90%
¿Se ha tenido que modificar algún objetivo?	No	No
¿Se han respetado los requisitos definidos?	90%	95%
¿Se ha logrado una implementación a bajo coste?	60%	100%
¿Se ha logrado un producto seguro?	90%	60%
¿Se ha logrado un producto robusto?	80%	80%
Escalabilidad de la solución	70%	95%
¿Facilita la administración y gestión de la red?	Sí	Sí
Riesgo de incidencias relacionadas con la solución	30%	5%
¿Se ha seguido la planificación?	Sí	Sí
¿Se han logrado los resultados previstos referentes a las dimensiones ético-social, de sostenibilidad y diversidad?	Sí	Sí

7. Conclusiones y trabajos futuros

Conclusiones

- Solución con enorme potencial de seguridad que no ha podido explotarse en su totalidad en este proyecto por falta de recursos y limitación del tiempo.
- Aunque se ha logrado un grado de seguridad aceptable, tiene margen de mejora.
- Herramienta de gestión intuitiva y funcional pero débil visualmente.
- Su implementación implica una alta criticidad para las operaciones de negocio. La caída del servicio supone pérdida de continuidad. Se requiere redundancia.
- Código de la aplicación poco optimizado.

Líneas de trabajo

- Implementar la autenticación 802.1x mediante certificados digitales (EAP-TLS).
- Lograr una red ZTNA integrando PacketFence con Nessus u OpenVass para analizar vulnerabilidades en los dispositivos antes de permitir el acceso a la red.
- Incluir redundancia para mejorar la robustez de la solución, escalabilidad y evitar interrupciones del servicio.
- Basar la autenticación de la aplicación en directorio activo y cifrar la comunicación mediante HTTPS.
- Simplificar código
- Agregar nuevas funcionalidades a la aplicación

Universitat Oberta de Catalunya

Gracias!

Daniel Pérez Torres

TFG Grado de Ingeniería Informática

01/2024