Diseño e implementación de una aplicación de cifrado de tráfico para una red móvil usando el algoritmo AES-GCM

F. E. Hurtado y L. C. Trujillo.

***Resumen*— En sistemas de telefonía móvil, la red de transporte o Mobile Backhaul es fundamental en la arquitectura de la red para el proveedor de servicios de internet, ya que esta red actúa como enlace entre la red de acceso de radio RAN (Radio Access Network) y la red troncal. Usualmente las redes de transporte que emplean fibra óptica se consideran seguras por la dificultad en la interceptación de datos, en muchos casos las redes no son de su propiedad, son gestionadas por un proveedor externo o pueden hacer parte de internet, lo cual genera riesgos de seguridad por el acceso público en dichas redes. En este artículo se formula una optimización en el uso de recursos de hardware y software en los procesos de operación de la red, empleando de una manera más eficiente la capacidad computacional disponible para el proceso de cifrado de datos en la transmisión dentro del ambiente de red de transporte móvil.**

***Palabras clave*— Telefonía móvil, red de transporte, operación de red, optimización de recursos, DPDK, IPSec, AES-GCM**

I. Introducción

En el ambiente de red móvil celular, la red de transporte o *Mobile Backhaul* [1] juega un rol vital dentro de la arquitectura de la red de un operador de servicios de internet [2], ya que actúa como enlace entre la red de acceso de radio RAN (*Radio Access Network*) y la red troncal. Esto significa que a través de esta red se transportan los datos y la voz del usuario hacia internet, otras redes móviles y la red telefónica [3]. A su vez, la rápida evolución del mercado de las telecomunicaciones ha enfrentado a los operadores de redes móviles con un aumento en la demanda de ancho de banda, debido a la proliferación de servicios de convergentes de voz y datos, y aplicaciones cada vez más exigentes en recursos de hardware y software. Como consecuencia, los operadores se ven obligados a reducir significativamente los costos de operación (*Operational Expenditure*, OPeX) para compensar el declive en la ganancia por usuario (*Average Revenue Per User*, ARPU) [4] debido a la competencia. Para esto, se aprovecha la aparición de nuevas tecnologías, como 4G/LTE (*Long Term Evolution*) y MPLS (*MultiProtocol Label Switching*), que optimizan el ancho de banda y la eficiencia en la transmisión. Sin embargo, al adoptar estas nuevas tecnologías en su red, se vuelven sensibles aspectos como la confidencialidad o la integridad de la información, de tal modo que deben considerarse medidas de seguridad adicionales.

Aunque usualmente las redes de transporte que emplean fibra óptica se consideran seguras por la dificultad en la interceptación de datos [5] o en algunos escenarios por estar en premisas privadas del operador, en muchos otros no son de su propiedad o son gestionadas por un proveedor externo, o inclusive pueden hacer parte de internet, lo cual genera riesgos de seguridad por el acceso público en dichas redes.

Como se muestra en la Figura 1, en las redes de tercera generación (3G) el tráfico de voz y datos está cifrado hasta el controlador de radio (Radio Network Controller, RNC), y mientras transita por la red de transporte no puede ser manipulado. Por otra parte, la red troncal se asume segura por estar bajo control del operador. No obstante, el RNC es vulnerable a ataques, de modo que el tráfico de control puede ser maliciosamente manipulado [4].

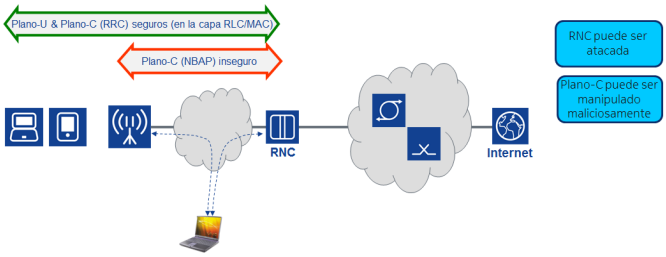


Fig. 1. Red de transporte 3G [4].

Por otra parte, la seguridad de datos y voz en las redes de cuarta y quinta generación depende de cuán segura es la red de transporte, que en la mayoría de los casos es prestada por un proveedor externo [5]. Además, como se observa en la Figura 2, en la arquitectura de red LTE los servicios están basados en IP (*Internet Protocol*) desde el usuario hasta internet [6], así que la información puede ser interceptada y analizada en cualquier punto de la red de transporte. Por este motivo, en el diseño de nuevas soluciones se incluyen esquemas de cifrado entre las redes de acceso de radio RAN (Radio Access Network) e IP usando el protocolo IPSec (*Internet Protocol Security*) [7] para hacer más seguras las comunicaciones y ofrecer así mejores niveles de confidencialidad e integridad en la información.

Este proceso de cifrado con IPSec agrega una carga de procesamiento en los dispositivos de red usados para este fin

F. E. Hurtado es estudiante de la maestría en Ingeniería Electrónica de la Universidad Javeriana (correos e.: f.hurtado@javeriana.edu.co).

L. C. Trujillo es profesor del departamento de Ingeniería Electrónica de la Universidad Javeriana, Bogotá, Colombia. (correo e. trujillo.luis@javeriana.edu.co).

(usualmente firewalls y terminadores de VPN - Virtual Private Network con IPSec) [8]–[10] y causa retardos (latencia) en la transmisión que pueden llegar a afectar las comunicaciones en tiempo real —como voz, video o telepresencia—. Generalmente el impacto del proceso de cifrado de datos en el desempeño de la red depende del tamaño y grado de complejidad de dichas redes en cada operador, además de las diferentes arquitecturas de hardware y software y más puntualmente los algoritmos criptográficos que implementan los distintos fabricantes de equipos de red.

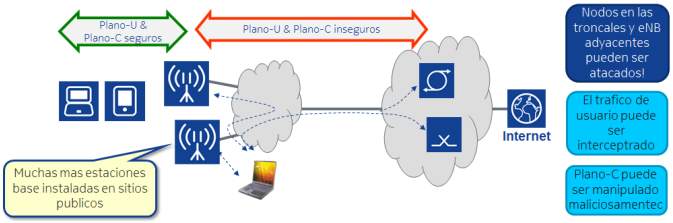


Fig. 2. Red de transporte 4G LTE [4].

Este proceso de cifrado con IPSec agrega una carga de procesamiento en los dispositivos de red usados para este fin y causa latencia en la transmisión que pueden llegar a afectar las comunicaciones en tiempo real. Además, cada fabricante de equipos de red trabaja en sus propios desarrollos tanto en hardware como en software para reducir el costo computacional haciendo más eficientes los procesos, algoritmos y arquitecturas en sus plataformas, con el reto presente de mantener la interoperabilidad. Estas mejoras se dan también en los criterios que requiere el mercado de las telecomunicaciones, como el nivel de desempeño de las plataformas en términos de cantidad de conexiones concurrentes que puede soportar y el volumen de tráfico que puede procesar la plataforma [8]–[10].

Para este fin, los fabricantes de equipos de red emplean dentro de sus plataformas los algoritmos que define la industria para cifrado, autenticación y validación de la integridad de la información según mejores prácticas y recomendaciones de entidades normativas como el NIST (*National Institute of Standards and Technology*) en Estados Unidos. Sin embargo, a medida que la capacidad de procesamiento de los computadores actuales crece, las especificaciones de seguridad de los algoritmos también se incrementan, ya que algoritmos complejos son menos susceptibles de ser decodificados. Por esta razón, el NIST recomienda, en su boletín SP 800-131A, el uso de algoritmos de nueva generación, dentro de los cuales se cuentan AES y GCM [11], donde se describe a los algoritmos de cifrado más recientes como la nueva visión de la criptografía, al ofrecer los últimos avances en tecnología ajustándose a las tendencias de la industria de forma segura, escalable e interoperable.

De igual forma, un borrador de estándar de la IETF (*Internet Engineering Task Force*) [12] propuso recientemente una actualización de los requerimientos de implementación de los algoritmos criptográficos para los protocolos ESP (*Encapsulation Security Payload*) y AH (*Authentication Header*) que hacen parte del protocolo IPSec, e incluyó una guía de mejores prácticas para el uso de estos algoritmos, en la cual se recomienda y enfatiza el uso y la importancia de AES (*Advanced Encryption Standard*) y GCM (*Galois Counter Mode*) en IPSec [13]; este conjunto de mejoras en el desempeño de los procesos de cifrado/descifrado es relevante en el ambiente de red móvil por sus características estrictas en cuanto a requerimientos de capacidad y velocidad de procesamiento para tráfico de aplicaciones en tiempo real, así como también cubre de forma eficiente los requerimientos de seguridad gracias al cifrado de datos.

En este artículo se formula una optimización en el uso de recursos de hardware y software en los procesos de operación de la red, empleando de una manera más eficiente la capacidad computacional disponible para el proceso de cifrado de datos en la transmisión dentro del ambiente de red de transporte móvil. Con este fin, se diseñó e implementó una aplicación de cifrado/descifrado de datos usando la combinación del algoritmo de cifrado AES en la pila de protocolos IPSec con el algoritmo de autenticación en bloques de llave simétrica GCM aprovechando los desarrollos de Intel® tanto en la arquitectura del procesador con el set de instrucciones AES-NI [14], como en software con la optimización de la implementación del algoritmo, para dar mayor capacidad y velocidad de transmisión de datos haciendo más eficiente el proceso de cifrado. La aplicación está desarrollada basada en la herramienta de software DPDK (*Data Plane Development Kit*) de Intel® [15], [16]. Para ello se emplean procesadores e interfaces de red con tecnología de Intel® como se referencia en [16], de manera que el operador de red móvil perciba una mejora en el desempeño de la red desde el punto de vista de experiencia del usuario al momento de implementar cifrado para protección y confidencialidad de los datos en comparación con otros esquemas usados en la actualidad, reduciendo costos de operación y mantenimiento.

II. Especificaciones del Software

El problema planteado es la implementación de cifrado en una red móvil de manera eficiente y sin afectar la experiencia del usuario final; para esto se definen unos lineamientos detallados en los objetivos del proyecto y se plantea una solución al problema inicial que deben estar alineada con los criterios bajo los cuales esta solución se debe comportar en un marco aceptable dentro del ambiente de este trabajo, que en este caso es la red del laboratorio de la universidad y el hardware disponible según la Tabla I.

TABLA I
Características Servidor

Ítem	Descripción
Plataforma	DELL PowerEdge R410
Chipset	2x Intel® Xeon® Processor L5640 *Westmere* 12M Cache, 2.26 GHz, 5.86 GT/s Intel® QPI https://ark.intel.com/products/47926/Intel-Xeon Processor-L5640-12M-Cache-2_26-GHz-5_86-GTs-Intel-QPI Number of cores 12, Number of threads 24.
Memoria	Total 32768 MBs over 8 channels @ 1333 MHz
PCIe	1 PCIe G2 slot + 1 storage slot
Network	2x 1Ge Broadcom + 2x Intel® 8255 Ethernet interfaces

Este marco de referencia está dado por una serie de normas usadas en la industria para garantizar interoperabilidad entre sistemas, fabricantes y plataformas, así como por los objetivos específicos definidos como alcance de este proyecto, con miras de ser una referencia para la industria criptográfica, tanto como para fabricantes de equipos de seguridad en redes como para operadores que requieran una solución de este tipo.

Para el diseño y la construcción del software se tiene en cuenta los procesos para definir la arquitectura, los procesos y componentes del software a ser desarrollado, así como el ambiente de desarrollo. Se cuenta con la plataforma de desarrollo de software DPDK, la cual ofrece herramientas de optimización de aplicaciones para procesamiento de datos en redes de comunicaciones, por lo cual se tienen condiciones iniciales que deben ser contempladas al momento de diseñar la aplicación solución. El diseño de software considera un proceso con dos etapas [17], tal como se presenta a continuación.

A. Diseño de Arquitectura

Comprende los pasos estructurales que se deben seguir y los módulos de software que se deben incluir para que la aplicación cumpla los objetivos definidos. De igual forma, se observan los parámetros, reglas y condiciones que impone la herramienta de desarrollo, en este caso DPDK. Dentro de la estructura de DPDK que es compatible con lenguaje C, existen librerías con módulos pre-configurados para funciones en capas bajas del modelo OSI (capa 2 y 3) tales como conmutación, enrutamiento de paquetes y verificación de tramas Ethernet entre otras, de manera que permite desarrollar la aplicación directamente sobre capas superiores. La secuencia de inicialización de la aplicación se define de acuerdo con los requerimientos definidos, así como con las condiciones de la herramienta de desarrollo DPDK, y se ejecuta a través de los siguientes pasos:

- Inicialización de controladores de hardware mediante variables EAL.
- Inicialización interface virtual para cifrado *cryptodev*.
- Configuración parámetros y algoritmos de cifrado.
- Inicialización de tabla de enrutamiento con LPM (*Longest Prefix Match*).
- Asociación de búferes de transmisión/recepción con procesadores.
- Inicialización de interfaces lógicas/físicas.
- Inicialización de la aplicación.

En la Figura 3 se presenta la secuencia de inicialización de la aplicación.

El diseño detallado describe el comportamiento de cada uno de los módulos que componen la solución.

La aplicación debe funcionar en dos escenarios, cifrando datos y descifrando datos. Para el escenario de cifrado, es decir en el sentido *outbound* se reciben los paquetes IP desde el generador *pktgen* en texto plano (sin cifrado) en la interface de entrada donde se leen los paquetes, se validan contra una lista de control de Acceso ACL (*Access Control List*) que decide que paquetes hacen parte del dominio de cifrado según su dirección IP origen y destino.

La ACL define que paquetes deben ser encapsulados dentro del túnel usando IPSec ESP. En este punto se cifran los datos usando los algoritmos definidos en la configuración de ESP (SA y SPI), y luego se hace una búsqueda de la interface de salida en la tabla de enrutamiento usando LPM de acuerdo con la IP destino, luego de esta decisión se envía el paquete a escritura a la interface de salida correcta.

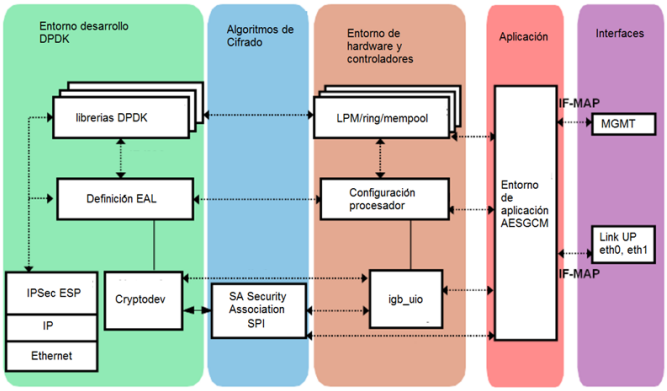


Fig. 3. Entorno de aplicación AESGCM.

En caso de no encontrar una coincidencia en la ACL el paquete puede transmitirse en texto plano, o si no hay una coincidencia en la tabla de enrutamiento el paquete puede descartarse. La Figura 4 muestra el diagrama de flujo del proceso de cifrado

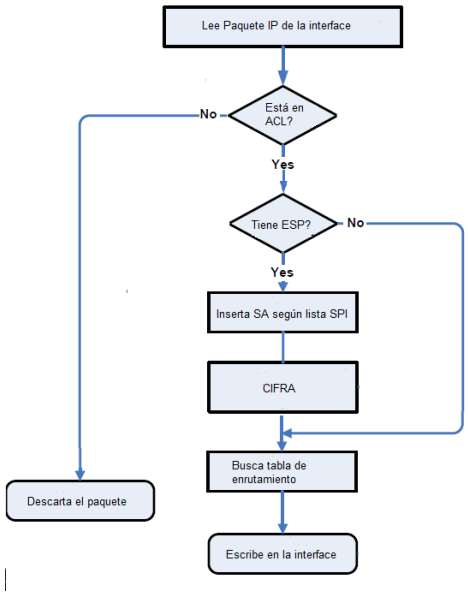


Fig. 4. Diagrama de flujo Cifrado.

Para el escenario de descifrado, es decir en el sentido *inbound* se reciben los paquetes IP cifrados con IPSec ESP en la interface de entrada donde se leen los paquetes y se define que paquetes van a ser des-encapsulados si contienen en la cabecera IP el campo de ESP. Luego se compara la información que contiene el campo ESP con respecto de los algoritmos de cifrado para comprobar la información en los SA y SPI. Si no hay coincidencia los paquetes se descartan. Si el paquete no tiene ESP en la cabecera IP pasan directamente en texto plano. Los paquetes con información

válida en los SA y SPI se descifra según los algoritmos de descifrado en la configuración y luego se hace una búsqueda de la interface de salida en la tabla de enrutamiento usando LPM de acuerdo con la IP destino, la ACL define que paquetes van a ser enviados a la tabla de enrutamiento, luego de esta decisión se envía el paquete a escritura a la interface de salida correcta. En caso de no encontrar una coincidencia en la tabla de enrutamiento o en la ACL el paquete puede descartarse.

Como referencia, se usan los siguientes módulos de DPDK [40]:

• esp.h: se encarga de encapsular los paquetes IP usando ESP

• rt.h: para crear la tabla de enrutamiento en IPv4 usando la función LPM

• ipsec.h: se encarga del cifrado de paquetes IP con el protocolo IPSec y de llamar la librería ESP para la creación del túnel

• parser.h: se encarga de validar la integridad y sanidad de los paquetes IP

• sa.h: se encarga de crear las SA (*Security Association*) [18] que asocian una dirección IP de origen, una dirección IP de destino como parte del túnel IPSec

• sp4.h: se encarga de crear los SPI (*Security Parameter Index*) [18] que hacen parte de las SA, asignando un conjunto de algoritmos de cifrado a un túnel IPSec

El programa *aesgcm.c* junto con los archivos de configuración *aead.cfg* (para AES-128-GCM) y *aescbc.cfg* (para AES-128-CBC y AES-256-CBC) se encarga de asociar las librerías predefinidas entre sí, enlazando cada módulo según la capa del modelo OSI que le corresponde siguiendo el diagrama de flujo. Es decir, el programa principal *aesgcm.c* se encarga de llamar cada una de las funciones de capas inferiores cuando lo requiera.

El archivo de configuración para IPSec incluye la información de llaves para autenticación y cifrado. IPSec se requiere dos llaves diferentes para inicializar el proceso de cifrado y el tamaño de cada llave depende del algoritmo de cifrado usado. La primera llave se usa para cifrar (*encryption key*) y la segunda llave se usa para autenticar (*authentication key*), en AES-CBC el tamaño de las llaves para esta aplicación es de 128 o 256 bits.

El archivo de configuración *aead.cfg* carga el modelo AEAD donde no se requiere llave de autenticación, únicamente de cifrado (*encryption key*) ya que la autenticación en este modo se realiza mediante un campo adicional en la cabecera IP de IPSec [19].

III. Desarrollos

El objetivo principal es implementar una aplicación de software capaz de cifrar y descifrar tráfico de datos en una red de transporte móvil simulada en el laboratorio, como un proceso en tiempo real. Esta aplicación se desarrolló usando la herramienta de desarrollo de software DPDK, introducida por Intel®, aprovechando su microcódigo dedicado para cifrado y aceleración de paquetes AES-NI basado en hardware, el cual facilita la implementación de los algoritmos AES-GCM para IPSec.

Además, se aprovecha el desarrollo de Intel®, desde la perspectiva de la optimización del proceso de cifrado de datos, de combinar el algoritmo de cifrado AES (*Advanced Encryption Standard*) en la pila de protocolos IPSec (*Internet Protocol Security*) y el algoritmo de autenticación en bloques de llave simétrica GCM (*Galois Counter Mode*). Para ello se emplearon procesadores e interfaces de red con tecnología de Intel®, así como instrucciones de software compatibles con lenguaje C bajo el esquema criptográfico de Linux [20].

A. Elementos de Hardware y Software

Para optimizar el uso de recursos en el proyecto y gracias a la abstracción de hardware que implementa VMware para la asignación dedicada de recursos a cada máquina virtual, se instaló el sistema operativo VMware ESXi-6.5.0 como hypervisor en un servidor Dell PowerEdge R410 [21]. En la Tabla II están descritas las características del hardware disponible para la implementación de las máquinas virtuales que alojan la aplicación.

TABLA II
CARACTERÍSTICAS HARDWARE Y SOFTWARE

Item	Descripción
Plataforma Servidor	DELL PowerEdge R410
Chipset	Intel® Xeon® Processor L5640 12M Cache, 2.26 GHz, 5.86 GT/s Intel® QPI https://ark.intel.com/products/47926/Intel-Xeon-Processor-L5640-12M-Cache-2_26-GHz-5_86-GTs-Intel-QPI Number of cores 12, Number of threads 24.
Memoria	Total 32768 MBs over 8 channels @ 1333 MHz
PCIe	1 PCIe G2 slot + 1 storage slot
QAT	PCI-e x16 mode
Sistema Operativo	Fedora 27
BIOS	PER410-011200
Linux kernel version	4.13.9-300.fc27.x86_64
GCC version	gcc (GCC) 7.3.1 20180130 (Red Hat 7.3.1-2)
DPDK version	18.05

Como referencia dentro de la implementación, se intentó usar KVM de Linux por ser software libre con licencia abierta, pero la eficiencia en procesos de creación de máquinas virtuales y la compatibilidad del controlador de red no fueron los más idóneos para continuar con la implementación en este hypervisor. Para el caso de VMware se solicitó una licencia de pruebas válida por un año. Como base para el desarrollo de la aplicación de cifrado se crearon dos máquinas virtuales con sistema operativo Linux Fedora 27 y se desarrolló la aplicación llamada AESGCM en la capa de aplicación usando DPDK, actuando como gateway de seguridad y cuya función es cifrar y descifrar el tráfico de datos TCP/IP que lo atraviesa usando IPSec. La Figura 5 muestra el listado de máquinas virtuales creadas.

Como punto de control para comparaciones de rendimiento se crean otras dos máquinas virtuales con sistema operativo Fedora 27 con la misma función de red usando el software libre "*strongSwan*" [22] que es una implementación de IPSec de referencia en la industria.

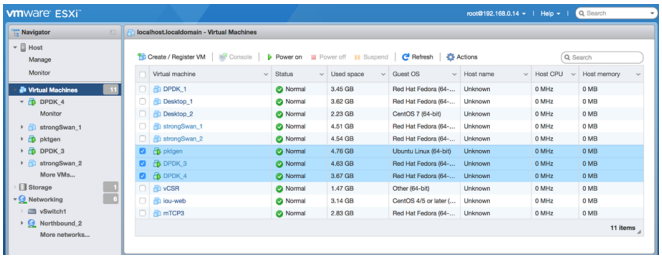


Fig. 5. Listado de Máquinas virtuales.

Cabe destacar que todas las máquinas virtuales fueron creadas asignado los mismos recursos de hardware:

- 4 vCPU organizados en 2 vCPU en 2 slots
- 8GB memoria RAM
- 20GB Disco Duro
- Tres interfaces de red: 1 asignada para gestión de 1Gbps y 2 para datos de 10Gbps cada una

Finalmente para efectos del plan de pruebas general, se crea una máquina virtual sobre el sistema operativo Linux Ubuntu 4.4.0-87, encargada de generar y recibir tráfico de manera que se pueda medir y analizar el funcionamiento y rendimiento de las aplicaciones de cifrado, usando un programa predefinido en DPDK para generación de tráfico TCP/IP llamado pktgen; esta aplicación puede generar paquetes TCP/UDP/ICMP con tamaños desde 64 hasta 2000 bytes y con capacidad generar hasta 10Gbps por interface [23].

B. *Arquitectura de la red*

La topología de red usada en las pruebas para ambos escenarios, tanto para el cifrado/descifrado de tráfico usando la aplicación AESGCM en DPDK, así como para la aplicación estándar *strongSwan* es presentada en la Figura 6. En ambos casos se mantiene el direccionamiento IP, así como las configuraciones y consideraciones para la red de transporte en el ambiente móvil.

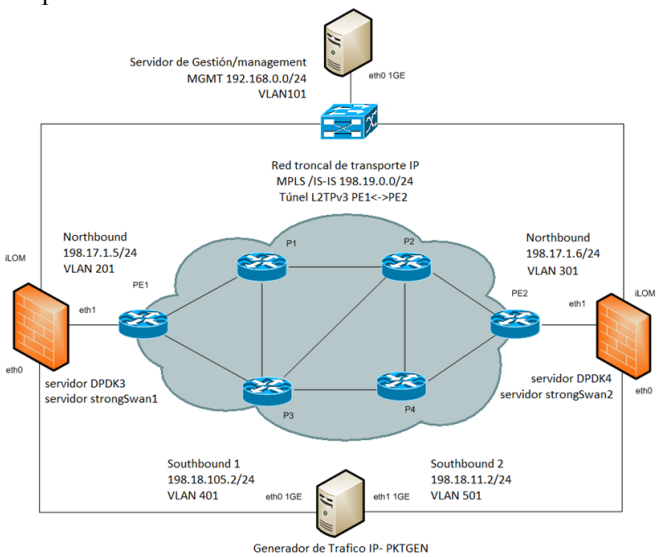


Fig. 6. Topología de red para las pruebas de rendimiento.

La infraestructura de red interna en el entorno virtual se basa en un modelo SDN (*Software Defined Networks*) [24], [25] sobre el cual se crean los segmentos de red que interconectan las máquinas virtuales entre sí y hacia el exterior, que en este caso es la red de transporte implementada en el laboratorio de redes de la Univesidad. Se definen los segmentos de red virtuales dentro de la plataforma de la siguiente forma:

Northbound: se encarga de conectar la interface externa de los Gateway de cifrado/descifrado con la red de transporte de pruebas en el laboratorio, como se trata de una red punto a punto, consta de dos segmentos:

- Northbound_1: hacia la interface externa del cifrador DPDK3
- Northbound_2: hacia la interface externa del cifrador DPDK4

Como enlace hacia la red de transporte se usa la interface física vmnic2 del servidor en modo troncal transportando las VLAN 201 y 401

Southbound1: se usa para interconectar la interface interna del cifrador/descifrador DPDK3 hacia el generador de tráfico en su interface Tx/Rx 0.

Southbound2: se usa para interconectar la interface interna del cifrador/descifrador DPDK4 hacia el generador de tráfico en su interface Tx/Rx 1.

Como enlace hacia el exterior se usa la interface física vmnic3 del servidor en modo troncal transportando las VLAN 301 y 501.

El generador de tráfico pktgen se conecta en ciclo cerrado para las pruebas de medición de latencia punto a punto en la red.

MGMT: se usa como red de gestión y operación

Como enlace hacia el equipo de gestión se usan las interfaces vmnic0 (activa) y vmnic1 en la VLAN 101 en modo acceso.

C. *Red de transporte*

SE implementó en ambiente de pruebas en el laboratorio redes de la Universidad Javeriana una red de transporte que emula el ambiente de producción usado por los operadores y proveedores de servicio de Internet en las redes móviles, de modo que se configuró una VPN (Virtual Private Network) en capa 2 del modelo OSI usando MPLS (Multi-Protocol Label Switching) [26] sobre un protocolo de enrutamiento dinámico IS-IS (Intermediate System-to-Intermediate System). La topología de red física se muestra en la Figura 7.

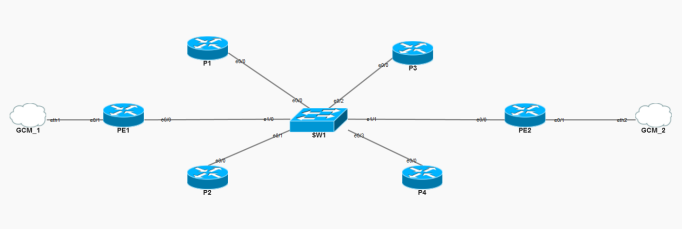


Fig. 7. Topología de red de transporte física.

Los enrutadores o routers y switches disponibles en el laboratorio tienen las siguientes características:

Routers:

- Hardware: Cisco 2821 (revision 53.51).

- Software: Cisco IOS Software, 2800 Software (C2800NM-ADVENTERPRISEK9-M), Version 12.4(16a), RELEASE SOFTWARE (fc2).

Switch:

- Hardware: Cisco WS-C2960-24TC-L (PowerPC405) processor (revision E0).
- Software: Cisco IOS Software, C2960 Software (C2960-LANBASE-M), Version 12.2(35)SE5, RELEASE SOFTWARE (fc1).

La infraestructura de red de transporte móvil se simula usando una VPN (*Virtual Private Network*) de capa 2 usando la tecnología L2TPv3 de acuerdo las funcionalidades soportadas por la versión de hardware y software de los routers del laboratorio. En el caso particular del ambiente del laboratorio de la Universidad solo hay 5 routers disponibles, por lo que la topología se mantuvo según el diseño original, pero sin contar con los enlaces y el dispositivo que representa el router faltante P3.

D. Implementación DPDK

El procedimiento para habilitar el sistema operativo Fedora 27 para ejecutar la herramienta de desarrollo DPDK es como sigue:

• Asignación espacios de memoria dedicados para la aplicación *hugepages*

• Instalación de las librerías y dependencias para desarrollo de aplicaciones en Linux

• Compilación e instalación del código fuente de la herramienta de desarrollo DPDK

• Definición de las variables de entorno EAL

• Definición enlaces simbólicos a las librerías

• Inicialización de los controladores PMD para las interfaces de red usando el controlador genérico igb_uio

• Carga, compilación y exportación de las librerías multi-buffer para IPSec: AESNI_MB_PMD y AESNI_GCM

• Activación y ejecución de pruebas sobre los controladores de interface virtuales para cifrado cryptodev

La aplicación aesgcm.c se ejecuta de la siguiente forma:

[root@dpdk3 aes-gcm]#./build/ipsec-secgw -l 1,2 -n 4 -v --vdev=crypto_aesni_gcm -- -p0x3 -u0x1 --config="(0,0,1),(1,0,2)" -P -f ./aead.cfg

Las variables EAL expresadas al ejecutar la aplicación definen los recursos de hardware y la configuración así:

-l define la cantidad de procesadores a inicializar

-n define la cantidad de canales de memoria

-v define el tipo de interface virtual cryptodev con *--vdev=crypto_aesni_gcm*,

-p define las interfaces habilitadas

-u define las interfaces protegidas o internas (usando el concepto de zona de seguridad)

--config="(0,0,1),(1,0,2)" define la asignación los búferes de transmisión y recepción a cada procesador para optimizar el uso de recursos

-f define el archivo de configuración con *./aead.cfg*

A nivel de red, la aplicación se encarga de recibir el tráfico en la interface de entrada o Southbound_1, procesar las cabeceras de los paquetes IP y separar la información de datos (payload del paquete) en bloques de 128 bits, cifrar usando AES y generar el hash con GCM, y finalmente enviar el tráfico cifrado a la interface de salida o Northbound_1 para transmitirlo por la red de transporte. A su vez, en el otro extremo, la aplicación se encarga de descifrar el tráfico recibido en la interface de entrada Northbound_2 y devolverlo en texto claro en la interface Southbound_2 para su análisis en las mediciones de desempeño.

Realizando una captura del tráfico en la interface Northbound_2 se pueden ver los paquetes cifrados IPSec/ESP transitar por la red:

```
[root@probe1~]# tcpdump -nn -i ens224 #interface Northbound_2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode listening on ens224, link-type EN10MB (Ethernet), capture size 262144 bytes
23:13:30.250275 IP 198.17.1.5 > 198.17.1.6: ESP(spi=0x00000005,seq=0xbcf2421), length 80
23:13:30.250281 IP 198.17.1.5 > 198.17.1.6: ESP(spi=0x00000005,seq=0xbcf2422), length 80
23:13:30.250283 IP 198.17.1.5 > 198.17.1.6: ESP(spi=0x00000005,seq=0xbcf2423), length 80
3 packets captured
```

IV. PROTOCOLO DE PRUEBAS

En los procesos de desarrollo de software se emplean los estándares de medición o benchmark para determinar la eficiencia y desempeño relativo de una aplicación, generalmente ejecutando una serie de pruebas estandarizadas sobre la misma. Estos estándares de medición se asocian con la medición de desempeño de ciertas características de hardware o software en un computador o elemento de red, y en este caso las pruebas de desempeño proveen un método de comparación entre varios elementos o subsistemas a lo largo de distintas configuraciones o parámetros de interés.

Para este efecto, se diseñan protocolos de pruebas para medir el desempeño del software en áreas o características específicas del software o la red, basados en la metodología propuesta por el RFC2544 [27].

Dentro de las pruebas a realizadas a la plataforma se tienen en cuenta los siguientes parámetros relevantes para la evaluación de la aplicación de software de red:

Tipos de benchmarks:

• I/O benchmark: usado para medir el desempeño de transmisión o throughput y los tiempos de respuesta de la aplicación de software o la red.

• Parallel benchmark: usado en arquitecturas de hardware con múltiples procesadores o núcleos.

Preparación de la prueba: El RFC2544 indica que la mejor forma de medir desempeño en aplicaciones en redes es usar un sensor generador de tráfico con interfaces con capacidad para transmitir y recibir datos de forma simultánea desde y hacia el dispositivo bajo pruebas (DUT, Device Under Test), de manera que se pueda tener un camino cerrado para los datos que atraviesan la plataforma [27]. Con este propósito se usa la aplicación pktgen basada también en DPDK para la generación del tráfico de prueba.

Se usan tramas/paquetes en formato Ethernet/TCP/IP con tamaño variable. Para Ethernet se usan los siguientes tamaños

de paquete durante las pruebas:

64, 128, 256, 512, 1024, 1280 y 1518 bytes

Estos tamaños incluyen los valores mínimos y máximos permitidos en el estándar de Ethernet. El generador de tráfico de prueba debe tener en cuenta los números de secuencia de los paquetes recibidos para verificar que se trata de los mismos paquetes transmitidos. En redes de transporte se tiene una carga moderada de tráfico de gestión (SNMP) y actualizaciones de enrutamiento de protocolos dinámicos (IS-IS, OSPF, MP-BGP), por lo que para la prueba se minimiza este tráfico para no alterar los resultados de las pruebas. Para las tramas/paquetes de broadcast se considera que un 1% del tráfico total es aceptable dentro de la prueba.

Para hacer las pruebas más apegadas a un ambiente de red de producción, se mide el desempeño del DUT con tráfico bidireccional. Se considera que en un ambiente de red de producción el tráfico puede generarse en ráfagas, por lo que las pruebas consideran los escenarios de red en estado estable y también durante episodios de tráfico en ráfagas, teniendo en cuenta que los paquetes mantengan el espacio entre tramas según el estándar. Los tamaños de tráfico de ráfagas (burst size) listados en la norma son 16, 64, 256 y 1024 tramas/paquetes. Para las pruebas se usa un tamaño de ráfaga de 64 paquetes. Se considera una prueba aceptable con una duración mínima de 60 segundos.

Para medir el desempeño de transmisión (throughput), se plantea como objetivo determinar el desempeño o throughput del DUT en la red, teniendo en cuenta las recomendaciones listadas en el RFC1242 [45]. Para la medición de throughput se requiere que se envíen tramas desde un generador de tráfico al DUT y luego contando las tramas recibidas de regreso en el generador en un periodo de tiempo. Los resultados se reportan en una gráfica que muestre las variables analizadas y el throughput máximo alcanzado por prueba.

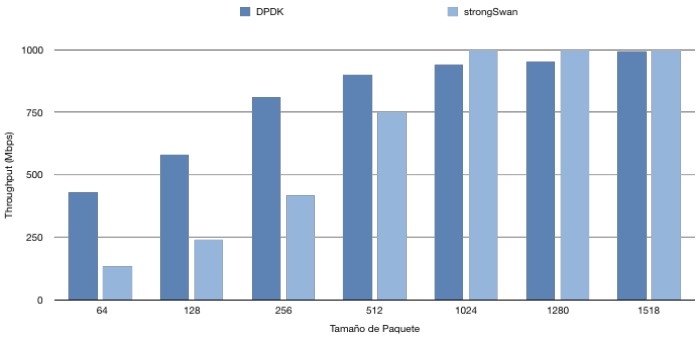
Para la medición de latencia se requiere que se envíen tramas desde un generador de tráfico al DUT, y luego se mide el tiempo que tarda la trama en llegar de regreso al generador. La prueba debe durar al menos 120 segundos y repetirse hasta 20 veces. Los resultados se reportan en una tabla por cada prueba con el tamaño de paquete y con cada variable analizada contabilizando en segundos o milisegundos el tiempo de latencia.

Para la medición tramas/paquetes perdidos en la red se requiere que se envíen tramas desde un generador de tráfico al DUT y a continuación, se cuenta la cantidad de tramas recibidas correctamente de regreso en el generador. Los resultados se reportan en una gráfica donde se muestren las tramas enviadas contra el porcentaje de tramas pérdidas para cada prueba con tamaño de trama/paquete variable.

V. Validación y Resultados

En el primer conjunto de pruebas para medición de capacidad de transmisión de datos cifrados para la aplicación DPDK con el algoritmo AES-GCM-128 para tráfico de datos en TCP, UDP e ICMP se evidencia una mejora de hasta un 56% en el desempeño con paquetes pequeños (> 512 bytes cada paquete) y un desempeño similar con paquetes grandes (< 1024 bytes) comparado con la aplicación de referencia strongSwan, como muestra la Figura 8.

Fig. 8. Comparación capacidad (throughput) en TCP con AES-GCM-128.



Dado que la red del laboratorio tiene un límite real de 100Mbps de capacidad de transmisión por las interfaces FastEthernet de los enrutadores Cisco, se ejecutan pruebas en dos escenarios, uno punto a punto con la red de transporte donde la capacidad máxima obtenida es cercana a los 97 Mbps, y otra con las máquinas virtuales con la aplicación DPDK conectadas back-to-back, obteniendo una capacidad máxima de 3286 Mbps sin errores.

En las redes móviles se tiene como referencia un tamaño de paquete promedio de 600 bytes IMIX (Internet Mix) con una distribución de 58.67% de paquetes de 90 bytes, 2% de paquetes de 92 bytes, 23.66% de paquetes de 594 bytes, y 15.67% de paquetes de 1418 bytes [28], por lo que el rendimiento de la aplicación para el caso de la red de transporte móvil en estas condiciones se ve optimizado en un 22% aprox.

La RFC 2679 [29] define las métricas para el indicador de latencia en un sentido para tráfico hacia Internet IPPM (IP Performance Measurement), y la industria de acuerdo a sus requerimientos define los límites aceptables para aplicaciones en tiempo real como voz y video así como para el entorno de red, que en este caso es la red de transporte móvil. Con estas condiciones para una experiencia de usuario aceptable con aplicaciones de alta exigencia de calidad de servicio, la latencia máxima debe ser de 10ms [30]. La Figura 9 muestra la comparación en el comportamiento de la latencia en la red de transporte del laboratorio. En este caso la latencia de la aplicación DPDK es mayor a la aplicación de referencia *strongSwan* en promedio en un 21%, en el peor caso la latencia es de 13ms con paquetes de 1518 bytes.

En el escenario de paquetes perdidos, la RFC 2680 [31] determina los mecanismos de medición de este indicador en redes de Internet IPPM, donde una tasa de paquetes perdidos menor a 1% se considera aceptable. En la Figura 10 se ilustra el comportamiento de paquetes perdidos. Se evidencia que para el throughput máximo que se alcanza en el laboratorio (100 Mbps) la tasa de paquetes perdidos es menor a 1% para todos los tamaños de paquete con la aplicación DPDK, mientras que en paquetes pequeños la aplicación *strongSwan* tiene hasta un 30% de paquetes perdidos. Esa tasa se incrementa aún más a medida que el throughput aumenta también.

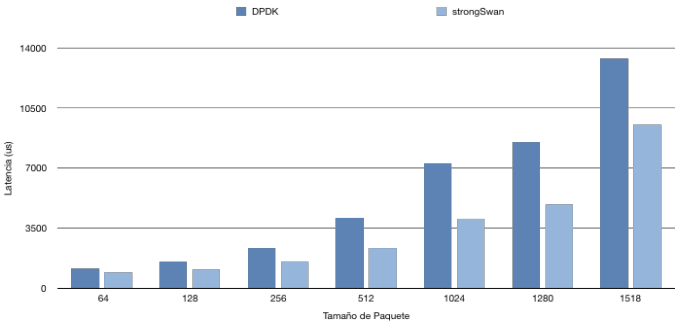


Fig. 9. Comportamiento latencia en red del laboratorio con AES-GCM-128.

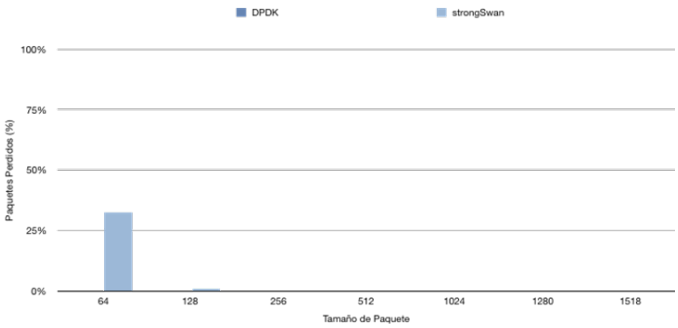


Fig. 10. Medición Paquetes perdidos en 100Mbps con AES-128-GCM.

La Figura 11 muestra el incremento en la tasa de paquetes perdidos con un throughput de 1Gbps back-to-back. Con paquetes pequeños y alto tráfico ambas plataformas sufren de una tasa muy alta de paquetes perdidos. En la práctica, esto es una dificultad que se presenta en el dimensionamiento de redes de comunicaciones y en una mayor medida en las redes móviles, ya que la mayoría (60%) de los paquetes son menores a 90 bytes [28] y las hojas de fabricante proveen datos de desempeño de sus equipos para tamaño de paquete promedio (600 bytes) o el máximo (1500 bytes) [8], [10].

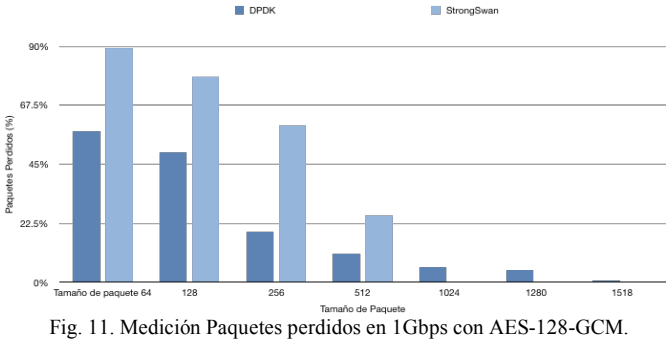


Fig. 11. Medición Paquetes perdidos en 1Gbps con AES-128-GCM.

VI. Conclusiones

En términos generales la aplicación desarrollada bajo DPDK es capaz de manejar una mayor cantidad de tráfico cifrado con menor tasa de error o paquetes perdidos, esto debido a la implementación de las optimizaciones incluidas en el set de instrucciones de Intel® para los algoritmos AES y GCM lo que genera valor en un escenario de red móvil comercial donde el operador de red experimenta más eficiencia en el uso del hardware en un entorno de red más seguro garantizando confidencialidad en la transmisión.

Sin embargo, se evidencia una latencia mayor en la aplicación DPDK comparada con la aplicación de referencia *strongSwan*. Esto puede ser causado por la implementación del código por parte del autor, al enlazar los procesos de enrutamiento LPM y asignación de variables para cifrado en la rutina ESP, lo cual es susceptible de ser optimizado también.

Como trabajo futuro, además de la optimización del código implementado para mejorar la latencia, se puede explorar la opción de adicionar el protocolo IKE a la aplicación para manejo de llaves dinámicamente de manera que la aplicación sea completamente interoperable con terminadores de VPN IPSec de la industria, ya que como parte de la implementación de la aplicación se manejan tanto las llaves como el vector de inicialización de forma estática.

Referencias

[1] Broadband Forum, “Use of MPLS in Mobile Backhaul Networks Introduction.”

[2] ADTRAN White Paper, “4G / LTE Mobile Backhaul Reference Architecture,” 2012.

[3] IP/MPLS FORUM, “White paper,” pp. 1–8, 2008.

[4] Nokia Networks, “LTE Transport v6.0,” 2014.

[5] SANS Institute, “Fiber Optics and its Security Vulnerabilities,” 2005.

[6] V. Fineberg, “A practical architecture for implementing end-to-end QoS in an IP network,” *IEEE Commun. Mag.*, vol. 40, no. 1, pp. 122–130, 2002.

[7] L. Yul, S. Jia, C. Xu, J. Guan, and D. Gaol, “An IPsec Seamless Switching Mechanism with High Availability and Scalability by Extending IKEv2 Protocol,” no. 61001122.

[8] Juniper Networks, “SRX Series Services Gateways for the Branch,” no. Cli. pp. 1–14, 2009.

[9] Cisco Systems, “ASA 5500 Next Generation Firewalls,” 2015. [Online]. Available: http://www.cisco.com/c/en/us/products/security/asa-5500-series-next-generation-firewalls/datasheet-listing.html. [Accessed: 02-Nov-2015].

[10] CheckPoint Software Technologies, “21400 Appliance Datasheet,” 2015. [Online]. Available: https://www.checkpoint.com/downloads/product-related/datasheets/21400-appliance-datasheet.pdf. [Accessed: 03-Nov-2015].

[11] E. Barker, A. Roginsky, G. Locke, and P. Gallagher, “Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths,” *NIST Spec. Publ.*, vol. 1, no. January, pp. 800–131, 2011.

[12] W. F. D. McGrew, “Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH),” 2014. [Online]. Available: http://www.ietf.org/internet-drafts/draft-mcgrew-ipsec-me-esp-ah-reqts-00.txt. [Accessed: 11-Feb-2015].

[13] G. L. Guo, Q. Qian, and R. Zhang, “Different implementations of AES cryptographic algorithm,” *Proc. - 2015 IEEE 17th Int. Conf. High Perform. Comput. Commun. 2015 IEEE 7th Int. Symp. Cybersp. Saf. Secur. 2015 IEEE 12th Int. Conf. Embed. Softw. Syst. H*, pp. 1848–1853, 2015.

[14] M. Dixon, “Optimized Galois- Implementation on Intel ® Architecture Processors,” no. August, 2010.

[15] Intel Corporation, “Intel ® Data Plane Development Kit (Intel ® DPDK),” no. January. 2014.

[16] A. Hoban, “Using Intel ® AES New Instructions and PCLMULQDQ to Significantly Improve IPSec Performance on Linux,” *Intel Corp.*, no. August, pp. 1–26, 2010.

[17] IEEE COMPUTER SOCIETY, *Swebok - Guía al cuerpo de conocimiento de la Ingeniería de Software*. 2004.

[18] P. Hoffman, “‘Cryptographic Suites for IPsec’, RFC 4308,” *DOI 10.17487/RFC4308*, 2005.

[19] Cisco Systems, “AEAD Authenticated Encryption with Replay prOtection (AERO),” 2014.

[20] Intel, “Intel DPDK - Getting Started Guide for Linux,” *Intel DPDK*, 2015.
[21] Dell Inc., “Dell ™ PowerEdge ™ r410 server,” pp. 3–4, 2011.
[22] A. Steffen, “strongSwan - The new IKEv2 VPN Solution,” pp. 1–19, 2007.
[23] K. Wiles, “Pktgen Documentation,” 2015.
[24] A. Gelberger, N. Yemini, and R. Giladi, “Performance analysis of Software-Defined Networking (SDN),” *Proc. - IEEE Comput. Soc. Annu. Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst. MASCOTS*, pp. 389–393, 2013.
[25] M. Furukawa, K. Kuroda, T. Ogawa, and N. Miyaho, “Highly Secure Communication Service Architecture using SDN Switch,” *10th Asia-Pacific Symp. Inf. Telecommun. Technol. (APSITT 2015)*, vol. 0086, no. c, pp. 1–3, 2015.
[26] D. Woods, “White Paper - Addressing Inter Provider Connections with MPLS-ICI,” *Ergonomics*, pp. 1–8, 2001.
[27] S. and J. M. Bradner, “‘Benchmarking Methodology for Network Interconnect Devices’, RFC 2544,” *DOI 10.17487/RFC2544*, 1999.
[28] Spirent Communications, “IPSec IMIX defined by Spirent Communications,” 2009.
[29] G. S. K. M. Z. Almes, “RFC 2679 A One-way Delay Metric for IPPM,” pp. 1–21, 1999.
[30] Zeljko Savic, “LTE Design and Deployment Strategies,” *Cisco Syst.*, 2010.
[31] G. S. K. M. Z. Almes, “RFC 2680 A One-way Packet Loss Metric for IPPM,” pp. 1–16, 1999.