SOFTWARE DEFINED NETWORKING EN SUBESTACIONES ELÉCTRICAS BASADAS EN IEC 61850

CESAR ALEJANDRO VARGAS CEDEÑO

UNIVERSIDAD PONTIFICIA BOLIVARIANA

ESCUELA INGENIERÍAS

FACULTAD DE INGENIERÍA EN TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

MAESTRÍA EN TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

MEDELLÍN

2016

SOFTWARE DEFINED NETWORKING EN SUBESTACIONES ELÉCTRICAS BASADAS EN IEC 61850

CESAR ALEJANDRO VARGAS CEDEÑO

Trabajo de grado para optar al título de…

Magister en Tecnologías de la Información y Telecomunicaciones (TIC)

Asesor

CLAUDIA STELLA CARMONA RODRÍGUEZ

Magister en Ingeniería de Telecomunicaciones

UNIVERSIDAD PONTIFICIA BOLIVARIANA

ESCUELA INGENIERÍAS

FACULTAD DE INGENIERÍA EN TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

MAESTRÍA EN TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

MEDELLÍN

2016

DECLARACIÓN ORIGINALIDAD

"Declaro que esta tesis (o trabajo de grado) no ha sido presentada para optar a un título, ya sea en igual forma o con variaciones, en esta o cualquier otra universidad". Art. 82 Régimen Discente de Formación Avanzada, Universidad Pontificia Bolivariana.

FIRMA AUTOR (ES) CESAR A. VARGAS

Medellín, 3 de Mayo de 2016

A la memoria de mi madre y abuela quienes me infundieron la ética y el rigor que guían mi transitar por la vida.

Cesar Alejandro Vargas Cedeño

AGRADECIMIENTOS

La Universidad Pontificia Bolivariana, un claustro universitario de reconocimiento nacional e internacional, generador y responsable de miles de profesionales.

Todos por haber contribuido en nuestra formación pero en especial:

A la Ingeniera Claudia Stela Carmona Rodríguez, por su asesoramiento, disposición, colaboración y estímulo para seguir creciendo intelectualmente.

CONTENIDO

LISTA DE ILUSTRACIONES

GLOSARIO

AIM: *Applications Implementation Methodology.*

CDMA: *Code Division Multiple Access.*

CLB: *Configurable Logic Block.*

DDR: *Double Data Rate type two.*

DLL: *Delay-Locked Loop.*

FDMA: *Frequency Division Multiple Access.*

FPGA: *Field Programmable Gate Array.*

GOOSE: *Generic Object Oriented Substation Event.*

HSR: *High-availability Seamless Redundancy.*

IEC: *International Electrotechnical Commission.*

IED: *Intelligent Electronic Device.*

LUT: *Configurable Logic Block.*

LLDP: *Link Layer Discovery Protocol.*

MMS: *Manufacturing Message Specification.*

Motherboard: La tarjeta madre es una tarjeta de circuito impreso a la que se conectan los componentes que constituyen el ordenador.

MSTP: *Multiple Spanning Tree Protocol.*

OFDM: *Orthogonal Frequency Division Multiplexing.*

OpenFlow: es una tecnología de switching que surgió a raíz del proyecto de Investigación: “OpenFlow: Enabling Innovation in Campus Networks” de 2008 en la Universidad de Stanford.

OpenStack: es una plataforma cloud computing de software libre.

PCI: *Peripheral Component Interconnect.*

PRP: *Parallel Redundancy Protocol.*

RAM: *Random Access Memory.*

RHEL: *Red Hat Enterprise Linux.*

RSTP: *Rapid Spanning Tree Protocol.*

SDN: *Software Defined Networking.*

SCD: *Substation Configuration Description.*

SCL: *Substation Configuration description Language.*

TDMA: *Time Division Multiple Access.*

TT: *Transfer Times.*

VLAN: *Virtual Local Area Networks.*

VPN: *Virtual Private Networks.*

RESUMEN

Este proyecto busca clarificar los conceptos acerca la norma IEC 61850 y observar hacia donde apuntan los nuevos desarrollos de la misma, para luego diseñar la red de telecomunicaciones de una subestación eléctrica genérica con una vida útil larga, facilidad de mantenimiento, reduciendo el cableado en cobre y con equipos tecnológicamente actualizados; haciendo uso Software Defined Networking para el manejo del flujo de datos y el tráfico en la red.

PALABRAS CLAVE: Bus de proceso; IEC 61850; redes definidas por software (SDN).

ABSTRACT

This project seeks to clarify the IEC 61850 concepts and observe what they point to the new developments of it, and then design the telecommunications network in a generic electrical substation with a long life, easy maintenance, reducing copper wiring and technologically updated equipment; using Software Defined Networking (SDN) to manage the flow of data traffic on the network.

KEY WORDS: Process bus; IEC 61850; Software Defined Networking (SDN).

1 INTRODUCCIÓN

En la actualidad el uso del internet ha aumentado y esto lleva a la necesidad de encontrar métodos para una mejor utilización de los recursos disponibles, creación de redes para mejorar la calidad y la interconexión entre usuarios o dispositivos [1]. En la actualidad las topologías de las redes de comunicaciones se han configurado para minimizar el riesgo de indisponibilidad de estas y proveer una conexión estable para usuario. La gestión de las topologías de red presenta limitaciones en la automatización en sus procesos [2].

Lo que buscaba este proyecto es comprender y evaluar el potencial del concepto SDN y realizar un prototipo práctico demostrativo haciendo uso de beneficios que nos proporciona SDN; como las rutas de desvío para cada flujo de paquetes, mejor escalabilidad, gestión de red, logrando una mayor agilidad para el envío y recepción de la información crítica; debido a que la inteligencia se encuentra en el controlador [3]. Con el controlador de la red SDN logramos manipular los parámetros al interior del Switch OpenFlow cuando se requiera, dando prioridad a información crítica para el buen funcionamiento de la red de comunicaciones de la subestación eléctrica.

Considerando lo anterior la implementación de la red de comunicaciones de la subestación se dispuso en una topología en estrella, donde se incluye el controlador para la administración de la red, un servidor con una tarjeta PCI (la NetFPGA) como Switch OpenFlow y algunos host haciendo de IED’s de la subestación eléctrica, donde se evaluó el comportamiento de los flujos de datos en la red de comunicaciones.

2 PLANTEAMIENTO DEL PROBLEMA

2.1 Problema

Desde la última década ha tomado fuerza el uso de redes de comunicaciones en las subestaciones eléctricas, por la facilidad de mantenimiento e implementación, escalabilidad, reducción del cableado en cobre y en el tiempo de montaje. Sin embargo, esto todavía deja algunos vacíos en las capacidades de los diseñadores de infraestructura crítica que la tecnología de red corporativa no proporciona. Ejemplos de estas deficiencias son los caminos pre-configurados, tráfico en la red, tolerancia a fallos en rutas end to end, la latencia debido a una reconfiguración de la red ante una falla, capacidad de supervisión, entre otros [3].

Los requisitos de rendimiento que están detalladas en el capítulo quinto del estándar IEC 61850 en lo que respecta a enclavamientos y las secuencias de maniobra han sido proporcionados por protocolos como RSTP, PRP y HSR. Pero los protocolos mencionados no han cumplido las expectativas en los tiempos de respuesta por la falta de gestión del tráfico en la red, en lo que se refiere a disparos de protección de IED a IED ya que los mensajes GOOSE de protección deben tener un tiempo de respuesta inferior a 4 milisegundos; y para la protección de barra y los valores muestreados los requisitos son aún más exigentes [4].

Por lo anterior, se requiere una comunicación sin perturbaciones para proporcionar alta disponibilidad en la red, en la entrega de mensajes GOOSE y valores muestreados de misión crítica para la subestación [5]; además, es necesario que la red de comunicaciones de la subestación eléctrica sea resistente a fallos.

2.2 Justificación

El diseño de redes basadas en SDN se configura fácilmente, ya que están orientados por principios de diseño de circuitos. Esto permite a los ingenieros de sistemas de energía hacer con enlaces de comunicación lo que están acostumbrados a hacer en líneas de transmisión; a través del diseño de rutas específicas por donde se quiere que fluyan datos. Además, la ingeniería de tráfico permite que el propietario de la red de comunicaciones tenga mayor control sobre el funcionamiento de esta, maximizando las capacidades de los activos de la red [3].

Por su parte, SDN permite definir rutas de desvío para cada flujo de paquetes, proporciona mejor escalabilidad, control de cambios mientras que se mejora el conocimiento de la red, capacidades de monitoreo cercanas a las de tiempo real a disposición de los operadores y permite un modelo de ciberseguridad de denegación por defecto [3]. Con SDN todos los puertos físicos se pueden utilizar para el envío de paquetes; esto ayuda a que haya un equilibrio en el ancho de banda y la segregación de los servicios, lo que maximiza el potencial de los activos de la red [3]. Otra razón importante donde SDN presenta un gran potencial para el sector eléctrico es la reducción de la administración de actualizaciones en los dispositivos de red; porque con SDN el mantenimiento necesario en parche o actualizaciones es menor y esto genera un mayor ahorro al propietario de la red del sistema de energía. Adicionalmente, la arquitectura SDN reduce la cantidad de código necesario en los equipos de la red de telecomunicaciones, ya que no es necesario para gestionar el servicio de descubrimiento de reenvío de paquetes, porque, este código reside en el controlador de flujo y no en el dispositivo de red de campo [3]. Combinadas estas capacidades hacen de SDN una opción atractiva para ser utilizado en la infraestructura de sistema de automatización de subestaciones eléctricas.

Considerando que este proyecto de investigación cuenta con gran parte de los recursos necesarios para su desarrollo y que el tema es novedoso para el sector eléctrico de nuestro país; donde se busca reducir costos y realizar diseños de subestaciones eléctricas escalables, interoperables, con una larga vida útil, logrando con esto que la inversión inicial sea atractiva para los clientes en los diferentes mega-proyectos del país. Además, esto es importante y de gran trascendencia en la ruta evolutiva que las tecnologías de redes de comunicaciones en subestaciones eléctricas seguirán en el corto y mediano plazo a nivel mundial. Por lo anterior expuesto, es favorable para

la Universidad Pontificia Bolivariana participar en estos desarrollos mediante la formulación de proyectos de investigación y de proyectos de postgrado que apoyen a éstos, sirviéndose así de los beneficios de ser considerada pionera en la propuesta y estudio de temas novedosos.

3 OBJETIVOS

3.1 Objetivo General.

Implementar un prototipo básico demostrativo de la red de telecomunicaciones SDN en una subestación eléctrica genérica IEC 61850.

3.2 Objetivos Específicos

Describir los principales aspectos técnicos funcionales de Software Defined Networking en una subestación eléctrica genérica IEC 61850.

Diseñar la arquitectura y la topología para la implementación del prototipo básico de la red de telecomunicaciones del sistema de automatización de una subestación eléctrica genérica IEC 61850.

Realizar un prototipo hardware y/o software demostrativo básico de SDN para ver el comportamiento de los mensajes críticos GOOSE de una subestación eléctrica genérica IEC 61850.

Analizar los resultados del comportamiento de los mensajes GOOSE críticos de la red de telecomunicaciones SDN del sistema de automatización de una Subestación eléctrica genérica IEC 61850.

4 MARCO REFERENCIAL

4.1 Marco contextual

Para satisfacer las necesidades mencionadas existen múltiples protocolos de redundancia de red que son capaces de reconfigurar la red ante un falla de algún equipo de esta; sin embargo, protocolos como RSTP y MSTP presentan una latencia (aproximadamente 100 ms) de interrupción o reconfiguración de la topología de la red [6], lo cual no es aceptable debido a la alta disponibilidad que tiene que tener la red de comunicaciones de una subestación eléctrica (Ver Ilustración 1). Existen otros mecanismos tales como la redundancia IP dual para aumentar aún más la disponibilidad de red; sin embargo, incrementa el coste y la complejidad de la solución.

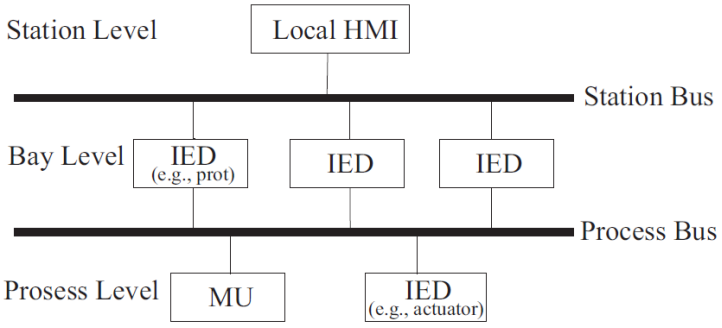

Ilustración 1 Redes IEC 61850 **[7]**.

El estándar IEC 61850 usa como elementos básicos de comunicación el modelo de datos GOOSE para la comunicación en el nivel de control y el bus de procesos [8]. Este modelo usa mensajes a nivel de capa 2 (Ethernet) y permite la interoperabilidad entre equipos de distintos fabricantes (ver Ilustración 2).

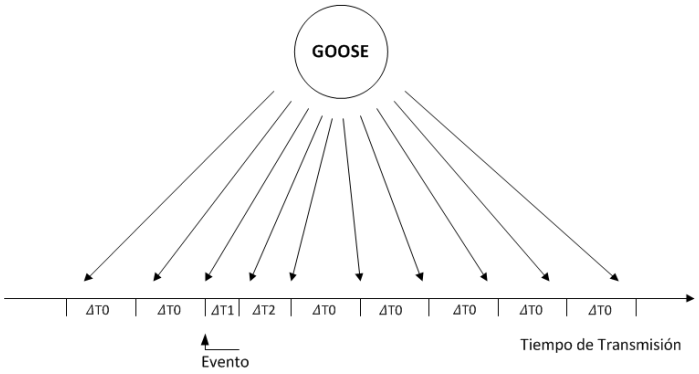

Ilustración 2 Modelo de transmisión de los mensajes GOOSE **[9]**.

La utilización de los mensajes GOOSE en la capa 2, ofrece importantes ventajas, pero carece de un mecanismo de confirmación de recepción del paquete; como lo ofrecen los protocolos de niveles superiores del modelo OSI. Por otra parte, para la comunicación con el Centro Nacional de Despacho (CND) se utiliza la norma la ISO 9506 (International Organization for Standardization), también llamada MMS. MMS es apropiado para cualquier aplicación que requiera un mecanismo común de comunicación para llevar a cabo una diversidad de funciones de comunicación relacionadas con el acceso en tiempo real, distribución de datos y control del proceso de supervisión [9].

4.2 Marco conceptual

4.2.1 Protocolos de comunicaciones redundantes

IEC 61850 en su primera edición no específica arquitectura o topología de la red de comunicaciones, ni los protocolos que permiten la protección del sistema de automatización; sin embargo, el desarrollo tecnológico incluyó al protocolo RSTP para este fin. En la edición 2 de la norma IEC 61850 publicada en el año 2010, se introducen dos protocolos para la protección de redes LAN dentro de los sistemas de automatización, estos son el PRP y HSR [10].

4.2.2 Rapid Spanning Tree Protocol (RSTP)

RSTP se usa en topologías de estrella o anillo de enlaces redundantes, y su tarea principal es evitar los lazos dentro de la red. Un lazo dentro de una red Ethernet se presenta cuando se tiene una doble conexión entre dos Switch’s, la presencia de un lazo dentro de la red hace que el tráfico no tenga salida y se vaya incrementando rápidamente hasta saturar y bloquear los Switch’s y demás equipos que conforman la red, para evitar este fenómeno el protocolo activa solo una de las dos conexiones (ver Ilustración 3).

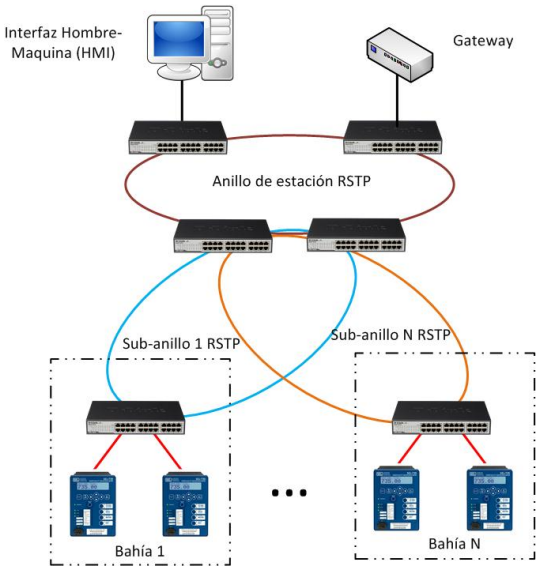

Ilustración 3 Topología de red basada en RSTP **[6]**.

Al detectarse una falla en uno de los equipos de la red, es necesario reconfigurar la red de comunicaciones. Esta función es realizada enviando mensajes a los diferentes nodos de la red para detectar cual es el enlace con problemas y para luego reconfigurarla. El tiempo de latencia o interrupción de restablecimiento para RSTP puede estar en el orden de 5 ms por cada nodo que conforma la red, dependiendo del número nodos que conforman la red esta reconfiguración puede tomar varios segundos [11]. Lo cual es inaceptable debido a la alta disponibilidad que tiene que tener la red de comunicaciones de una subestación eléctrica.

El control de cambios de la red de comunicaciones es difícil con protocolos dinámicos como RSTP, ya que estos toman decisiones de envío de paquetes en la red basados en topologías físicas o lógicas, y no los servicios que se ejecutan en los circuitos [11]. SDN, por el contrario, permite que las decisiones de envío de paquetes se basen en las aplicaciones que requieren comunicaciones y no en la topología. Los circuitos de reenvío son independientes de la topología, lo que significa que no importa cómo están conectados los Switch's, se selecciona la configuración de la ruta de reenvío y la base de los atributos deseados de transporte [3].

4.2.3 Parallel Redundancy Protocol (PRP)

Los IED's que poseen certificados de conformidad con el protocolo PRP poseen dos puertos Ethernet [8] [12]. Cada puerto está conectado a dos

redes Ethernet independientes (LAN A y LAN B). Los puertos envían las tramas Ethernet por ambas redes (incluyendo la información necesaria para asegurar el adecuado tratamiento de las mismas), los nodos destino reciben la primera trama y descartar la segunda trama; en caso de un fallo en una de las redes se sigue transmitiendo las tramas por la otra red LAN [6]. Una de las grandes ventajas del protocolo PRP es que sus tramas pueden circular por la infraestructura de red Ethernet convencional (ver Ilustración 4).

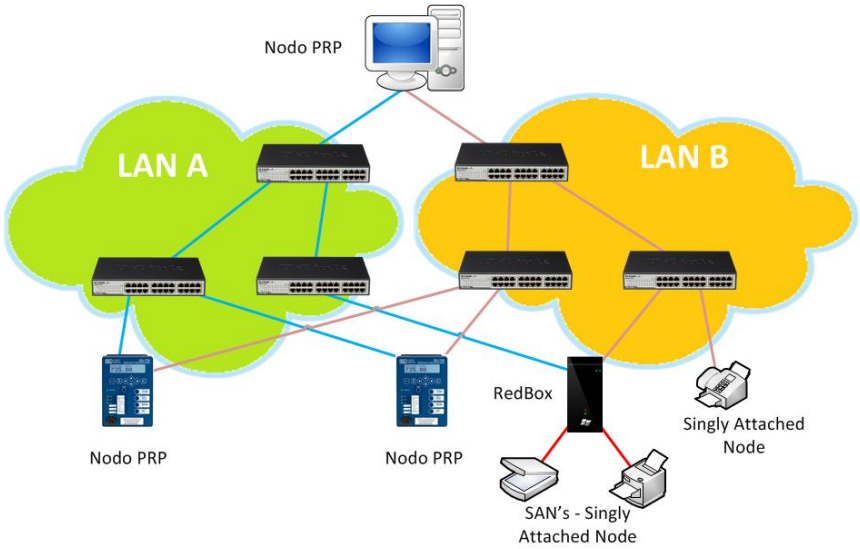

Ilustración 4 Topología de red basada en PRP **[6]**.

Los equipos Single Attached Nodes (SAN´s) que se muestran en la Ilustración 4, solo poseen un puerto con conexión Ethernet, por esto no presentan redundancia; un claro ejemplo de estos equipos son: servidores SCADA, estaciones de operación e ingeniería, impresoras, contadores de energía, etc. Aunque estos equipos pueden seguir funcionando de la forma convencional, por medio de un Redundancy Box (RedBox) se les puede dotar de redundancia y conectarlos a las dos redes LAN [9].

4.2.3.1 High-availability Seamless Redundancy (HSR)

HSR ofrece redundancia enviando los paquetes en las dos direcciones de un anillo Ethernet. Al igual que PRP, los equipos de una red HSR también deben disponer de dos puertos Ethernet, pero HSR incorpora el nodo DANH (Doubly Attached Node for HSR) que conecta sus dos interfaces a una misma red conformando una topología de anillo [11] [13]. En este caso los nodos de la red, además de gestionar la eliminación de tramas redundadas y la emisión de tramas de supervisión, deben reenviar las tramas que no sean

destinatarios. Los tiempos de latencia fijados en el estándar para la retransmisión de tramas son muy exigente (<5 us para Fast Ethernet) con el fin de poder conectar un número razonable de nodos en el anillo (ver Ilustración 5).

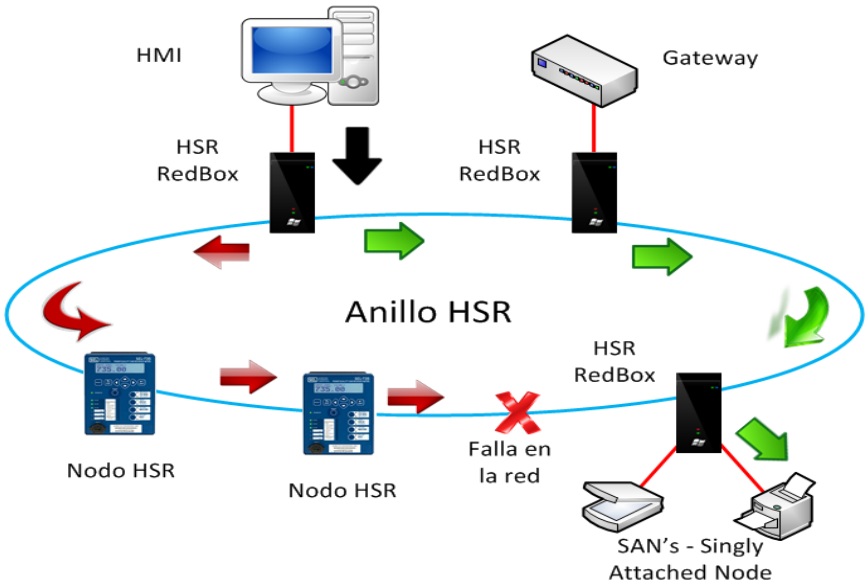

Ilustración 5 Topología de red basada en HSR **[6]**.

Con el fin de facilitar la implementación hardware de los mecanismos de gestión de tramas de baja latencia, la información específica HSR se sitúa al inicio de la trama Ethernet. Por este motivo y a diferencia de PRP, las tramas HSR no pueden ser retransmitidas por equipos Ethernet convencionales. Adicionalmente, los equipos SAN's solo podrán ser conectados a la red por medio de un RedBox [13].

Como se puede observar cada protocolo de comunicaciones tiene ventajas y desventajas con respecto a los otros. La norma IEC 61850 busca ampliar los conceptos eléctricos utilizados hasta ahora para la modernización o puesta en marcha de una subestación eléctricas, integrando las nuevas tecnologías de redes de comunicaciones. Los protocolos PRP y HSR son un claro ejemplo de esto, ya que presentan un tiempo de reconfiguración de cero ante la falla de un elemento de la red de comunicaciones. A la hora de una modernización del sistema de control de una subestación existente; el protocolo PRP es una muy buena alternativa, porque se podrían reutilizar los Switch's existentes de la red LAN del nivel de subestación. Y luego implementar una red RSTP para la comunicación con los niveles superiores, ya que estos no soportan PRP. Adicionalmente, solo es necesaria una latencia baja en el bus de subestación y de proceso para la acción de control

y protección de la subestación, no para la supervisión. Los esquemas PRP y HSR representan un reto para la sincronización de tiempo debido a que los retardos sobre las dos redes redundantes son diferentes y demandan la mejora de la robustez y precisión del reloj del sistema haciendo uso del protocolo IEEE 1588 [14]. Estos protocolos son idóneos para la modernización y puesta en servicio del sistema de supervisión y control de una subestación eléctrica, pero para el sistema de protecciones el panorama es muy diferente ya que carecen de ciberseguridad, control de tráfico, gestión de la red, segmentación de flujo y un menor tiempo de respuesta; estos servicios nos garantizarían disponibilidad para las funciones de protección de la subestación eléctrica.

4.2.4 Redes Virtuales

Un entorno de red soporta virtualización de la red si permite la coexistencia de múltiples redes virtuales en la misma entidad física. Cada red virtual en un entorno de virtualización de red es un conjunto de nodos virtuales y enlaces virtuales. Una red virtual es un subconjunto de los recursos físicos de la red, lo que propone la virtualización es el desacoplamiento de funcionalidades en un entorno de red mediante la separación de la función de los proveedores de servicios de internet en dos: los proveedores de infraestructura, que gestionan la infraestructura física y los proveedores de servicios, que crean redes virtuales por agregación de recursos de múltiples proveedores de infraestructura y servicios de red de extremo a extremo según la oferta [15]. Explicándolo de otra manera, la virtualización de la red es un entorno de red que permite a los proveedores de servicios múltiples para componer dinámicamente múltiples redes virtuales heterogéneas que coexisten juntas en forma aislada unas de otras. Los proveedores de servicios pueden desplegar y gestionar servicios personalizados de extremo a extremo en las redes virtuales para los usuarios finales mediante el intercambio eficaz y la utilización de los recursos de red subyacentes arrendados a múltiples proveedores de infraestructura [15]. Existen diferentes tecnologías de virtualización de la red como son VLAN, VPN, active and programmable networks y overlay networks.

4.2.5 Software Defined Networking

Una instancia SDN consta de tres partes principales: la aplicación, plano de control y plano de datos (ver Ilustración 5). La aplicación es la parte que

explota el control desacoplado y el plano de datos para lograr objetivos específicos, como un mecanismo de seguridad o una solución de medición de la red de comunicaciones [16].

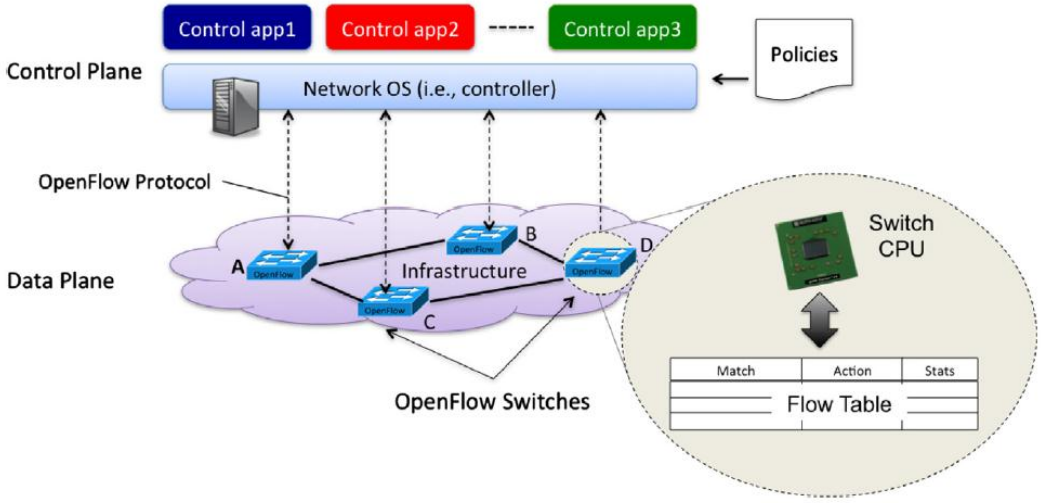

Ilustración 6 Componentes de SDN **[17]**.

Las aplicaciones se comunican con un controlador en el plano de control a través de la interfaz de este. El plano de control es la parte que manipula los dispositivos de reenvío a través de un controlador para lograr el objetivo específico de la aplicación de destino. El controlador utiliza la interfaz del Switch con capacidad para SDN para conectar con el plano de datos. El plano de datos es la parte que soporta un protocolo compartido (por ejemplo, OpenFlow) con el controlador y maneja los paquetes reales sobre la base de las configuraciones que son manipuladas por el controlador [17] [18].

4.3 Marco legal

La protección de las subestaciones eléctricas de alta y extra alta tensión ha sido y sigue siendo de gran importancia para las empresas de energía eléctrica, ya que estas subestaciones deben ser de alta disponibilidad. Un factor importante a tener en cuenta en la protección del sistema eléctrico es el desempeño de este ante la presencia de una falla (como los cortos circuitos que originan corrientes elevadas y esfuerzos mecánicos). En este proyecto se propone el estudio del desempeño de una red de comunicaciones genérica basada en IEC 61850, la cual se diseñará bajo estándares que garanticen tanto la seguridad como la confiabilidad; por lo anterior, esta deberá cumplir lo establecido en el Reglamento Técnico de Instalaciones Eléctricas (RETIE) [19] y adicionalmente los equipos de

comunicaciones requeridos en la subestación eléctrica deberán garantizar el intercambio de la información de supervisión, control y protección, entre el Usuario, el Transportador, el Centro Regional de Despacho (CRD) y el Centro Nacional de Despacho (CND), necesaria para la operación confiable del Sistema de Transmisión Nacional (STN); tal como se indica en el código de redes de la CREG (Comisión de Regulación de Energía y Gas) que hace parte del Reglamento de Operación del Sistema Interconectado Nacional [20].

4.4 Estado del arte

4.4.1 PRP y HSR en subestaciones electicas basada en IEC 61850

Las redes basadas en Ethernet han ganado aceptación para muchas aplicaciones de automatización industrial. Un ejemplo claro de esto, es la adopción por la IEC de los protocolos HSR y PRP para la automatización de subestaciones de energía (La Ilustración 8 muestra el flujo de paquetes de una red HSR). Estos protocolos no presentan retardo al reconfigurarse la red de comunicaciones y ni tampoco pérdida de tramas [21].

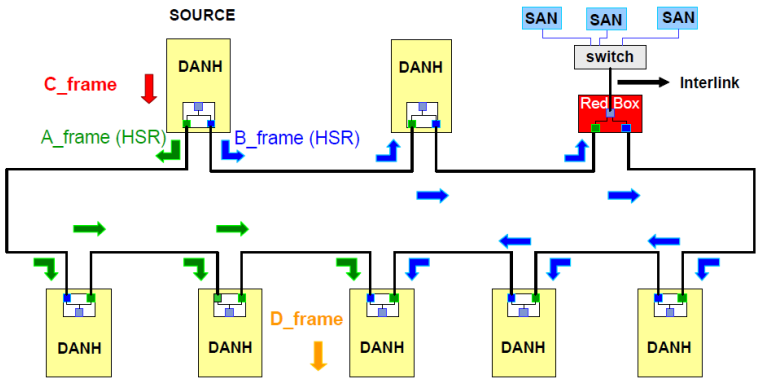

Ilustración 7 Configuración HSR para el tráfico unicast **[21]**

El protocolo HSR, descrito en la norma IEC 62439-3, se definió con el mecanismo de conmutación Cut-Through. La Ilustración 8 muestra el formato de trama definido para HSR, en el campo de identificación HSR EtherType se ha fijado en el comienzo de la trama; esta etiqueta en combinación con direcciones MAC de origen y destino da suficiente información para que el Switch Cut-Through pueda realizar la conmutación [21].

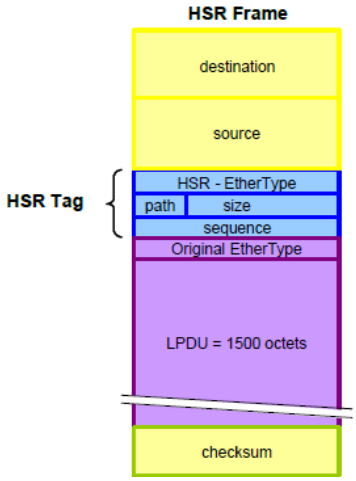

Ilustración 8 Formato de la trama HSR **[21]**

En la operación Cut-Through, la recepción de la etiqueta (TAG) de HSR es recibida aproximadamente en 2.4 µs a 100 Mbits/s; por su parte, la recepción en Store-and-Forward de la trama se realiza aproximadamente en 123 µs en 100 Mbit/s para el tamaño máximo de la trama (1.522 octetos). Las arquitecturas de conmutación IP diseñadas específicamente para los protocolos PRP y HSR, combinan la baja latencia de conmutación de Cut-Through a través de los puertos redundantes con un Store-and-Forward optimizando el modo de operación con el esquema de memoria distribuida para los puertos restantes. El tiempo de latencia teórico se ha calculado teniendo en cuenta los campos obligatorios de las tramas Ethernet que deben ser analizados para tomar la decisión de conmutación [21].

PRP y HSR son protocolos que ofrecen confiabilidad en muchas aplicaciones industriales de la capa 2. Sin embargo, los tiempos de respuesta requeridos por los procesos de control y protección en tiempo real, como el bus de proceso en automatización de subestaciones de energía, necesitan implementaciones de hardware de baja latencia [21].

4.4.2 Proyecto experimental OFELIA basado en NetFPGA's

La red experimental OFELIA basada en OpenFlow está diseñada para redes flexibles, configurables y separación del recursos (ver Ilustración 10), sin limitar las capacidades de experimentación, especialmente las características de OpenFlow, en los usuarios de OFELIA. Para aprovechar al máximo la infraestructura, la mayoría de los diseños en islas permiten topologías virtuales arbitrarias conectando físicamente los Switch OpenFlow en una

topología de malla parcial o total, y mediante la conexión de cada uno de los anfitriones finales o servidores virtuales para múltiples Switch's. Como muchos investigadores ejecutan diferentes experimentos sobre el mismo banco de pruebas, la interferencia entre los recursos de la red que se utiliza es posible [22].

Ilustración 9 Mapa de conexiones del proyecto OFELIA **[22]**

El objetivo de este proyecto es dar al experimentador la flexibilidad para ejecutar experimentos sin verse limitados por corte de prácticas, y particularmente en el de corte de red de comunicaciones. En una red OpenFlow una segmentación puede ser definida por un subconjunto o parte del tráfico que lleva la red que se controla a través del protocolo OpenFlow. Este concepto se conoce como flowspace. Se evaluaron varias estrategias para segmentar la red por aislamiento de tráfico, tales como el corte basada en MAC, cortes basados en VLAN y la asignación arbitraria flowspace incluso bajo demanda [22].

El banco de pruebas OFELIA se ha construido con el fin de albergar múltiples capas y múltiples tecnologías a través de una estructura de red pan-europea, el paradigma SDN siendo la principal filosofía a seguir y el protocolo OpenFlow utilizado como factor clave. El diseño e implementación de OFELIA se basa en tres pilares fundamentales: la flexibilidad, diversidad de recursos para el investigador, facilidad de organización del experimento, y la facilidad de control y gestión de los administradores [22].

4.4.3 Evaluación de Software Defined Networking en sistemas eléctricos basados en IEC 61850

A continuación se plantearán dos escenarios de pruebas para el tráfico de mensajes GOOSE. En el primer escenario se generarán dos flujos de mensajes GOOSE que van desde el cliente 1 al cliente 2, uno de los flujos de mensajes se priorizará (la velocidad de datos de un flujo de mensajes es de aproximadamente 200 kbit/s). El flujo en la red de telecomunicaciones entre el cliente 1 y el cliente 2 es muy alto, ya que el cliente 1 también genera un flujo adicional hacia el cliente 2 con una velocidad de datos de aproximada de 440 Mbit/s. El segundo escenario de prueba utiliza un único flujo de tráfico GOOSE desde el cliente 1 al cliente 2. La duración de ambos escenarios de prueba es de unos 300 segundos [23].

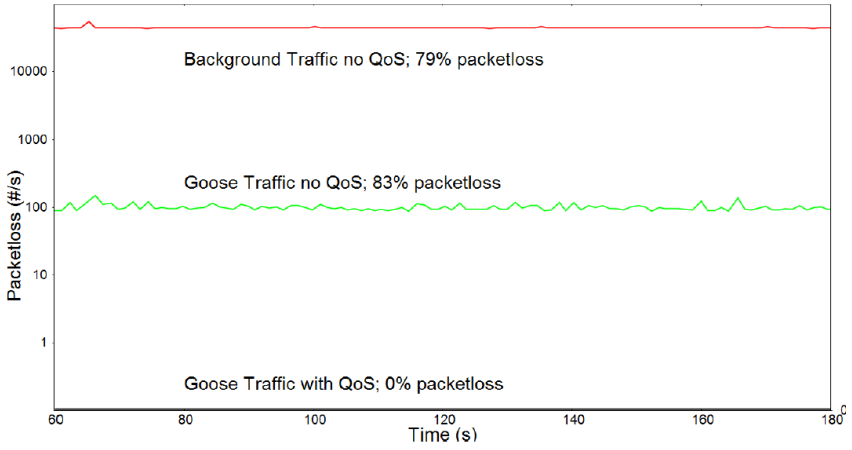

Ilustración 10 Escenario 1 con priorización de paquetes **[23]**.

La Ilustración 10 muestra la pérdida de paquetes por segundo de los flujos generados para el escenario 1, en donde se observa que en el flujo de mensajes GOOSE con prioridad no se presenta pérdida de paquetes. Su demora mínima durante la medición era 0.061 ms, el retraso máximo para algunos paquetes fueron 152 ms. Por otra parte, los flujos sin prioridad tenían una pérdida de paquetes de aproximadamente 80%. El resultado de escenario de prueba 1 muestra que una red SDN OpenFlow generalmente se puede utilizar para transportar tráfico de alta prioridad [23].

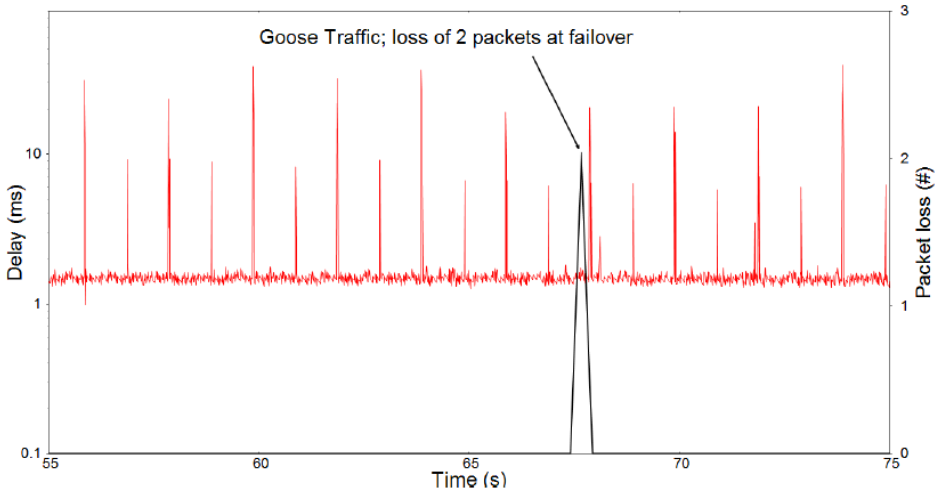

Ilustración 11 Escenario 2 sin priorización de paquetes **[23]**.

En la Ilustración 11 se puede observar la simulación del tráfico de los mensajes GOOSE entre el cliente 1 y el cliente 2, donde el retardo promedio es de aproximadamente 2 ms, este valor es significativamente mayor al del escenario 1 (con priorización de paquetes).

Como se puede observar, SDN y OpenFlow ofrecen importantes ventajas conceptuales a través de redes convencionales y funcionalidades básicas, como la priorización del tráfico en la red. Una ventaja conceptual de utilizar OpenFlow en una red de comunicación de energía es que los flujos de control y protecciones pueden ser totalmente definidos de antemano, y en caso de una falla, la red de comunicaciones de la subestación electrica se comporta de una manera bien definida.

4.4.4 SDN en subestaciones eléctricas basada en IEC 61850

En redes industriales o de misión crítica, es necesario desarrollar mecanismos para asegurar el funcionamiento de la red de tal manera que los servicios prestados no se vean comprometidos ante una falla. En la actualidad, el concepto Smart Grid es un caso de uso crítico relevante, que incluye la protección, automatización y control de sistemas de energía eléctrica, y que se apoya en los servicios de las TIC (Tecnologías de Información y Comunicaciones) que deben cumplir con los requisitos de tiempo real de estas aplicaciones con alta fiabilidad y seguridad. La Comisión Electrotécnica Internacional es el organismo de normalización más importante en esta área, siendo destacable la norma IEC 61850 para aplicaciones emergentes de servicios públicos de energía, que propone una

serie de especificaciones que definen globalmente la configuración de una subestación de energía, incluyendo nuevos servicios de comunicación, requisitos y prioridades [24].

4.4.4.1 Integración OpenFlow e IEC 61850

Durante el proceso de integración de las comunicaciones y el modelo de datos IEC 61850, los archivos SCD se traducen en información que puede ser utilizada por el controlador OpenFlow, la obtención de información precisa acerca de la configuración de la subestación. De esta forma, el controlador puede saber los dispositivos existentes y su flujo de datos en la red de la subestación, por lo que puede determinar automáticamente la topología lógica de esta. El controlador tiene que decidir cómo enrutar, modificar o descartar los flujos de tráfico [16], separando físicamente el tráfico de baja prioridad con el de alta prioridad (mensajes GOOSE), lo que minimiza fácilmente desviaciones de latencia (Ver Ilustración 12).

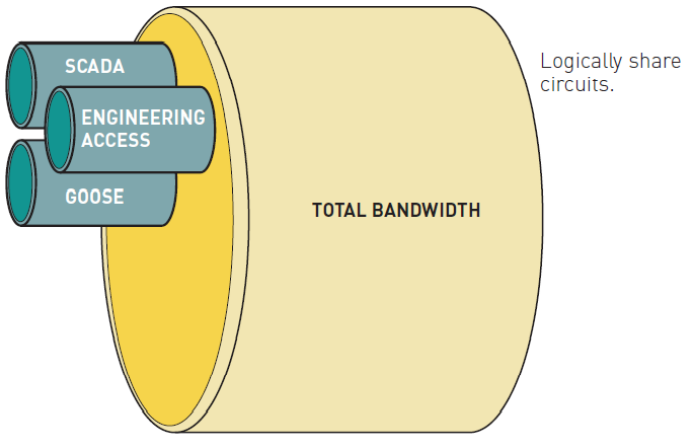

Ilustración 12 Ancho de banda compartido lógicamente **[25]**.

Varias propiedades se obtienen después de un análisis correcto de los archivos SCD (ver Ilustración 13). Estos nos permiten construir entradas de reenvío de flujo que se instalan en los Switch, permitiendo a los IED's publicar y suscribirse a la información.

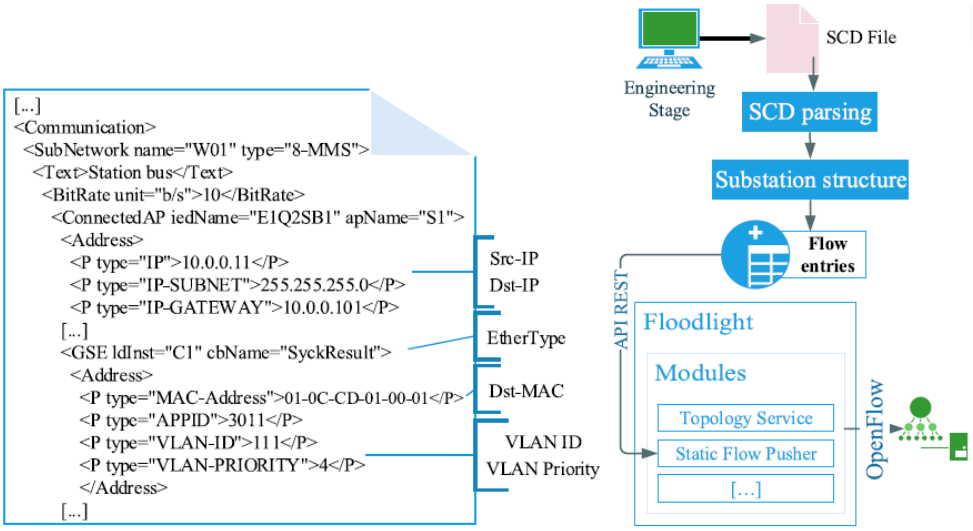

Ilustración 13 Integración OpenFlow y IEC 61850 **[24]**.

El controlador tiene que conocer la conexión entre los Switch's y los IED (puertos y topología física). Luego de traducir los archivos SCL, se debe tener en cuenta que se conoce la conexión de cada dispositivo con cada Switch, por lo que esta información se incorpora de forma estática en la lógica del controlador. Es decir, el controlador tiene un conocimiento a priori de las conexiones del IED y la conexión de un cada IED con cada Switch de acceso. Por lo anterior, SDN proporcionaría a la red de comunicaciones de la subestación IEC 61850 mayor ciberseguridad, gestión de red, disponibilidad, segmentación de flujo, robustez y un menor tiempo de respuesta en las funciones de supervisión, control, protección y medida [24].

5 METODOLOGÍA

El proyecto se basó en la metodología AIM usada por Oracle en proyectos de software. AIM presenta varias fases (Ver Ilustración 14), pero para este proyecto solo se usaron cuatro (4) de ellas, las cuales se describen a continuación:

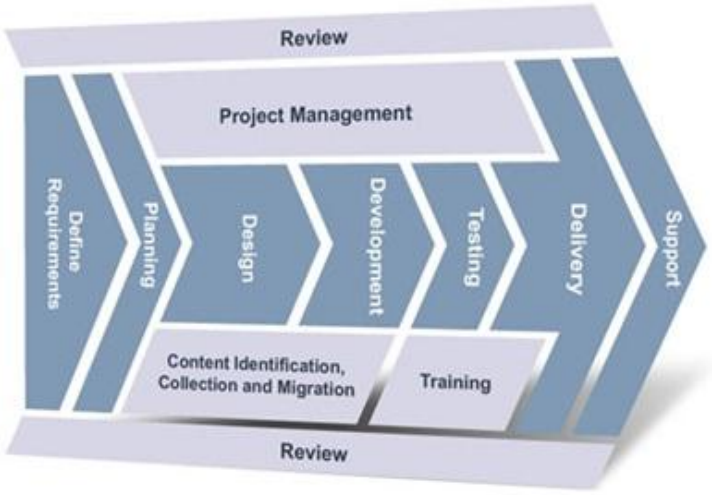

Ilustración 14 Metodología AIM usada por Oracle.

Fase de Definición y planeación: En esta fase se realizó la búsqueda bibliográfica sobre el concepto Software Defined Networking, y luego se describió en el capítulo "*Aspectos técnicos del concepto Software Defined Networking*" del documento de tesis y en el artículo publicable; como se aplicaron los aspectos técnicos principales de SDN en el diseño de la red de telecomunicaciones del sistema de automatización de una subestación eléctrica genérica IEC 61850.

Fase de diseño: En esta fase se propuso la arquitectura y topología de la red de telecomunicaciones del sistema de automatización de una subestación eléctrica genérica IEC 61850 y se describió en el capítulo "*Arquitectura y topología de la red de telecomunicaciones del sistema de automatización de una subestación eléctrica genérica IEC 61850*" del documento de tesis, teniendo en cuenta los aspectos técnico principales de las Software Defined Networking.

Fase de desarrollo: Se definió el controlador y Switch OpenFlow usados en el diseño de la red de telecomunicaciones SDN del sistema de automatización de una subestación eléctrica genérica IEC 61850. Y luego se redactó el

capítulo “*Pruebas del comportamiento de la red de telecomunicaciones SDN del sistema de automatización de una subestación eléctrica genérica IEC 61850*” del documento de tesis. En esta fase se realizaron algunos ajustes al diseño de la arquitectura y la topología.

Fase de entrega: Para finalizar se analizaron las pruebas donde se observó el comportamiento de la red de telecomunicaciones SDN del sistema de automatización de una subestación eléctrica genérica IEC 61850 el análisis y conclusiones de las pruebas quedaron redactadas en el capítulo “*Análisis y conclusiones del comportamiento de la red de telecomunicaciones SDN del sistema de automatización de una subestación eléctrica genérica IEC 61850*” del documento de tesis y en el artículo publicable.

6 PRESENTACIÓN Y ANÁLISIS DE RESULTADOS

En este capítulo se enuncian los aspectos técnicos de Software Defined Networking, como arquitecturas y topologías de redes de comunicaciones de subestaciones eléctricas; y se describe a detalle cómo se implementó el Switch OpenFlow en el dispositivo NetFPGA, donde se pudo observar el comportamiento de los paquetes de mensajes críticos para la subestación eléctrica genérica basada en IEC 61850.

6.1 Aspectos técnicos del concepto Software Defined Networking

Las arquitecturas de red actuales son estáticas, no son programables y los desarrolladores de redes no pueden agregar nuevos servicios de manera arbitraria debido a la limitación de las tecnologías de red, a la complejidad, a la incapacidad de escalar la red y a la dependencia a proveedores. La razón principal de la creación de SDN es eliminar estas dependencias hacia los proveedores de equipos, proporcionando una manera más eficiente de gestionar la red, una mayor flexibilidad de respuesta a las demandas y maneras de innovación de forma más rápida [26]. A continuación se enuncian las principales características de SDN:

- El plano de control y el plano de datos se encuentran desacoplados y abstraídos el uno del otro [27] [16].
- La inteligencia de la red se encuentra lógicamente centralizada, lo cual permite tener una visión global de la red y de los cambios en las demandas de la infraestructura de red, lo que hace posible el desarrollo de diferentes aplicaciones.
- El plano de datos al ser programable aporta facilidades en la automatización y flexibilidad a las redes.

SDN simplifica el diseño de la red y las operaciones; permitiendo a los investigadores y los proveedores de red que puedan programar la red sin interrumpir el tráfico de la producción, desarrollar rápidamente nuevos servicios y de forma independiente. Además, la flexibilidad de SDN permite a los administradores de red configurar, administrar, asegurar y optimizar los recursos de red de forma automática. SDN convierte las redes estáticas en una plataforma de prestación de servicios capaz de responder rápidamente a las necesidades cambiantes del negocio, de los usuarios finales y del mercado [26]. Por lo tanto, actualmente una variedad de dispositivos de red y software se han adaptado OpenFlow basándose en SDN porque permite gestión y control de red centralizado, automatización de la gestión de la red,

la configuración de la red se realiza mediante programación, aumentando la fiabilidad y seguridad en la red, más control granular de la red y mejor experiencia del usuario final.

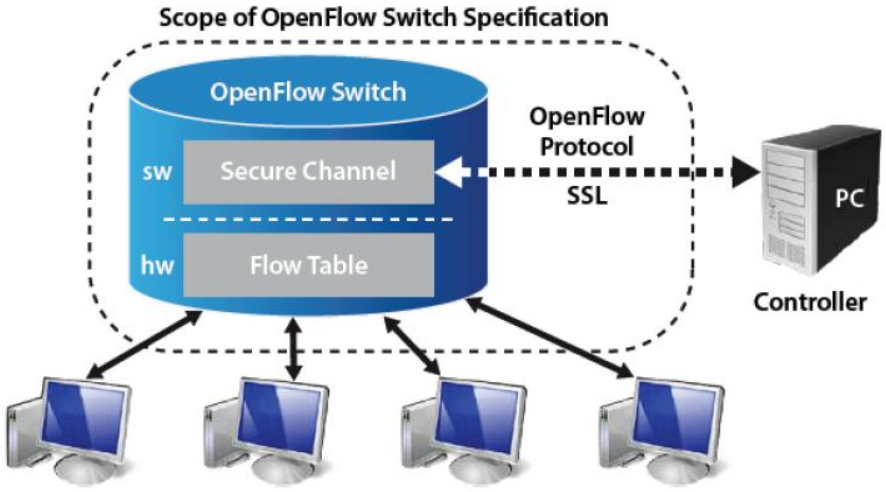

Ilustración 15 Host conectados a un Switch OpenFlow **[28]**.

SDN proporciona a los administradores de red muchas ventajas, ya que estos expanden SDN a la red, lo cual permite que los recursos de red se pueden compartir de manera segura por múltiples grupos de usuarios. Además, los administradores de red pueden mantener fácilmente redes virtuales con sus equipos de cómputo asociados y los recursos de almacenamiento. Los administradores pueden implementar balanceo de carga con un Switch OpenFlow y un servidor de productos básicos. Esta solución permite a administradores controlar mejor el flujo de tráfico en toda la red para mejorar el rendimiento de esta. Además, ya que los administradores se esfuerzan por ampliar los beneficios de la virtualización de servidores y almacenamiento a la red, están limitados por la infraestructura de la red física (Ver Ilustración 15). Sin embargo, una red OpenFlow virtualizada elimina estas limitaciones, lo que permite a los administradores crear una abstracción de red virtual, basándose en el flujo el cual amplía los beneficios de la virtualización a nivel de red [26].

6.1.1 Controlador POX

POX proporciona un marco para la comunicación entre el Switch y el controlador utilizando el protocolo OpenFlow. POX es un controlador que fue

desarrollado para cubrir las necesidades de redes SDN usando Python como lenguaje de programación y es usado en sistemas operativos como Windows, Mac OS y Linux. POX es capaz de detectar la topología de la red de comunicaciones y seleccionar rutas de acceso, posee interfaz gráfica y componentes de visualización. POX fue desarrollado para controlar Switch's OpenFlow en versión 1.0 [2]. La configuración e instalación del controlador POX se detalla en el Anexo 1.

POX fue diseñado con varios componentes en los que se destacan samples.pretty_log, openflow.webservice, forwarding.l2_learning, py, forwarding.l3_learning, openflow.of_01, entre otros. En [29] se describen a detalle cada uno de los componentes del controlador POX.

6.1.2 Switch OPENFLOW

Switch consiste en una o más tablas de flujo [26], que realizan búsquedas de paquetes y por un canal seguro se conecta el Switch Openflow a un proceso remoto con el controlador. El controlador gestiona el Switch permitiendo que los comandos y los paquetes se intercambien a través del protocolo OpenFlow, que proporciona una interfaz estándar para la comunicación. El controlador, a través del protocolo OpenFlow, puede añadir, eliminar y actualizar entradas del flujo de datos que pasan a través del Switch [28] [30]. La tabla de flujo está constituida por campos de comparación, contadores, y un conjunto de instrucciones que se aplicarán a los paquetes correspondientes. Las acciones básicas asociadas con la entrada de flujo son:

- Reenvío de paquete a un puerto de salida; esto permite que los paquetes se enruten a través de la red de comunicaciones.
- Encapsular el paquete y enviarlo al controlador a través de la interfaz de comunicación. Por lo general, esta acción se aplica al primer paquete de un nuevo flujo.
- Paquetes de descarte que pertenecen al flujo, o que son utilizados para la seguridad, para contener los ataques de denegación de servicio [28].

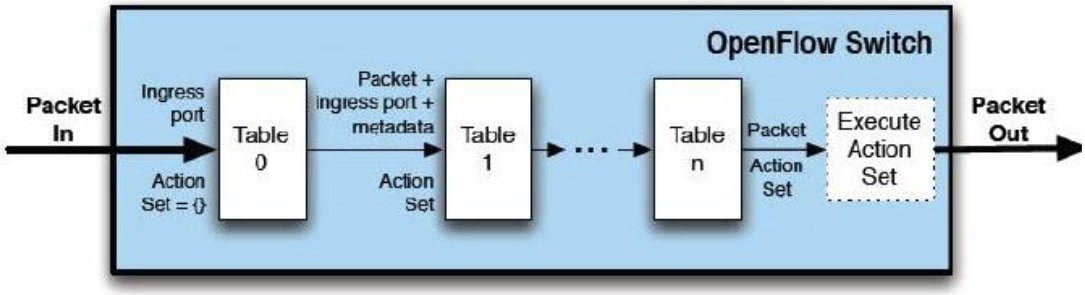

Ilustración 16 Paquetes de datos a través de tablas de flujo **[30] [2]**.

6.1.2.1 Operación del plano de control

El plano de control puede ser reactivo por el manejo de los nuevos flujos sólo después de que llegan mediante la inserción de las entradas de flujo, incluso antes de que llegue el flujo de datos, o se puede utilizar una combinación de los enfoques basados en los flujos específicos. OpenFlow es un estándar abierto que se aplica en Software Defined Networking, el cual separa el plano de control y el plano de datos en equipos de red tradicional, como Switch's y Router's. El plano de control se rige por una entidad dedicada y centralizada llamada controlador. El controlador se comunica con switch OpenFlow a través de un canal de comunicación seguro que se especifica por el protocolo OpenFlow (a través de un mensaje stats_request). Un flujo de red se puede definir como una combinación de cabeceras de los paquetes a través de diferentes capas. Por consiguiente, las decisiones de conmutación y enrutamiento de flujos de red se pueden definir por una amplia gama de aplicaciones que se ejecuta en el controlador [31] [32]. El modo reactivo,

stats_request, y los algoritmos de descubrimiento de topología pueden proporcionarle al plano de control sobre la red dinámica (ver Ilustración 17).

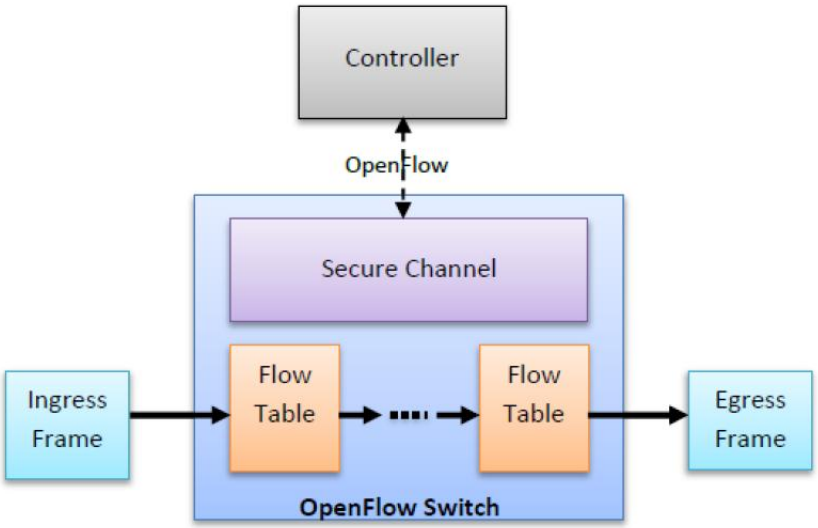

Ilustración 17 Principios de OpenFlow **[28]**.

El plano de control puede comportarse de una manera centralizada y administrar una gran cantidad de Switch's interconectados que forman una isla OpenFlow [33] [34]. En ese despliegue, el controlador puede utilizar el mensaje packet_out para instruir el plano de datos para enviar paquetes LLDP desde los puertos del Switch a los enlaces de red de comunicaciones. Hay que tener en cuenta que un paquete LLDP entrante en un puerto del Switch se enviará al controlador como un mensaje packet_in. Sobre la base de estos packet_ins LLDP, el plano de control puede inferir la conectividad dentro de la isla y en la construcción de la topología de la red [17].

6.1.2.2 Plano de datos

Una función básica del plano de datos es el reenvío de paquetes. Además, el plano de datos puede apoyar algunas tareas dentro de la red para realización de diversos servicios como el almacenamiento en caché de la red y la trans-codificación. Dentro de la red de servicios se requiere un procesamiento adicional que consume recursos del Switch, como CPU, memoria y almacenamiento [18]. Por lo anterior, es necesitamos una mayor robustez en el hardware del Switch OpenFlow para que su rendimiento no sea degradado [18].

6.1.2.3 PLATAFORMA DE DESARROLLO

6.1.2.3.1 Sistema Operativo CentOS 5.11

El Switch Openflow fue configurado en el sistema operativo CentOS 5.11, el cual es una distribución de Linux estable, predecible, manejable, que no posee costo alguno, de libre distribución y una plataforma reproducible derivada de las fuentes de RHEL. CentOS es una distribución apoyada por la comunidad de fuentes proporcionadas libremente al público por Red Hat desde el año de 2004. El propósito de CentOS es ser funcionalmente compatible con RHEL y es desarrollado por un pequeño grupo de desarrolladores principales; que a su vez son apoyados por una comunidad de usuarios activa que incluye desde administradores de sistemas, administradores de red, principales contribuyentes de Linux, y los desarrolladores de Linux de todo el mundo (para un mayor detalle acerca de CentOS ver la referencia [35]).

6.1.2.3.2 Plataforma NetFPGA

A continuación se presenta las características principales de la plataforma "*NetFPGA D247089*", la cual fue utilizada para la implementación del Switch OpenFlow; esta posee el integrado Virtex-II Pro 50, el cual es un FPGA de Xilinx. En la Tabla 1 se detalla las principales características del FPGA.

Logic Cells	Configurable Logic Blocks		18 X 18 Bit Multiplier Blocks	Block Select RAM+		DCM's	Maximum User I/O Pads
	Slices	Max Distr RAM (Kb)		18 Kb Blocks	Max Block RAM (Kb)		
53,136	23,616	738	232	232	4,176	8	852

Tabla 1 Principales características de Virtex-II Pro 50 (XC2VP50) [1]

Los FPGA son arreglos de celdas lógicas cuyos componentes lógicos e interconexiones entre éstas pueden ser programadas luego de su fabricación. En la Ilustración 18 se muestra la arquitectura genérica de una FPGA [36].

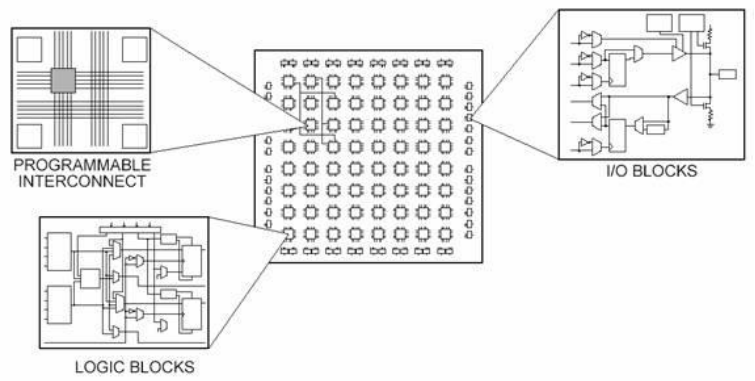

Ilustración 18 Arquitectura de un FPGA.

Los FPGA están constituidos por diferentes bloques funcionales como son:

- DLL: este bloque permite manejar señales de reloj con el fin de garantizar que la señal de reloj llegue en el mismo instante a todos los CLB de la arquitectura interna del FPGA.

- CLB: Está constituido por 4 LUT y 4 flip-flop.

- LUT: Son usualmente configuradas en memorias RAM estática, donde se guardan todos los posibles resultados de la función que se desea simular en las diferentes posiciones de la misma. Entonces, para poder simular dicha función, las señales de entrada van conectadas directamente en las entradas de direccionamiento de la memoria y como previamente se guarda para cada posición de memoria un resultado, al ingresar el valor de las entradas, realmente se obtiene el valor de una posición de memoria en donde previamente se ha calculado, el cual es entregado por la memoria como el resultado para dichas entradas.

- Flip-Flops: Los cuales están destinados para almacenar datos de las señales y lograr así el desarrollo de máquinas de estado en caso de ser necesarias [36].

Ilustración 19 Plataforma NetFPGA-1G [37]

Los procesadores de señal están siendo implementados en dispositivos FPGA para diversas aplicaciones, con el fin de mejorar el rendimiento, la economía, la flexibilidad y el consumo de potencia. La tecnología de FPGA ha tenido gran aceptación en la industria de telecomunicaciones. Casi el 50% de toda la producción de FPGA se encuentra en equipos de telecomunicaciones (estaciones de base inalámbricas, multiplexores/conmutadores, enrutadores y módems). Los FPGA brindan gran flexibilidad, que puede permitir atender múltiples estándares. Un ejemplo es un micro-teléfono celular universal que reconozca los diferentes patrones de señal de sistemas TDMA, CDMA, FDMA u OFDM automáticamente y se reconfigure para utilizar el protocolo identificado. La flexibilidad y el rendimiento que proveen los FPGA también permiten que los diseñadores estén fácilmente al día con estándares en evolución [38].

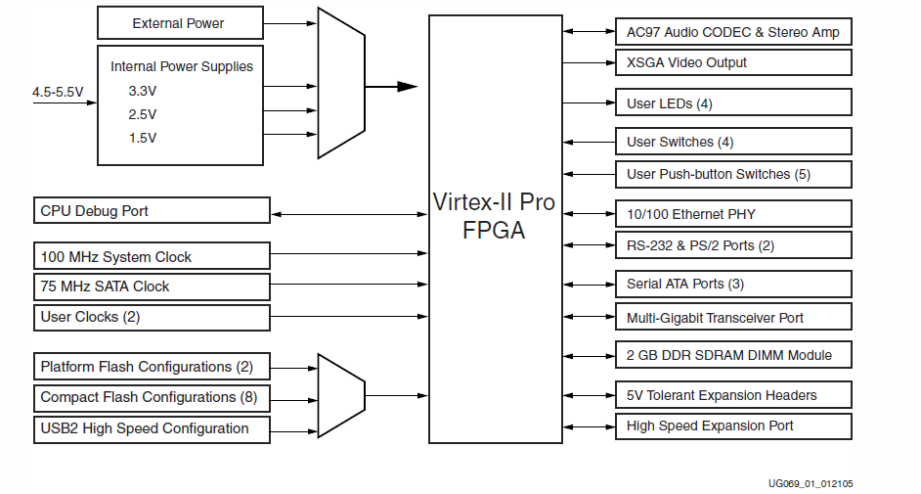

Ilustración 20 Diagrama de bloques típico de la Virtex-II Pro [39]

La plataforma de la Ilustración 19 posee cuatro puertos Ethernet de 1 Gigabits cada uno con conector RJ45, compatible con 10/100/1000 Ethernet PHY (MII / GMII / RMII). Además de una estructura lógica de alto rendimiento, la Virtex-II Pro contienen muchos bloques funcionales a nivel de sistema integradas (Ver Ilustración 20). Estas características permiten a los desarrolladores construir los más altos niveles de rendimiento y funcionalidad en los sistemas basados en FPGA [1] [39].

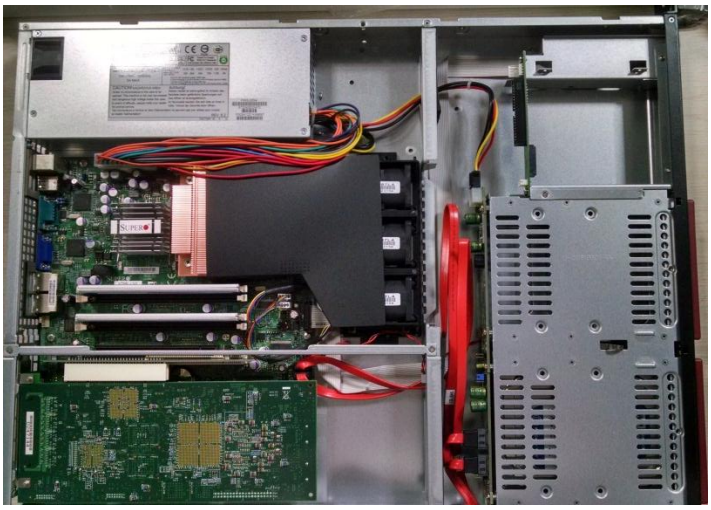

Ilustración 21 NetFPGA en el PC Dell 2950 [40]

En este proyecto en particular, la NetFPGA y el computador son dos secciones de hardware; la NetFPGA fue conectada al computador a través de la PCI (como se muestra en Ilustración 21). El entorno de diseño del Switch OpenFlow se describe a continuación:

- NetFPGA XC2VP50 (Digilent NetFPGA Board).
- Procesador: Intel Dual Core E6420.
- Sistema Operativo: CentOS Installed (free).
- Motherboard: PDSBM-LN2.
- RAM: 2048mb (2gb) DDR2.

En el Anexo 2 y Anexo 3 se describió a detalle la instalación y configuración de los driver's de la NetFPGA y la configuración del Switch OpenFlow respectivamente.

6.2 Arquitectura y topología de la red de telecomunicaciones del sistema de automatización de una subestación eléctrica genérica IEC 61850

En la actualidad existen muchas topologías de redes de comunicaciones en subestaciones eléctricas, las más comunes son las mostradas en el numeral 4.2.1 o una combinación de ellas. A la hora de modernizar el sistema de control y protección de una subestación existente; el protocolo PRP es una muy buena alternativa, porque se podrían reutilizar los Switch's existentes de la red LAN del nivel de subestación. Y luego implementar una red RSTP para la comunicación con los niveles superiores, ya que estos no soportan PRP (ver Ilustración 22). Adicionalmente, solo es necesaria una latencia baja en el bus de subestación y de proceso para la acción de control y protección de la subestación, no para la supervisión.

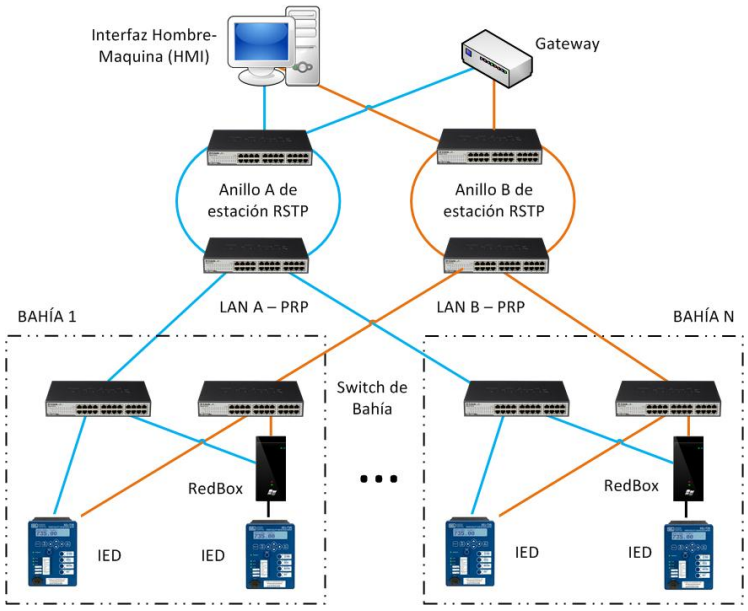

Ilustración 22 Topología de red basada en PRP y RSTP [6].

Por el contrario, para la construcción de una subestación nueva; la implementación del protocolo PRP requiere la habilitación de una red totalmente redundante, convirtiéndola en una opción costosa, y resultando HSR más conveniente o una combinación de los protocolos antes enunciados, y por medio de RedBox o QuadBox conectar las redes (ver Ilustración 23).

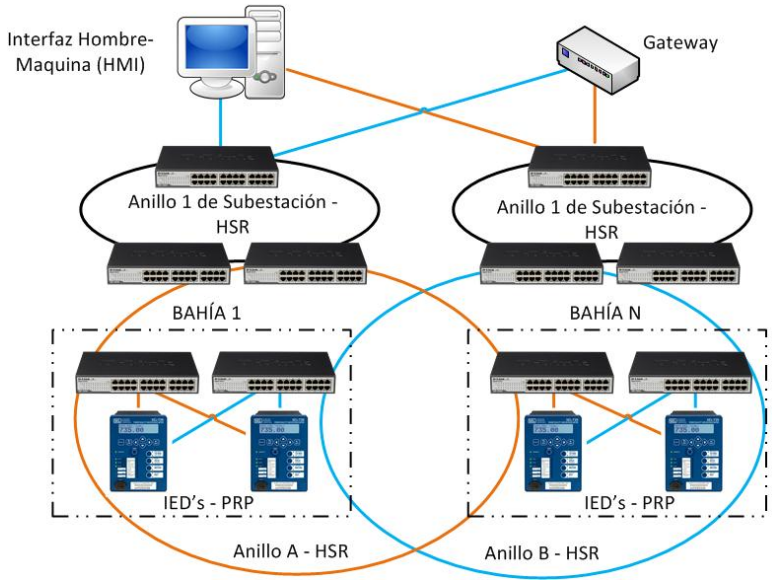

Ilustración 23 Topología de red basada en PRP y HSR [6].

La elección de uno u otro protocolo depende de varios factores, entre estos la disponibilidad requerida para la subestación, el presupuesto, existencia de equipos que cumplan los protocolos, etc., no es necesario que todas las subestaciones manejen el mismo protocolo [41]. Los requisitos de rendimiento para las funciones pueden ser diferentes dependiendo del nivel de tensión y del papel que desempeña la subestación eléctrica, es decir, en distribución y en transmisión [4]. Existen varios tipos de mensajes en la red de comunicaciones de las subestaciones eléctricas, algunos con mayor prioridad que otros como se muestra en la Tabla 2:

Performance class	**Transfer time [ms]**	**Application examples: Transfer of**	**Examples**
TT0	>1000	Files, events, log contents	Recierre Habilitado Trifásico, archivos de osciloperturbografía, relé no disponible.
TT1	1000	Events, alarms	Alarma por baja presión de un interruptor, relé no disponible.
TT2	500	Operator commands	Comando de apertura/cierre del interruptor
TT3	100	Slow automatic interactions	Secuencias de maniobra.
TT4	20	Fast automatic interactions	Bloqueo por oscilación de potencia, bloqueo por recierre.
TT5	10	Releases, Status changes	Protección de distancia, señal permisiva de una protección, seccionador abierto/cerrado.
TT6	3	Trips, Blockings	Disparo baja frecuencia, disparo sobre tensión, disparo diferencial.

Tabla 2 Requerimientos para las diferentes clases de mensajes [4].

Para este proyecto en particular se enfocó en la priorización de los mensajes de críticos, estos mensajes típicamente están compuestos por un código binario simple, que contiene un comando o un mensaje, por ejemplo, "Trip", "Close", “Reclose order”, "Start", "Stop", "Block", "Unblock", “Trigger”, “Release”, “State change” [4]. El IED que recibe el mensaje, ejecutará acciones inmediatamente. Todos estos mensajes se refieren a momentos críticos, como las funciones de protección de los equipos de patio como son interruptores, seccionadores y transformadores [4]. El Trip es el mensaje más rápido en subestación eléctricas; por lo tanto, este mensaje tiene requisitos más exigentes en comparación con todos los demás mensajes (Ver Tabla 2). El mismo rendimiento puede ser solicitado para el enclavamiento y lógicas entre las funciones de protección [4]. Como la finalidad de este proyecto es observar el comportamiento de la información crítica se usará una topología en estrella como se muestra en Ilustración 24.

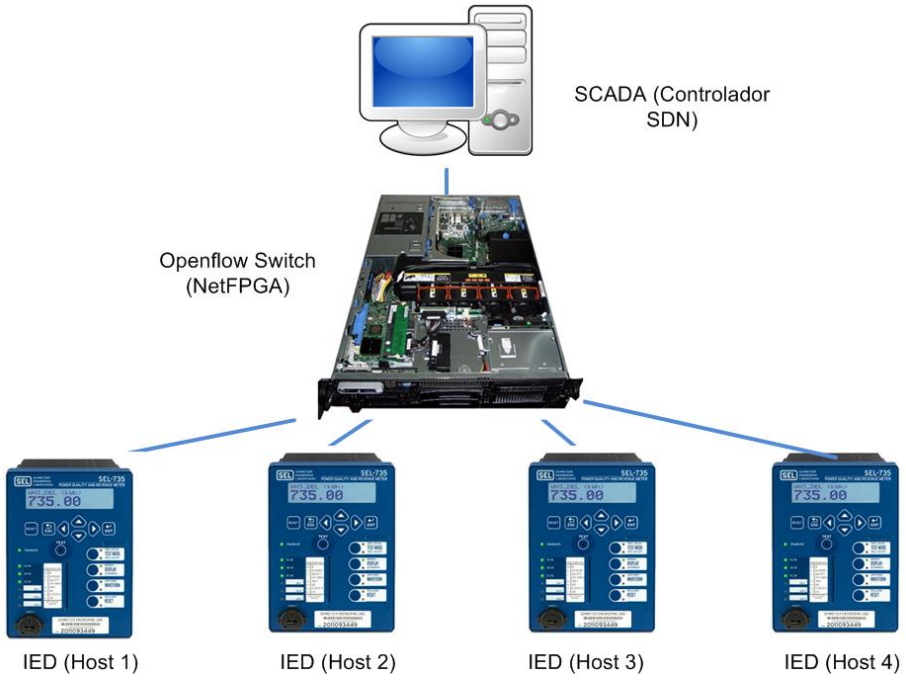

Ilustración 24 Topología en estrella de SDN.

Los equipos de control y protección (transformadores, interruptores, relés de control y relés de protección) que conforman la arquitectura de la red SDN de la subestación eléctrica son representados por los Host de la Ilustración 24 y presentan las siguientes características:

Equipo	Sistema Operativo	Dirección IP	
Controlador SDN	Ubuntu 16.04 (64 bits)	eth1:	192.168.100.100
Switch OpenFlow (con NetFPGA)	Centos 5.11 (32 bits)	eth1: nf2c0: nf2c1: nf2c2: nf2c3:	192.168.100.22 192.168.0.10 192.168.0.11 192.168.0.12 192.168.0.13
Host 1	Ubuntu 12.04 (32 bits)	eth0:	192.168.0.200
Host 2	Ubuntu 12.04 (32 bits)	eth0:	192.168.0.201
Host 3	Ubuntu 12.04 (32 bits)	eth0:	192.168.0.202
Host 4	Ubuntu 12.04 (32 bits)	eth0:	192.168.0.203

Tabla 3 Características de los equipos que conforman la red SDN de la bahía de la subestación.

6.3 Pruebas del comportamiento de la red de comunicaciones SDN de la subestación

En esta sección se describió a detalle las diferentes pruebas que se realizaron a la red SDN de la subestación eléctrica genérica, con el fin de observar el comportamiento de esta. En las pruebas se usó el comando *hping3* como herramienta de generación de paquetes de datos; ya que este comando permitió configurar parámetros como número de paquetes enviados, protocolo (TCP, UDP, ICMP), inundar la red, intervalos de envío de paquetes, dirección IP destino, entre otros y proporcionó datos como número de paquetes transmitido, número de paquetes recibidos, tiempo de ida y vuelta, entre otros.

Una vez instaladas las herramientas necesarias para el buen funcionamiento del Switch y el controlador, se conecta el controlador POX con el Switch OpenFlow, mediante los siguientes comandos.

En el controlador se ejecutó:

sudo ifconfig eth1 192.168.100.100

sudo ./pox.py samples.pretty_log forwarding.l2_learning py

En el Switch OpenFlow se asignaron las direcciones IP a las Interfaces

ifconfig eth1 192.168.100.22

ifconfig nf2c0 192.168.0.10

ifconfig nf2c1 192.168.0.11

ifconfig nf2c2 192.168.0.12

ifconfig nf2c3 192.168.0.13

Con este comando se programó la PCI y se descargó el archivo .bit a la NetFPGA

nf_regress_test.pl --project openflow_switch

Luego se eliminaron los procesos ofdatapath y ofprotocol

killall ofdatapath

killall ofprotocol

Después se reiniciaron las interfaces de red

/etc/init.d/network restart

cd /usr/local/netfpga/openflow/udatapath

./ofdatapath --detach punix:/var/run/dp0 -d 40169FF344C0 -i nf2c0, nf2c1, nf2c2 ,nf2c3

Por último se conectó el Switch OpenFlow con el controlador de dirección IP 192.168.100.100

cd /usr/local/netfpga/openflow/secchan

./ofprotocol unix:/var/run/dp0 tcp:192.168.100.100:6633

```
an|INFO|OpenFlow reference implementation version 1.0.0
an|INFO|OpenFlow protocol version 0x01
an|WARN|new management connection will receive asynchronous messages
|INFO|unix:/var/run/dp0: connecting...
|INFO|tcp:192.168.100.100:6633: connecting...
|INFO|unix:/var/run/dp0: connected
watcher|INFO|Datapath id is 40169ff344c0
watcher|INFO|Identified data path local port as "tap0".
|INFO|tcp:192.168.100.100:6633: connection failed (Connection refuse
|WARN|tcp:192.168.100.100:6633: connection dropped (Connection refus
|INFO|tcp:192.168.100.100:6633: connecting...
|INFO|tcp:192.168.100.100:6633: connection failed (Connection refuse
|WARN|tcp:192.168.100.100:6633: connection dropped (Connection refus
|INFO|tcp:192.168.100.100:6633: waiting 2 seconds before reconnect
|INFO|tcp:192.168.100.100:6633: connecting...
|INFO|tcp:192.168.100.100:6633: connected
```

Ilustración 25 Switch OpenFlow Conectado

Una vez el Switch OpenFlow fue conectado al controlador POX, mostró el mensaje de *Connected* como se muestra en la Ilustración 25. Luego se probó la conexión entre dos host's conectados en el puerto 1 y 3 (interfaces nf2c0 y nf2c2 de la NetFPGA respectivamente) mediante el comando ping.

El Host 3 se conectó a la interfaz nf2c2, con dirección IP 192.168.0.202 y dirección MAC 44:37:E6:31:6c:9a.

Sudo ifconfig eth0 192.168.0.202

ping 192.168.0.200

```
upb1@upb1-ThinkCentre-M70e:~$ ifconfig eth0
eth0      Link encap:Ethernet  direcciónHW 44:37:e6:31:6c:74
          Direc. inet:192.168.0.202  Difus.:192.168.0.255  Más
          Dirección inet6: fe80::4637:e6ff:fe31:6c74/64 Alcanc
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Mét
          Paquetes RX:787 errores:0 perdidos:0 overruns:0 fram
          Paquetes TX:204 errores:0 perdidos:0 overruns:0 carr
          colisiones:0 long.colaTX:1000
          Bytes RX:52170 (52.1 KB)  TX bytes:22800 (22.8 KB)
          Interrupción:17

upb1@upb1-ThinkCentre-M70e:~$ ping 192.168.0.200
PING 192.168.0.200 (192.168.0.200) 56(84) bytes of data.
64 bytes from 192.168.0.200: icmp_req=1 ttl=64 time=0.303 ms
64 bytes from 192.168.0.200: icmp_req=2 ttl=64 time=0.287 ms
64 bytes from 192.168.0.200: icmp_req=3 ttl=64 time=0.289 ms
```

Ilustración 26 Ping al host1 desde el host2

El Host 1 se conectó a la interfaz nf2c0, con dirección IP 192.168.0.200 y dirección MAC 44:37:E6:31:6c:74.

Sudo ifconfig eth0 192.168.0.200

ping 192.168.0.202

```
upb2@upb2-ThinkCentre-M70e:~$ ifconfig eth0
eth0      Link encap:Ethernet  direcciónHW 44:37:e6:31:6c:9a
          Direc. inet:192.168.0.200  Difus.:192.168.0.255  Másc:255
          Dirección inet6: fe80::4637:e6ff:fe31:6c9a/64 Alcance:Enl
          ACTIVO DIFUSIÓN FUNCIONANDO MULTICAST  MTU:1500  Métrica:
          Paquetes RX:884 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:5303 errores:0 perdidos:0 overruns:0 carrier:
          colisiones:0 long.colaTX:1000
          Bytes RX:80239 (80.2 KB)  TX bytes:237667 (237.6 KB)
          Interrupción:17

upb2@upb2-ThinkCentre-M70e:~$ ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data.
64 bytes from 192.168.0.202: icmp_req=1 ttl=64 time=0.298 ms
64 bytes from 192.168.0.202: icmp_req=2 ttl=64 time=0.279 ms
64 bytes from 192.168.0.202: icmp_req=3 ttl=64 time=0.274 ms
```

Ilustración 27 Ping al host2 desde el host1

Las figuras 6 y 7 muestran la comunicación del Host 1 y el Host 3 a través del Switch OpenFlow haciendo uso de la NetFPGA. Hay que aclarar que los cables UTP usados fueron categoría 5, 5E ó 6 debido a que las interfaces de todos los Host conectados a la NetFPGA deben ser Gigabit Ethernet. A continuación, se describen las pruebas realizadas en la red SDN de la subestación eléctrica genérica.

6.3.1 Prueba 1: Envío de paquetes con priorización de paquetes

En esta prueba se enviaron paquetes de datos de dos tipos, ICMP y TCP desde el Host 3 al Host 1, priorizado únicamente los TCP's; adicionalmente se generaron otros paquetes con el fin de generar tráfico en la red. A continuación, se describe los comandos usados:

Host 3: Las siguientes líneas generan 10 paquetes ICMP cada segundo a la dirección IP destino 192.168.0.200 y a la 192.168.0.203, con el fin de generar tráfico en la red:

sudo hping3 192.168.0.200 --icmp --fast

sudo hping3 192.168.0.203 –icmp --fast

Host 3: La siguiente línea genera un paquete TCP cada milisegundo a la dirección IP destino 192.168.0.200:

sudo hping3 192.168.0.200 --tcp-timestamp -i u1000 –c 100

Host 1: La siguiente línea genera 10 paquetes UDP e ICMP cada segundo a la dirección IP destino 192.168.0.203 y a la 192.168.0.201 respectivamente, con el fin de generar tráfico en la red:

sudo hping3 192.168.0.203 --udp --fast

sudo hping3 192.168.0.201 --icmp --fast

Host 2: La siguiente línea genera 10 paquetes ICMP cada segundo a la dirección IP destino 192.168.0.203, con el fin de generar tráfico en la red:

sudo hping3 192.168.0.200 --icmp --fast

Se tomaron muestras sin priorización de paquetes y el comando *hping3* da los siguientes resultados a los paquetes TCP enviados desde el Host 3:

--- 192.168.0.200 hping statistic ---

8586 packets transmitted, 366 packets received, 96% packet loss

round-trip min/avg/max = 0.2/0.3/4.0 ms

Y para los otros tipos de paquetes los resultados fueron:

162 packets transmitted, 120 packets received, 26% packet loss

round-trip min/avg/max = 0.1/11.7/694.8 ms

Luego se priorizaron los paquetes TCP que se generaron desde el Host 3 al Host 1 y viceversa. Para esto fue necesario configurar la tabla de flujos con los siguientes comandos que se ejecutan en la consola del controlador POX:

Librerías

from pox.lib.addresses import IPAddr

from pox.lib.addresses import EthAddr

import pox.openflow.libopenflow_01 as of

El controlador pregunta el ID de Switch OpenFlow

core.openflow.connections.keys()

La respuesta del Switch OpenFlow es un ID para saber a cual Switch OpenFlow se refiere, ya que pueden haber varios Switch's conectados.

[70465916978368]

Los siguientes comandos priorizaron el flujo de paquetes TCP desde el puerto 3 hasta el puerto 1.

ofs1_msg1=of.ofp_flow_mod(command=0)

ofs1_msg1.priority=3

ofs1_msg1.match.in_port=3

ofs1_msg1.idle_timeout = 0

ofs1_msg1.hard_timeout = 0

ofs1_msg1.match.dl_type = pkt.ethernet.IP_TYPE

ofs1_msg1.match.nw_proto = pkt.ipv4.TCP_PROTOCOL

ofs1_msg1.match.nw_dst="192.168.0.200/32"

ofs1_msg1.actions.append(of.ofp_action_dl_addr.set_dst("44:37:E6:31 :6c:9a"))

ofs1_msg1.actions.append(of.ofp_action_output(port=1))

core.openflow.connections[70465916978368].send(ofs1_msg1)

Los siguientes comandos priorizaron el flujo de paquetes TCP desde el puerto 1 hasta el puerto 3.

ofs1_msg2=of.ofp_flow_mod(command=0)

ofs1_msg2.priority=3

ofs1_msg2.match.in_port=1

ofs1_msg2.idle_timeout = 0

ofs1_msg2.hard_timeout = 0

ofs1_msg2.match.dl_type=pkt.ethernet.IP_TYPE

ofs1_msg2.match.nw_proto = pkt.ipv4.TCP_PROTOCOL

ofs1_msg2.match.nw_dst="192.168.0.202/32"

ofs1_msg2.actions.append(of.ofp_action_dl_addr.set_dst("44:37:E6:31 :6c:74"))

ofs1_msg2.actions.append(of.ofp_action_output(port=3))

core.openflow.connections[70465916978368].send(ofs1_msg2)

```
POX> ofs1_msg1=of.ofp_flow_mod(command=0)
POX> ofs1_msg1.priority=3
POX> ofs1_msg1.match.in_port=3
POX> ofs1_msg1.match.dl_type=0x800
POX> ofs1_msg1.match.nw_dst="192.168.0.200/32"
POX> ofs1_msg1.actions.append(of.ofp_action_dl_addr.set_dst("44:37:E6:31:6c:9a")
POX> ofs1_msg1.actions.append(of.ofp_action_output(port=1))
POX> core.openflow.connections[70465916978368].send(ofs1_msg1)
POX>
POX> ofs1_msg2=of.ofp_flow_mod(command=0)
POX> ofs1_msg2.priority=3
POX> ofs1_msg2.match.in_port=1
POX> ofs1_msg2.match.dl_type=0x0800
POX> ofs1_msg2.match.nw_dst="192.168.0.202/32"
POX> ofs1_msg2.actions.append(of.ofp_action_dl_addr.set_dst("44:37:E6:31:6c:74")
POX> ofs1_msg2.actions.append(of.ofp_action_output(port=3))
POX> core.openflow.connections[70465916978368].send(ofs1_msg2)
```

Ilustración 28 Configuración Switch OpenFlow (Priorización de paquetes).

La configuración del Switch OpenFlow a través del controlador POX se muestra en la Ilustración 28. Pero en el Switch OpenFlow (NetFPGA) se puede observar la tabla de flujo, por medio de los comandos (Ver Ilustración 29):

cd /usr/local/netfpga/openflow/utilities

./dpctl dump-flows unix:/var/run/dp0

```
[root@localhost utilities]# ./dpctl dump-flows unix:/var/run/dp0
stats_reply (xid=0x23a8664b): flags=none type=1(flow)
  cookie=0, duration_sec=1084s, duration_nsec=771000000s, table_id=0, priority=3, n_packets=103, n_bytes=10094, idle_timeout=
0,hard_timeout=0,ip,in_port=1,nw_dst=192.168.0.202,actions=mod_dl_dst:44:37:e6:31:6c:74,output:3
  cookie=0, duration_sec=1109s, duration_nsec=512000000s, table_id=0, priority=3, n_packets=84, n_bytes=8232, idle_timeout=0,
hard_timeout=0,ip,in_port=3,nw_dst=192.168.0.200,actions=mod_dl_dst:44:37:e6:31:6c:9a,output:1
[root@localhost utilities]#
```

Ilustración 29 Configuración de la tabla de flujo del Switch OpenFlow.

La prueba da como resultado que los 100 paquetes TCP transmitido fueron recibidos por Host 1 (Ver el Debug en el Anexo 4).

--- 192.168.0.200 hping statistic ---

100 packets transmitted, 100 packets received, 0% packet loss

round-trip min/avg/max = 0.2/0.3/0.3 ms

Y para los paquetes sin priorización los resultados fueron:

357 packets transmitted, 142 packets received, 61% packet loss

round-trip min/avg/max = 0.1/47.4/168.2 ms

6.3.2 Prueba 2: Observar el comportamiento de la red en el tiempo

En esta prueba se realizó el envío de paquetes de datos de dos tipos, ICMP y TCP desde el Host 3 al Host 1, con el fin de observar el comportamiento de la red de comunicaciones con y sin priorización de paquetes críticos en el tiempo. A continuación, se describe los comandos para la generación de paquetes en el Host 3:

La siguiente línea de comandos inundó la red con paquetes ICMP a la dirección IP destino 192.168.0.200, con el fin de generar tráfico en la red:

sudo hping3 192.168.0.200 --icmp --flood

Y en otra consola se ejecutó la siguiente línea, la cual genera un paquete TCP cada 100 milisegundos a la dirección IP destino 192.168.0.200. No se realizó el envío de paquetes TCP con el mismo intervalo tiempo porque se bloquea el software *WireShark* por la cantidad de paquetes que debería procesar.

sudo hping3 192.168.0.200 --tcp-timestamp -i u100000

Luego en el Host 1 se ejecutó el software *WireShack* que capturó los paquetes enviados desde el Host 3 con el fin de ver el comportamiento de la red de comunicaciones en el tiempo.

6.4 Análisis del comportamiento de la red de comunicaciones SDN de la subestación

En esta sección se analizaron los resultados de las pruebas realizadas a la red de comunicaciones de una subestación eléctrica genérica basada en IEC 61850. La prueba 1 sin priorización de paquetes descrita en el numeral 6.3.1 proporcionó los siguientes resultados:

	Paquetes			**Tiempo de ida y vuelta de los paquetes recibidos**		
	Transmitidos	**Recibidos**	**Perdidos**	**Mínimo (ms)**	**Promedio (ms)**	**Máximo (ms)**
Paquetes críticos	8586	366	96%	0.2	0.3	4.0
Otros paquetes	162	120	26%	0.1	11.7	694.8

Tabla 4 Resultados de la prueba 1 sin priorización.

Aunque el tiempo que tardan los paquetes de mensaje críticos recibidos fue bastante bueno, en la Tabla 4 se observó que el 96% de los paquetes críticos se perdieron por el tráfico en la red de comunicaciones, lo cual no es aceptable por los requisitos de rendimiento para las funciones de misión crítica en la red de comunicaciones de una subestación eléctrica descritos en el capítulo 5 de la norma IEC 61850 (Ver Tabla 2). Adicionalmente, solo se perdieron el 26% de los paquetes que no son prioritarios para el buen funcionamiento de la subestación. A continuación, se presentan los resultados de la misma prueba pero priorizado los paquetes de mensajes críticos.

	Paquetes			Tiempo de ida y vuelta de los paquetes recibidos		
	Transmitidos	**Recibidos**	**Perdidos**	**Mínimo (ms)**	**Promedio (ms)**	**Máximo (ms)**
Paquetes críticos	100	100	0%	0.2	0.3	0.3
Otros paquetes	357	142	61%	0.1	47.4	168.2

Tabla 5 Resultados de la prueba 1 con priorización de paquetes.

Al priorizar los paquetes de mensajes de misión crítica no hubieron pérdidas de paquetes, ya que el Switch OpenFlow prioriza los paquetes configurados en la Ilustración 28, asignándole a los paquetes de misión crítica una prioridad de 3 (Ver Ilustración 29) y a los otros paquetes la prioridad de 65535. Hay que tener en cuenta que al priorizar los paquetes críticos, se generó un mayor retardo en los paquetes sin priorizar.

La captura de la prueba 2 que se describió en el numeral 6.3.2 haciendo uso del software WireShark se muestra en la Ilustración 30, donde se observó que algunos paquetes críticos tienen retardos, y al priorizar estos paquetes después de transcurridos cinco segundos, estos dejan de presentar los retardos.

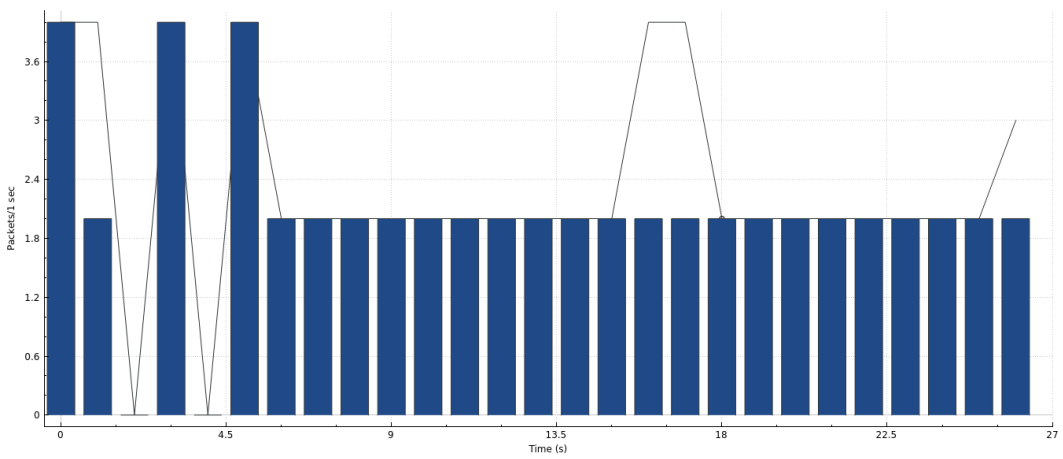

Ilustración 30 Priorización de paquetes TCP a los 5 segundos.

Luego se realizó la misma prueba pero se priorizan los paquetes en el segundo 12 (Ver la Ilustración 31), donde de igual manera hay retardos en los paquetes críticos hasta que se realiza la priorización de estos paquetes en el segundo 12.

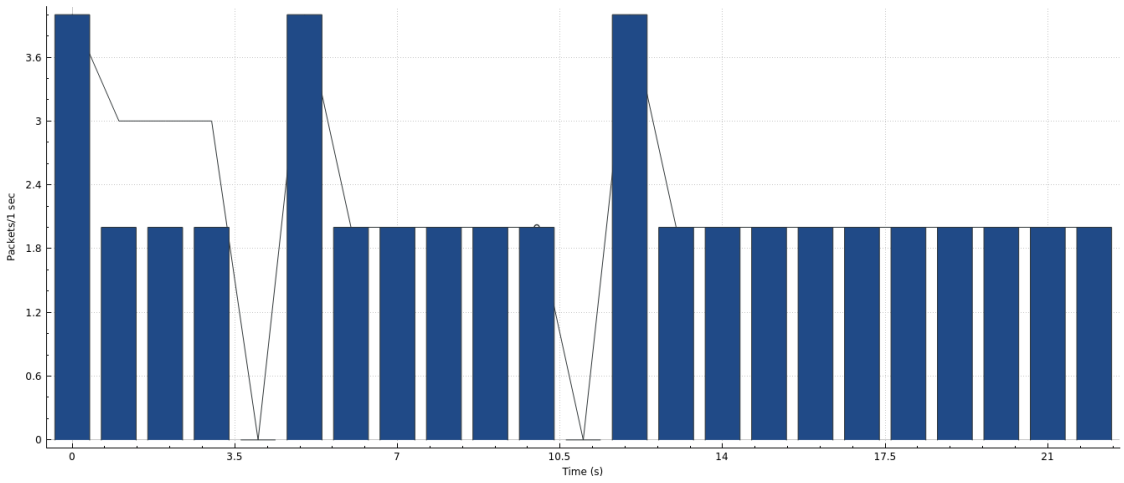

Ilustración 31 Priorización de paquetes TCP a los 12 segundos.

Como se observó, el Switch OpenFlow permitió priorizar el flujo de los paquetes de datos, teniendo como parámetros de discriminación la dirección IP fuente, dirección IP destino, el protocolo, interfaz de entrada del paquete, interfaz de salida del paquete, tipo, entre otros.

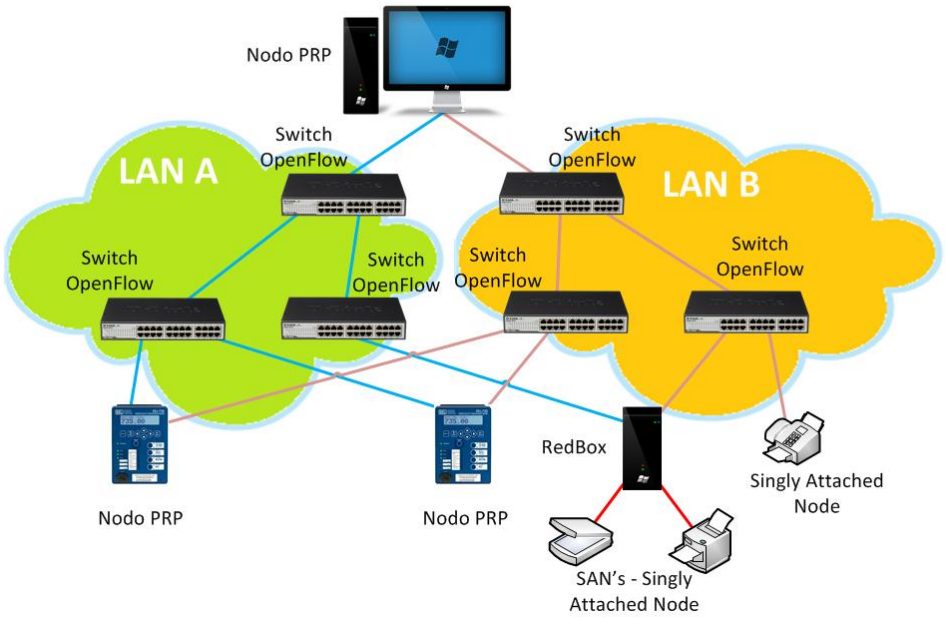

Ilustración 32 Red de comunicaciones SDN y PRP.

Hay que tener en cuenta que el protocolo Software Defined Networking no ofrece ningún beneficio en redundancia de la red de comunicaciones, como el que ofrecen los protocolos PRP y HSR, ya que el flujo de los paquetes de

datos en estos protocolos pueden ir por dos rutas independientes hacia su destino como se muestra en la Ilustración 4 e Ilustración 5. SDN presentaría la pérdida de comunicación con uno de los nodos ante una falla en uno de los enlaces de la red; por lo anterior, la arquitectura y topología de la red de comunicaciones de la subestación eléctrica genérica propuesta en este proyecto debería hacer uso del protocolo PRP para generarle redundancia a la red como se muestra en la Ilustración 32. Esta red de comunicaciones está compuesta por dos redes LAN independientes haciendo uso del protocolo PRP y el control de flujo de datos de la red de comunicaciones se haría mediante el protocolo SDN.

7 CONCLUSIONES

Se realizó un seguimiento a las actividades y al trabajo de la Universidad de Stanford, ya en esta se desarrollaron las librerías y componentes del controlador POX y el Switch OpenFlow que se utilizaron para el desarrollo de este proyecto. Además, esta universidad se encuentra trabajando para el desarrollo y la divulgación de SDN.

Se realizaron las configuraciones necesarias en el sistema operativo CentOS 5.11 para el correcto funcionamiento del Switch sobre la plataforma NetFPGA, basándose en los aspectos fundamentales de OpenFlow.

SDN proporciona a la red de comunicaciones de la subestación IEC 61850 gestión del tráfico en la red, disponibilidad, segmentación de flujo, robustez y un menor tiempo de respuesta en las funciones de misión crítica de la subestación eléctrica.

SDN al tener la inteligencia lógicamente centralizada, permite tener una visión global de la red de comunicaciones de la subestación eléctrica y de los cambios en la demanda de la infraestructura, lo que hace posible el desarrollo de diferentes aplicaciones. Además, SDN simplifica el diseño de la red de comunicaciones de la subestación; lo que permite programar la red de comunicaciones sin interrumpir el tráfico.

EL Switch OpenFlow permite parametrizar el flujo de datos de la red de comunicaciones con el fin de asignar prioridades a los paquetes de mensajes críticos como son los disparos del interruptor, valores muestreados, entre otros.

Una ventaja de utilizar OpenFlow en la red de comunicación de las subestaciones eléctricas es que los flujos de control y protecciones pueden ser definidos de antemano, y en caso de una falla, el sistema se comporta de una manera esperada.

La traducción de los archivos SCL daría al controlador un conocimiento a priori de las conexiones entre IED’s y la conexión de estos con los Switch de acceso y distribución.

Para que el rendimiento del Switch OpenFlow no sea degradado, ya que la red requiere un procesamiento adicional que consume recursos de esta, como CPU, memoria y almacenamiento; es necesaria una mayor robustez en

el hardware del Switch Por lo anterior, el Switch OpenFlow tendrá un mayor valor comercial que un Switch tradicional.

En general, las principales limitaciones y problemas encontrados en el desarrollo provenían de las continuas actualizaciones de los paquetes del sistema operativo CentOS 5.11; adicionalmente, la plataforma NetFPGA utilizada en este proyecto se encuentra descontinuada y sus librerías son difíciles de conseguir en la Internet. Se recomienda la elaboración de un ISO personalizado del sistema operativo CentOS 5.11 con las herramientas descritas en este proyecto ya pre-configuradas.

8 TRABAJOS FUTUROS

Implementación de Switch’s OpenFlow en una bahía de una subestación eléctrica, con el fin de ver cómo se comporta el tráfico en la red de una subestación real.

Implementar una red SDN con las cuatro NetFPGA que posee la Universidad Pontificia Bolivariana e inundar el tráfico de la red, para ver el comportamiento de los paquetes priorizados.

En este proyecto se configuró el controlador POX, este es específicamente para desarrollos en Software Defined Networking. Se podría configurar otro tipo de controlador, como NOX, que fue desarrollado por la Universidad de Stanford para redes OpenFlow.

La NetFPGA es una plataforma que permite desarrollar varios proyectos de pregrado y postgrado, como son Router’s, proyectos de IPv4/IPv6, generador de paquetes, Switch con soporte de tiempo real, túneles, entre otros.

9 REFERENCIAS

[1] XILINX, «Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet,» pp. 1 - 11, 2011.

[2] C. A. C. PARDO, «IMPLEMENTACIÓN DE UN OPENFLOW CONTROLLER PARA EL MANEJO DE OPENFLOW SWITCHES,» PONTIFICIA UNIVERSIDAD JAVERIANA, Bogotá, 2014.

[3] R. Bobba, «Software-Defined Networking Addresses Control System Requirements,» pp. 1 - 9, 2014.

[4] International Electrotechnical Commission (IEC), «Communications networks and systems in substations,» de *INTERNATIONAL STANDARD IEC 61850 Edición 1*, 2003, pp. 26 - 33.

[5] D. Dolezilek, «Using Information From Relays to Improve the Power System – Revisited,» *Schweitzer Engineering Laboratories,* pp. 1 - 2, 2010.

[6] M. Goraj, «Migration paths for IEC 61850 substation communication networks towards superb redundancy based on hybrid PRP and HSR topologies,» p. Pag 3, 2013.

[7] B. Lee, «Role-based access control for substation automation systems using XACML,» *El Sevier,* p. 2, 2015.

[8] J. Chang, «Protection and Control System Upgrade Based on IEC- 61850 and PRP,» p. Pag 1, 2014.

[9] B. L. ,. J. T. J. Mo, «Dynamic Simulation and Test of IEC 61850-9-2 Process Bus Applications,» p. Pag 2, 2012.

[10] IEC-62439-3, «Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR),» 2010.

[11] G. G. R. A. F. Zurita, «Arquitecturas de red LAN para la automatización de subestaciones, basadas en la norma IEC 61850 (RSTP, PRP y HSR),» pp. 4 - 7, 2014.

[12] N. Liu, «Reliability evaluation of a substation automation system communication network based on IEC 61850,» *IEEE,* 2014.

[13] J. Á. Araujo, «PRP and HSR for High Availability Networks in Power Utility Automation: A

Method for Redundant Frames Discarding,» *IEEE,* p. 2, 2014.

[14] P. F. A. F. S. R. C. M. De Dominicis, «On the use of IEEE1588 in Existing IEC61850 based SASs: Current Behavior and Future Challenges,» p. Pag 6, 2010.

[15] N. M. K. Chowdhury, «A survey of network virtualization,» *El Sevier,* Vols. %1 de %22 - 4, 2010.

[16] A. Sydney, «Using GENI for experimental evaluation of Software Defined Networking in smart grids,» *El Sevier,* pp. 7 - 9, 2014.

[17] M. Kobayashi, «Maturing of OpenFlow and Software-defined Networking through deployments,» *El Sevier,* pp. 2 - 5, 2014.

[18] H. L. H. Farhady, «Software-Defined Networking: A survey,» *El Sevier,* pp. 2 - 5, 2015.

[19] Ministerio de Minas y Energía, «Reglamento Técnico de Instalaciones Eléctricas – RETIE.,» 2009.

[20] LA COMISION DE REGULACION DE ENERGIA Y GAS, «Leyes 142 y 143 de 1994 - Codigo de redes,» Colombia, 1994.

[21] A. Astarloa, «FPGA Implemented Cut-Through vs Store-and-Forward Switches for Reliable Ethernet Networks,» *IEEE,* 2014.

[22] M. Suñé, «Design and implementation of the OFELIA FP7 facility: The European OpenFlow testbed,» *El Sevier,* pp. 2 - 5, 2014.

[23] T. Pfeiffenberger, «Evaluation of Software-Defined Networking for Power Systems,» *IEEE,* 2014.

[24] E. J. J. M. N. M. A. A. Elias Molina, «Using Software Defined Networking to manage and control IEC 61850-based systems,» *ELSEVIER,* pp. 1 - 13, 2014.

[25] SCHWEITZER ENGINEERING LABORATORIES, INC, «SOFTWARE-DEFINED NETWORK SOLUTIONS FOR CRITICAL INFRASTRUCTURE,» 25 10 2014. [En línea]. Available: https://www.selinc.com/WorkArea/DownloadAsset.aspx?id=106309.

[26] T. Liu, «Implementing Open flow switch using FPGA based platform,» pp. 5 - 20, 2014.

[27] M. Kobayashi, «Maturing of OpenFlow and Software-defined Networking through deployments,» *El Sevier,* pp. 2 - 5, 2014.

[28] G. CRESTANI, «PROBLEMATICHE DI SICUREZZA NELLE SOFTWARE DEFINED NETWORKS,» UNIVERSIT`A DI BOLOGNA, Bolonia, 2013 - 2014.

[29] Universidad Stanford, «POX Wiki,» 05 03 2015. [En línea]. Available: https://openflow.stanford.edu/display/ONL/POX+Wiki. [Último acceso: 10 11 2015].

[30] A. N. Ιεροδιακόνου, «ΑΝΑΠΤΥΞΗ APPLICATION AWARE CONTROLLER ΣΕ ΠΕΡΙΒΑΛΛΟΝ OPENFLOW,» ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ, Atenas, 2013.

[31] Y.-K. Lai, "Real-time detection of changes in network with OpenFlow based on NetFPGA implementation," *El Sevier,* p. 3, 2014.

[32] S. Sharma, «OpenFlow: Meeting carrier-grade recovery requirements,» *ELSevier,* p. 2, 2013.

[33] A. Hakiri, "Software-Defined Networking: Challenges and research opportunities for Future Internet," *El Sevier,* pp. 2 - 3, 2014.

[34] I. Ryoo, «Information exchange architecture based on software defined networking for cooperative intelligent transportation systems,» *Spinger,* 2015.

[35] CentOS, «Sistema Operativo CentOS,» [En línea]. Available: https://www.centos.org/about/. [Último acceso: 26 10 2015].

[36] C. A. Vargas, «DESARROLLO DE UN SMART RADIO PROTOTIPO BASADO EN EL CONCEPTO COGNITIVE RADIO UTILIZANDO DISPOSITIVOS FPGA,» Medellìn, 2008.

[37] M. Happe, «Bringing Programmability to the Hardware: Field-Programmable Gate Arrays (FPGAs),» p. 20, 2015.

[38] D. C. H., «Design and Implementation of High-Performance FPGA Signal Processing Datapaths for Software Defined Radios,» p. 29, 2003.

[39] Xilinx , «Xilinx University Program Virtex-II Pro Development System - Hardware Reference Manual,» nº UG069 (v1.0), pp. 1 - 138, 2005.

[40] G. Gibb, «Obtain Hardware And Software - Purchase a Dell 2950,» 14 Enero 2013. [En línea]. Available: https://github.com/NetFPGA/netfpga/wiki/ObtainHardwareAndSoftware. [Último acceso: 20 Octubre 2015].

[41] G. G. R. A. F. Zurita, «Arquitecturas de red LAN para la automatización de

subestaciones, basadas en la norma IEC 61850 (RSTP, PRP y HSR),» pp. Pag 4 - 7, 2014.

ANEXO 1

INSTALACIÓN CONTROLADOR POX

Para la descarga e instalación del controlador POX es necesario ejecutar las siguientes líneas de comandos en la consola de Ubuntu:

git clone https://github.com/noxrepo/pox.git
cd pox
git checkout dart

ANEXO 2

INSTALACIÓN Y CONFIGURACIÓN DE LOS DRIVER DE LA NETFPGA

Luego de haber instalado la NetFPGA en el slot PCI del computador y se haya instalado el sistema operativo Centos 5.11 es conveniente realizar una actualización de los repositorios *yum* mediante el siguiente comando:

yum update -y

A continuación, se procede a la instalación de los driver's de la NetFPGA:

uname -r

yum -y install gcc

yum -y install ncurses-devel

yum -y install libnet

yum -y install compat-libstdc++-296.i386

yum -y install libpcap-devel

rpm -i perl-Net-Pcap-0.16-1.el5.rf.i386.rpm

rpm -i perl-Error-0.17019-1.el5.rf.noarch.rpm

rpm -i perl-Net-RawIP-0.25-1.el5.rf.i386.rpm

yum -y install perl-XML-Simple

rpm -qa | grep -i kernel

*yum -y install kernel-devel**

*yum -y install kernel-PAE**

rpm -e kernel-$(uname -r)

Reboot

rpm -Uhv http://netfpga.org/yum/el5/RPMS/noarch/netfpga-repo-1-1_CentOS5.noarch.rpm

yum install netfpga-base

reboot

Luego de reiniciar el PC donde se encuentra instalada la NetFPGA, se hace la verificación de que el driver esté bien instalado con los siguientes comandos:

/sbin/modprobe nf2

lsmod |grep nf2

 # Muestra sí reconoce las interfaces de la NetFPGA nf2cX.

ifconfig –a |grep nf2

ANEXO 3

CONFIGURACIÓN DEL SWITCH OPENFLOW

Existen varias pruebas de verificación del correcto funcionamiento de los driver's como son SelfTest y la prueba de Regresión. Una vez verificado el correcto funcionamiento de la NetFPGA, se instala el autoconf, con la siguiente serie de comandos que descargan y compilan las librerías de este paquete:

sudo yum -y install git automake pkgconfig libtool gcc

wget http://ftp.gnu.org/gnu/autoconf/autoconf-2.63.tar.gz

tar xvzf autoconf-2.63.tar.gz

cd autoconf-2.63

./configure --prefix=/usr

make

sudo make install

La instalación del *autoconf* es requerida en la versión 2.6 o mayor para poder compilar las librerías del Switch OpenFlow, los comandos para la instalación del Switch se enuncian a continuación:

 # Instala las librerías del Switch OpenFlow

yum install netfpga-openflow_switch

 # Descarga la carpeta del Switch OpenFlow

git clone git://gitosis.stanford.edu/openflow.git

cd openflow

git checkout -b 1.0.0-netfpga origin/devel/tyabe/1.0.0-netfpga

./boot.sh

cd hw-lib/nf2

wget http://openflow.org/downloads/netfpga/openflow_switch.bit.100_3.tar.gz

tar xfvz openflow_switch.bit.100_3.tar.gz

cd ../../

./configure --enable-hw-lib=nf2

make

sudo make install

gedit regress/bin/nf2.map &

cp netfpga/projects/openflow/regress/scripts/env_vars .

export OF_ROOT=/root/netfpga/projects/openflow/

export PERL5LIB=${OFT_ROOT}/lib/Perl5:${PERL5LIB}

export PERL5LIB=${OFT_ROOT}/lib/Perl5

sudo /sbin/chkconfig avahi-daemon off

sudo /etc/rc.d/init.d/avahi-daemon stop

sudo /sbin/chkconfig ip6tables off

echo 'options ipv6 disable=1' > /etc/modprobe.d/disable-ipv6.conf

gedit /etc/sysconfig/network

 # Crear un enlace simbólico entre las dos carpetas

ln -s /usr/local/netfpga/projects/openflow_switch/ /root/netfpga/projects/openflow_switch

 # Hay que cambiar el nombre de eth0 por eth2

gedit /etc/sysconfig/network-scripts

ANEXO 4

DEBUG PRUEBA 1: ENVIO DE PAQUETES CON PRIORIZACIÓN

```
upb1@upb1-ThinkCentre-M70e:~$ sudo hping3 192.168.0.200 --tcp-
timestamp -i u1000 -c 100
HPING 192.168.0.200 (eth0 192.168.0.200): NO FLAGS are set, 40
headers + 0 data bytes
len=46 ip=192.168.0.200 ttl=64 DF id=27614 sport=0 flags=RA
seq=0 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27615 sport=0 flags=RA
seq=1 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27616 sport=0 flags=RA
seq=2 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27617 sport=0 flags=RA
seq=3 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27618 sport=0 flags=RA
seq=4 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27619 sport=0 flags=RA
seq=5 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27620 sport=0 flags=RA
seq=6 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27621 sport=0 flags=RA
seq=7 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27622 sport=0 flags=RA
seq=8 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27623 sport=0 flags=RA
seq=9 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27624 sport=0 flags=RA
seq=10 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27625 sport=0 flags=RA
seq=11 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27626 sport=0 flags=RA
seq=12 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27627 sport=0 flags=RA
seq=13 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27628 sport=0 flags=RA
seq=14 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27629 sport=0 flags=RA
seq=15 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27630 sport=0 flags=RA
seq=16 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27631 sport=0 flags=RA
seq=17 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27632 sport=0 flags=RA
seq=18 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27633 sport=0 flags=RA
seq=19 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27634 sport=0 flags=RA
seq=20 win=0 rtt=0.2 ms
```

```
len=46 ip=192.168.0.200 ttl=64 DF id=27635 sport=0 flags=RA
seq=21 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27636 sport=0 flags=RA
seq=22 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27637 sport=0 flags=RA
seq=23 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27638 sport=0 flags=RA
seq=24 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27639 sport=0 flags=RA
seq=25 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27640 sport=0 flags=RA
seq=26 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27641 sport=0 flags=RA
seq=27 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27642 sport=0 flags=RA
seq=28 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27643 sport=0 flags=RA
seq=29 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27644 sport=0 flags=RA
seq=30 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27645 sport=0 flags=RA
seq=31 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27646 sport=0 flags=RA
seq=32 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27647 sport=0 flags=RA
seq=33 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27648 sport=0 flags=RA
seq=34 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27649 sport=0 flags=RA
seq=35 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27650 sport=0 flags=RA
seq=36 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27651 sport=0 flags=RA
seq=37 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27652 sport=0 flags=RA
seq=38 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27653 sport=0 flags=RA
seq=39 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27654 sport=0 flags=RA
seq=40 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27655 sport=0 flags=RA
seq=41 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27656 sport=0 flags=RA
seq=42 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27657 sport=0 flags=RA
seq=43 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27658 sport=0 flags=RA
seq=44 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27659 sport=0 flags=RA
seq=45 win=0 rtt=0.2 ms
```

```
len=46 ip=192.168.0.200 ttl=64 DF id=27660 sport=0 flags=RA
seq=46 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27661 sport=0 flags=RA
seq=47 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27662 sport=0 flags=RA
seq=48 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27663 sport=0 flags=RA
seq=49 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27664 sport=0 flags=RA
seq=50 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27665 sport=0 flags=RA
seq=51 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27666 sport=0 flags=RA
seq=52 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27667 sport=0 flags=RA
seq=53 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27668 sport=0 flags=RA
seq=54 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27669 sport=0 flags=RA
seq=55 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27670 sport=0 flags=RA
seq=56 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27671 sport=0 flags=RA
seq=57 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27672 sport=0 flags=RA
seq=58 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27673 sport=0 flags=RA
seq=59 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27674 sport=0 flags=RA
seq=60 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27675 sport=0 flags=RA
seq=61 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27676 sport=0 flags=RA
seq=62 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27677 sport=0 flags=RA
seq=63 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27678 sport=0 flags=RA
seq=64 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27679 sport=0 flags=RA
seq=65 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27680 sport=0 flags=RA
seq=66 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27681 sport=0 flags=RA
seq=67 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27682 sport=0 flags=RA
seq=68 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27683 sport=0 flags=RA
seq=69 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27684 sport=0 flags=RA
seq=70 win=0 rtt=0.2 ms
```

```
len=46 ip=192.168.0.200 ttl=64 DF id=27685 sport=0 flags=RA
seq=71 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27686 sport=0 flags=RA
seq=72 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27687 sport=0 flags=RA
seq=73 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27688 sport=0 flags=RA
seq=74 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27689 sport=0 flags=RA
seq=75 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27690 sport=0 flags=RA
seq=76 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27691 sport=0 flags=RA
seq=77 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27692 sport=0 flags=RA
seq=78 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27693 sport=0 flags=RA
seq=79 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27694 sport=0 flags=RA
seq=80 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27695 sport=0 flags=RA
seq=81 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27696 sport=0 flags=RA
seq=82 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27697 sport=0 flags=RA
seq=83 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27698 sport=0 flags=RA
seq=84 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27699 sport=0 flags=RA
seq=85 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27700 sport=0 flags=RA
seq=86 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27701 sport=0 flags=RA
seq=87 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27702 sport=0 flags=RA
seq=88 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27703 sport=0 flags=RA
seq=89 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27704 sport=0 flags=RA
seq=90 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27705 sport=0 flags=RA
seq=91 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27706 sport=0 flags=RA
seq=92 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27707 sport=0 flags=RA
seq=93 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27708 sport=0 flags=RA
seq=94 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27709 sport=0 flags=RA
seq=95 win=0 rtt=0.3 ms
```

```
len=46 ip=192.168.0.200 ttl=64 DF id=27710 sport=0 flags=RA
seq=96 win=0 rtt=0.2 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27711 sport=0 flags=RA
seq=97 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27712 sport=0 flags=RA
seq=98 win=0 rtt=0.3 ms
len=46 ip=192.168.0.200 ttl=64 DF id=27713 sport=0 flags=RA
seq=99 win=0 rtt=0.2 ms

--- 192.168.0.200 hping statistic ---
100 packets transmitted, 100 packets received, 0% packet loss
round-trip min/avg/max = 0.2/0.3/0.3 ms
```