Desarrollo de un entorno de virtualización de redes con fines docentes

UNIVERSIDAD
COMPLUTENSE
MADRID

Víctor Fernández Duque
Director - Juan Carlos Fabero Jiménez

Trabajo de fin de grado del Grado en Ingeniería del Software

Introducción

En esta sección se habla detalladamente sobre el documento SRS(Especificación de requisitos software) el cual consta de varias subsecciones: propósito, el alcance de la aplicación, las referencias tomadas y una visión general del documento.

Propósito

El propósito del documento de la SRS es definir de forma detallado y precisa los requisitos necesarios para el desarrollo del producto software requerido por el cliente. A continuación se detalla a quién va dirigido este documento y los participantes:

- Cliente: Juan Carlos Fabero Jiménez.
- Desarrollador: Víctor Fernández Duque.
- Usuario: Todos aquellos usuarios que decidan usar nuestro controlador.

Alcance

El producto llevará a cabo la función de controlador para gestionar un entorno de virtualización de redes, empleando OpenVSwitch. Nuestra aplicación consistirá en un panel donde convivirán dos partes la de creación de la topología red, y otra el control de flujos de la misma.

Definiciones, acrónimos y abreviaturas

- **NCOVS:** Nuestro controlador OpenVSwitch.
- **SRS**: es una descripción completa del comportamiento del sistema que se va a desarrollar.

Referencias

- SRS Wikipedia

Resumen

Construir posteriormente.

Descripción general

En esta sección se dará una breve descripción para mostrar en que se basa nuestro controlador.

Perspectiva del producto

El software que se va a desarrollar es un controlador con el cual se gestionarán redes y se controlará su flujo de datos, el controlador podrá usarse en una red física, pero nosotros lo desarrollaremos mediante virtualización en la plataforma debian.

El controlador deberá gestionar la creación, carga, y guardados de los estados de las interfaces, puentes y cables virtuales, además de la creación, eliminación y listado de flujos de datos dentro de la red.

Funciones del producto

Como se ha dicho con anterioridad el controlador tiene dos funciones principales la creación de la topología de red y el control de la misma, entonces hemos visto necesario la creación de tres módulos independientes.

- **Red:** es la parte encargada de gestionar la creación y modificación de la topología de red.
- **Control:** es la parte encargada de gestión el flujo de datos dentro de la red anteriormente creada.
- **Archivo**: es la parte encargada de la creación y lectura de ficheros para guardar el estado de la aplicación.

En cuanto a las funciones de estos módulos:

- **Red:**
    - **Añadir puente:** permite agregar un puente (switch) donde conectaremos nuestras máquinas virtuales.
    - **Añadir interfaz:** añade una interfaz virtual para las máquinas.
    - **Añadir puerto:** añade un elemento al puerto de un switch.
    - **Añadir cable:** añade un cable virtual para conectar los puentes.
    - **Eliminar puente:** permite eliminar un puente de nuestra topología de red.
    - **Eliminar interfaz:** elimina una interfaz virtual.
    - **Eliminar puerto:** elimina un puerto de un switch.
    - **Eliminar cable:** elimina un cable de memoria.
    - **Mostrar:** muestra los switches y los elementos conectados a los puertos..
    - **Guardar:** recoge el estado actual del controlador y lo traduce a un formato json para después pasarlo a la función de guardar del fichero..
    - **Cargar:** recibe un estado guardado mediante la lectura de un fichero json previamente creado y lo carga en la aplicación.
    - **Limpiar:** elimina toda la configuración del controlador y la memoria.

- **Control:**
    - **Añadir flujo:** agrega controles de flujo al switch introducido..
    - **Eliminar flujo:** elimina los controles de flujo del switch introducido.
    - **Mostrar:** muestra los flujos del switch introducido.

    - **Guardar flujo:** guarda el control de flujo del switch introducido.
    - **Cargar flujo:** carga un estado de control de flujo en un switch existente.
    - **Limpiar:** limpia el control de flujos de un switch.
- **Archivo:**
    - **Guardar json:** Guarda un fichero cuyo nombre y contenido recibirá.
    - **Cargar json:** Lee el contenido de un archivo y devuelve el contenido.
    - **Guardar control de flujo:** Guarda el estado del control de flujo de un switch.

Características del usuario

La aplicación estará dirigida a un usuario especializado en gestión de redes, no es que sean necesarios altos conocimientos debido a que el controlador simplifica la gestión de redes, pero si es necesario conocimientos básicos sobre gestión y control de redes. La aplicación contará con un único tipo de usuario que será el administrador de la red.

Restricciones

A la hora de desarrollar la presente aplicación, la única restricción de nuestro cliente es que sea una aplicación portable, eficiente y sencilla de usar, por lo tanto decidimos que la mejor opción será realizar nuestro desarrollo en el lenguaje tcl-tk que nos da una forma simple de general la parte gráfica para de esta de forma centrarnos en la complejidad del proyecto, además, respecto a la forma de almacenar la información utilizaremos archivos de texto plano, ya que es la forma más cómoda de guardar el estado de nuestro controlador, y no necesitamos un uso abusivo del mismo.

Supuestos y dependencias

La aplicación está pensada para utilizarse como una aplicación portable, el desarrollo será en tcl-tk. de modo que será muy simple gestionar una capa de presentación y negocio debido a las facilidades que aporta tcl-tk con los perfiles gráficos y la forma de gestionar nuestras funciones mediante la modularización.

Requisitos específicos

En esta sección se hablará sobre los requisitos que tendrá nuestro controlador.

Requisitos funcionales

En esta sección se definirá las funciones de nuestro controlador, especificando su prioridad, estabilidad, descripción, entrada, salida, necesita, acciones, precondición, postcondición, efectos laterales y dependencias.

Red

En esta sección hablaremos sobre los requisitos funcionales de la topología de red.

RF01 - Añadir puente	
Prioridad	Alta
Estabilidad	Alta
Descripción	Permite agregar un switch virtual o físico.
Entrada	Nombre del switch
Salida	Confirmación del éxito o fracaso de la operación
Necesita	Conexión a OpenVSwitch
Acciones	Agrega a nuestro entorno Debian y al controlador un nuevo puente
Precondición	Que no exista ningún switch con el mismo nombre
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF02 - Añadir interfaz	
Prioridad	Alta
Estabilidad	Alta
Descripción	añade una interfaz virtual para las máquinas internas
Entrada	nombre del puente
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Agrega a nuestro entorno debian una nueva interfaz de red
Precondición	Que no exista ninguna interfaz con el mismo nombre.
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF03 - Añadir Puerto	
Prioridad	Alta
Estabilidad	Alta
Descripción	añade un interfaz a un switch
Entrada	nombre del switch e interfaz
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Agrega un puerto al switch introducido.
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF04 - Añadir cable	
Prioridad	Alta
Estabilidad	Alta
Descripción	Añade un cable virtual para conectar los puentes
Entrada	Nombre de la conexión y puentes que se quieren conectar
Salida	Ninguno
Necesita	Conexión a nuestro sistema virtualizado
Acciones	Agrega un cable de red para conectar los puentes entre si
Precondición	Debe existir los puentes introducidos para conectarlos
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	RF01

RF05 - Eliminar puente	
Prioridad	Media
Estabilidad	Media
Descripción	Permite eliminar un puente de nuestra topología de red
Entrada	Nombre del puente que se desea eliminar
Salida	Ninguno
Necesita	Conexión a nuestro sistema virtualizado
Acciones	Elimina un puente o switch
Precondición	Debe existir el puente
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF06 - Eliminar interfaz	
Prioridad	Media
Estabilidad	Media
Descripción	elimina una interfaz virtual
Entrada	Nombre de la interfaz
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Elimina una interfaz de red
Precondición	Debe existir la interfaz de red
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF07 - Eliminar puerto	
Prioridad	Alta
Estabilidad	Alta
Descripción	Elimina un puerto a un switch
Entrada	nombre del switch y del puerto
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Elimina un puerto al switch introducido
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF08 - Eliminar cable	
Prioridad	Media
Estabilidad	Media
Descripción	Elimina una conexión entre puentes de nuestro sistema
Entrada	Nombre de la conexión
Salida	Ninguno
Necesita	Conexión a nuestro sistema virtualizado
Acciones	Elimina la conexión que existe entre dos puentes
Precondición	Debe existir el cable
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF09 - Mostrar	
Prioridad	Alta
Estabilidad	Media
Descripción	Muestra todos los switches y los elementos conectados a este
Entrada	Ninguno
Salida	Interfaces y switches que están actualmente en el OpenVSwitch
Necesita	Conexión a OpenVSwitch
Acciones	Muestra toda la información de OpenVSwitch
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF10 - Guardar	
Prioridad	Baja
Estabilidad	Alta
Descripción	guarda el estado actual de la red, bridges, interfaces, cables, control de flujos.
Entrada	Nombre del archivo para guardar
Salida	Archivo con la información del programa
Necesita	Conexión a OpenVSwitch y sistema virtualizado
Acciones	Guarda el estado de OpenVSwitch en un documento json
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF11 - Cargar	
Prioridad	Baja
Estabilidad	Alta
Descripción	carga un estado de red anteriormente guardado
Entrada	Nombre del archivo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch y sistema virtualizado
Acciones	introduce el estado de openVSwitch antes guardado
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF12 - Limpiar	
Prioridad	Baja
Estabilidad	Alta
Descripción	elimina toda la configuración del controlador y la memoria
Entrada	Ninguno
Salida	Ninguno
Necesita	Conexión a OpenVSwitch y sistema virtualizado
Acciones	elimina la configuración de OpenVSwitch y de nuestro sistema virtualizado
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF11 - Actualizar

Prioridad	Alta
Estabilidad	Media
Descripción	coge el estado actual del controlador y lo traemos a la memoria de nuestra aplicación
Entrada	Ninguno
Salida	Ninguno
Necesita	Conexión a OpenVSwitch y sistema virtualizado
Acciones	Se conecta a OpenVSwitch y obtiene el estado actual para trasladarlo al controlador
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

Control

En esta sección hablaremos sobre los requisitos funcionales del control de la red.

RF12 - Añadir flujo	
Prioridad	Alta
Estabilidad	Media
Descripción	Añadirá un control de flujo a un switch
Entrada	Control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Añade un control al flujo de un switch
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno

Dependencia	Ninguno

RF13 - Eliminar flujo	
Prioridad	Media
Estabilidad	Media
Descripción	elimina un control de flujo de un switch
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Eliminará un control de flujo
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF14 - Mostrar flujo	
Prioridad	bajo
Estabilidad	Media
Descripción	muestra el control de flujo de un switch
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Devuelve el control de flujo de un switch para mostrarlo
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF15 - Guardar flujo	
Prioridad	Media
Estabilidad	Alta
Descripción	Guarda el control de flujo de un switch
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Obtiene el flujo de control de un switch y lo guarda en un fichero
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF16 - Cargar flujo	
Prioridad	Media
Estabilidad	Media
Descripción	carga un estado de control de flujo en un switch existe
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	obtiene el control de flujo de un switch y lo carga en el introducido por el usuario
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF18 -Limpiar flujo	
Prioridad	Media
Estabilidad	Media
Descripción	elimina el control de flujo de un switch
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Elimina todo el control de flujos de un switch introducido
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

Archivo

En esta sección hablaremos sobre los requisitos funcionales del control de la red.

RF19 - Guardar json	
Prioridad	Media
Estabilidad	Media
Descripción	Recibe el estado actual de la aplicación en formato json y lo guarda en la carpeta de “save”
Entrada	Nombre fichero y contenido en formato json
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Guarda el estado de la aplicación
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno

Dependencia	Ninguno

RF20 - Cargar flujo	
Prioridad	Media
Estabilidad	Media
Descripción	lee un fichero de estado de la aplicación
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Lee un fichero previamente guardado y lo devuelve a el modulo de topologia
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno
Dependencia	Ninguno

RF21 - Guardar flujo	
Prioridad	Media
Estabilidad	Media
Descripción	guarda el flujo de un switch
Entrada	Switch y control de flujo
Salida	Ninguno
Necesita	Conexión a OpenVSwitch
Acciones	Recibe el control de flujo de un switch y lo guarda en un fichero
Precondición	Ninguno
Postcondición	Ninguno
Efectos laterales	Ninguno

Dependencia	Ninguno

Requisitos no funcionales

En este apartado hablamos de los requisitos ligados a las características de funcionamiento de nuestro controlador.

Eficiencia

RNF01: nuestro controlador deberá responder a las peticiones del usuario en un tiempo máximo de cinco segundos.
RNF02: el sistema no deberá consumir demasiados recursos, para ello se sitúa un techo de memoria en un gigabyte de memoria.

Seguridad

RNF03: el controlador se conectará a la aplicación mediante un sistema cifrado.
RNF04: solo usuarios con permisos de administrador podrán utilizar el controlador.

Usabilidad

RNF05: el tiempo para aprender cómo se utiliza el controlador no será de más de dos horas.
RNF06: el sistema ofrecerá mensajes de error informativos orientados al usuario final.
RNF07: el controlador tendrá una interfaz gráfica básica e intuitiva.

Requisitos de rendimiento

La aplicación necesitará conexión a OpenVSwitch y al sistema virtualizado para poder realizar todas las operaciones.
El entorno de prueba será una maquina virtual Debian en la cual está el sistema OpenVSwitch y máquinas virtuales UML, de bajo coste en memoria para probar la red.

Restricciones de diseño

El lenguaje que se utilizará será tcl-tk debido a su fácil portabilidad y sencillez.
Los estados guardados se almacenarán en documentos con extensión “.nco”

Atributos del sistema software

A continuación se hablará de las cualidades que debe tener nuestro controlador:

- Eficiencia: será capaz de utilizar eficazmente los recursos con los que cuenta.
- Robustez: será capaz de reaccionar ante condiciones excepcionales.
- Extensibilidad: será escalable.
- Corrección: cumplirá los objetivos que se han marcado en la especificación.
- Verificabilidad: se creará métodos sencillos y fáciles de verificar.
- Facilidad de uso: uso del sistema será fácil para cualquier usuario.

- Legibilidad: código será sencillo y legible así como con un buen estilo de programación.
- Fácil mantenimiento: se podrá realizar modificaciones en el software sin dificultad.