INSTITUTO POLITÉCNICO NACIONAL

CENTRO DE INVESTIGACIÓN EN COMPUTACIÓN

Análisis de seguridad de RINA

TESIS

QUE PARA OBTENER EL GRADO DE:

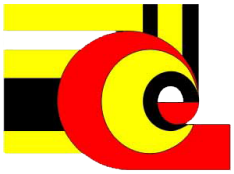
MAESTRÍA EN CIENCIAS EN INGENIERÍA DE CÓMPUTO CON OPCIÓN EN SISTEMAS DIGITALES

P R E S E N T A:

Ing. Patricio Mercado Capistran

Directores de tesis:
Dr. Eleazar Aguirre Anaya
Dr. Raúl Acosta Bermejo

Mexico, D.F. Julio 2015



SIP-14 bis

INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D.F. siendo las 10:00 horas del día 5 del mes de junio de 2015 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación del:

Centro de Investigación en Computación

para examinar la tesis titulada:

"Análisis de seguridad de RINA"

Presentada por el alumno(a):

Mercado	**Capistran**	**Patricio**
Apellido paterno	Apellido materno	Nombre(s)

Con registro:

A	1	3	0	2	4	2

aspirante de: **MAESTRÍA EN CIENCIAS EN INGENIERÍA DE CÓMPUTO CON OPCIÓN EN SISTEMAS DIGITALES**

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***APROBAR LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Directores de Tesis

Dr. Eleazar Aguirre Anaya

Dr. Raúl Acosta Bermejo

Dr. Ponciano Jorge Escamilla Ambrosio

Dra. Nareli Cruz Cortés

Dr. Moisés Salinas Rosales

1

PRESIDENTE DEL COLEGIO DE PROFESORES

Dr. Luis Alfonso Villa Vargas

INSTITUTO POLITÉCNICO NACIONAL
DIRECCIÓN

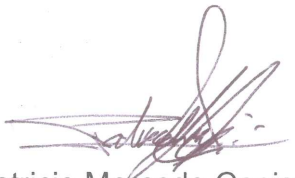
INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México el día 9 del mes Julio del año 2015, el (la) que suscribe Patricio Mercado Capistran alumno (a) del Programa de Maestría en Ciencias en Ingeniería de Cómputo con opción en sistemas digitales con número de registro A130242, adscrito al Centro de Investigación en Computación, manifiesta que es autor (a) intelectual del presente trabajo de Tesis bajo la dirección del Dr. Eleazar Aguirre Anaya y Dr. Raúl Acosta Bermejo y cede los derechos del trabajo intitulado "Análisis de seguridad de RINA", al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección patriciomc@yahoo.com.mx. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.



Patricio Mercado Capistran

Nombre y firma

Resumen

El presente trabajo lleva a cabo el análisis de las propiedades de la seguridad de Integridad y Disponibilidad en el "flujo de información" entre dos entidades en una implementación básica de RINA en su versión 1.9. Buscando con esto comprobar que la información no pueda ser falseada y se encuentre disponible.

Éste análisis de seguridad se lleva a cabo con apoyo del diseño de un instrumento de seguridad; instrumento que está integrado por una metodología de evaluación; el diseño de escenarios de pruebas; el diseño de ataques y la selección o diseño de herramientas para explotar vulnerabilidades que comprometen la integridad y la disponibilidad de la implementación de RINA.

La metodología de evaluación es una propuesta que surge del análisis de dos metodologías de evaluación, ATAM y SAAM. Las cuales son aplicables a la arquitectura TCP/IP. Se retoman elementos de cada una de ellas, hacer una adecuación de los mismos y de esta manera generar una evaluación de seguridad que es aplicable a la implementación de RINA.

Escenarios y herramientas para evaluar la integridad y la disponibilidad del flujo de información en una implementación de RINA como tal no existen.

El instrumento de evaluación se aplica sobre una implementación de la arquitectura RINA en un sistema operativo Debian 7 "Wheezy", capturando el flujo de información entre dos entidades para un análisis e interpretación, siendo éste uno de los elementos necesarios para el diseño de los ataques que comprometen integridad y disponibilidad. Determinandó de esta manera con el instrumento de evaluación que la implementación básica de RINA en su versión 1.9 no es segura.

El capítulo uno describe los problemas que presenta el Internet de hoy y los problemas que debe atender el Internet del futuro. El segundo capítulo refiere las metodologías base para el diseño de la metodología aplicable a la implementación de RINA. El tercer capítulo, conforma la metodología de evaluación aplicable a la implementación de RINA, la cual surge con base en el análisis de las metodologías base. El capítulo cuarto refiere a la ejecución del instrumento de evaluación sobre la implementación. El quinto y último capítulo presenta las conclusiones a las que se llegaron en el presente trabajo.

Abstract

This investigation carried out the analysis of integrity and availability of information flow between two entities in a deployment of RINA version 1.9. The idea is to ensure that the information can't be distorted and is available

This safety analysis is performed to support the design of a security instrument; instrument consists of an evaluation methodology; design test scenarios; and the selection and design tools to exploit vulnerabilities that compromise the integrity and availability of implementing RINA.

The evaluation methodology is a proposal that emerges from analysis of two evaluation methodologies, ATAM and SAAM. Which they're applicable to the TCP/IP architecture. It's necessary to return certain elements of each, to an adaptation of the same and thus generate the safety assessment is applicable to the implementation of RINA. Scenarios and tools to assess the integrity and availability of information flow in an implementation of RINA as such doesn't exist.

The assessment tool it's applied to an implementation of the RINA architecture. In a Debian 7 Wheezy Operating System, capturing the flow of information between the two entities for analysis and interpretation, this being one of the elements for attacks designed to compromise integrity and availability. Determining the assessment instrument that implementation in version 1.9 of RINA is not secure.

Chapter one describes the TCP/IP architecture with major security problems and present solutions that have been implemented. The second chapter refers to the methodologies basis for designing the methodology for the implementation of RINA. The third chapter presents the methodology applicable to the implementation of RINA, which arises based on the analysis of the SAAM and ATAM evaluation methodologies. The fourth chapter refers to the execution of the instrument on implementation. The fifth and final chapter presents the conclusions that come in this paper.

Agradecimientos

Le agradezco a Dios por haberme acompañado y guiado a lo largo de mi carrera, por ser mi fortaleza en los momentos de debilidad y por brindarme una vida llena de aprendizajes, experiencias y sobre todo felicidad.

Le doy gracias a mis padres y hermano por apoyarme en todo momento, por los valores que me han inculcado, y por haberme dado la oportunidad de tener una excelente educación en el transcurso de mi vida.

Deseo expresar mi agradecimiento a los directores de esta tesis, al Dr. Eleazar Aguirre Anaya y al Dr. Raúl Acosta Bermejo, por la dedicación y apoyo que han brindado a este trabajo, por el respeto a mis sugerencias e ideas, por la dirección y el rigor que ha facilitado a las mismas. Gracias por la confianza ofrecida desde que llegué a este centro de investigación.

Por su orientación y atención a mis consultas, mi agradecimiento al Dr. Moisés Salinas Rosales, a la Dra. Nareli Cruz Cortés, al Dr. Ponciano Jorge Escamilla Ambrosio, a la Dra. Gina Gallegos García de la ESIME Culhuacan por las sugerencias recibidas. Gracias a todos por sus valiosas sugerencias en momentos de duda.

Quiero expresar mi gratitud al CONACYT y a todas las personas que forman parte de esta casa de estudios, quienes a través de sus funciones brindan la formación de estudios de excelencia en investigación y posgrado. Por haber sido un segundo hogar donde día a día viví agradables experiencias académicas.

Pero un trabajo de investigación es también fruto del reconocimiento y del apoyo vital que nos ofrecen las personas que nos estiman, sin el cual no tendríamos la fuerza y energía que nos anima a crecer como personas y como profesionales.

Agradezco a mis amigos Oscar Ruíz, Alejandro Escorcia, Rolando Sanchez, Rubén Hernández, Rodrigo Jurado, Antonio (Toño), Miriam Barboza, Sergio Proa, Rodrigo Roman, Javier Lopez, Sara, Eduardo Pacheco, César Hernández, Alfonso Calvo.

Índice general

List of Figures IX

1. El Internet del futuro 1
1.1. El Internet de hoy . . . 1
1.1.1. Generaciones de Internet . . . 2
1.1.2. Caracteristicas necesarias para la siguiente Arquitectura de Internet 3
1.2. Arquitecturas para el Internet del futuro . . . 4
1.2.1. Service Oriented Network Architectures (SONATE) . . . 5
1.2.2. NENA . . . 6
1.2.3. XIA (eXpressive Intenetwork Architecture) . . . 7
1.2.4. MobilityFirst . . . 8
1.2.5. NDN . . . 10
1.2.6. NEBULA . . . 11
1.2.7. RINA . . . 12
1.3. Administración de Procesos . . . 14
1.3.1. Procesos . . . 14
1.3.2. Jerarquía de procesos . . . 14
1.3.3. Estado de un procesos . . . 15
1.4. Arquitectura RINA . . . 17
1.4.1. Comunicación en RINA . . . 18

2. Evaluaciones de seguridad para arquitecturas de comunicación 23
2.1. Evaluación de seguridad . . . 23
2.2. Metodologías de evaluación . . . 25
2.3. Evaluación con ATAM . . . 25
2.4. Evaluación con SAAM . . . 28
2.5. Propuesta de solución . . . 31

3. Diseño de la evaluación de seguridad de RINA 34
3.1. Etapa de Presentación . . . 34
3.1.1. Presentación de la metodología . . . 34
3.1.2. Descripción de la arquitectura . . . 36
3.1.3. Identificación del objetivo . . . 37
3.2. Fase de Análisis e Investigación . . . 38

3.2.1. Análisis de la arquitectura . . . 41
3.2.2. Diseño y descripción del escenario de pruebas . . . 43
3.2.3. Herramientas . . . 48
3.3. Etapa de Pruebas . . . 49
3.3.1. Ejecución individual de la evaluación en el escenario de pruebas . . . 49
3.4. Etapa de resultados . . . 50
3.4.1. Análisis de resultados . . . 50
3.4.2. Presentación de resultados . . . 51

4. Aplicación de la metodologia de evaluación sobre la implementación de RINA 52
4.1. Etapa de presentación . . . 52
4.1.1. Presentación de la metodología de evaluación . . . 52
4.1.2. Descripción de la arquitectura . . . 53
4.1.3. Identificación del objetivo . . . 54
4.2. Etapa de análisis e investigación . . . 55
4.2.1. Análisis de la arquitectura . . . 55
4.2.2. Diseño y descripción de los escenarios de pruebas . . . 56
4.3. Etapa de pruebas . . . 65
4.3.1. Ejecución individual de la evaluación en los escenarios de pruebas 65

5. Resultados 71
5.1. Presentación de resultados . . . 71
5.1.1. Resultados . . . 71
5.1.2. Análisis de resultados . . . 74

A. Instalación del sistema operativo 83

B. Instalación de la implementación 84

C. Configuración del sistema 1 86

D. Configuración del sistema 2 87

Índice de figuras

1.1. Arquitectura RINA 18

3.1. Elementos en RINA 35
3.2. Objeto de evaluación 36
3.3. Capas en RINA 37
3.4. Flujo de comunicación de un proceso 40
3.5. Comunicación entre aplicaciones 41
3.6. Ataques informáticos sobre RINA 45

4.1. Modelo básico de RINA 54
4.2. Figura 18 .Modelo OSI 58
4.3. Figura 19 .Modelo OSI 58
4.4. Figura 8 .Modelo OSI 61
4.5. Figura 21 .Modelo OSI 61
4.6. Figura 22 .Modelo OSI 62
4.7. Figura 23 .Modelo OSI 62
4.8. Figura 3 .Modelo OSI 63
4.9. Figura 3 .Modelo OSI 63
4.10. Figura 3 .Modelo OSI 64
4.11. Figura 3 .Modelo OSI 64
4.12. Figura 28 .Modelo OSI 66
4.13. Figura 29 .Modelo OSI 67
4.14. Figura 30 .Modelo OSI 68
4.15. Figura 31 .Modelo OSI 68
4.16. Figura 32 .Modelo OSI 69
4.17. Figura 33 .Modelo OSI 70

5.1. Figura 34 .Modelo OSI 72
5.2. Figura 35 .Modelo OSI 72
5.3. Figura 36 .Modelo OSI 73
5.4. Figura 37 .Modelo OSI 73
5.5. Figura 39 .Modelo OSI 74
5.6. Figura 40 .Modelo OSI 74

Acrónimos, abreviaturas y siglas

AP **L**ist **A**bbreviations **H**ere
API **O**rdinary **W**izarding **L**evel
ARP **L**ist **A**bbreviations **H**ere
CDAP **O**rdinary **W**izarding **L**evel
CEP **L**ist **A**bbreviations **H**ere
DIF **O**rdinary **W**izarding **L**evel
DNS **L**ist **A**bbreviations **H**ere
DTP **O**rdinary **W**izarding **L**evel
DTCP **L**ist **A**bbreviations **H**ere
EFCP **O**rdinary **W**izarding **L**evel
FA **L**ist **A**bbreviations **H**ere
IP **O**rdinary **W**izarding **L**evel
IPC **L**ist **A**bbreviations **H**ere
IPCP **O**rdinary **W**izarding **L**evel
IPCM **L**ist **A**bbreviations **H**ere
IRATI **O**rdinary **W**izarding **L**evel
ISP **L**ist **A**bbreviations **H**ere
OWL **O**rdinary **W**izarding **L**evel
LAN **L**ist **A**bbreviations **H**ere
MTU **O**rdinary **W**izarding **L**evel
PDU **L**ist **A**bbreviations **H**ere
QoS **O**rdinary **W**izarding **L**evel
RIB **L**ist **A**bbreviations **H**ere
RINA **O**rdinary **W**izarding **L**evel
RINARP **L**ist **A**bbreviations **H**ere
RMT **O**rdinary **W**izarding **L**evel
SDU **L**ist **A**bbreviations **H**ere
TCP **O**rdinary **W**izarding **L**evel
TTL **L**ist **A**bbreviations **H**ere
UDP **O**rdinary **W**izarding **L**evel
VLAN **L**ist **A**bbreviations **H**ere

Capítulo 1

El Internet del futuro

El Internet que se conoce actualmente crece cada día que pasa. Cuando se concibió, no estaba pesado que alcanzase el tamaño actual. La idea básica de la arquitectura del Internet de hoy en día fue diseñada hace 30 años. A través de los cuales se ha aprendido mucho sobre la comunicación entre redes y el intercambio de paquetes.

1.1. El Internet de hoy

Internet ha cambiado la forma de trabajar y vivir, y ha contribuido positivamente al crecimiento de los negocios. No obstante, muchos elementos de la arquitectura de Internet de hoy en día se desarrollaron hace más de 30 años. La escala de los problemas causados por una tarea administrativa rutinaria realizada por un operador pone de manifiesto el hecho que de Internet no fue producto de un plan, sino que creció a partir de un conjunto de soluciones ad hoc. Desde su nacimiento, Internet ha tenido un crecimiento orgánico. Y cuando surgen los problemas se deben plantear y encontrar soluciones para solventarlos. Pero en ocasiones esos ajustes se convierten en problemas en sí mismos años después [1].

Internet es esencialmente un espacio único de interconexión gigantesco ya que todos los dispositivos con una dirección IP pública son parte de este espacio. Funciona con tecnología antigua que además es muy difícil de actualizar ya que Internet es demasiado grande y hay demasiadas personas implicadas en realizar los cambios para llevar a cabo esa actualización. Como consecuencia de todo ello, las limitaciones que impone la tecnología antigua no se pueden ajustar y se requieren más y más soluciones para hacer frente a los problemas causados por dichas limitaciones. Todo esto hace que Internet sea

cada vez más compleja y por eso mismo más frágil. Al estar todo en el mismo espacio, la escalabilidad sólo se puede lograr por la fuerza bruta: más memoria en los routers y CPUs más rápidas para hacer frente al tamaño de las tablas de enrutamiento. Aunque el problema del agotamiento de las direcciones IP se debería resolver con el nuevo protocolo IPv6, dado que este sigue usando un único espacio de direccionamiento, se sigue sin resolver el problema de tamaño de tablas de enrutamiento. De hecho, lo hace peor a largo plazo, ya que los routers tendrán que almacenar más entradas en la tabla de enrutado y cada entrada consumirá más memoria ya que las direcciones IPv6 son también más largas. Tener un espacio único de direccionamiento también tiene el problema de no permitir el aislamiento entre las diferentes redes de Internet. Todo el mundo opera en ese espacio y eso ha producido algunos problemas. La próxima generación de Internet debe ser segura. Debe permitir establecer límites y hacer cumplir sus políticas dentro de sus límites. Se debe permitir establecer políticas de cómo y dónde recibir la información. Se debe tener la libertad de seleccionar sus nombres, las identificaciones y direcciones con tan poco control centralizado como sea posible. La próxima generación de Internet debe ser diseñado para dispositivos móviles, personas y ordenadores.

Los nombres y el direccionamiento en la arquitectura tienen que permitir a los dispositivos moverse y decidir cómo y dónde desea recibir el tráfico de Internet con todos los derechos de privacidad de su ubicación si se desea

1.1.1. Generaciones de Internet

EL Internet de hoy tiene ya casi 40 años de edad. El primer RFC de la Internet Engineering Task Force data de abril de 1969. Desde sus inicios, Internet ha pasado por dos generaciones mayores con una duración de unos 20 años.

Durante las dos primeras décadas, Internet era sobre todo un proyecto de investigación. La industria se dividió competían desarrollando tecnologías de red: SNA de IBM, Digitals DECnet, XNS Xeroxs y AppleTalk por nombrar algunos. Los grupos de estándares estaban ocupados desarrollando los protocolos de interconexión de sistemas abiertos (OSI). Esta fase duró hasta alrededor de 1989 y se puede llamar a Internet 1.0 o Internet investigación.

A partir de 1989, Internet entró en una nueva fase con la puesta en marcha de la industria y la adopción del Internet para el comercio. Una serie de cuestiones que no fueron considerados importantes hasta ese entonces comenzaron a surgir como resultado de esta adopción. La primer RFC de seguridad data de 1989. Los problemas de escalabilidad requirieron dividir en rutas y dominios. Open Shortest Path First (OSPF) y Border Gateway Protocol (BGP) se desarrollaron como resultado. La escasez de direcciones

IP condujo al desarrollo de una serie de soluciones que incluyen direcciones privadas, traducción de direcciones de red (NAT) e IPv6. El manejo del tráfico, gestión de control, y la calidad de los servicios se convirtieron en problemas importantes. Llamando a esto como Internet 2.0 o la Internet comercial.

Estamos entrando en una nueva fase, donde Internet se ha convertido en una parte integral de la vida, las empresas y el gobierno. Se ha aprendido mucho acerca de la creación de redes en los últimos 40 años. Este conocimiento debe ser la base para el diseño de la próxima generación de Internet: la Internet 3.0[2].

1.1.2. Caracteristicas necesarias para la siguiente Arquitectura de Internet

Algunas de las características que podrían ayudar a eliminar algunos de los problemas que enfrentan los usuarios actuales de Internet son las siguientes:

a) Comunicación eficiente

La arquitectura actual de Internet requiere un origen y un destino para para que la comunicación tenga lugar. Todos los paquetes recibidos cuando el destino está apagado se eliminan. Con los dispositivos inalámbricos, esta restricción se relajó al permitir que las estaciones base almacenen los paquetes mientras que el dispositivo este apagado o no disponible. La comunicación eficiente se debe generalizar o extender a los dispositivos con cable.

b) Separación de la identidad y la dirección.

En Internet actual un sistema se identifica por su dirección IP. Como resultado, cuando un sistema cambia su punto de unión, la dirección cambia. Esto hace difícil llegar a los sistemas móviles. Este es un problema bien conocido y un número de intentos y propuestas se han hecho en el pasado para resolver este problema - incluyendo IP móvil, Infraestructura de Internet Indirección, el protocolo de identidad de anfitrión y otros.

c) Conocimiento de ubicación

Las direcciones IP no están relacionados con la ubicación geográfica. Esto se puede considerar la fuerza de IP. Sin embargo, una gran parte de las aplicaciones de transferencia de información, como cualquier otro sistema de transporte, requiere encontrar el servidor

más cercano. Además, los nodos móviles necesitan saber su ubicación. La siguiente generación de Internet debe permitir que el receptor decida por la privacidad de su ubicación.

d) Comunicación Persona – Persona

El Internet ha sido diseñado para la comunicación del equipo. Pero el verdadero objetivo de la comunicación es a menudo un ser humano. Una persona puede ser alcanzable por un ordenador, un ordenador portátil, un teléfono celular o un teléfono con cable. El objetivo es llegar a la persona y no al ordenador, portátil o los teléfonos. Puesto que la persona no tiene una dirección IP, los usuarios se ven obligados a elegir una parada intermedia como el destino de nuestra comunicación en lugar del verdadero destino de la persona.

e) Seguridad

Los problemas de seguridad de Internet actual son bien conocidos. Es necesario que la próxima generación permite la opción de autenticación de fuentes / destinos / sistemas intermedios, la privacidad de la ubicación, la privacidad de los datos, y las garantías de integridad de datos.

En este contexto han surgido varias propuestas que pretenden dar soluciones a los problemas del Internet de hoy en día. Propuestas que han sido clasificadas en dos tipos, 1) Evolutivas y 2) Borrón y cuenta nueva. Estás propuestas son presentadas en el siguiente apartado de manera breve para dar a conocer el trabajo que hoy en día se está haciendo para el futuro internet.

1.2. Arquitecturas para el Internet del futuro

Para investigadores como John Day las soluciones con que se han resuelto algunos de los problemas anteriores, no son más que paliativos de corto plazo que aumentan la complejidad de la red, y la hacen cada vez más difícil de gestionar y mantener, aumentando su fragilidad. Hay una opinión creciente de que es necesario crear tecnologías que vayan más allá de lo que ofrece IP o IPv6. Investigadores como David D. Clark considera que "es tiempo de repensar la arquitectura básica de Internet, de comenzar con un diseño fresco e igual de importante, una estrategia para verificar su viabilidad tal que permita una oportunidad de implementación" (Talbot, 2005).

Actualmente existen dos métodos de diseño para el desarrollo de una arquitectura para el Internet del futuro;

1) Método de "Borrón y Cuenta nueva (Clean Slate)
2) "Evolutivo" (Evolutionary).

Desde el punto de vista del método "Clean Slate", la arquitectura se ha diseñado desde cero; para el método "Evolutionary" se añaden nuevos componentes de diseño a la arquitectura ya existente[3].

La selección de siete arquitecturas (Como se observa en la tabla 1) de red es con base en la madurez del proyecto, además, se cuenta con una demostración o un prototipo. Dentro del método de "Clean Slate" se encuentran NENA, SONATE y RINA, el resto de ellas pertenecen al método "Evolutionary". Para implementar una arquitectura de Internet, es un requisito previo saber si cumplen o no los requisitos de seguridad[4].

Arquitecturas para el futuro Internet							
Criterios	**XIA**	**RINA**	**SONATE**	**NENA**	**MobilityFirst**	**NEBULA**	**NDN**
Inicio del Proyecto	2010	2010	2009	2009	2010	2010	2010
Demo	✓	✓	✓	✓	✓	✓	x
Prototipo	✓	✓	✓	✓	✓	x	✓

Tabla 1. Arquitecturas para el Internet del futuro.

1.2.1. Service Oriented Network Architectures (SONATE)

SONATE es un acrónimo de "Arquitectura de Red orientado al Servicio". En SONATE se seleccionan los servicios prestados por la construcción de bloques (la implementación de un protocolo o un mecanismo como CRC, retransmisión, etc) son seleccionados y compuestos por una composición de algoritmos para crear un protocolo gráfico durante el tiempo de ejecución con base en los requerimientos de la aplicación[5].

Los mecanismos de seguridad en SONATE son:

1. SONATE es capaz de seleccionar bloques de construcción que:
a) Habilitan el cifrado de datos (Por ejemplo, el cifrado de datos micro protocolo)
b) Proporcionar autenticación de datos (Por ejemplo, firmas digitales, MAC)
c) Proporcionar un servicio de control de flujo

2. Cada sesión de comunicación está obligado por un protocolo gráfico.
SONATE no es capaz de mitigar el ataque de "Análisis de tráfico" y el ataque de "Masquerading" ya que SONATE no proporciona comunicación anónima y un atacante sólo necesita conocer el puerto y la dirección del objetivo para iniciar ambos ataques.

SONATE tampoco puede mitigar ataques de repudio, ya que no tiene una tercera parte de confianza para demostrar que la comunicación entre dos usuarios se ha terminado, por lo tanto, no puede crear un mensaje de no repudio (Como se observa en la tabla 2).

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	Cifrado de Datos
	Análisis de Tráfico	Ninguno: SONATE no proporciona comunicación anónima
Integridad	Modificación	Firma Digital
	Repudio	Ninguna: SONATA no tiene una tercera parte de confianza para proporcionar mensajes de no repudio
Disponibilidad	Denegación de Servicio	Control de Flujo
Autenticación	Hombre en el medio	Firma Digital
	Ataque de reflexión	Firma Digital
	"Masquerading"	Ninguno: SONATE no proporciona comunicación anónima
	"Replaying"	Un protocolo gráfico por sesión

Tabla 2. Arquitectura SONATE.

1.2.2. NENA

En NENA, los servicios proporcionados por los bloques construidos (La implementación de un protocolo o mecanismo como CRC, retransmisión, etc.) son seleccionados y compuestos por una composición de algoritmos para crear un protocolo gráfico (llamado netlet) durante el tiempo de diseño (por un desarrollador o asistido por un software) asumiendo los requisitos de una aplicación, las limitaciones del administrador y redes[6].

Los mecanismos de seguridad de NENA son:
1. NENA utiliza implementaciones seguras de protocolos. Cada protocolo un identificador de protocolo único.
2. NENA tiene un mecanismo de colaboración de detección de ataques
3. Similar a SONATE, NENA es capaz de seleccionar un protocolo que ofrezca:
a) Cifrado de Datos
b) Autenticación de datos

NENA no es capaz de prevenir el análisis de tráfico y ataques "Masquerading" ya que NENA no proporciona comunicación anónima y un atacante sólo necesita conocer el puerto y la dirección del objetivo que pretende ser atacado. Además, NENA no puede evitar el ataque de repudio, ya que no tiene un servidor tercero de confianza (Como se observa en la tabla 3)

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	Cifrado de Datos Netlet
	Análisis de Tráfico	Ninguno: NENA no proporciona comunicación anónima
Integridad	Modificación	Firma Digital Netlet y Protocolo de Señalización
	Repudio	Ninguna: NENA no tiene una tercera parte de confianza
Disponibilidad	Denegación de Servicio	Detección Colaborativa de Ataque
Autenticación	Hombre en el medio	Firma Digital Netlet
	Ataque de reflexión	Firma Digital Netlet
	"Masquerading"	Ninguno: NENA no proporciona comunicación anónima
	"Replaying"	Timestamp Netlet

Tabla 3. Arquitectura NENA.

NENA es capaz de evitar otros ataques mediante el uso de implementaciones seguras de protocolos, detección de colaboración de ataques o seleccionando un Netlet adecuado para contrarrestar los ataques. Ejemplos de Netlets que pueden ser utilizados para contrarrestar los ataques son el cifrado de datos y las firmas digitales Netlet. Mientras tanto, para mitigar el ataque de denegación de Servicios NENA utiliza el servicio de colaboración de detección de ataques.

1.2.3. XIA (eXpressive Intenetwork Architecture)

Mientras que en el Internet, una dirección IP es utilizada tanto para dirigir el host y el contenido, XIA utiliza tres tipos principales de identificadores para recuperar el contenido: ID de contenido, ID de Host y un ID de Servicio. El ID de contenido, el hash de contenido, es utilizado para recuperar el contenido sin necesidad de conocer su localización. El ID de host, el hash de la llave pública, se utiliza para comunicarse con el host que proporciona el contenido. El ID de servicio, el hash de llave pública de servicio, se utiliza para obtener el servicio que proporciona el contenido[7].

Los mecanismos de seguridad en XIA son:

1. La arquitectura utiliza Contenido/ Host/ ID de Servicio (CID/HID/SID) para recuperar el contenido. CID es la hash de contenido, HID es la hash de llave pública del Host y SID es el hash de llave pública del servicio.
2. El LAP (Lightweight Anonimato y la privacidad) mecanismo de defensa que permite la comunicación anónima para evitar el seguimiento a distancia.
3. El mecanismo de defensa STRIDE asigna el ancho de banda disponible en una topología de árbol basado
4. La AKI (Infraestructura de Clave Responsable) es un mecanismo de defensa que proporciona procesos de autenticación de datos
XIA es capaz de mitigar todos los ataques revisados, excepto los ataques de "Replaying", porque XIA no tiene algún mecanismo que obligue a una sesión de comunicación (Como se observa en la tabla 4).

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	Llave pública de Servicio (Utilizado en SID) también puede ser utilizado por un mecanismo de cifrado.
	Análisis de Tráfico	Mecanismo LAP
Integridad	Modificación	Hashes Principales (CID/HID/SID)
	Repudio	Hashed Principales (CID/HID/SID) y un mecanismo AKI
Disponibilidad	Denegación de Servicio	Mecanismo STRIDE
Autenticación	Hombre en el medio	Mecanismo AKI
	Ataque de reflexión	Mecanismo AKI
	"Masquerading"	Mecanismo LAP y AKI
	"Replaying"	NONE: XIA no tiene algún mecanismo

Tabla 4. Arquitectura XIA.

Para ataques de "snooping" XIA los mitiga utilizando una llave pública y una llave privada de un servicio (SID) para hacer cifrado de mecanismos. Para otros ataques, XIA los mitiga utilizando "Hash" ID (CID/HID/SID) o mediante el uso de los mecanismos de defensa proporcionados por la arquitectura SCION (LAP, AKI o STRIDE son los mecanismos de defensa).

1.2.4. MobilityFirst

En este enfoque, el usuario final puede solicitar un servicio utilizando el "Huma- Readable Name (HRN)". La arquitectura de nomenclatura MobilityFirst tiene tres identificadores: dirección de red (NA), identificador único global (GUID), y HRN. Se asegura la movilidad mediante la separación de la información de ubicación de red llamado NA de su identidad llamado GUID. Al igual que con XIA, GUID hace hash del contenido en sí. MobilityFirst tiene dos servicios asignados: "Name Assignment Service" (NAS) y la

"Global Name Resolution Service" (BGN). El NAS se une a un HRN con GUID y el GNRS asigna GUID a NA. GNRS funciona como un directorio de ubicación de contenido, ya que se unen dinámicamente el nombre y la ubicación. Cuando el contenido está disponible en más de una ubicación, GNRS elige el contenido para el solicitante de la ubicación más cercana[8].

Los mecanismos de seguridad en MobilityFirst son:
1. Para recuperar el contenido, MobilityFirst utiliza "Identificador Único Global (GUID) que se asigna a cada contenido como una dirección. GUID es el resultado de hacer hash al contenido y puede ser utilizado como una llave pública para el mecanismo de cifrado.
2. MobilityFirst permite actualizar el direccionamiento frecuentemente con el uso de la función GNRS.
3. MobilityFirst utiliza un protocolo integrado que permite a los nombres de llave pública auto-certificarse.

MobilityFirst es robusto contra el ataque "snooping" porque tiene un mecanismo para cifrar los paquetes utilizando el nombre o GUID del contenido como identificador. Para el ataque modificación, MobilityFirst mitiga mediante la asignación de un GUID único a cada contenido. GUID es un hash de un contenido, por lo tanto, el receptor puede comprobar la exactitud del contenido. Para los ataques "man-in-the-middle", reflexión, MobilityFirst los mitiga mediante el uso de un protocolo que es capaz de autenticar a un usuario y que tiene un GUID único para cada contenido. Para los ataques DoS, MobilityFirst los evita mediante la utilización de la función GNRS para la actualización de direccionamiento. Para el ataque de repudio, MobilityFirst los mitiga generando un mensaje de "no repudio" generado por la PKI (Como se observa en la tabla 5)

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	GUID puede ser usado para cifrado de datos
	Análisis de Tráfico	Ninguna: MibilityFirst no puede proporcionar comunicación anónima.
Integridad	Modificación	Hashs GUID
	Repudio	Tiene un PKI para generar mensajes de "No repudio"
Disponibilidad	Denegación de Servicio	Actualización de direccionamiento con GNRS
Autenticación	Hombre en el medio	Auto-certificación de Llave pública
	Ataque de reflexión	Auto-Certificación de Llave Pública
	"Masquerading"	Auto-Certificación de Llave Pública y cada contenido tiene un único GUID
	"Replaying"	NINGUNO: MobilityFirst no tiene algún mecanismo para unir a una sesión de comunicación

Tabla 5. Arquitectura MobilityFirst.

Sin embargo, MobilityFirst no puede mitigar el "Análisis de tráfico" y el ataque de "Replaying" debido a los siguientes motivos:

1. MovilityFirst no puede mitigar el "Análisis de Tráfico" porque no tiene un mecanismo que permita la comunicación anónima.
2. El ataque de "Replaying" es posible llevarlo a cabo en MobilityFirst ya que no tiene un mecanismo para unir los mensajes con la sesión.

1.2.5. NDN

NDN define dos tipos de paquetes: uno es para solicitud (llamado paquete e interés) y otro es para "replay" (paquete de datos). El paquete de interés principalmente tiene dos campos: Nombre de Contenido y "nonce". El nombre de contenido identifica los datos a ser recuperados y "nonce" une cada sesión de comunicación. El paquete de datos lleva el nombre y el contenido de los datos, junto con la firma digital y la información firmada[9].

Los mecanismos de seguridad en NDN son:
1. Un cifrado de extremo a extremo se puede utilizar. Este es utilizado para cifrar los datos en NDN
2. Los paquetes de datos son firmados utilizando una firma digital

NDN es vulnerable a un ataque de análisis de tráfico, a pesar de que NDN es una red centrada en el contenido. Siendo contenido centrado no es suficiente para evitar que ataquen, se necesita un mecanismo tal como un mecanismo para ocultar el paquete,

y este mecanismo no es proporcionado por NDN. La investigación de un método para mitigar un ataque DDoS está en curso, por lo tanto, NDN es vulnerable a un ataque de DoS (Como se observa en la tabla 6).

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	GUID puede ser utilizado para cifrar los datos
	Análisis de Tráfico	Ninguno: MobilityFirst no puede proporcionar comunicación anónima
Integridad	Modificación	Hashed GUID
	Repudio	Tiene un PKI para proporcionar mensajes de no repudio
Disponibilidad	Denegación de Servicio	Actualización de direccionamiento por BGN
Autenticación	Hombre en el medio	Auto certificación de llave pública
	Ataque de reflexión	Auto certificación de llave pública
	"Masquerading"	Auto certificación de llave pública y cada contenido tiene un GUID único
	"Replaying"	Ninguno: MobilityFirst no tiene ningún mecanismo para obligar a una sesión de comunicación.

Tabla 6. Arquitectura NDN.

NDN es capaz de mitigar un ataque de "replaying" ya que tiene un "nonce" en su paquete de interés. Este "nonce" distinguirá el viejo y el nuevo paquete de interés. Para la modificación, el repudio, man-in-the-middle, y los ataques de "replaying", NDN los impide mediante el uso de una firma digital. Para el ataque de "masquerading", NDN lo evita a través de una firma digital y el uso de un nombre único para cada contenido.

1.2.6. NEBULA

NEBULA facilita los centros de datos en un entorno de nube para comunicarse de una manera fiable. NEBULA consiste de tres componentes: NEBULA Core (NCore), NEBULA Data Plane, y NEBULA virtual y técnicas de redes extensibles (NEVENT). NCORE interconecta los centros de datos utilizando un mecanismo de direccionamiento confiable. NDP es un plan de datos que proporciona control de acceso flexible y mecanismos de seguridad. NEVENT, un plano de control, es responsable de determinar las rutas de los paquetes para llegar al destino[10].

Los mecanismos de seguridad en NEBULA son:
1. Prueba de Consentimiento (PoC) mecanismo para autorizar un paquete y un camino.
2. Prueba de ruta (PoP) con el fin de asegurarse de que el paquete sólo siga el camino autorizado.
3. NEBULA utiliza token par a enlazar una sesión de comunicación autorizada

4. Hay un servidor en NEBULA que puede actuar como un tercero de confianza. Este servidor demuestra que la comunicación entre dos usuarios tuvo lugar realmente porque los usuarios que quieren enviar un paquete se pondrán en contacto con él para obtener el PoC.

NEBULA no fue diseñado para mitigar los ataques de "snooping" y Análisis de tráfico. Para mitigar estos ataques, mecanismos como el cifrado de extremo a extremo o direccionamiento de cebolla debe ser aplicado en la parte superior de NEBULA. NEBULA tampoco puede evitar el ataque de "masquerading" porque el atacante puede hacerse pasar por un usuario autorizado para conseguir la dirección de los usuarios (Como se observa en la tabla 7).

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	Ninguno: No proporciona ningún mecanismo de cifrado
	Análisis de Tráfico	Ninguno: No proporciona comunicación anónima
Integridad	Modificación	Prueba de consentimiento: Para autorizar los paquetes. Prueba de ruta: Para garantizar que los paquetes fluyen en el camino autorizado.
	Repudio	Hay un servidor de consentimiento como tercero de confianza.
Disponibilidad	Denegación de Servicio	Prueba de ruta: No hay paquetes no autorizados en una ruta
Autenticación	Hombre en el medio	Prueba de consentimiento: Para autenticar el emisor y el receptor de los datos
	Ataque de reflexión	Prueba de consentimiento
	"Masquerading"	Ninguno: No proporciona comunicación anónima
	"Replaying"	Un Token es utilizado para unir una sesión de comunicación

Tabla 7. Arquitectura NEBULA.

NEBULA es capaz de mitigar los otros ataques mediante el uso de la función de un PoC y un PoP que residen en el PND. Por otra parte, NEBULA es capaz de prevenir el ataque de repudio, ya que tiene un servidor de consentimiento como tercero de confianza para crear un mensaje de no repudio.

1.2.7. RINA

RINA es una arquitectura de red que trata de aplicar los principios generales de interconexión de procesos a las redes. Como tal, no trata de arreglar Internet con una capa más de soluciones sino que cambia su diseño de manera que esto permita superar

las limitaciones actuales, haciendo que las conexiones entre redes sean más fiables y predecibles.

RINA se basa en el principio de que la asignación de direcciones y otros aspectos de las operaciones de red deben ser automatizados para minimizar los errores humanos y permitir un mayor grado de verificación y validación. Se trata de abordar la causa fundamental de los problemas, en vez de inventar otros caminos complejos para intentar evitar sus consecuencias.

Con RINA, no hay un espacio único y global de direcciones, hay un espacio de direccionamiento por capa – DIF en la jerga de RINA-. Así, una mala configuración administrativa se trata como un apagón de una capa o DIF, de manera que si se estropea definitivamente el enrutamiento en una capa, entonces, las capas de encima encontrarían una ruta alternativa tan rápido como lo permita la señalización[11].

El principio básico de diseño de RINA es que "La comunicación entre redes sólo es una comunicación entre procesos (IPC)". IPC es una función que permite a dos procesos comunicarse (uno emisor y otro receptor) entre sí. Ejemplos de la función IPC son: Localización de procesos, determinar permisos, pasar información, programación, y gestión de la memoria. Por ejemplo, un proceso de aplicación de origen solicita un servicio utilizando el nombre del proceso de la aplicación de destino. Ellos se comunican entre sí mediante la utilización de los servicios de "Distributed IPC Facility (DIF)"[12].

Los mecanismos de seguridad en RINA son:
1. Todos los miembros in la misma DIF deben ser autenticados primero antes de que ellos puedan unirse.
2. Si el atacante está dentro de la DIF, el todavía necesita escanear todas los posibles ID de conexiones de punto final (CEP-id) del objetivo, y la probabilidad es 2exp16 (Dado que el CEP-id es de 16 bits).
3. RINA tiene un módulo SDU de protección que es capaz de proporcionar funciones como: funciones de cifrado, funciones de compresión, y función de detección de errores.
4. El CEP-ids en RINA son utilizados para distinguir entre una nueva y una vieja conexión de datos.
RINA es capaz de mitigar todos los ataques revisados, excepto los ataques de DoS (Como se observa en la tabla 8)

Objetivo de Seguridad	Ataque de Seguridad	Mecanismo para mitigar el ataque
Confidencialidad	Snooping	Módulo SDU de protección (Cifrado de Datos)
	Análisis de Tráfico	Módulo SDU de protección y Proceso de Autenticación de Miembros
Integridad	Modificación	Módulo SDU de protección (Función HASH)
	Repudio	NINGUNO: RINA no tiene un tercero de confianza
Disponibilidad	Denegación de Servicio	NINGUNO: El ataque desde el interior no puede ser mitigado
Autenticación	Hombre en el medio	Proceso de autenticación de miembros
	Ataque de reflexión	Proceso de autenticación de miembros
	"Masquerading"	Proceso de autenticación de miembros
	"Replaying"	Único CEP-id en cas sesión

Tabla 8. Arquitectura RINA.

Para los otros ataques RINA los mitiga mediante la utilización de la función del módulo de protección SDU, por el proceso de autenticación del IPC Manager y un único CEP-id que es asignado a cada usuario. El CEP-Id también se puede utilizar para distinguir las viejas y las nuevas comunicaciones. El CEP-Id puede mitigar el ataque "replaying" al distinguir los mensajes de una nueva sesión y los mensajes de una sesión anterior.

1.3. Administración de Procesos

Un proceso es un programa en ejecución. Un programa ejecutable es un conjunto de instrucciones y datos almacenados en un fichero. Cuando lo que tiene ese programa se carga en la memoria y se pone en ejecución se convierte en un proceso[13]

1.3.1. Procesos

Un concepto clave en todos los sistemas operativos es el proceso. Un proceso es en esencia un programa en ejecución. Cada proceso tiene asociado un espacio de direcciones, una lista de ubicaciones de memoria, un conjunto de recursos, que comúnmente incluye registros (el contador de programa y el apuntador de pila, entre ellos), una lista de archivos abiertos, alarmas pendientes, listas de procesos relacionados y toda la demás información necesaria para ejecutar el programa. En esencia, un proceso es un recipiente que guarda toda la información necesaria para ejecutar un programa.

1.3.2. Jerarquía de procesos

Los Sistemas Operativos deben disponer de una forma de crear y destruir procesos cuando se requiere durante la operación, teniendo además presente que los procesos pueden

generar procesos hijos mediante llamadas al Sistema Operativo Los Procesos se manejan en forma jerárquica: cada proceso es lanzado desde un proceso "padre", dicho proceso se le llama proceso "hijo". Entonces podremos deducir que todos los procesos son hijos del proceso padre por excelencia: el proceso init.
Una vez que se crea un proceso, empieza a ejecutarse y realiza el trabajo al que está destinado. Sin embargo, nada dura para siempre, ni siquiera los procesos. Tarde o temprano el nuevo proceso terminará, por lo general debido a una de las siguientes condiciones:

- Salida normal (Voluntaria)
- Salida por error (Voluntaria)
- Error Fatal (Involuntaria)
- Eliminado por otro proceso (Involuntaria La mayoría de los procesos terminan debido a que han concluido su trabajo. Cuando un compilador ha compilado el programa que recibe, ejecuta una llamada al sistema para indicar al sistema operativo que ha terminado. Esta llamada es "exit" en UNIX y "ExitProcess" en Windows.

1.3.3. Estado de un procesos

Durante su existencia pasa por una serie de estados discretos, siendo varias las circunstancias que pueden hacer que el mismo cambie de estado. Debido a ello se puede establecer una "Lista de Listos" para los procesos "listos" y una "Lista de Bloqueados" para los "bloqueados".

La "Lista de Listos" se mantiene en orden prioritario y la "Lista de Bloqueados" está desordenada, ya que los procesos se desbloquean en el orden en que tienen lugar los eventos que están esperando.

Al admitirse un trabajo en el sistema se crea un proceso equivalente y es insertado en la última parte de la "La Lista de Listos". La asignación de la CPU al primer proceso de la "Lista de Listos" se denomina "Despacho", que es ejecutado para una entidad del Sistema Operativo llamada "Despachador".

El "Bloqueo" es la única transición de estado iniciada por el propio proceso del usuario, puesto que las otras transiciones son iniciadas por entidades ajenas al proceso.

La manifestación de un proceso en un Sistema Operativo es un "Bloque de Control de Procesos" (PCB) con información que incluye:

- Estado actual del Proceso
- Identificación única del Proceso
- Prioridad del Proceso
- Apuntadores para localizar la memoria del Proceso
- Apuntadores para asignar recursos
- Área para preservar registros

Cuando el sistema operativo cambia la atención de la CPU entre los procesos, utiliza las áreas de preservación del PCB para mantener la información que necesita para reiniciar el proceso cuando consiga de nuevo la CPU. Los Sistemas que administran los procesos deben poder crear, destruir, suspender, reanudar, cambiar la prioridad, bloquear, despertar y despachar un proceso. La "creación" de un proceso significa:

- Dar un nombre al proceso
- Insertar un proceso en la lista del sistema de procesos conocidos
- Determinar la prioridad inicial del proceso
- Crear el bloque de control del proceso
- Asignar los recursos iniciales del proceso

Un proceso puede "crear un nuevo proceso", en cuyo caso el proceso creador se denomina "proceso padre" y el proceso creado "proceso hijo" y se obtiene una "estructura jerárquica de procesos". La destrucción de un proceso implica:

- Borrarlo del sistema
- Devolver sus recursos al sistema

- Purgarlo de todas las listas o tablas del sistema
- Borrar su bloque de control de procesos

Un proceso "suspendido" no puede proseguir hasta que otro proceso lo reanude. Reanudar (reactivar) un proceso implica reiniciarlo en el punto donde fue suspendido.

La "destrucción" de un proceso puede o no significar la destrucción de los procesos hijos, según el sistema operativo. Generalmente se llama "Tabla de Procesos" al conjunto de información de control sobre los distintos procesos[14]

1.4. Arquitectura RINA

RINA es una arquitectura de red que propone que la comunicación entre redes puede verse como un conjunto de capas recursivas que proporciona un servicio de comunicación entre procesos distribuidos con un alcance determinado

La comunicación entre redes en RINA tiene como principio fundamental únicamente la comunicación entre procesos. El elemento principal del modelo de red es una única capa conocida como "DIF" (Distributed IPC Facility) que se utiliza simplemente para agrupar aplicaciones de procesos en diferentes sistemas. Una DIF puede ser vista como una capa, pero no en el mismo sentido como en la arquitectura TCP/IP. En RINA todas las capas utilizan el mismo protocolo para ejecutar un conjunto coordinado de mecanismos de política gestionada y alcanzar el servicio IPC deseado, asignándolo de la mejor manera posible los requisitos de las aplicaciones de usuario [15]

En la figura 1.1 se muestra la arquitectura RINA con tres niveles de capas DIFs. Cada capa proporcionando servicios IPC sobre un cierto alcance. El primer nivel DIF opera por encima del medio físico y sus políticas son optimizar el trato con las características del medio físico. Este primer nivel de DIF proporciona servicios IPC al segundo nivel de capa DIF, el segundo y tercer nivel operan de la misma manera[16].

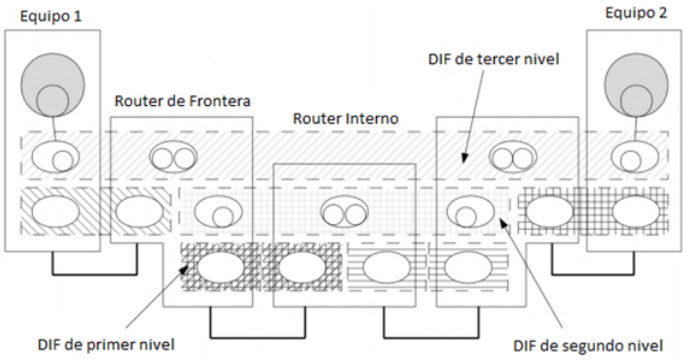


FIGURA 1.1: Arquitectura RINA

1.4.1. Comunicación en RINA

En un Sistema Operativo para permitir a dos procesos comunicarse, la funcionalidad de IPC requiere ciertas funciones como localización de los procesos afectados, determinación de permisos, transferencia de información, aprovisionamiento de recursos y administración de la memoria. El conjunto de componentes del Sistema Operativo que realizan esta función podría ser consecuentemente llamado IPC Facility

Una DIF es una estructura organizativa que agrupa procesos de aplicaciones que proveen servicios de IPC y está configurada mediante ciertas políticas específicas. Según esta visión, la comunicación entre redes no se trata de una serie de capas que agrupan distintas funcionalidades, sino de una única capa de IPC distribuido que se repite con distintos alcances ofreciendo los mismos mecanismos o funcionalidades, pero especialmente configuradas mediante un conjunto de políticas propias para operar a distintos niveles de rendimiento (capacidad, pérdidas, retrasos, etc). En esencia, una DIF es únicamente una aplicación distribuida, cuyos miembros, procesos de aplicaciones llamados procesos IPC, se especializan en ofrecer servicios IPC distribuidos. De esta manera, una DIF no es estructuralmente distinta a cualquier otra aplicación distribuida, únicamente se centra en realizar bien una única tarea [17].

Siendo parte de los elementos y tareas más importante que se llevan a cabo dentro de una DIF las siguientes actividades:

Nombres y direcciones en RINA

En RINA todos los procesos de aplicaciones (incluidos los IPC Processes) tienen un nombre único que los identifica en el espacio de nombres de aplicaciones. Con tal de

facilitar su operación dentro de la DIF, cada "IPC Process" en una DIF obtiene un sinónimo que puede tener significado dentro de la DIF una dirección (El abasto de la dirección es la propia DIF, no son visibles fuera de la DIF)
Cada DIF tiene un directorio que asigna el nombre de la aplicación de destino al nombre del IPC Process en una DIF mediante el cual se llega a la aplicación.

La capa Shim DIF

Es una capa que se posiciona sobre una capa no RINA (p.ej. Cable, Internet, Ethernet) y presenta una API RINA (quizá parcial) para que la aplicación encima pueda tratarla como una DIF normal.

Las principales responsabilidades de una Shim DIF son:
i) Asignar nombres de aplicaciones de la capa N+1 a direcciones dentro de la DIF;
ii) Crear y destruir flujos (flows) dentro de la Shim DIF (El flow es un recurso de comunicación bien definido en la DIF, por ejemplo: conexión TCP, UDP, conjunto de tramas Ethernet con las mismas MACs de origen/destino)

El manejador de procesos (IPC Manager Daemon)

El IPC Manager es el coordinador y administrador de los diferentes componentes de RINA, tanto en el espacio de usuario (user-space) como en el kernel. La librería librina-ipc-manager encapsula todas las operaciones de administración locales disponibles para el IPC Manager. Estas operaciones se pueden dividir en:

- Administración de procesos IPC locales - Creación/destrucción de procesos IPC.
  - Asignación de proceso IPC a DIFs dando información suficiente al IPC process para que pueda empezar a operar como miembro de la DIF.
  - Registro de procesos IPC a DIFs N-1 permitiendo a un IPC process de nivel N ser accesible vía una o más DIFs de N-1.
  - Forzar un proceso IPC a hacer "enrollment" con un proceso IPC remoto.
  - El proceso IPC local puede ser o no ser parte de la DIF.
  - Inspeccionar el RIB de un proceso IPC

- Servicios a procesos de aplicaciones - Redireccionamiento de peticiones de “flows” al “daemon” del proceso IPC más adecuado. En esta situación, o bien el IPC Manager cuenta con información suficiente para realizar esta acción, o bien debe consultar al IDD para identificar la DIF correcta para usar.
  - Redireccionamiento de peticiones de registro de las aplicaciones al proceso IPC que es miembro de la DIF objetivo.
  - Listado de DIFs en el sistema disponible para una cierta aplicación junto con sus características.

- Creación de procesos IPC
  Cuando el IPC Manager daemon necesita crear un nuevo proceso IPC utiliza la operación ïpcm-create”de librina-ipc-manager. Esta operación supone la invocación de dos funciones: I) la ”syscall-system”para instanciar un nuevo proceso en el espacio de usuario y II) ïpc-process-create”, una llamada a sistema que será atendida en el kernel por el KIPCM y creará las estructuras necesarias en el kernel para el nuevo proceso IPC.

- Destrucción de procesos IPC
  Para destruir un proceso IPC, el IPC Manager daemon invoca ipcm-destroy de librina-ipc-manager. Esta operación supone la invocación de varias funciones: la syscall kill para enviar la orrespondiente señal al proceso en user-space, y la syscall ipc-process-destroy, que será atendida en el kernel por el KIPCM quien destruirá las estructuras en el kernel del proceso IPC.

- Asignación de un proceso a una DIF
  Este proceso hace que un IPC process anteriormente creado se enliste como miembro de una DIF, aportando al nuevo miembro toda la información necesaria para operar en dicha DIF. El IPC Manager invoca ipcm-assign de la librería pasando el ID del IPC process y la información de la DIF. Esto genera un mensaje assign-request hacia el IPC process daemon. El IPC process daemon rellena su RIB con la información y devuelve un mensaje assign-response.

- Registro de un proceso a una DIF N-1
  Después de crear un proceso IPC y registrarlo en una DIF, el siguiente paso es registrarlo en una N-1 DIF para que éste pueda ser accedido (igual que los procesos de aplicaciones). La acción la realilza el IPC Manager en nombre del IPC process a registrarse, hablando con aquellos IPC processes de las N-1 DIF a las que el IPC

process necesita registrarse.

La tarea de asignación

Todas las comunicaciones experimentan tres fases: Asginacion de Flujo, Reserva (establecimiento) y Transferencia. La asignación de flujo es el proceso vía el cual un proceso IPC se conecta a una DIF y recibe información suficiente para empezar a operar como miembro.

Comienza cuando el proceso IPC establece una conexión de aplicación con otro proceso IPC que ya es parte de la DIF. Durante la fase de establecimiento de la conexión, puede pasar que el proceso IPC que se conecta sea requerido a autenticarse si las políticas de seguridad de la DIF así lo establecen. Una vez establecida la conexión se le pasa al proceso IPC la información de la DIF que necesita (asignación de dirección, políticas, etc).

Asignación de flujos (Flow Allocator FA)

El FA es el componente responsable de administrar los flujos (flows) y su ciclo de vida: reserva, monitoreo, y destrucción. Las tareas del FA son:

- Encontrar el proceso IPC vía el cual la aplicación de destino es accesible.
- Asignar las políticas de QoS requeridas con el correspondiente flow.
- Negociar la reserva del flow con el FA del IPC process de destino (control de acceso, políticas, etc).
- Liberar los recursos reservados para el flow cuando éste es terminado o la aplicación termina inesperadamente.

A diferencia de TCP, en RINA, los flows y las conexiones son independientes. El FA reserva puertos y crea flows. Los port-IDs son los extremos del flow, únicos en el sistema. Un flow consiste en: i) el binding local entre el port-ID de origen y destino y el punto de conexión final (connection end point-ID -CEP-ID) de origen y destino respectivamente; y ii) la conexión o el potencial de conexión entre instancias de EFCP (DTP/DTCP).

Protocolo de control de flujo y error (Error and Flow Control Protocol-EFCP)

EFCP es el protocolo de transmisión para la comunicación entre dos procesos IPC. El protocolo asegura fiabilidad, orden y control de flow si es requerido por el QoS. EFCP está basado en Delta-T, diseñado por Richard Whatson.

La cadena de octetos intercambiados entre dos máquinas de estado es una Protocol Data Unit (PDU), la cual está compuesta por una Protocol Protocol Control Information (PCI), entendida por la DIF; y la Service Data Unit (SDU), que contiene la información de usuario que no es entendida por la DIF y es pasada al usuario.
La máquina de estados de EFCP consiste en dos submáquinas con binding débil entre sí:

- 1) La máquina de estados de Data Transfer Protocol (DTP)
  Realiza la transmisión de PDUs, los mecanismos de binding fuerte como secuenciación, fragmentación/reensablado y concatenación/separación. Cada instancia de flow cuenta con una instacia de DTP.

- 2) La máquina de estados de Data Transfer Control Protocol (DTCP)
  Realiza los mecanismos de binding débil (feedback) como: retransmisión y control de flow. Una instancia por flow es creada si éste requiere retransmisión o control de flow. DTCP utiliza los números de secuencia de las PDUs y tres timers para asegurar sincronismo con la instancia peer de DTCP: MPL, tiempo máximo que el receptor mantendrá una PDU antes de enviar un ack (A), y máximo tiempo que un emisor esperando un ack intentará la transmisión antes de desistir (R).
  Siendo esto los elementos principales que participan en la comunicación de los IPC Process dentro de una capa DIF. El prototipo se presenta como un contenedor seguro de los elementos antes mencionados, motivo por el cual es necesario evaluar la seguridad que presenta el prototipo a través de alguna metodología de seguridad[18].

Capítulo 2

Evaluaciones de seguridad para arquitecturas de comunicación

La seguridad en las tecnologías de la información y comunicaciones, se hace tan indispensable como su funcionalidad misma. Preservar la disponibilidad, integridad y confidencialidad, de sus datos y operaciones, es un reto que se hace cada día más complejo, por su misma evolución y los riesgos que cada día se vuelven más sofisticados, al estar cada vez los usuarios mejor conectados y menos controlados.

Desde la consolidación de Internet como medio de interconexión global, los incidentes de seguridad relacionados con sistemas informáticos vienen incrementándose de manera alarmante. Este hecho, unido a la progresiva dependencia de la mayoría de las organizaciones hacía sus sistemas de información, vienen provocando una creciente necesidad de implantar mecanismos de protección que reduzcan al mínimo los riesgos asociados a los incidentes de seguridad.

2.1. Evaluación de seguridad

Una evaluación de seguridad constituye un método eficaz para garantizar un nivel de seguridad superior en el sistema, ya que en éstas se tiene en cuenta la seguridad en forma interna. Permite detectar e identificar problemas, vulnerabilidades y debilidades en un determinado sistema. Los sistemas tecnológicos son vulnerables en temas muy distintos y diversos, desde servidores, notebook, estaciones de trabajo, firewall perimetral, bases de datos, redes wifi, usuarios móviles, etc. Determinar cada vulnerabilidad para cada

uno de estos sistemas requiere de técnicas específicas, herramientas, conocimiento y experiencia en los sistemas [19].

Una evaluación de seguridad consiste en probar los métodos de protección del sistema de información sometiendo el sistema a una situación real. Generalmente, se utilizan dos métodos:

- Caja negra: El cual consiste en intentar lograr una intrusión en la red sin tener conocimiento del sistema para generar una situación realista.
- Caja blanca: Consiste en intentar lograr una intrusión en el sistema conociéndolo por completo para poner a prueba al máximo los límites de seguridad de la red

La evaluación consiste de pruebas ofensivas contra los mecanismos de defensa existentes en el entorno que se está analizando. El objetivo de estas pruebas es verificar bajo situaciones extremas cuál es el comportamiento de los mecanismos de defensa, específicamente, se busca detectar vulnerabilidades en los mismos. Además, se identifican aquellas faltas de controles y las brechas que pueden existir entre la información crítica y los controles existentes.

Una evaluación de seguridad comprende múltiples etapas con diferentes tipos de actividades en distintos ámbitos y entornos.

- Fase de reconocimiento: Posiblemente, la etapa que más tiempo demanda. Se definen los objetivos y se recopila toda la información posible que luego será utilizada a lo largo de las siguientes fases. La información que se busca abarca desde nombres, topología de la red, direcciones IP, entre otros. El tipo de información o la profundidad de la investigación dependerá de los objetivos que se hayan fijado.
- Fase de escaneo: Utilizando la información obtenida previamente se buscan posibles vectores de ataque. Esta etapa involucra el escaneo de puertos y servicios. Posteriormente se realiza el escaneo de vulnerabilidades que permitirá definir los vectores de ataque.
- Fase de enumeración: El objetivo de esta etapa es la obtención de los datos referente a los usuarios, nombres de equipos, servicios de red, entre otros.
- Fase de acceso: En esta etapa finalmente se realiza el acceso al sistema. Esta tarea se logra a partir de la explotación de aquellas vulnerabilidades detectadas que fueron aprovechadas por el auditor para comprometer el sistema.

- Fase de mantenimiento de acceso: Luego de haberse obtenido el acceso al sistema, se busca la manera de preservar el sistema comprometido a disposición de quien lo ha atacado.

Los sistemas o aplicaciones, por muy bien o mal protegidos que se encuentren, pueden ser objeto de una intrusión con o sin consentimiento en cualquier momento, por lo que es importante entender que descubrir las fallas de los mismos mediante el uso de las herramientas para ello, puede ser una gran ventaja a la hora de defenderse de futuros intentos de intrusión.

2.2. Metodologías de evaluación

Una metodología de evaluación es un conjunto de reglas y lineamientos para "¿Cuándo?", "¿Qué?" y "¿Cuáles?" eventos son evaluados. Una metodología es uno de los estándares profesionales más completos y comúnmente utilizados en auditorías de seguridad para revisar la seguridad de los sistemas desde Internet. Incluye un marco de trabajo que describe las fases que habría de realizar para la ejecución de la auditoría

Una metodología es un documento que refiere, de forma estandarizada y ordenada, las diversas verificaciones y pruebas que debe realizar durante el desarrollo de las auditorias y verificaciones de la seguridad.
Son pocas las evaluaciones que se pueden utilizar para evaluaciones de Arquitecturas de Red, para el análisis de este trabajo uno puede enfocarse en dos que pueden adaptarse para evaluar el modelo de red de RINA, estas son: 1) ATAM, y 2) SAAM

2.3. Evaluación con ATAM

ATAM es un método de arquitectura basada en escenarios para evaluar los atributos de calidad, tales como: el grado de modificación, portabilidad, extensibilidad, e integridad[20].

Los Factores claves en la metodología de ATAM son:

- La falta de métodos de evaluación que consideren el impacto de las decisiones en la arquitectura, como la disponibilidad, rendimiento, seguridad, cambios, facilidad de uso, etc.

El método de evaluación ATAM consiste de cuatro fases:

1) Presentación
2 Investigación y Análisis
3) Pruebas
4) Reportes.

Cada fase consiste de una colección de pasos. La fase de presentación consiste en el intercambio de información a través de presentaciones. La fase de investigación y análisis se refiere a la evaluación de los atributos claves de la calidad frente a los enfoques de la arquitectura. En la fase de prueba se comparan los resultados de la fase anterior con las necesidades de las partes interesadas. Por último, en la fase de presentación de informes se resumen los resultados obtenidos al aplicar ATAM[21].

Para un mayor conocimiento de la metodología de ATAM, se presenta una descripción más de tallada de las fases que comprenden la metodología.

**FASE DE PRESENTACIÓN

- Paso 1. Presentación de ATAM. Inicialmente, el líder del grupo de evaluación describe ATAM a los participantes.
- Paso 2. Presentación del Negocio Un portavoz del proyecto describe los objetivos del negocio.
- Paso 3. Presentación de la Arquitectura El arquitecto describe la arquitectura de software del sistema.

**FASE DE ANÁLISIS E INVESTIGACIÓN

- Paso 4. Identificación de los objetivos de la Arquitectura Identificar los enfoques de la arquitectura del paso anterior, pero no se analizan todavía.
- Paso 5. Generar Atributos de calidad a través de un árbol de utilidades. Se generan los atributos de calidad que conforman la ütilidad"del sistema.

- Paso 6. Análisis de los objetivos de la Arquitectura Analizar los enfoques de la arquitectura con base en escenarios de alta prioridad, los cuales han sido identificados en el paso anterior, los enfoques arquitectónicos que abordan esos escenarios se generan y analizan.

**FASE DE PRUEBAS

- Paso 7. Lluvia de ideas y prioridad de escenarios Durante una sesión de lluvia de ideas los interesados proporcionan un gran grupo de escenarios. El equipo de ATAM, junto con el equipo de interesados dan una prioridad a los escenarios mediante una votación.
- Paso 8. Nuevo análisis de objetivos de la Arquitectura Se vuelve a realizar un análisis de los enfoques de la arquitectura. Los escenarios priorizados del paso anterior se utilizan como entrada para iteraciones del paso seis. Este conjunto de escenarios son los más importantes. El objetivo es identificar y documentar cualquier otro enfoque de la arquitectura, cuáles son riesgos y cuáles no lo son, a esto se llama puntos de sensibilidad.

**FASE DE REPORTE

- Paso 9. Presentación de Resultados En la última fase, con base en la información recopilada durante las tres primeras fases de la sesión ATAM, el equipo de evaluación resume y presenta los resultados a las partes interesadas.

Resultados y Fortalezas de ATAM

Las fortalezas generales de una sesión ATAM son:

- Existe una mayor comprensión por parte de las personas interesadas Mejora de la documentación de la arquitectura de software. En algunos casos, la documentación de la arquitectura debe ser recreada. Mejora de la comunicación de las partes interesadas. En términos de resultados prácticos ATAM ofrece:
- Escenarios de calidad producidos por las partes interesadas con base en los atributos de calidad los requisitos.

- Resultados con base en la calidad de los escenarios y casos de uso.
- Clasificación de los atributos de calidad, que proporcionan evaluadores con un catálogo de los parámetros de la arquitectura y estímulos apropiados para rastrear diferentes atributos de calidad y sus interdependencias

2.4. Evaluación con SAAM

SAAM es el primer método de análisis de arquitectura que se basada en escenarios. Fue creado para evaluar el grado de modificación de las arquitecturas. Los creadores de SAAM buscaron un método capaz de expresar las diferentes demandas de calidad de las arquitecturas de software (como la modificación, flexibilidad, facilidad de mantenimiento, etc.) por medio de escenarios y evaluarlos contra los reales[22].

En la práctica SAAM ha demostrado ser útil para evaluar rápidamente muchos atributos de calidad como el nivel de modificación, portabilidad, extensibilidad, integridad, así como la cobertura funcional.
El método también se puede utilizar para evaluar los aspectos de calidad de arquitecturas de software tales como el rendimiento o la fiabilidad. Si se analiza una sola arquitectura, SAAM indica los puntos débiles o fuertes, junto con los puntos de donde la arquitectura no cumple con sus requisitos modificados[23].

Factores claves en el desarrollo de SAAM

El desarrollo de SAAM fue motivado por una variedad de opiniones sobre las arquitecturas de software y la falta de métodos y bases común para hacer frente a ellos. En consecuencia, los atributos comunes de calidad como el grado de modificación, flexibilidad o de mantenimiento, no se asociaron con artefactos de software directos que pueden ser analizados y medidos.

Pasos en una sesión de evaluación SAAM

El método consiste en seis pasos principales, que normalmente están precedidos por una breve descripción del contexto general de negocios y requieren la funcionalidad del sistema.

- Paso 1. Desarrollo de escenarios. El primer paso en una sesión SAAM es una lluvia de ideas con el fin de identificar el tipo de actividades que el sistema puede

soportar. Estas actividades, junto con las posibles modificaciones que las partes interesadas puedan anticipar se agrupan en los llamados escenarios del sistema

- Paso 2. Describir la arquitectura. En el segundo paso de la sesión SAAM son presentados los candidatos de la arquitectura. Las notaciones arquitectónicas utilizadas deben ser bien entendidas por los participantes y deberán indicar la representación estática del sistema (componentes, sus interconexiones y la relación con el medio ambiente), así como el comportamiento dinámico del sistema.

- Paso 3. Clasificación y Prioridad de los escenarios En este punto del análisis los escenarios son clasificados en escenarios directos y escenarios indirectas (sus equivalentes en notación UML son los casos de uso)

- Paso 4. Evaluación individual de los Escenarios Indirectos En caso de un escenario directo el arquitecto demuestra cómo el escenario sería ejecutado por la arquitectura. Para el caso de un escenario indirecto el arquitecto describe cómo la arquitectura necesitaría ser cambiada para adaptarse a la situación.

- Paso 5. Evaluar la Interacción del Escenario Cuando dos o más escenarios están solicitando cambios a lo largo del mismo componente(s) de la arquitectura, se dice que interactúan.

- Paso 6. Crear una evaluación Global Finalmente se le asigna un peso a cada escenario en términos de su importancia relativa para el éxito del sistema.

Resultados y Fortalezas de SAAM

Los puntos fuertes de SAAM son

- La comprensión a profundidad de las partes interesadas acerca de la arquitectura que se está analizando.

- En algunos casos, después de una sesión de evaluación la documentación del software de la arquitectura es mejorado.

- Mejora la comunicación entre las partes interesadas.

Observaciones sobre SAAM

- La generación de escenarios es con base en la visión de las partes interesadas. Se necesita un esfuerzo muy pequeño por parte de las partes interesadas para imaginar cualquiera de los "escenarios indirectos"
- SAAM no proporciona una métrica clara de calidad para que analicen los atributos de la arquitectura.
- En SAAM se especifica que debe haber una arquitectura candidata, lo que "debería ser descrito en una notación arquitectónica que sea bien entendido por las partes"
- El equipo de evaluación se basa únicamente en la experiencia de los arquitectos en la propuesta de diferentes arquitecturas (si las hay).
- SAAM es un método paso a paso para realizar el análisis de la arquitectura de software. Sin embargo, proporciona algunas técnicas para la realización de las diferentes etapas, basándose principalmente en la experiencia de analista del evaluador.
- SAAM no faculta al equipo para una preparación previa con el fin de facilitar una posible sesión de SAAM, por tanto, tendrá un gran esfuerzo para el equipo de evaluación ser aceptada por los arquitectos de sistemas o diseñadores.

En resumen, de estos métodos de evaluación se pueden destacar características como se observa en la tabla 9.

Método	Calidad Evaluada	Métricas y Herramientas Soportadas	Descripción del Proceso	Fortalezas	Debilidades	Tipo del Sistema al que es aplicable
SAAM	Modificación en la Arquitectura	Clasificación de escenarios (Directo vs Indirecto)	Razonable	- Identifica las áreas de alta complejidad - Abierto para la descripción de cualquier arquitectura	* No es un indicador claro de calidad * No es compatible con las técnicas para realizar las etapas	Todos
ATAM	Modificación en la Arquitectura	Puntos sensibles, Puntos de Compensación,	Bueno	- Generación de escenarios basados en los requisitos - Aplicables a propiedades estáticas y dinámicas	Se requiere conocimiento técnico detallado	Todos

Tabla 9. Metodologías SAAM y ATAM.

Siendo estás las metodologías base para generar una propuesta de solución, la cual consiste en el diseño de una metodología que sea parte del instrumento de evaluación con la cual se evalúa la implementación de RINA.

2.5. Propuesta de solución

Las metodologías antes mencionadas son aplicables a la arquitectura de red TCP/IP, siendo el modelo de trabajo de TCP/IP un modelo a capas, donde las capas están jerarquizadas. Cada capa se construye sobre su predecesora. El número de capas y, en cada una de ellas, sus servicios y funciones son variables con cada tipo de red. Sin embargo, en cualquier red, la misión de cada capa es proveer servicios a las capas superiores haciéndoles transparentes el modo en que esos servicios se llevan a cabo. RINA se trata de un modelo en el que sólo hay un tipo de capa y además es configurable. Una capa es una aplicación distribuida que proporciona servicios de comunicación entre procesos a través de un cierto ámbito (como un enlace punto a punto, una red local, una red regional, una red de redes...). Estas capas (denominadas DIFs en terminología RINA) son recursivas, ya que se proporcionan servicios la una a la otra, y se pueden utilizar tantas como el diseñador de red considere oportuno.

Siendo muy diferentes cada una de las arquitecturas, es necesario retomar, después de un previo análisis de cada una de las etapas, ciertas elementos de las metodologías para diseñar y adecuar una nueva metodología que sea aplicable al modelo de arquitectura de RINA.

De la propuesta de SAAM se pueden retomar los siguientes pasos:

a) Paso 1: Desarrollo de los escenarios:
b) Paso 2: Descripción de la arquitectura. Esta etapa se retoma o puede aplicar en RINA para dar una explicación de la arquitectura, sus
componentes y los elementos que serán evaluados.
c) Paso 3: Clasificación y prioridad de los escenarios: Para este trabajo, únicamente se cuenta con un solo escenario.
d) Paso 4: Evaluación individual de los escenarios De las fases que comprenden la metodología ATAM se puede retomar los siguientes pasos:

1) De la fase de Presentación:
- Presentación de la metodología

- Presentación de la arquitectura

2) De la Fase de "Análisis e investigación"
- Identificación de los objetivos de la arquitectura
- Análisis de los enfoques de la Arquitectura

3) De la fase de Presentación de resultados
- Presentación de resultados

El análisis previo de las metodologías antes mencionadas, da lugar a la propuesta de solución (Como se observa en la Tabla 10), la cual consta de cuatro fases con etapas diferentes en cada una de las fases

Fase	Actividad	Descripción
1. PRESENTACIÓN	Presentación de La Metodología de Evaluación	Alcances que se buscan alcanzar con La Evaluación de Seguridad
	Descripción de la Arquitectura	Descripción breve de la arquitectura
	Fase de Identificación de los objetivos	Identificación del Objetivo
2. ANÁLISIS E INVESTIGACIÓN	Análisis de los enfoques de la Arquitectura	Análisis de los datos con los que se cuenta para desarrollo de posibles ataques
	Diseño y descripción del escenario de pruebas	Diseño del escenario de pruebas
3. PRUEBAS	Ejecución individual de la evaluación en el escenario de pruebas	Ejecución de los ataques diseñados sobre el escenario de pruebas
4. RESULTADOS	Análisis de resultados	Análisis de la ejecución del o los ataques sobre el escenario de pruebas
	Presentación de resultados	Presentación final de los resultados

Tabla 10. Propuesta de solución.

Con apoyo de las metodologías ATAM y SAAM es la manera en que surge la metodología de evaluación que se aplica sobre la implementación de la arquitectura RINA. Metodología que va a será aplicable, para este trabajo, a un objetivo en específico, el cual busca evaluar la Integridad y la Disponibilidad.

La metodología es uno de los elementos que integran el instrumento de evaluación, el cual consta de cuatro etapas principales. Etapas a través de las cuales se lleva a cabo la recolección de información para buscar comprometer la integridad y la disponibilidad.

En el capítulo número tres se presenta una descripción más detallada de cada una de las etapas del instrumento de evaluación, el cual se encuentra:

1) Una metodología de evaluación (propuesta en el capítulo 2)
2) El diseño de escenarios de pruebas
3) Diseño de ataques para comprometer la Disponibilidad e Integridad del flujo de comunicación
4) Diseño y/o selección de herramientas para los ataques.

Un punto importante es el diseño de ataques y selección de herramientas o diseño de las mismas, para lo cual se requiere de una descripción detallada que busca arrojar la mayor cantidad de información posible y así detectar posibles vulnerabilidades para ser explotadas

Capítulo 3

Diseño de la evaluación de seguridad de RINA

3.1. Etapa de Presentación

Durante la etapa de presentación se hace referencia al tipo de prueba o pruebas que se llevan cabo, el tipo de evaluación a a ejecutar puede ser de caja blanca, gris o negra. Con base en el tipo de evaluación es la información con la que se cuenta para llevar a cabo la evaluación de seguridad.

Así como la definición de los alcances y objetivos que se buscan lograr de la evaluación. Con base en esto es la aplicación del instrumento de evaluación.

3.1.1. Presentación de la metodología

Es una evaluación de caja blanca que se aplica sobre una implementación básica de RINA en su versión 1.9 cuyo objetivo es una capa DIF que comunica dos aplicaciones a través de dos procesos.

Es objeto de estudio la comunicación de la aplicación “rina-echo-time (c)” con la aplicación “rina-echo-time (s)”. La comunicación se lleva a cabo a través de dos procesos situados en la misma DIF. Cada proceso se localiza en un sistema distinto y los sistemas están conectados a través de un canal de comunicación directo llamado “Normal.DIF”. El IPC Manager es el responsable de localizar el o los procesos, así como de asignar recursos para establecer el canal de comunicación (Como se observa en la Figura 3.1)

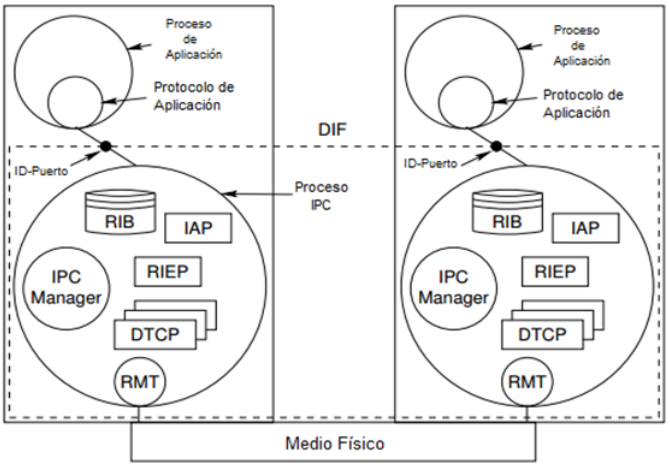


FIGURA 3.1: IPC Manager

Se busca evaluar dentro del flujo de información las propiedades de la seguridad de:

a) La Integridad: Integridad de la información se refiere a la protección de la información de ser modificada por personas no autorizadas. Corroborar que el flujo de la información entre dos entidades no se ha modificado, es decir, que los datos recibidos entre dos entidades son exactamente los mismos que fueron enviados sin que se haya producido ninguna modificación de la información transmitida

b) La Disponibilidad: El sistema se mantiene funcionando eficientemente y es capaz de recuperarseDisponibilidad de la información se refiere a garantizar que las partes autorizadas pueden acceder a la información cuando sea necesario. El sistema se mantiene funcionando eficientemente y es capaz de recuperarse rápidamente en caso de fallo

Objeto a evaluar

Esta sección hace referencia a la implementación básica de RINA con la cual se evalúan las propiedades de la seguridad de Integridad y Disponibilidad" (Como se observa en la figura 3.2).

El objetivo de evaluación consta de:

a) Dos aplicaciones, 1) "rina-echo-time (s)" y 2) "rina-echo-time (c)"
b) Una capa DIF de nombre "Normal.DIF"

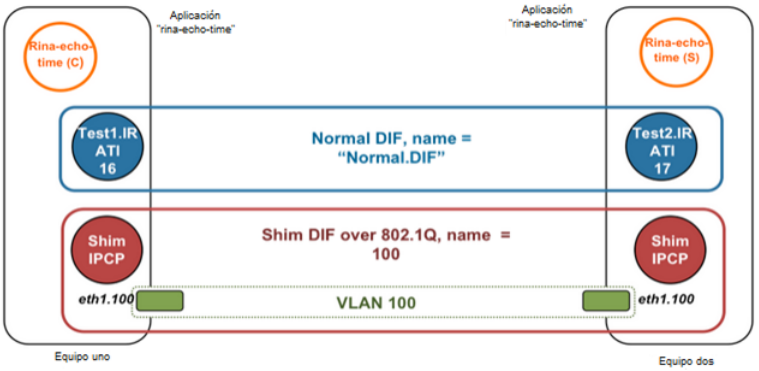


FIGURA 3.2: Objetivo a evaluar

c) Una capa Shim de nombre 100

A través de la aplicación de la metodología es posible determinará si la integridad y la Disponibilidad del flujo de información entre dos entidades en una implementación básica del modelo de RINA es vulnerable o no frente a los ataques propuestos en la sección 3.2.2.

3.1.2. Descripción de la arquitectura

En esta etapa se describe la arquitectura sobre la cual se va a aplicar la metodología de evaluación. Para esta evaluación se hace una descripción breve de la arquitectura de RINA, ya que la misma fue descrita más ampliamente en el capítulo uno del presente trabajo.

Arquitectura de RINA

Como se plantea en el capítulo uno, en el apartado dedicado al modelo de RINA, se plantea una estructura recursiva de capas que provee servicios de comunicación entre procesos a las aplicaciones de la capa superior (Como se observa en la figura 3.3).
Sólo existe un único tipo de capa que se repite tantas veces como decida el diseñador de la red y existe una separación de mecanismos y políticas.
Todas las capas tienen las mismas funciones con distinto rango. No todas las capas pueden necesitar todas las funciones, pero no requerirán más. Una capa es una aplicación distribuida que realiza y gestiona procesos IPC.

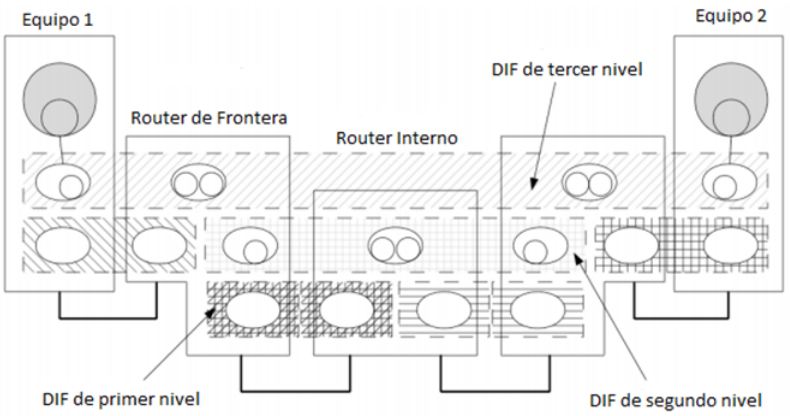


FIGURA 3.3: Modelo a capas de RINA

Nombres y Direcciones en RINA

Todos los procesos de aplicaciones tienen un nombre único que los identifica en el espacio de nombres de aplicaciones. Con tal de facilitar su operación dentro de la DIF, cada "IPC Process" en una DIF obtiene un sinónimo que puede tener significado topológico dentro de la DIF: una dirección

El abasto de direcciones es la propia DIF, no son visibles fuera de la DIF. Cada DIF tiene un directorio que asigna el nombre de la aplicación de destino al nombre del "IPC Process" en una DIF mediante el cual se llega a la aplicación. Para una DIF el nivel N, el proceso en la capa N+1 es una aplicación y el proceso en la capa N-1 es un PoA (Point of Attachment).

3.1.3. Identificación del objetivo

Posiblemente, esta sea una de las etapas que más tiempo demanda. Asimismo, se definen objetivos y se recopila toda la información posible que luego será utilizada a los largo de la siguiente fase. La información que se busca abarca desde nombres de aplicación y direcciones, hasta la topología de la red, entre otros. El tipo de información o profundidad de la recopilación de información depende del objetivo fijado.

Se ha hecho mención, en la etapa de "Presentación de la Metodología"que La Evaluación de Seguridad corrobora la Integridad y la Disponibilidad del flujo de la información entre dos entidades en una implementación básica del modelo de RINA en su versión 1.9 para conocer si es vulnerable o no a los ataque propuesto en la sección 3.2.2 (Ataques a ser

usados en la evaluación de seguridad).

Al ser una evaluación de caja blanca, información como direcciones, puertos, nombres de procesos y otros identificadores pueden ser consultados a través de la consulta de archivos.de configuración, flujos capturados y especificaciones publicadas por parte del equipo de desarrollo, buscando con toda esta información verificar las propiedades de la seguridad ya antes mencionadas.

3.2. Fase de Análisis e Investigación

En esta fase es donde se lleva a cabo la investigación referente al flujo de la comunicación que se da entre las aplicaciones a través de los procesos que se encuentran en la misma DIF, las entidades que participan en la comunicación y la información que se genera de realizar una solicitud de asignación de flujo entre las aplicaciones.

El flujo de datos

En la figura 3.4 se puede observar el flujo de datos en el objetivo sobre el que se aplica la evaluación.

1. La aplicación “rina.apps.echotime.client:1” solicita una asignación flujo a la aplicación “rina.apps.echotime.server:1” sin algún requerimiento en particular (Por ejemplo, no importa si los datos llegan en orden o de otro lado e incluso si algunos datos se pueden perder). A diferencia del internet de hoy, la aplicación no tiene una dirección en específico, algún número de puerto o estar consciente del protocolo de la capa.

2. La solicitud de asignación de flujo es dirigido al “ICP Manager”, el cual localiza la DIF a través de cual la aplicación destino se puede alcanzar (Para este caso la búsqueda es simple ya que “rina.apps.echotime.server:1” se encuentra registrado en la única DIF disponible en el “Sistema 1”. El “IPC Manager” envía la solicitud de asignación de flujo al “IPC Process Daemon”

3. El “IPC Process Daemon” solicita la creación dinámica de un “Port-id” al Kernel. Este puerto se maneja localmente al flujo que posteriormente será regresado a la aplicación y a diferencia del Internet no se utiliza como el ID de conexión de punto final de la

conexión que soporta el flujo.

4 Y 5. El kernel usa un algoritmo que dinámicamente calcula un número de puerto disponible y lo regresa al "IPC Process Daemon".

6. El "IPC Process Daemon" crea una "instancia de asignación de flujo" (FAI), quien será el responsable de manejar el flujo durante su tiempo de vida. Entonces el FAI compara la solicitud especificada por la aplicación con el "QoS cubes" que el "IPC Process" puede soportar. Los "QoS cubes" define una región en el espacio de rendimiento que un conjunto específico de políticas puede cubrir. Cada "IPC Process" soporta uno o más "QoS Cubes". En este caso, el "IPC Processes" en la "normal.DIF" soporta dos tipos de "QoS cubes", llamado "unreliable with flow control" y "reliable with flow control". El "IPC Process Daemon"primero – ya que no se especificó de inicio ningún requerimiento – y ahora está listo para crear y configurar la conexión "EFCP" que soportará el flujo.

7, 8 y 9. El "IPC Process Daemon" envía un mensaje al kernel con el fin de crear y configurar una conexión "EFCP" para el flujo. El "binding" entre el flujo y la conexión es temporal, y el mismo flujo puede ser soportado por múltiples conexiones secuenciales de "EFCP" sin que la aplicación lo note o se dé cuenta (La aplicación se mantiene utilizando el mismo puerto). El "IPC Process Daemon" proporciona toda la información en todas las políticas requeridas para la configuración del "EFCP": en este caso en su mayoría relacionado con el control de flujo ya que el control de retransmisión no está activo".

10 El "IPC Process Daemon" revisa su directorio y ve con el fin de alcanzar la aplicación destino debe reenviar la solicitud de flujo al "IPC Process" con dirección 17. Por lo tanto envía un CDAP CREATE dirigido al "IPC Process" 17 con toda la información del flujo (nombre de aplicación fuente, destino, dirección fuente, destino, fuente CEP-id, QoS-id, políticas) codificado como un objeto de flujo. Esta acción es necesaria para i) asegurar que la aplicación destino está disponible todavía en la DIF; ii) asegurar que la aplicación fuente se le permite la comunicación con la aplicación destino y iii) negociar dinámicamente algunas de las características del flujo (políticas EFCP y CEP-id).

11, 12, 13 y 14. El "IPC Process Daemon" recibe de vuelta el mensaje "CDAP" (respuesta CREATE), aceptando el flujo que contiene el valor de destino CEP-id. Lo primero que hace es comunicar el kernel acerca del destino CEP-id para la conexión EFCP, entonces el paquete que pertenece a la conexión, puede propiamente ser identificado.

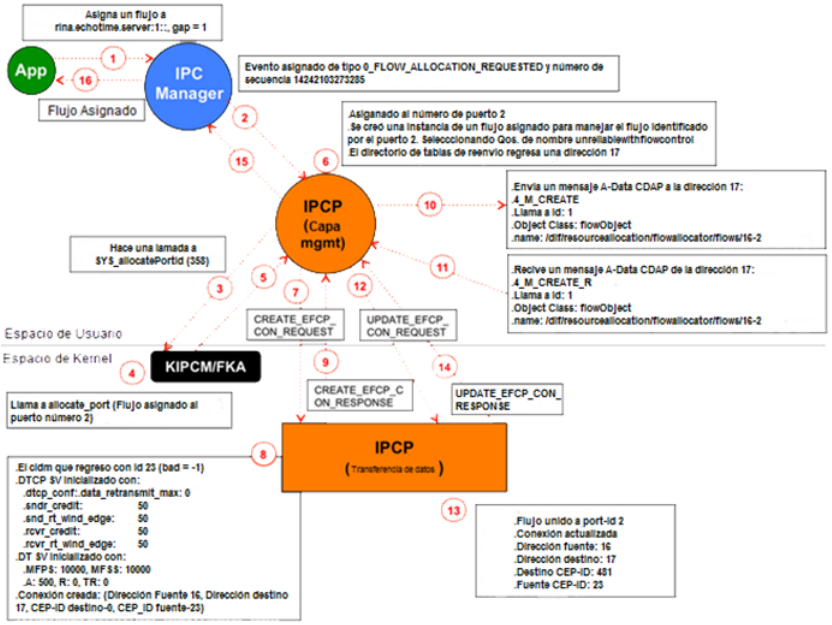


FIGURA 3.4: Flujo de comunicación en el Manejador de Procesos

15, 16. "IPC Process Daemon" responde al "IPC Manager Daemon" sobre el éxito del flujo asignado, comunicando de nuevo el "Port-id" que será utilizado por el flujo. El "IPC Manager Daemon" lleva a cabo un procedimiento similar con la aplicación fuente, quien puede ahora comenzar a iniciar a utilizar el flujo

Cuando intercambian mensajes, una DIF es similar a una capa TCP/IP en el que el protocolo máquina en un nodo (en este caso un Proceso IPC) se comunica con otro protocolo máquina dentro de la misma capa DIF (usualmente en otros nodos). Por ejemplo en TCP/IP, el protocolo TCP habla con otro protocolo TCP en otro nodo, pero no habla a un protocolo IP. Un caso similar ocurre con IP; IP sólo se comunica con otro protocolo IP

Este caso es el mismo en RINA en el que un Proceso IPC en una DIF sólo se comunica con otro Proceso IPC en la misma DIF. El mensaje que pasa entre los procesos IPC son llamados Protocolo de Unidad de Datos (PDUs), este consiste de un Protocolo de Control de Información (PCI) y una carga que RINA llama un SDU. Un SDU a menudo contiene un PDU de la capa superior.
La secuencia que sigue el intercambio de mensaje de dos procesos que son miembros de la misma DIF se puede observar en la figura 3.5

Cada capa DIF consiste de uno o más procesos IPC, los cuales contienen los siguientes

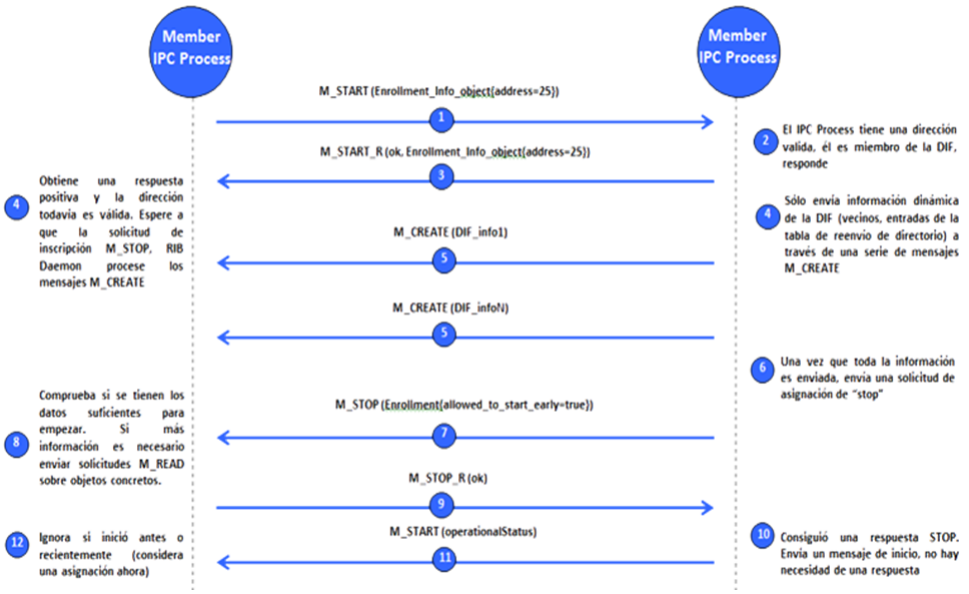


FIGURA 3.5: Comunicación entre aplicaciones

componentes: un "Resource Information Base (RIB), IPC Management Tasks (Enrollment, Routing, Directory, Resource Allocation, Security Management), Relaying y Multiplexing Task (RMT), una Service Data Unit (SDU) Delimiter task, un SDU protection task, y el Error and FlowControl Protocol (EFCP).
Cada proceso Proceso IPC puede también tener uno o más "Application Processes (AP)" ejecutándose por encima de la DIF que contará con la DIF para comunicarse con otras APs que tienen acceso a la misma DIF. El Proceso IPC en sí es un punto de acceso, utiliza el DIF de abajo para la comunicación y así sucesivamente hasta el medio de transmisión físico.

3.2.1. Análisis de la arquitectura

Con la información previamente recabada y analizada se lleva a cabo un nuevo análisis de la implementación básica de RINA en su versión 1.9, en la cual se tiene una idea más clara y precisa de la información con la cual se está trabajando, para de esta manera generar un escenario de pruebas y un conjunto de ataques que sean aplicables a la implementación.

Se entiende que una DIF es un contenedor que guarda información de los nodos de red; así como información sobre la comunicación entre ellos. Para este trabajo en particular se considera atacar a un miembro específico que ya es parte de la misma DIF.

La información necesaria para poner en marcha un ataque es la siguiente:

- Nombre de la DIF
- Nombre de un miembro en específico
- Dirección de un miembro en específico
- Credenciales de autenticación (Aún si estás son nulas)
- Llaves (Para HMAC, cifrado, etc.)
- Políticas de protección de un SDU

Con el fin de poner en marcha un ataque, el evaluador o atacante debe ser capaz de enviar tráfico por medio de de la red a la máquina de destino. Con el fin de interceptar, modificar o fabricar tráfico de red de y hacía un miembro específico, tendría que ser capaz de identificar al miembro de alguna manera. Esto significa que un identificador para la DIF así como el nodo o aplicación en cuestión tendría que ser conocido por el atacante. Una forma alternativa de obtener PDUs de un objetivo específico es a través de un "hop" intermedio o un switch en modo espejo; para este caso en particular la adquisición del flujo de información se lleva a cabo a través de un "Switch Catalyst 3500 Serie XL" con un puerto en modo espejo. El atacante debe conocer la ruta que sigue los PDUs a través de la DIF. Sin embargo, antes de interceptar o modificar un PDU, el atacante debe situarse a lo largo de la ruta que conecta la aplicación fuente y la aplicación destino. Para ello es necesario conocer por lo menos uno de los procesos destino. Para interrumpir el tráfico de un miembro específico de una DIF sin afectar al tráfico de otros nodos, el atacante debe identificar al miembro. Si el atacante no requiere interrumpir el tráfico de otros miembros, puede entonces ser necesaria menos información. Con el fin de atacar el tráfico entre las "Aps" específicas, el atacante también debe conocer o ser capaz de predecir los identificadores de conexión del tráfico en cuestión.

El atacante puede desear capturar el tráfico de una instancia o entre dos nodos específicos.

a) Si un atacante sólo desea capturar el tráfico de una o más instancias de la aplicación (AP) bajo ataque, entonces se tiene que descubrir al menos un conjunto de identificadores de puerto local que se utilizan para esa AP.

b) Si el atacante está interesado en el tráfico entre dos nodos en específico, entonces de debe descubrir la identificación del puerto que está siendo utilizado por la otra AP por el proceso IPC remoto.
Además de los identificadores de red, el atacante debe conocer (o ser capaz de eludir)

las credenciales de autenticación y las políticas de protección de SDU para ser capaz de descifrar el tráfico.

Una vez conocido esto y para poder llevar a cabo la "Evaluación de Seguridad" se requiere el diseño de un escenario de pruebas en específico, esto es, sólo se considera un objetivo en particular para llevar a cabo la evaluación, en la siguiente sección se describen los escenarios que serán utilizados, así como la manera, con base a la información que se conoce, puede ser atacado y comprometida esta implementación.

3.2.2. Diseño y descripción del escenario de pruebas

El diseño del escenario de pruebas requiere la participación de las entidades cliente y servidor, atacante o evaluador y un observador situados en el escenario de la implementación básica de RINA o en algún lugar de nuestro escenario. Su ubicación juega un papel importante, ya que es la manera de capturar el tráfico entre los procesos de la DIF así como de verificar que es posible o no ejecutar un ataque. Se deben ubicar de tal manera que su participación no altere los resultados de la aplicación de la metodología o consuma recursos de sistemas que pudieran alterar la información.

Participantes en la Evaluación de Seguridad

El entorno de desarrollo y pruebas en local se lleva a cabo mediante la utilización de equipos dedicados para la implementación de RINA en su versión 1.9. Donde cada uno de los participantes cumple la siguiente función:

La entidad número uno (entidad dedicada para la implementación de RINA con un sistema operativo Debian) hace la función de cliente, en el cual radia la aplicación origen (rina-echo-time (C)) que solicita una asignación de flujo para poder comunicarse con la aplicación destino (rina-echo-time (S)) que se encuentra situada en la entidad número dos o entidad servidor.

La entidad número dos (entidad dedicada también para el modelo de RINA, con un sistema operativo Debian) hace la función de servidor, en el cual se encuentra la aplicación destino con la cual se desea comunicar la aplicación origen ubicada en la entidad número uno a través de una solicitud de flujo

Una tercera entidad, que hace la función de Evaluador o Atacante es el encargado de conseguir una muestra del flujo de información que se genera al comunicarse la aplicación cliente con la aplicación destino, así como de ejecutar un ataque que pueda comprometer la seguridad de la implementación de RINA. El ataque que se lleva a cabo para este trabajo en particular, está dirigido únicamente contra la entidad número dos también conocida como servidor.

Un cuarto participante, su función principal, como su nombre lo indica, es observar o verificar que la ejecución de los ataques sobre el modelo de RINA es efectivo o no, con base en el análisis de la información que consigue a través del observador es posible indicar si es seguro o no la implementación de RINA frente a los ataques propuesto en este trabajo.

Para comunicar cada uno de los participantes del presente escenario, así como hacer una adquisición del flujo de información que servirá para realizar el diseño de uno o varios de los ataques que comprometan la implementación de RINA es a través de un "hob" o un "switch". Para este último se requiere que sea un switch administrable, configurando por lo menos un puerto en modo espejo y así adquirir la muestra que se requiere para el diseño de un vector de ataque.

Muestra del flujo de información

La aplicación "echo" ha sido diseñada para probar el desempeño del "stack" del prototipo de IRATI, especialmente la "Shim DIF". Se desarrolló una versión más ligera bautizada como "echo", cuyo comportamiento puede resumirse de la siguiente manera:

- La aplicación puede operar en modo servidor, modo en que soporta tests concurrentes o modo cliente (echo-server, echo-client respectivamente).
- Cuando el cliente de la aplicación se ejecuta se registra en una DIF, reserva un flow al servidor y espera por el resultado. Si es satisfactorio, negocia los parámetros del test con el servidor (número y tamaño de SDUs a ser enviadas) y comienza la transferencia de datos. Cuando el servidor recibe una SDU, hace el "echo" de la misma SDU al cliente.
- El test se completa cuando todas las SDUs fueron enviadas y recibidas o cuando transcurre un cierto valor de tiempo sin recibir una SDU por parte de alguno de los extremos.

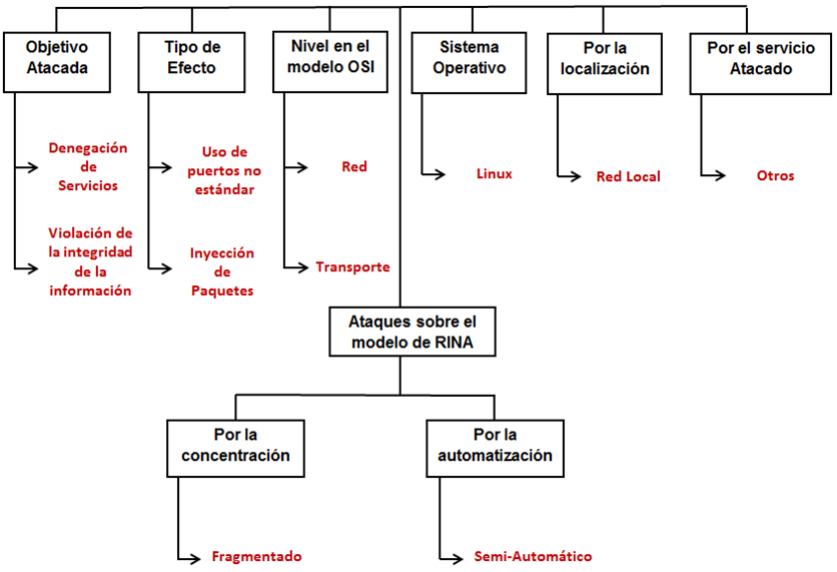


FIGURA 3.6: Posibles ataques a la implementación de RINA

- Tanto el cliente como el servidor reportan estadísticas dela transferencia realizada como el número de SDUs recibidas y el tiempo abarcado.

Ataques sobre la implementación de RINA

Con base en la información recaba en el capítulo número uno, en la sección de "Amenazas y ataques a TCP/IP" se hace una selección de los posibles ataques que se pueden ejecutar para buscar comprometer el flujo de la información en la implementación de RINA, de dicha selección se puede ver en la figura 3.6

Detallando cada uno de los ataques se tiene que:

1) Manipulación de Paquetes

En este tipo de ataque, una entidad no autorizada, que para este caso, ya es parte de la DIF, corroborar si es posible o no insertar datos falsificados en el sistema. El objetivo de este ataque es la intercepción de datos y el análisis de tráfico, una técnica más sutil para obtener información de la comunicación, que puede consistir en obtención del origen y destinatario de la comunicación, leyendo las cabeceras de los paquetes monitorizados.

Es un tipo de ataque activo ya que implica una modificación del flujo de datos transmitido o la creación de un falso flujo de datos.

Con base en el flujo capturado por parte del Atacante o el Observador, en la Fase 3.2.2 de la etapa "Muestra del flujo de información" y a través de su análisis, es formar paquetes sobre para ser inyectados al servidor desde el atacante.
La forma de hacerse de un paquete para conocer cómo está formado es a través del uso del "Switch", trabajando con uno de sus puertos en modo espejo.

La aplicación en el cliente se intenta comunicar con la aplicación situada en el servidor a través de dos procesos que se encuentran en la misma DIF. Haciendo uso del "rina-echo-ping" lanzado desde el cliente solicita la asignación de un flujo, solicitud en donde viaja la información requerida para conocer cómo está formado un paquete. El servidor responde al "rina-echo-ping" del cliente, en la respuesta viajan los demás parámetros que se requieren para iniciar la comunicación entre las aplicaciones.

- C - S: Crea una solicitud (Nombre de Servicio, A, S, fuente CEP-id, QoS)
- S - C: Crea una respuesta (OK, Destino CEP-id)
- C - S: ACK (Destino CED-id), ISNc.
- S - C: Challenge (. . .)
- C - S: Respuesta (. . .)
- C - S: Datos

Para este escenario de ataque, se considera que el atacante ha frustrado la autenticación y es miembro de la DIF como lo son el usuario "C" y el usuario "S". Bajo este caso, el atacante es capaz de conocer las direcciones de A y S, es decir, el atacante envía el ataque de manera interna.

Como se trata de una fase de establecimiento de la conexión, el atacante puede utilizar cualquier fuente de CEP-id. Y puesto que en el modelo de RINA IRATI no hay necesidad de sincronizar los números de secuencia, el remitente también puede utilizar cualquier número de secuencia inicial. Suponiendo longitudes de campo estándar, tomamos la longitud del "CEP-id" a ser el mismo que el de un puerto-id (es decir, 16 bits), por lo tanto, adivinar el "CEP-id" implica 2 a la 16 posibilidades. Este tipo de ataque equivale a ataque de escaneo de puertos, en el que un intruso puede intentar una identificación sin asignar del CEP-id destino.

2) Inyección de Paquetes

Este tipo de ataque es aquel en el que el atacante no tiene acceso a los paquetes de datos de la conexión de la víctima, pero intenta inyectar paquetes que parecen legítimos. La formación de un paquete legítimo requiere adivinar diferentes campos en la cabecera del paquete.

En este caso en particular se tiene acceso a los paquetes a través de la captura por parte del Observador o por parte del Atacante utilizando uno de los puertos en modo espejo, con esto, es posible generar una mayor cantidad de paquetes que se buscan inyectar en el servidor desde el atacante; con el apoyo del observador se puede detectar si es posible llevar a cabo la inyección de paquetes en el servidor.

Para este trabajo en particular el atacante lleva a cabo la inyección de paquetes durante la fase de establecimiento de la conexión, es decir: Después de que la "asignación de recursos" se ha completado y antes de que la fase de transferencia de datos comience.

Los datos que se deben de conocer para llevar a cabo este ataque son: adivinar y/o conocer la fuente CEP-id y el destino CEP-id. El atacante también debe conocer o adivinar otros parámetros de la conexión como lo son "QoS-id. Se considera que la fase de transferencia de datos no ha iniciado, por lo tanto el atacante puede usar cualquier ISN. Dados CEP-ids de 16-bit y QoS-id de 8 bits, el atacante tiene 216+16+8=240 posibilidades para adivinar el CEP-ids y QoS-id para la conexión víctima.

3) Denegación de Servicio

DoS, son las siglas de Denegación de Servicio ("Denial Of Service"), éste tipo de ataques consiste en hacer Ïnaccesibleün servicio para otro usuarios (Entidad uno o cliente), colapsando, saturando y/o sobrecargando el servicio, buscando provocar la caída total del sistema (Entidad dos o Servidora).

La reinyección de tráfico que busca consumir toda la memoria RAM de la entidad número dos, también llamada entidad "Servidor", haciendo inaccesible la comunicación de la entidad número uno, también llamada cliente. Se requiere conocer datos como el "PorID" asignado durante la asignación de flujo, "Dirección de los procesos", puerto de la

aplicación que se busca hacer inaccesible.

Para los ataques planteados, la teoría del modelo de RINA indica que no es posible llevarlos a cabo ya que la comunicación entre aplicaciones se lleva a cabo a través de los procesos que forman parte de la misma capa DIF, haciendo de esta manera el flujo de información seguro, lo cual es lo que se busca comprobar, ya que el usuario donde se lanzan cada uno de los ataques no forma parte de la capa DIF.

3.2.3. Herramientas

Para ejecutar cada uno de los ataques previamente se requiere hacer uso de herramientas previamente diseñadas como:

- Hping: La herramienta hping es un analizador/ensamblador de paquetes TCP/IP de uso en modo consola. Está inspirado en el comando ping de unix. hping es capaz de enviar paquetes ICMP, TCP, UDP, y RAW-IP. Generando paquetes TCP/IP a medida, que se contengan la información que se requiera.
- Scapy: Scapy es un manipulador de paquetes interactivo escrito en Python que permite generar paquetes, enviar paquetes de red, probar equipamiento, descubrir y escanear redes, así como desarrollar nuevos protocolos.

Para monitorear la memoria del sistema, podemos utilizar:

- htop: Muestra en tiempo real el estado de los procesos del sistema. Muestra el uso de CPU, memoria RAM y memoria de intercambio (swap). Además, lista los procesos actualmente en ejecución en todo el sistema, ordenados por uso de CPU
- Monitor: El monitor del sistema es la herramienta gráfica principal de monitorización en tiempo real y de análisis de datos registrados

Para el análisis del tráfico y captura, se emplea:

- TCPDUMP: Es un herramienta en línea de comandos cuya utilidad principal es analizar el tráfico que circula por la red. Permite al usuario capturar y mostrar a tiempo real los paquetes transmitidos y recibidos en la red a la cual el ordenador está conectado.

- TShark: Es una herramienta de análisis de tráfico, permite usar filtros como los de wireshark pero en la consola, puede ser usado en sin que se requiera el entorno gráfico.
- Wireshark: Es una herramienta multiplataforma utilizada para realizar análisis sobre paquetes de red.

3.3. Etapa de Pruebas

Esta fase se enfoca en llevar a cabo cada una de las pruebas sobre la implementación básica de RINA en su versión 1.9, exponiendo la implementación de RINA a los ataques diseñados en la fase anterior a través de secuencias controladas y definidas. Durante esta etapa se prueba la eficiencia de los ataques diseñados previamente. Generado de esta forma un escenario para comprobar la Integridad del flujo de la información, así como un escenario para la Disponibilidad de la información en el servidor.

Recordando que es una evaluación de caja blanca, tenemos acceso a la información, misma que es utilizada para el diseño de los ataques de Manipulación de paquetes, Inyección de paquetes y un ataque de Denegación de Servicios. Siendo necesario para los mismos la siguiente información.

- Nombre de aplicaciones
- Nombre de Procesos
- Dirección de procesos
- Nombre de la DIF
- Nombre de la SHIM

3.3.1. Ejecución individual de la evaluación en el escenario de pruebas

Esta etapa de la evaluación corresponde a la parte práctica. Es decir, en este punto ya se cuenta con un escenario de pruebas que previamente fue diseñado. Ya está bien identificada cada una de las características físicas de los participantes en el escenario de pruebas; Se conoce de los participantes datos como la Memoria RAM disponible, el Sistema Operativo, Versión de Kernel, el Procesador con el que cuentan. Así como las herramientas que se requieren por parte del Atacante y el observador para poder

implementar los ataques.

Sobre el escenario de pruebas previamente diseñado se va a ejecutar cada uno de los ataques diseñados. Cada ataque se debe realizar sobre el mismo escenario de pruebas y bajo las mismas condiciones de operación, esto para que los resultados no se vean afectados por algún cambio de versión o equipos con mayor o menor capacidad.

Para poder llevar a cabo un análisis posterior de los resultados arrojados de la ejecución de cada uno de los ataques y poder determinar si el modelo de RINA es seguro o no, se almacenan en archivos ya sea como un txt, un pcap o de otro tipo dependiendo de lo que se desea reportar.

3.4. Etapa de resultados

Finalmente en esta sección de la evaluación se presentan los resultados obtenidos de llevar a cabo la ejecución de cada uno de los ataques diseñados sobre el cada uno de los escenarios sobre los cuales se llevó a cabo la implementación de la versión 1.9 de RINA.

3.4.1. Análisis de resultados

Para cada uno de los ataques efectuados sobre la implementación de RINA en su versión 1.9 se debe realizar un análisis, para saber si el flujo de información se ve afectado o no por los ataques diseñados en la etapa 3.2.2, es decir, para:

1) Manipulación de Paquetes: Con ayuda del observador se busca capturar flujo para poder dar interpretación a cada uno de los octetos que conforman ese flujo y poder obtener información.

2) Inyección de Paquetes: Aquí es necesario hacer uso del observador o apoyarse de los servicios del observador, ya que de esta manera es posible determinar si es posible llevar a cabo la inyección de tráfico en el servidor por parte del atacante.

3) Denegación de servicios: Valores como la memoria RAM de la entidad sobre la que se va a dirigir el ataque es un dato a tomar en cuenta, así como la cantidad de paquetes necesarios para poder consumir la memoria RAM, duración del ataque; con esto

determinar si es posible consumir la memoria con un solo atacante o se requieren más, información como esta es la que se debe considerar para poder llevar a cabo un ataque de Denegación de Servicios.

3.4.2. Presentación de resultados

En esta etapa de la evaluación de seguridad se presentan los resultados obtenidos de la ejecución de los ataques diseñados sobre la implementación de RINA en su versión 1.9. Es decir, se presenta de manera breve, clara y concisa si la hipótesis planteada al inicio de la investigación es correcta o no.

Dando por terminado el trabajo respondiendo al objetivo planteado, validando o no la hipótesis, si la metodología planteada en el presente trabajo fue correcta o no, así como los problemas que se presentaron durante la presente investigación. De esta manera se da por terminada la ejecución del instrumento de seguridad para, pudiendo demostrar si es seguro o no el modelo de red de RINA en sus versión 1.9 frente a los ataques más representativos de TCP/IP.

Finalmente se tienen identificadas cada una de las etapas, participantes, herramientas y ataques, es decir, el Instrumento de evaluación ha sido formado y corresponde a la siguiente la ejecución del Instrumento de Evaluación para verificar, al menos para este trabajo, la Integridad y la Disponibilidad.

Capítulo 4

Aplicación de la metodologia de evaluación sobre la implementación de RINA

4.1. Etapa de presentación

La metodología se aplica sobre una implementación básica de RINA, de la cual se conoce el código fuente, así como el diseño de la arquitectura, siendo esta información de utilidad para verificar cada uno de los valores y/o parámetros que se vayan recabando a través del observador y así poder dar correcta interpretación a cada uno de los elementos recabados.
De la misma manera, la metodología busca comprobar que no es posible comprometer las propiedades de la seguridad de la información que refieren a la integridad y a la disponibilidad en el flujo de la información que se produce al comunicarse las dos aplicaciones entre sí.

4.1.1. Presentación de la metodología de evaluación

La metodología de evaluación se aplica en una implementación de RINA con número de versión 1.9. Se plantea la generación de escenarios para evaluar la integridad así como la disponibilidad de la información. El objeto de evaluación es el flujo de la comunicación entre dos aplicaciones situadas en entidades diferentes, las aplicaciones se conectan a través de una DIF en común que contiene sólo dos procesos; uno de los procesos solicita la asignación de un flujo al "IPC Manager" el cual es el encargado de gestionar y asignar los parámetros de configuración para comenzar la comunicación entre las aplicaciones.

Es necesaria la participación de entidades como 1) Observador(es) y 2) Atacante(s); La función del observador es monitorear el flujo resultante de la comunicación entre las aplicaciones. La función del atacante es la ejecución de ataques diseñados contra el objetivo de evaluación. Para este trabajo el objetivo de las pruebas es la capa DIF, la cual contiene los procesos que serán utilizados por las aplicaciones de la capa N+1. También es objeto de la investigación comprometer la disponibilidad de la información en la aplicación destino, es decir, la aplicación que toma el papel de servidor.

4.1.2. Descripción de la arquitectura

La arquitectura RINA consta de una sola capa que se repite de forma recursiva llamada DIF, la cual se repite tantas veces como el diseñador la requiera; dicha capa es la que contiene los procesos a través de los cuales se comunican las aplicaciones que se encuentran en la capa N+1. La aplicación, a través de un proceso que pertenece a la DIF que se encuentra en la capa N-1 solicita una asignación de flujo al IPC Manager, el cual asigna los recursos necesarios para poder comenzar la comunicación entre los procesos. A cada proceso asigna un número de puerto, una dirección, un nombre, un número de puerto (que es el que servirá para comunicarse la aplicación con el proceso).

El IPC Maganer, también es el encargado de buscar a procesos que se encuentren dentro de la misma capa y así poder alcanzar un procesos a través del cual pueda comunicarse con la aplicación destino, en caso de no encontrar un proceso dentro de la misma capa, busca en la capa N-1 hasta poder alcanzar un proceso final a través del cual pueda alcanzar la aplicación destino; esto lo hace a través de un algoritmo llamado algoritmo de “dijkstra” (también llamado algoritmo de caminos mínimos, es un algoritmo para la determinación del camino más corto dado un vértice origen al resto de los vértices en un grafo con pesos en cada arista) aplicable en este caso a los procesos de la capa DIF. Los procesos pueden crearse o unirse a una capa ya existente, siendo las políticas de seguridad de cada una de las capas las encargadas de determinar si puede o no hacerlo, estas políticas son definidas por el diseñador. Si los procesos son autenticados, es decir, se les permite unirse a la misma capa, se les asigna una dirección, un puerto, un nombre y CEP-Id que son necesarios para establecer la comunicación entre los procesos.

Una vez que se ha superado la etapa de autenticación los procesos comenzarán a comunicarse e intercambiar información entre ellos. Dentro del archivo de configuración “IPC Config” se definen los parámetros que servirán para establecer las condiciones bajo las

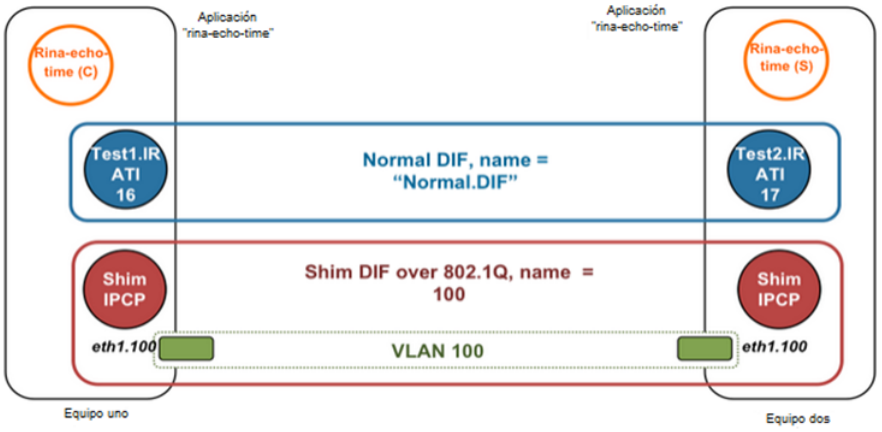


FIGURA 4.1: Modelo básico de RINA

cuales se llevará a cabo la comunicación entre las aplicaciones.

Ya que se han comunicado las aplicaciones, el proceso origen solicita al IPC Manager liberar el flujo que fue asignado al inicio del establecimiento de la comunicación, liberando de esta manera los recursos asignados al inicio de la comunicación. Quedando de esta manera liberados puertos y direcciones para ser asignados a otros procesos cuando lo soliciten. Fuera de la capa DIF, todos estos recursos que asignados pierden sentido, siendo valores validos únicamente para la capa a la que pertenecen.

Para el presente trabajo, el objetivo de evaluación consiste de dos aplicaciones situadas en la capa N+1 de la capa "Normal.DIF", la cual contiene dos procesos dentro de la capa, procesos que serán utilizados para poder comunicar las aplicaciones (Como se puede ver en la figura 4.1)

4.1.3. Identificación del objetivo

El objetivo a ser evaluado consiste de dos aplicaciones que desean comunicarse, las aplicaciones se encuentran en entidades diferentes, la aplicación en el equipo uno "rina-echo-time (c)" solicita una asignación de flujo para comunicarse con la aplicación "rina-echo-time (s)". En cuanto a los procesos, a la entidad número uno pertenece el proceso "test1.IRATI", con dirección de 16 y a la entidad número dos pertenece el proceso "test2.IRATI" con dirección 17, ambos procesos se encuentran dentro de la misma capa DIF de nombre "Normal.DIF".

Es necesario evaluar la Integridad y Disponibilidad del flujo de la comunicación de los procesos antes mencionados. Para determinar si es o no comprometer las propiedades de la seguridad antes mencionadas.

En la capa N – 1 se encuentra la capa "SHIM", la cual es una capa "no RINA" cuyo objetivo es adaptar las tecnologías actuales a la arquitectura RINA, haciendo de esta manera más fácil la adopción de RINA por parte de otras tecnologías.

4.2. Etapa de análisis e investigación

En esta fase hay que recabar toda la información necesaria para conocer a detalle cada uno de los elementos que servirán para conocer los elementos que aporten información la forma en que funciona la arquitectura y descubrir posibles vulnerabilidades que se están presentes en esta arquitectura, las cuales, en caso de encontrarlas, servirán para demostrar que esta arquitectura, al igual que otras, presenta problemas de seguridad.

4.2.1. Análisis de la arquitectura

La metodología lleva a cabo la ejecución de pruebas de tipo caja blanca, se tiene acceso al código fuente y al diseño de la arquitectura RINA, para ejecutar pruebas sobre la arquitectura RINA, de manera específica en la implementación de RINA en su versión 1.9.

Módulos del Kernel

Es requisito hacer uso de una capa "shim" y una capa "DIF", por lo tanto se requiere trabajar con módulos a nivel de kernel los cuales proporcionan soporte para la funcionalidad de las capas antes mencionadas, dicho módulos son: "modprobe shim-eth-vlan" y "modprobe normal-ipcp".

En el archivo de configuración del "IPC Manager" se definen las capas SHIM y DIF. Para la Capa Shim se define una capa tipo "shim-eth-vlan" de nombre "100". Para la capa DIF se define una capa tipo "normal-ipc" de nombre "normal.DIF" El IPC Manager es el encargado de gestionar los procesos que forman parte de una capa DIF a través de los cuales se comunican las aplicaciones. Se les asigna un nombre a cada uno de los

procesos, una dirección, número de puerto. Todo esto se lleva a cabo dentro de la DIF creada.

Capa Shim

La capa Shim es la encargada de hacer transparente la arquitectura RINA para las tecnologías no RINA lográndose de esta manera comunicarse y funcionar.
En el archivo "difConfigurations" se define la configuración de la "Shim", se asigna por nombre "Shim" 100, su tipo "shim-eth-vlan" y el nombre de la interfaz de comunicación "eth1". El único parámetro importante de la capa shim requerido por la DIF, es el "inteface-name".

Capa DIF

La capa "DIF" es más compleja, iniciando con la constante de transferencia de datos. La sección "dataTransferConstants" asigna un tamaño de dirección (en este caso con un tamaño de 2), un tamaño de CEP-id (en este caso con un tamaño de 2), un tamaño de puerto de 2, tamaño de ID para los QoS (en este caso con un tamaño de 2), un tamaño para el número de secuencia (en este caso de un tamaño de 4), un tamaño máximo de PDU (en este caso un tamaño de 10000) y un tiempo de vida máximo para el PDU (en este caso un tamaño de 30). Esta información es propiamente manejada por la sección que corresponda al protocolo de Control de Flujo y Error que se utiliza en la DIF.

El archivo de configuración "ipcmanager.conf" contiene los parámetros de inicio de la comunicación entre los procesos que se van a comunicar.
Los escenarios de prueba sobre los que se va a aplicar a aplicar la metodología de evaluación son un modelo cliente – servidor, en el cual se busca comprometer tanto la seguridad de la integridad, así como la disponibilidad de la misma para la entidad número dos, también conocida como servidor. Siendo objeto de ataque el servidor.

4.2.2. Diseño y descripción de los escenarios de pruebas

Para cada uno de los escenarios de pruebas se debe considerar que la implementación se ejecuta sobre un sistema operativo Debian Wheezy 7, así como:

1) Para el sistema número uno (cliente); se encuentra integrado por una aplicación y proceso llamado "test1.IRATI", con dirección 16. Esta aplicación a través del proceso es la que realiza la solicitud de asignación de flujo para comunicarse con la aplicación

que se encuentra en el servidor, haciendo uso del "rina-echo-time".

2) En el sistema número dos (servidor); se encuentra una aplicación y un proceso llamado "test2.IRATI", con dirección 17. Esta entidad es la encargada de responder a la solicitud "rina-echo-time" que genera el cliente.

3) EEl Observador: es el encargado de llevar a cabo el monitoreo de cada uno de los flujos que se producen, así como de llevar a cabo la captura del paquetes del "rina-echo-time" y darle interpretación a los octetos que forman parte de ese flujo, de esta manera formar paquetes.

4) El Atacante: su función dentro del escenario de pruebas es ejecutar cada uno de los ataques diseñados para verificar las propiedades de integridad y disponibilidad en el flujo de la comunicación.

5) CCada uno de los elementos que participan en la evaluación necesitan ser conectados para comunicarse entre sí, para el presente trabajo se hace uso de un "Switch Catalyst 3500" administrables, en el cual al puerto 1 se conecta el sistema uno, al puerto número 2 se conecta el sistema dos. El puerto número tres es el puerto espejo, puerto donde se conecta el observador y en otro puerto del switch se conecta el atacante para ejecutar cada uno de los ataques.

6) Existen dos escenarios, uno busca analizar la integridad y otro la disponibilidad del flujo de la información.

Escenario para verificar la Integridad de la información

En este escenario la función que cumple el observador es la de capturar un paquete RINA con el fin de conocer cómo es un paquete en esta arquitectura de red (Como se puede ver en la figura 4.2)

En RINA si un proceso no forma parte de la capa DIF, no es posible tener acceso al flujo que se produce de comunicarse entre ellos. Dado que el "Observador" no es parte de la capa DIF, la manera de capturar un paquete RINA es a través de un "switch". Para el presente trabajo se hace uso de uno administrable, configurando uno de sus puertos

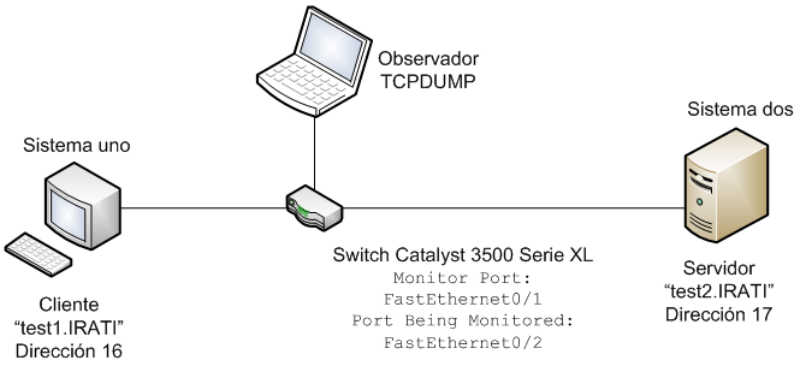


FIGURA 4.2: Escenario para la Integridad

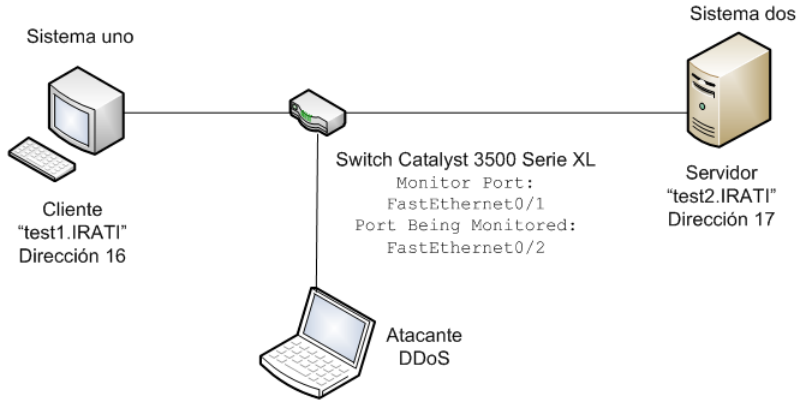


FIGURA 4.3: Escenario para la disponibilidad

como puerto espejo y capturando el flujo de comunicación entre el cliente y el servidor.

Escenario para verificar la Disponibilidad de la información

El escenario para garantizar la disponibilidad de la información es el que se observa en la figura 4.3

Para este escenario no hay necesidad de un observador, para este caso se requiere de otra entidad que cumpla las funciones de atacante, la cual va a ser la encargada de ejecutar el ataque contra la entidad número dos (también llamada servidor), esto con el fin de negar la comunicación del cliente con el servidor, logrando de esta manera demostrar que es posible comprometer la disponibilidad de la información.

Un factor importante es conocer los equipos con los que se va a trabajar, en términos generales las características de cada uno de los equipos que forman parte de los escenarios de prueba se observan en la tabla 11.

	Equipos utilizados en el experimento			
	Sistema 1	**Sistema 2**	**Sistema 3**	**Sistema 4**
S. O.	Debian/Linux 7.0 (Wheezy)	Debian/Linux 7.0 (Wheezy)	Ubuntu 14.10	Ubuntu
Tipo	64 Bit	64 Bit	64 Bit	64 bit
Versión	GNOME Versión 3.4.2	GNOME Versión 3.4.2		
Memoria	1.9 GiB	3.8 GiB	5.7 GiB	
Procesador	Intel Core 2 Duo CPU E6550 @ 2.33 GHZ x 2	Intel Core i5-2500 CPU @ 3.30 GHz x 4	CPU M 640 @ 2.80 GHz x 4	
Gráficos	Intel G33	Gallium 0.4 on NVA8	Gallium 04 on VAS	
Disco	153.2 GB	976.1 GB	238.2 GB	
Funciones	Cliente	Servidor	Atacante	Observador
Herramienta	tcpdump/tshark	tcpdump/tshark	tcpdump/tshark	Wireshark

Tabla 11. Datos de los equipos del experimento.

EL IPCManager.conf

Siendo una prueba de caja blanca, tenemos acceso al archivo de configuración de RINA, el cual contiene información sobre los procesos, la asignación de flujos, la capa DIF, la capa Shim. Información que sirve para conocer información referente a cada uno de los paquetes que se requieren para establecer la comunicación entre los procesos.

En lo que respecta a la información que podemos recabar del archivo de configuración referente a la capa DIF, se obtienen los datos que se muestran en la tabla 12.

Propiedad	Proceso 1	Proceso 2
apInstance	1	1
difName	normal.DIF	normal.DIF
difsToRegisterAt	100	100
interface-name	eth0	eth0
address	16	17

Tabla 12. Informacion de la capa DIF.

Para el control de flujo, dentro del archivo de configuración podemos consultar las longitudes de cada uno de los campos que se intercambian en la asignación de flujo, siendo esto parte de la capa DIF.

En esta sección del archivo de configuración se observa el tamaño de los campos, la capa DIF a la cual pertenecen, tamaño máximo de PDU y el tiempo máximo del PDU (Como se puede ver en la Tabla 13)

	Constante de Transferencia de Datos	
Parámetro	**Sistema 1**	**Sistema 2**
difName	normal.DIF	normal.DIF
difType	normal-ipc	normal-ipc
addressLenght	2	2
cepIdLength	2	2
lengthLength	2	2
portIdLength	2	2
qosIdLength	2	2
sequenceNumberLength	4	4
maxPduSize	10000	10000 ms
maxPduLifeTime	30	30 ms

Tabla 13. Parámetros de la tranferencia de datos.

Parte del proceso de la implementación de RINA es configurar una VLAN, para esto se debe tomar en cuenta lo siguiente:

Sistema 1

Para la configuración de la VLAN son necesarias las siguientes características:

ip link add link eth1 name eth1.100 type vlan id 100
ip link set dev eth1 up
ip link set dev eth1.100 up

Además de configurar la VLAN, es necesario proporcionar soporte para la funcionalidad de las capas “shim” y “DIF” en el Kernel, haciendo necesario agregar los siguientes módulos:

modprobe shim-eth-vlan
modprobe rina-default-plugin
modprobe normal-ipcp

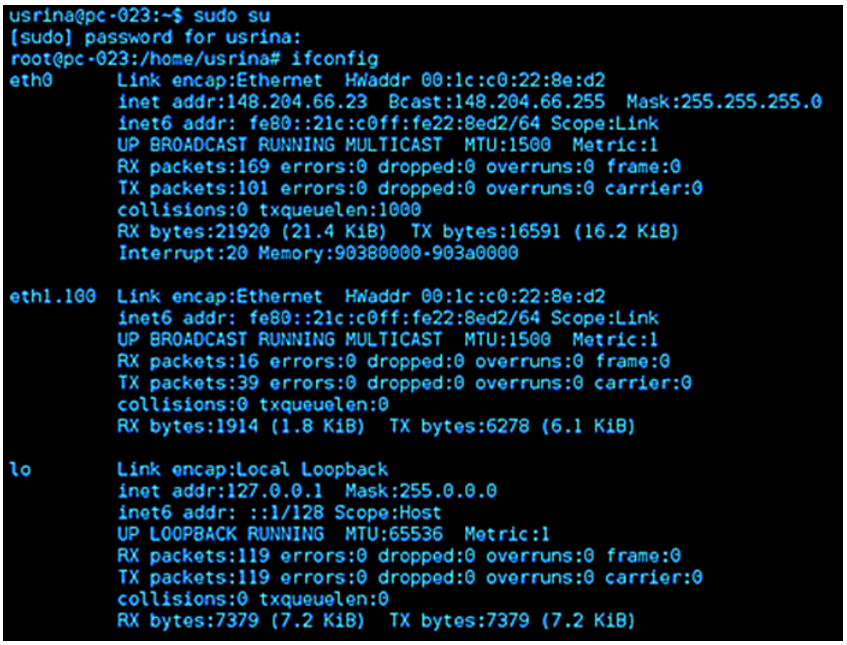


FIGURA 4.4: Creación de la VLAN

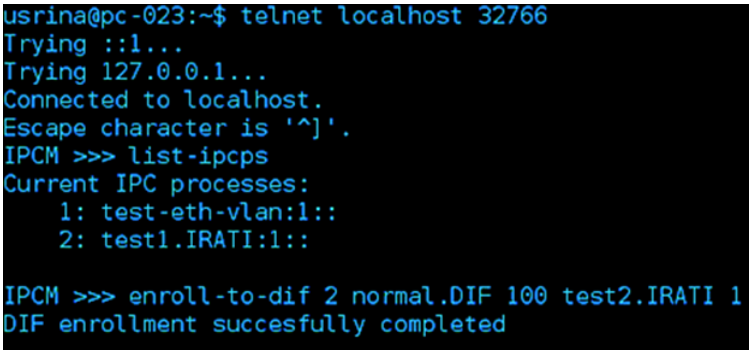


FIGURA 4.5: Función de .[en]roll-to-dif"de los dos procesos

Como se puede observar en la figura 4.4 ha quedado de esta manera lista la VLAN y el soporte para las capas.

Los procesos que residen dentro del archivo de configuración deben saber que estos existen y están en la misma capa, para ello es necesario hacer uso de la siguiente "enroll-to-dif", de esta forma los procesos ahora saben que existe el proceso uno y el proceso dos, así como que ambos procesos pertenecen a la misma capa DIF (Como se puede ver en la figura 4.5)

Parte del archivo de configuración son los datos que corresponden a la capa DIF. Como se puede observar en la imagen 4.6, los datos que refieren a la capa DIF, como el nombre, el tipo de capa y el nombre de la interfaz se puede consultar en el archivo de

```
DIF configuration/properties:
        DIF name: 100:::
        DIF type: shim-eth-vlan
        Parameters:
                interface-name:eth1
DIF configuration/properties:
        DIF name: normal.DIF:::
        DIF type: normal-ipc
        Parameters:
Application --> DIF mappings:
        rina.utils.apps.echo.client-1--: normal.DIF:::
        rina.utils.apps.echo.server-1--: normal.DIF:::
        rina.utils.apps.rinaperf.client-1--: normal.DIF:::
        rina.utils.apps.rinaperf.server-1--: normal.DIF:::
```

FIGURA 4.6: Información de la capa DIF en el sistema uno

```
Local Configuration
        Installation path: /bin
        Library path: /lib
        Log path: /var/log
        Console port: 32766
        CDAP timeout in ms: 10000
        Enrollment timeout in ms: 10000
        Flow allocator timeout in ms:  15000
        Watchdog period in ms: 60000
        Declared dead interval in ms: 120000
        Neighbors enroller period in ms: 10000
IPC process to create:
        Name: test-eth-vlan:1::
        DIF Name: 100:::
        DIFs to register at:
        Host name:
        SDU protection options:
        Parameters:
IPC process to create:
        Name: test1.IRATI:1::
        DIF Name: normal.DIF:::
        DIFs to register at:
                100:::
```

FIGURA 4.7: Parámetros de establecimiento de comunicación

configuración.

Así como se puede consultar información respecto a las capas DIF y shim, también se puede consultar información referente a las condiciones bajo las cuales se lleva a cabo el intercambio de información, es decir, la configuración local de los paquetes que se van a transmitir (Como se puede ver en la figura 4.7)

Sistema 2

De la misma forma en que fue configurada la VLAN para el sistema número uno se debe hacer lo mismo para el sistema número dos. Siendo necesario crearla con la misma información y los mismos módulos para dar soporte a las capas shim y DIF. La VLAN

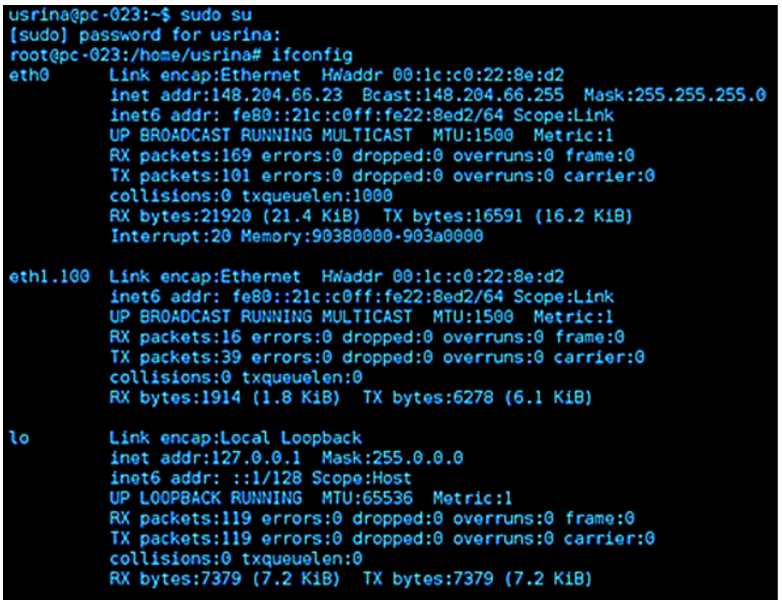


FIGURA 4.8: VLAN en el sistema dos

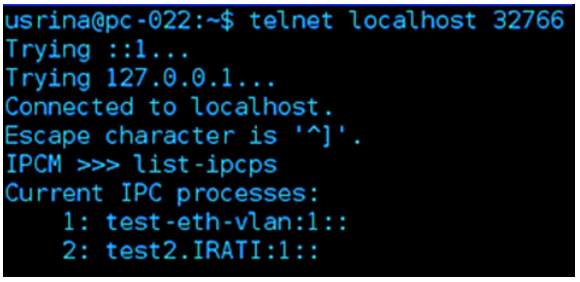


FIGURA 4.9: Creando el proceso dos

para el sistema dos se puede observar en la figura 4.8

Para el sistema dos se hace necesario crear el proceso destino o proceso que dará respuesta a la solicitud del proceso "test1.IRATI". Asignándole el nombre de "test2.IRATI" a este nuevo proceso (Como se puede ver en la figura 4.9)

Para consultar la información que refiere a este nuevo proceso podemos consultar el archivo de configuración en el cual, al igual que en el proceso uno, nos muestra el nombre de la DIF a la que pertenece, el tipo de DIF y la interfaz. Para este ejemplo, los dos procesos deben de tener los mismos nombres de DIF, el mismo tipo y la misma interfaz, ya que ambos procesos pertenecen a la misma capa (Como se puede ver en la figura 4.10)

Al igual que en el proceso anterior, los datos referentes a las condiciones bajos las cuales

```
DIF configuration/properties:
        DIF name: 100:::
        DIF type: shim-eth-vlan
        Parameters:
                interface-name:eth1
DIF configuration/properties:
        DIF name: normal.DIF:::
        DIF type: normal-ipc
        Parameters:
Application --> DIF mappings:
        rina.utils.apps.echo.client-1--: normal.DIF:::
        rina.utils.apps.echo.server-1--: normal.DIF:::
        rina.utils.apps.rinaperf.client-1--: normal.DIF:::
        rina.utils.apps.rinaperf.server-1--: normal.DIF:::
```

FIGURA 4.10: Información de la capa DIF en el sistema dos

```
Local Configuration
        Installation path: /bin
        Library path: /lib
        Log path: /var/log
        Console port: 32766
        CDAP timeout in ms: 10000
        Enrollment timeout in ms: 10000
        Flow allocator timeout in ms:  15000
        Watchdog period in ms: 60000
        Declared dead interval in ms: 120000
        Neighbors enroller period in ms: 10000
IPC process to create:
        Name: test-eth-vlan:1::
        DIF Name: 100:::
        DIFs to register at:
        Host name:
        SDU protection options:
        Parameters:
IPC process to create:
        Name: test2.IRATI:1::
        DIF Name: normal.DIF:::
        DIFs to register at:
                100:::
        Host name:
        SDU protection options:
        Parameters:
```

FIGURA 4.11: Archivo de configuración dos

se va a llevar a cabo el flujo de información, debe ser el mismo, lo cual se puede comprobar en el archivo de configuración del proceso dos (Como se puede ver en la figura 4.11) La información antes recabada corresponde a la etapa de análisis, en la cual se recaba la mayor cantidad de información posible para poder detectar alguna vulnerabilidad en la arquitectura.

Al ser una evaluación de caja blanca se tiene acceso al archivo de configuración, pudiendo cotejar cada uno de los bytes del flujo capturados con el archivo de configuración, pudiendo de esta manera validar que la información que se interpreta es de forma correcta.

4.3. Etapa de pruebas

Para conocer la forma de un paquete de RINA se requiere realizar la captura del "rina-echo-time" que es ejecutado desde la entidad uno y la respuesta que regresa el servidor al "rina-echo-time".

Para esto se requiere hacer uso del escenario que busca comprometer la integridad de la información. Primeramente el cliente envía un "rina-echo-time" a la cual responde el servidor. Con ayuda del puerto espejo del switch se puede capturar este paquete y así conocer su contenido.

Ahora, para llevar a cabo el ataque de Denegación de Servicio se hace uso del escenario diseñado para Disponibilidad de la información, siendo el atacante el encargado de generar el ataque. Un valor muy importante es del atacante es la memoria RAM con la que cuenta. Ya que con este ataque se busca consumir los recursos del servidor, es decir, se busca consumir toda la memoria RAM de la entidad dos con el fin de que la entidad número uno no pueda seguir enviando paquetes dado que el servidor no es capaz de atender a sus solicitudes ya que está siendo consumida por el atacante.

4.3.1. Ejecución individual de la evaluación en los escenarios de pruebas

Para la captura de paquetes se utiliza la herramienta TCPDUMP, guardandolo en un archivo del tipo pcap. Se envía por parte del cliente una solicitud "rina- echo-time". Con ayuda de esta herramienta es posible probar el flujo sobre la capa "DIF". En la aplicación, el cliente envía un número configurable de paquetes (SDUs) a una velocidad configurable hacía el servidor.

Ejecutando "rina-echo-time" en el servidor

En el Servidor, el "Sistema 2", dentro de la carpeta ÏNSTALLATION_PATH/bin"se ejecuta:

./rina-echo-time -1

```
12:31:19.727754 00:1c:c0:22:8e:d2 (oui Unknown) > e8:39:35:3f:43:26 (oui
Unknown), ethertype 802.1Q (0x8100), length 724:
      0x0000:  0111 0010 0001 0000 0000 0040 00c2 0200  ...........@....
      0x0010:  0000 0008 0010 0c18 002a 0661 5f64 6174  .........*.a_dat
      0x0020:  6132 0661 5f64 6174 6138 0042 e604 32e3  a2.a_data8.B..2.
      0x0030:  0408 1010 111a dc04 0800 1004 1805 2a04  ..............*.
      0x0040:  666c 6f77 3230 2f64 6966 2f72 6573 6f75  flow20/dif/resou
      0x0050:  7263 6561 6c6c 6f63 6174 696f 6e2f 666c  rceallocation/fl
      0x0060:  6f77 616c 6c6f 6361 746f 722f 666c 6f77  owallocator/flow
      0x0070:  732f 3136 2d32 3800 42eb 0332 e803 0a22  s/16-28.B..2...."
      0x0080:  0a19 7269 6e61 2e61 7070 732e 6563 686f  ..rina.apps.echo
      0x0090:  7469 6d65 2e63 6c69 656e 7412 0131 1a00  time.client..1..
      0x00a0:  2200 1222 0a19 7269 6e61 2e61 7070 732e  "..".rina.apps.
      0x00b0:  6563 686f 7469 6d65 2e73 6572 7665 7212  echotime.server.
      0x00c0:  0131 1a00 2200 1802 2000 2810 3011 3a06  .1..".....(.0.:.
      0x00d0:  0801 1000 1800 4000 4801 5224 1800 2000  ......@.H.R$....
      0x00e0:  2800 3000 3900 0000 0000 0000 0040 0148  (.0.9........@.H
      0x00f0:  0050 ffff ffff ffff ffff ff01 5800 6000  .P..........X.`.
      0x0100:  5add 0208 0112 bc02 0801 1292 0108 0112  Z...............
      0x0110:  1508 c801 1032 1a06 0a00 1200 1a00 2206  .....2........".
      0x0120:  0a00 1200 1a00 1800 2249 0800 1000 1a15  ........"I......
      0x0130:  0a07 6465 6661 756c 7412 0764 6566 6175  ..default..defau
      0x0140:  6c74 1a01 3022 150a 0764 6566 6175 6c74  lt..0"...default
      0x0150:  1207 6465 6661 756c 741a 0130 2a15 0a07  ..default..0*...
      0x0160:  6465 6661 756c 7412 0764 6566 6175 6c74  default..default
      0x0170:  1a01 3028 0030 0038 0040 0048 0050 005a  ..0(.0.8.@.H.P.Z
      0x0180:  060a 0012 001a 0062 060a 0012 001a 006a  .......b.......j
      0x0190:  060a 0012 001a 0072 060a 0012 001a 0018  .......r........
      0x01a0:  0022 9001 0800 1000 1a15 0a07 6465 6661  ."..........defa
      0x01b0:  756c 7412 0764 6566 6175 6c74 1a01 3022  ult..default..0"
      0x01c0:  150a 0764 6566 6175 6c74 1207 6465 6661  ...default..defa
      0x01d0:  756c 741a 0130 2a15 0a07 6465 6661 756c  ult..0*...defaul
      0x01e0:  7412 0764 6566 6175 6c74 1a01 3032 150a  t..default..02..
      0x01f0:  0764 6566 6175 6c74 1207 6465 6661 756c  .default..defaul
      0x0200:  741a 0130 3a15 0a07 6465 6661 756c 7412  t..0:...default.
      0x0210:  0764 6566 6175 6c74 1a01 3042 150a 0764  .default..0B...d
      0x0220:  6566 6175 6c74 1207 6465 6661 756c 741a  efault..default.
      0x0230:  0130 4800 2a06 0a00 1200 1a00 3206 0a00  .0H.*.......2...
      0x0240:  1200 1a00 1a06 0a00 1200 1a00 2206 0a00  ............"...
      0x0250:  1200 1a00 2a06 0a00 1200 1a00 3000 3800  ....*.......0.8.
      0x0260:  6801 7000 7803 4800 5000 8801 0092 0106  h.p.x.H.P.......
      0x0270:  0a00 1200 1a00 9a01 00a2 0100 aa01 00b2  ................
      0x0280:  0100 ba01 00c2 0100 ca01 00d2 0100 da01  ................
      0x0290:  00e0 0100 4800 5000 8801 0092 0106 0a00  ....H.P.........
      0x02a0:  1200 1a00 9a01 00a2 0100 aa01 00b2 0100  ................
      0x02b0:  ba01 00c2 0100 ca01 00d2 0100 da01 00e0  ................
      0x02c0:  0100
```

FIGURA 4.12: Forma de un paquete de RINA

Con esto, la Aplicación del Servidor ha quedado registrada en la capa DIF. Lo hace a través de la inspección de la RIB del IPC Process "test2.IRATI" en el sistema 2 mediante la consola del "IPC Manager", dando como resultado.

Name:/dif/management/flowallocator/directoryforwardingtableentries/rina.apps.echotime.server-1–; Class: directoryforwardingtableentry; Instance: 20 Value: App name: rina.apps.echotime.server-1–; Address: 17; Timestamp: 0

Con esto el IPC Process "test2.IRATI" ha compartido el registro de esta aplicación a través de la DIF, por lo tanto la RIB de "test1.IRATI" también contendrá esta entrada. La forma que tiene el primer paquete que envía el proceso cliente o el proceso que solicita la asignación de flujo haciendo uso de la aplicación "rina-echo-time" aparece en la figura 4.12 Siendo un paquete de una longitud de 724 bytes que incluye la cabecera de Ethernet, la de la VLAN y la parte que corresponde al paquete RINA.

Ejecutando "rina-echo-time" en el cliente

En el Cliente (la entidad número uno o Sistema uno), se ejecuta el "rina-echo- time", la cual se encuentra dentro de la carpeta ÏNSTALLATION_PATH/bin", ejecutando esto,

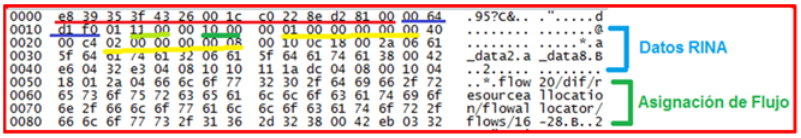


FIGURA 4.13: Datos útiles del paquete de RINA

se envía la respuesta a la solicitud que genera el cliente, respondiendo con un paquete de 726 bytes, en general presenta la misma forma que el paquete que se envía cuando se crea la solicitud, con la diferencia de que hay dos octetos más que indican que es una respuesta por parte de la aplicación destino que para este caso es la aplicación servidor.

./rina-echo-time - c 100

La forma del paquete se puede ver en la figura 4.13

Identificando los siguientes octetos de la solicitud que genera el cliente, octetos que pueden ser también identifados en la respuesta del servidor:

Octeto 0 – Octeto 14 / Ethernet
Octeto 15 – Octeto 17 / VLAN
Octeto 18 / Versión/Bandera
Octeto 19 y 20 / Dirección F/D
Octeto 21 / Versión/Bandera
Octeto 22 y 23 / Dirección D/F
Octeto 24 / Versión/Bandera
Octeto 25 y 30 / CEP-id F/D
Octeto 34 y 39 / CEP-id D/F

Otra forma de dar validez a los octetos presentes en el flujo es a través de la inspecciona del archivo “RIB” de alguno de los dos procesos, archivo en el cual se observa una nueva entrada que describe el flujo que soportado la capa “DIF” (Como se puede ver en la figura 4.14)

Este archivo muestra información sobre el proceso, información como direcciones, CEP-id, Qos, puerto asignado. Información que sirve de apoyo para dar interpretación a los octetos.

Mientras esta solicitud de asignación de flujo se lleva a cabo, el observador se encarga de capturar el tráfico con TCPDUMP en un archivo en formato PCAP para su posterior

```
Name: /dif/resourceallocation/flowallocator/flows/16-2; Class: flow; Instance: 21
Value: * State: 2
* Is this IPC Process the requestor of the flow? 1
* Max create flow retries: 1
* Hop count: 3
* Source AP Naming Info: rina.apps.echotime.client:1::
* Source address: 16
* Source port id: 2
* Destination AP Naming Info: rina.apps.echotime.server:1::res:
Destination port id:
* Connection ids of the connection supporting this flow: +
Src CEP-id 0; Dest CEP-id 0; Qos-id 1
* Index of the current active connection for this flow: 0
```

FIGURA 4.14: Contenido del archivo RIB

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	IntelCor_22:8e:d2	HewlettP_3f:43:26	0xd1f0	724	PRI: 0 CFI: 0 ID: 100
2	0.001885	HewlettP_3f:43:26	IntelCor_22:8e:d2	0xd1f0	726	PRI: 0 CFI: 0 ID: 100
3	0.002483	IntelCor_22:8e:d2	HewlettP_3f:43:26	0xd1f0	57	PRI: 0 CFI: 0 ID: 100
4	0.003137	HewlettP_3f:43:26	IntelCor_22:8e:d2	0xd1f0	60	PRI: 0 CFI: 0 ID: 100
5	0.003175	HewlettP_3f:43:26	IntelCor_22:8e:d2	0xd1f0	60	PRI: 0 CFI: 0 ID: 100
6	0.003211	IntelCor_22:8e:d2	HewlettP_3f:43:26	0xd1f0	49	PRI: 0 CFI: 0 ID: 100
7	0.005217	IntelCor_22:8e:d2	HewlettP_3f:43:26	0xd1f0	227	PRI: 0 CFI: 0 ID: 100

FIGURA 4.15: Handshake de RINA

análisis e interpretación. Wireshark no es capaz de interpretar el flujo de RINA. Utiliza protocolos que no son conocidos por esta herramienta, marcando dichos protocolos como “0Xd1f0”, dando lugar a la interpretación en crudo de los datos por parte del atacante

El flujo que comprende el establecimiento de la comunicación entre los procesos se puede ver en la figura 4.15, son siete los pasos que comprenden el establecimiento de la comunicación entre estos dos procesos. En los primeros paquetes se envía la información necesaria para establecer la comunicación entre cada uno de los procesos. El primer paquete tiene una longitud de 724 octetos y el archivo de respuesta del servidor un archivo de 726 octetos, hay dos octetos de diferencia, los cuales indican que es una respuesta por parte del servidor.

Con el PCAP capturado y haciendo uso de la información que se conoce se forman los paquetes para buscar vulnerar la Integridad y Disponibilidad de la implementación de RINA. Para comprometer la Integridad se hace a través de un ataque de Ïnyección de paquetesz para la Disponibilidad es a través de un ataque de ”Denegación de Servicios”. Cadaos ataques. A continuación se describe de manera detallada cada uno de los mismos.

Integridad (Inyección de Paquetes)

FIGURA 4.16: Consumo de memoria de un rina-echo-time.[en] el servidor

Con ayuda del escenario diseñado para "Verificar la Integridad", el Observador se apoya de la herrienta "Wireshark" para escuchar el puerto espejo del "switch" y así capturar el tráfico que intercambia la entidad número uno (cliente) y la entidad número dos (Servidor). Para de esta manera realizar el análisis de cada uno de los paquetes y tratar de dar interpretación a cada uno de los octetos que forman parte del tráfico.

Interpretación los octetos se vulnera la propiedad de Confidencialidad. Así mismo con la información que se interpreta da paso a conocer la manera en que un paquete está formado. Y de esta manera llevar a cabo la fabricación de paquetes. Un paquete RINA tiene la siguiente forma:

"Ethernet + VLAN + Bandera + Dirección Fuente + Bandera + Dirección destino + Bandera + CEP-id Fuente + CEP-id destino"

Disponibilidad (DoS)

Al ejecutar el "rina-echo-time" se observa que existe un consumo de memoria del 0.1 % de la memoria RAM disponible (Como se observa en la figura 4.16).

Este 0.1 % representan 3.92 Mb consumidos por un "rina-echo-time".
Para ejecutar el ataque de una "Denegación de Servicio" se requiere determinar la cantidad de paquetes necesarios para consumir los recursos de memoria RAM disponibles en el servidor (Como se puede ver en la figura 4.17).
1 "rina-echo-time", consume 3.92 Mb, para consumir la memoria RAM se requiere enviar 99949.0446 paquetes "rina-echo-time"

La etapa de .ªnálisis e identificación"permite conocer las vulnerabilidades presentes en la implementación básica de RINA en su versión 1.9. A través de este análisis es posible

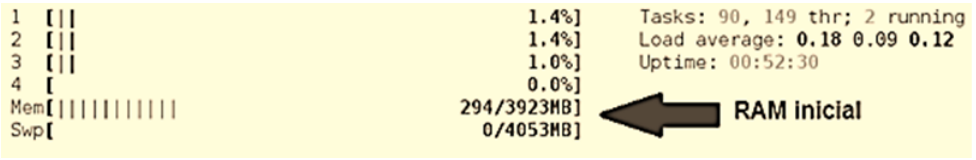


FIGURA 4.17: RAM en condiciones iniciales

conocer que las medidas de seguridad como lo son el módulo que refiere a las políticas de seguridad y autenticación aún no se encuentra presentes en la implementación. Lo que permitió llevar a cabo con éxito los ataques propuestos en el presente trabajo de tesis.

En la implemenatación básica de RINA para su versión 1.9 es posible vulnerar tanto la propiedad de Integridad, así como la de Disponibilidad y de manera transitiva la Condifencialidad. Los datos de establecimiento de la conexión vijan en claro, es decir, se pueden capturar siendo o no parte de la capa DIF, no presentan algún método de cifrado para su protección. En el siguiente capítulo se presenta un análisis de los restulados obtenidos de aplicar el instrumento de evaluación sobre la implementación básica de RINA para su versión 1.9 y las causas de la ejecución de cada uno de los ataques sobre esta implementación básica.

Capítulo 5

Resultados

5.1. Presentación de resultados

El quinto y último capítulo corresponde a la presentación de los resultados obtenidos en el desarrollo del presente trabajo. Resultados que informan sobre la ejecución de los ataques a la implementación básica del modelo de RINA en su versión 1.9.
Siendo satisfactorios los ataques planteados. Se logró la inyección de paquetes, así como consumir los recursos disponibles de la entidad que cumple las funciones de servidor en el modelo básico de RINA.

5.1.1. Resultados

Para el ataque referente a la integridad de la información, se logró observar que es factible llevarlo a cabo, ya que la integridad de la información se ve comprometida al dar interpretación a la información en el flujo de datos; datos que viajan sin cifrado o protección alguna logrando ser leídos e interpretados. Adquiriendo significado los octetos que hacen referencia a los puertos de origen y destino, direcciones fuente y destino, parámetros como el CEP-id.

- Integridad

Un flujo normal de comunicación es el que se encuentra presente en la figura 5.1

Cuando se lleva a cabo la inyección de paquetes en un flujo de la implementación de RINA, el flujo normal se ve afectado por la inyección de los paquetes, presentado la

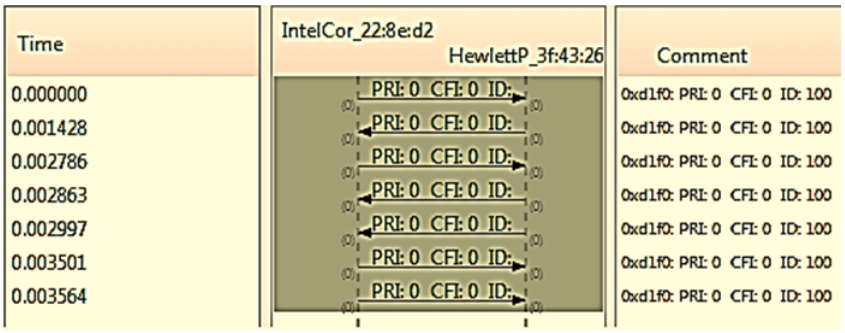


FIGURA 5.1: Comunicación normal entre los procesos uno y dos

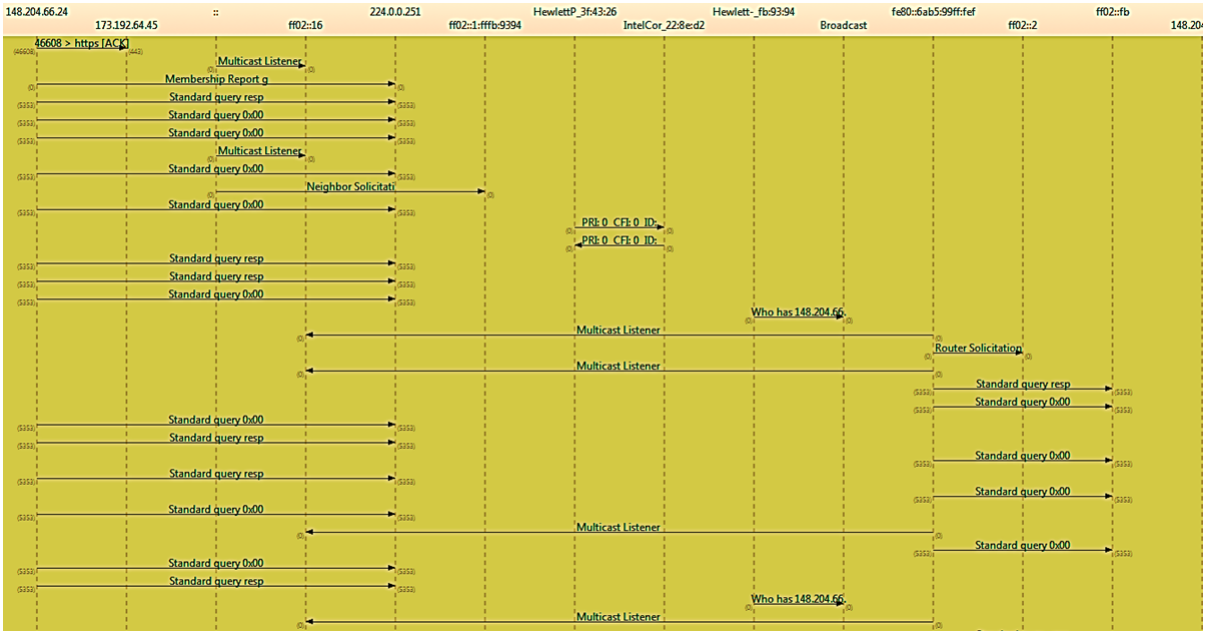


FIGURA 5.2: Flujo bajo inyección de datos

forma que se ve en la figura 5.2

Donde el flujo de la información se ve altera por una tercera entidad, situación que no se debería de dar ya que la teoría de RINA indica que si un proceso o elemeno no se es parte de la misma capa DIF no es posible capturar el tráfico ni comunicarme con otros procesos, ya que no es una entidad autorizada.

- Disponibilidad

En lo que respecta a la Denegación de Servicio, se logró afectar la disponibilidad de la información en el servidor mediante el envío masivo de paquetes con ayuda de la herramienta "Hping3"logrando consumir todos los recursos de la memoria RAM, haciendo

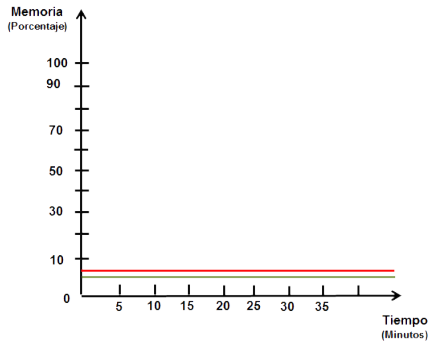


FIGURA 5.3: Condiciones iniciales de la memoria RAM y SWAP

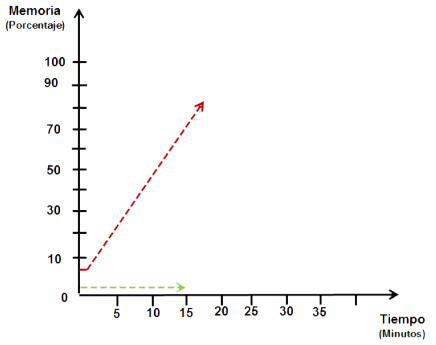


FIGURA 5.4: Ataque en progreso

imposible la comunicación del cliente con el servidor.

Por otra parte, con ayuda de la herramienta "Monitor" que proporciona el sistema Debian, se lleva a cabo el monitoreo de la memoria del sistema y así dar seguimiento al ataque DoS en progreso, dando los siguientes resultados.

- RAM inicial

Las condiciones iniciales de la memoria RAM son monitoreadas como se muestra en la figura 5.3

La línea marcada en color rojo la representación de la memoria RAM y la línea en color verde la memoria SWAP.

- Ataque en progreso (minuto quince)

La memoria RAM bajo ataque se comienza a consumir, esto se aprecia en la figura 5.4

- Inicia la operación de la memoria SWAP

Una vez que la memora física ha alcanzado sus límites, la memoria SWAP comienza a funcionar, ésta lo hace al minuto veintitrés (Como se ve en la figura 5.5)

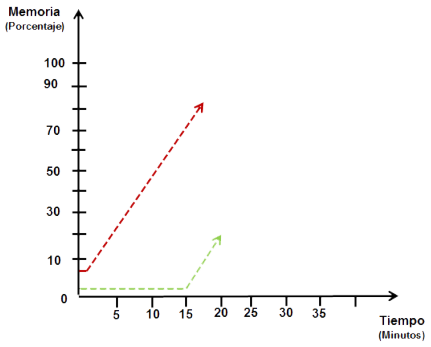


FIGURA 5.5: Comienza a trabajar la memoria SWAP

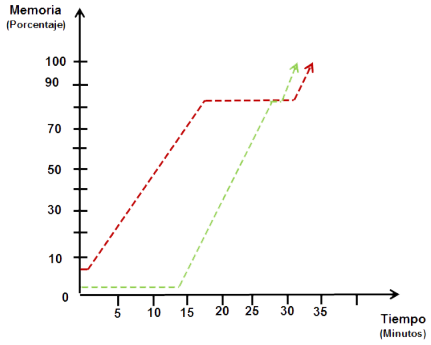


FIGURA 5.6: Consumo total de la memoria RAM y SWAP

- RAM final

Finalmente el ataque DoS ha tenido resultados satisfactorios, consumiendo tanto la memoria RAM como la memoria SWAP, esto pasa el minuto treinta y cuadro (Como se ve en la figura 5.6).

5.1.2. Análisis de resultados

Este capítulo se presenta el análisis que se obtuvo de realizar este trabajo, llevando a cabo el análisis de cada una de las etapas de la metodología. De esta forma dando por concluido el trabajo, reportando los resultados obtenidos de la aplicación de la metodología sobre la implementación disponible hoy en día de RINA.

En la implementación de RINA en su modelo 1.9 es posible comprometer la integridad de la información así como de la disponibilidad de la información, haciendo de esta manera inseguro el envío y recepción de información, ya que es posible interpretar información de los flujos que se capturan, información que es útil para llevar a cabo el diseño de ataques contra la arquitectura de RINA. Actualmente, el equipo encargado del proyecto de IRATI que es el encargado de dar a conocer la arquitectura de RINA ha liberado una

versión en la cual hacen mención que se incorporan nuevas funcionalidades, así como el trabajado de las políticas de seguridad de la capa DIF. Siendo esto de interés ya que se podría evaluar esta nueva implementación con la metodología diseñada y sometiéndola a un ataque de negación de servicio para saber si con estas nuevas aportaciones RINA realmente es capaz de brindar la seguridad que hoy en día requiere el Internet, de serlo así.

Llevar a cabo una selección y análisis de las metodologías de evaluación con las que se logró formar una metodología con la cual se evaluó la seguridad de la implementación de RINA. Ejecutando cada una de las etapas de la metodología diseñada sobre la implementación se descubrió que es posible adquirir información sensible como direcciones, puertos, número de secuencia y CEP-id que viaja en claro dentro del flujo de la comunicación entre los procesos. Además, la función "Watchdog" genera paquetes en los cuales se envía información sobre cada uno de los procesos que forman parte de la capa DIF; capturando estos paquetes es posible conocer más información sobre los integrantes de cada una de las capa.

Para esta versión de implementación el CEP-ID siempre de inicio tiene un valor de cero; no soporta la asignación de direcciones de forma dinámico para los procesos; existen valores predeterminado para su buen funcionamiento, siendo el valor el tamaño máximo de PDU de 10000 y el "tiempo de vida máximo" definido por default de 30.

A través de herramientas como "TCPDUMP" y "Wireshark" es posible capturar el tráfico que viaja dentro de la capa DIF al comunicarse la aplicación cliente con la aplicación ubicada en el servidor, es posible identificar la fase de establecimiento de la comunicación entre las aplicaciones, las cuales consta de siete fases. En las cuales los primeros dos paquetes son los encargados de llevar la información necesaria para que las aplicaciones puedan comunicarse. Los siguientes cinco paquetes son de confirmación de recepción de información de cada uno de los procesos para poder iniciar la comunicación entre las aplicaciones. Este es el caso básico donde los procesos se encuentran dentro de la misma DIF, no fue necesario validar las políticas de seguridad de la capa.

La etapa de "Análisis" permitió recabar información que compromete las propiedades de la seguridad, la captura e interpretación de esta información se debe a que en esta implementación las políticas de seguridad de la capa DIF de RINA no están trabajadas o aún no han sido implementadas. Hoy en día existe una nueva versión que se presenta

como mejorada, en la cual, nuevos elementos de seguridad han sido incorporados.

Del análisis de la información se desprende:

- El identificador de CEP-id, para esta implementación, siempre de origen tiene un valor de cero, haciendo posible enviar un paquete con este identificador en cero. El CEP-id que regresa el servidor es el único que cambia.
- RINA no utiliza un número de secuencia para los paquetes, permitiendo asignar cualquier valor.
- La versión 1.9 de RINA no soporta la asignación dinámica de direcciones, el diseñador debe asignarlas de manera estática, respetando las longitudes que se asignan en el archivo de configuración de RINA, ya que de no respetar estas longitudes, existen problemas en cuanto a la información que asigna a las direcciones impidiendo a un proceso localizar a otro.
- Para esta implementación el tamaño máximo de PDU válido es de 10000, así como también el tiempo máximo de PDU es de 30 ms.
- En la fase de establecimiento de la conexión, una vez completado, se lleva a cabo el intercambio de los siete paquetes que involucran dicho proceso, la función "watchDog" envía cada 60 segundo información sobre los procesos que se encuentran en la capa DIF, paquetes que incluye información como nombre de los procesos, direcciones, puertos, CEP-ID, información que puede ser capturada con un analizador de red como wireshark o TCPDUMP, siendo posible la interpretación de esta información.

Conclusiones

El propósito del trabajo de tesis fue evaluar la Integridad y Disponibilidad en el flujo de información de una comunicación entre dos aplicaciones a través de procesos sobre una implementación básica de RINA en su versión 1.9. Para demostrarlo fue necesario llevar a cabo el diseño de un instrumento de evaluación; instrumento que esta formado por una metodología, herramientas de ataque, escenarios de pruebas y el diseño de ataques. La metodología de evaluación fue diseñada a partir de las metodologías SAAM y ATAM. Con la metodología se logró conocer más a fondo la arquitectura de RINA e identificar posibles vulnerabilidades de la misma para ser explotadas. Los ataques de İnyección de datosƶ "Denegación de Servicio"fueron diseñados con base en la información recaba en las etapas de .ᴬnálisis e investigación.ªsí como de la de .ᴬnálisis del objetivo". Con en el fin de comprometer la Integridad y Disponibilidad, cabe mencionar que no sólo éstas fueron vulneradas, también la confidencialidad se vio vulnerada al capturar el flujo entre los procesos y dar interpretación a los octetos que hacen referencia a los procesos que participan en el establecimiento de la comunicación. La interpretación fue posible ya que al ser una evaluación de caja blanca se tiene acceso a los archivos de configuración, para este caso la consulta se hizo a los archivos RIBƶ al archivo de configuración çonfig"de RINA; estos fueron de apoyo para corroborar la interpretación que se daba a cada uno de los octetos. Dado los resultados obtenidos se logra llegar a las siguientes conclusiones:

- La arquitectura TCP/IP es vulnerable a diversos ataques, siendo los ataques de escucha ilegal, inyección de paquetes y denegación de servicio de utilidad para este trabajo de tesis. Con la escucha de paquete se logra identificar cómo esta formado un paquete RINA; con base en la información de la estructura de un paquete es posible armar paquetes para su inyección. La misma captura de paquetes muestra informacón como direcciones y puertos origen/destino, así como nombre de procesos y de la capa DIF, datos que son útiles para el ataque de Denegación de servicios y de esta manera ejecutarlo con exito comprometiendo Integridad y Disponibilidad en el flujo de la información.

- RINA es una propuesta de arquitectura de red para el Internet del futuro con poca investigación sobre la evaluación de su seguridad. Como tal, no existe una metodología aplicable para evaluarla, haciendo necesario proponer una con base en metodologías ya existentes aplicables al modelo TCP/IP. Sirviendo como base elementos que se retoman de las metodologías SAAM y ATAM para proponerla.

Logrando de esta manera aplicarla sobre la implementación básica de RINA y así evaluar la Integridad y Disponibilidad del flujo de la información, demostrando que ambas pueden ser vulneradas por lo menos para esta implementación.

- A partir de las metodologías SAAM y ATAM se generó la propuesta de metodología de evaluación; La metodología prpuesta consta de cuatro etapas principales, siedo estas la etapa de: 1) Presentación, 2) Análisis, 3) Pruebas y 4) Resultados. A través de la aplicación de estas etapas sobre la implementación fue posible explotar vulnerabilidades y generar ataques que exponen la debilidad de esta implementación de RINA frente a un ataque de inyección de paquetes, así como a uno de denegación de servicios.

La metodología fue diseñada con base en metodologías que ya han sido probadas y cuentan con un sustento científico, generando una metodología aplicable a RINA, la misma, cn base en sus etapas, aporta elementos útiles como direcciones y puertos para el diseño de ataques. Parte del éxito de los ataque se debe a que el módulo referente a la seguridad y la autenticación que operan en la capa DIF aún no han sido implementadoss para esta versión. Siendo de interés ejecutar el mismo instrumento de evaluación sobre la nueva versión liberada por parte del equipo de desarrollo del proyecto IRATI con el fin de corroborar si la arquitectura presenta elementos sólidos para decir que es una solución al problema de seguridad del Internet del futuro.

- Para la implementación básica de RINA en su versión 1.9 no fue necesario un escenario complejo, fue suficiente con los escenarios propuestos ya que sólo se comunican dos aplicaciones a través de dos procesos que se encuentran dentro de la misma capa DIF, el escenario de prueba se vuelve más complejo conforme aumenten las capas, los procesos y los participantes. En RINA aún no están en función todos sus módulos, en específico el módulo de seguridad motivo pon el cual con herramientas como Wireshark y TCPDUMP es posible capturar el trafico y darle interpretacion, para posteriormente formar paquetes con la herramienta Scapy y enviarlos con Hping fue posible capturar. El estado actual que presenta RINA es que sigue utilzando como base el protocolo de "TCP/IP"para adarptarla a las tecnologias de hoy en día siendo posible diseñar o ejecutar ataques presentes en el Internet de hoy en dia. El equipo de desarrollo habla de liberar un prototipo de RINA con un mayor trabajo en los módulos de seguídad, motivo por el cual resulta de interes implementar la siguiente versión y someter a pruebas la misma

para determinar, una vez que los módulos de seguridad se encuentren en función si RINA es segura o no, ya que de momento para una implementación básica de RINA en su versión 1.9 no lo es, al no garantizar la Integridad, Disponibilidad y Confidencialidad.

Trabajos Futuros

Resultado de la etapa de .ᴬnalisis e Investigacionz con base en los resultados obtenidos surge la idea de crear más procesos dentro de la misma capa DIF que soliciten comunicarse con una sola aplicación destino a traves de un solo proceso, proponiendo sean tal vez 50 o 100 procesos. Así como también procesos que se encuentren en DIFs diferentes, para conocer si un proceso destino es capaz de responder a varias solicitudes y asignarles un flujo a cada una de ellas al mismo tiempo sin consumir los recursos, de ser incapaz de dar respuesta a cada una de ellas se podría representar como una denegación de servicio.

Otro trabajo a futuro es llevar a cabo la ejecución de los ataques diseñados en este trabajo de tesis en la nueva implementación de RINA que ha sido liberar por el equipo de desarrollo de, implementación que tal vez no incorpora seguridad en su totalidad, pero sí presenta trabajo en el área, verificacndo así, si los ataques planteados en este trabajo pueden vulnerar la seguridad o no dentro de las capas DIF.

El instrumento de evaluación fue aplicado a un solo objetivo de la implementación básica de RINA, para corroborar la seguridad de otros elementos de la arquitectura, es necesario definir a qué elemtos será aplicada y ejecutar el instrumento sobre cada uno de ellos, logrando de esta manera evaluar otros elementos y determinar su seguridad.

En este trabajo únicamente han sido evaludas la Integridad y Disponibilidad, siendo necesario para la nueva implementación plantear una mayor cantidad de ataques, buscando con esto ya no sólo corroborar Integridad y Disponibilidad, sino también Confidencialidad, Autenticación y el no repudio. Resultado una evaluación de seguridad a RINA más completa. Al presentar un trabajo en los módulos de seguridad para la nueva implementación, se requiere un mayor análisis del objetivo a ser atacado, siendo necesario ahondar más en la etapa de .ᴬnálisis e Investigación", esto con el fin de descubrir alguna vulnerabilidad para ser explotada y así determinar si la nueva implementación es segura o no con respecto a la anterior.

Otro trabajo de interés es proponer una red más grande, en la cual se encuentren interconactos otros equipos a través de un switch. Con el fin de que una aplicación pueda comunicarse con otra aplicación distante pasando por varios equipos, que ya no sea sólo por varias capas. Tratar de capturar el flujo y analizar el intercambio de información entre procesos, aplicaciones y capas.

Bibliografía.

[1] Diego R. López (14.12.2014), Consultado Febrero 2015 [En línea]. Disponible: http://blogthinkbig.com/rina/

[2] Raj Jain. 2006. Internet 3.0: Ten Problems with Current Internet Architecture and Solutions for the Next Generation. In Proc. IEEE Military Communications Conference (Milcom 2006), Washington DC, October 23-25, 2006.

[3] Jennifer Rexford and Constantine Dovrolis. "Future Internet Architecture:
Clean-Slate Versus Evolutionary Research. Communications of the ACM". 2010. Vol. 53 No. 9, Pages 36-40

[4] Jianli Pan, Subharthi Paul, and Raj Jain. "A survey of the research on future internet Architectures". 0163-6804 July 2011 IEEE

[5] Paul Mueller, Bernd Reuther, Markus Hillenbrand. Future Internet Architecture:
A Service-Oriented Approach. 50 (2008) 6 / DOI 10.1524/itit.2008.0510

[6] Kpatcha Bayarou, Rahamatullah Khondoker, Ronald Marx. "Future of Internet Architectures - moving FIA forward". February 2014.

[7] Beny Nugraha, Rahamatullah Khondoker, Ronald Marx y Kpatcha Bayarou.
Detecting and Mitigating Repaying Attack in Expressive Internet Architecture (XIA). Octubre 2014

[8] Ivan Seskar, Kiran Nagaraja, Sam Nelson and Dipankar Raychaudhuri. "MobilityFirst Future Internet Architecture Project". 2011

[9] Hongyu Hu, Jun Bi, Tao Feng, Sen Wang, Pingping Lin, You Wang. "A Survey on New Architecture Design of Internet". 2011

[10] Tom Anderson, Ken Birman, Robert Broberg, Matthew Caesar. "A Brief Overview of the NEBULA Future Internet Architecture".

[11] Diego R. Lopez. Diciembre 2014. El día que "se rompió" Internet (un poquito). [en línea]. [Fecha de consulta: Marzo 2015]. Disponible en: http://blogthinkbig.com/rina/

[12] IRATI, Investigating RINA. Researching and Prototyping the recursive Internetwork Architecture to support Distributed Computing. 2014. The Recursive InterNetwork Architecture.
[Fecha de consulta: Noviembre 2014]. Disponible en: http://irati.eu/the-recursive-internetwork-architecture/

[13] Andrew S. Tanenbaum. "Sistemas Operativos Modernos". Tercera Edición 2008. Pag 37, 38 y 39.

[14] Francisco Javier Serrano Castaño. "Gestión de procesos en los sistemas operativos".

[15] John Day. “Patterns in Network Architecture. A return to Fundamentals”. 2008

[16] Eduard Grasa. “RINA: redefining the network fundamentals”. November 2010 [Fecha de consulta: Noviembre 2014]. Disponible en: https://i2catblogctx.wordpress.com/2010/11/08/rina-redefining-the-network-fundamentals/

[17] Eleni Trouva, Eduard Grasa, John Day y Steve Bunch. “Layer Discovery in RINA networks”. Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), September 2012 IEEE 17th International Workshop on

[18] Leonard Bergesio, Miguel Martín y Eduard Grasa. “Prototipo de RINA sobre Ethernet”. Proyecto Final de Máster de Software libre. UOC. Diciembre 2013

[19] “Implementing a Successful Security Assessment Process”, SANS Institute InfoSec Reading Room. Bradley Hartm, GSEC Version 1.2e August 21, 2001

[20] Kazman, Rick ; Klein, Mark ; Clements, Pau. “ATAM: Method for Architecture Evaluation”. August 2000.

[21] Mugurel T. Ionita, Dieter K. Hammer, Henk Obbink. “Scenario-Based Software Architecture Evaluation Methods: An Overview”. Pag 3, 4 y 5.

[22] Rick Kazman, Len Bass, Gregory Abowd y Mike Webb. “SAAM: A Method for Analyzing the Properties of Software Architectures”.

[23] Mugurel T. Ionita, Dieter K. Hammer, Henk Obbink. “Scenario-Based Software Architecture Evaluation Methods: An Overview”. Pag 1, 2 y 3

Apéndice A

Instalación del sistema operativo

Instalación de Debian/Linux 7.0 (Wheezy)

La implementación de RINA se puede ejecutar sobre todas las plataformas GNU/Linux

Apéndice B

Instalación de la implementación

Implementación de RINA
- Configuración a nivel Kernel
Para las partes del kernel, el siguiente grupo de paquetes son requeridos: Kernel-Package libncurses5-dev
- Configuración a nivel Espacio de Usuario
Para las partes del espacio de usuario , los siguientes paquetes son requeridos:
sudo apt-get install:
autoconf
automake
libtool
pkg-config
git
g++
Protobuf-compiler, libprotobuf-dev (Se requi e re de la versi ón ¿= 2.5.0)
1) wget https: / /protobuf.googlecode.com/svn/rc/protobuf-2.6.0.tar.gz (for version 2.6.0)
2) tarxzvf protobuf-2.6.0.tar.gz
3) cd protobuf-2.6.0
4) ./configure –prefix=/usr
5) make
6) makeclean
7) makeinstall
sudo apt-get install:
openjdk-6-jdk
maven
SWIG versión 2.x, se puede bajar de http://swig.org
1) Se requiere una versión ¿2.0.8, la versión 2.0.12 es la más recomendada

2) Dependiendo de la instalación se puede requerir la instalación de libpcre3-dev, libnl-genl-3-dev, libnl-3-dev (En una versión requerida debe ser ¿=3.2.14)
1) Agrega 'deb http://ftp.de.debian.org/debian jessiemain' en la siguiente ruta /etc/apt/source.list
2) Actualiza apt-get update
3) Ejecuta la siguiente línea de comando: apt-get install libnl-genl-3-dev libnl-3-dev
Se requiere del módulo base 'vlan', para lo cual se debe instalarlo sudo apt-get install vlan
Una vez que se cuenta con los paquetes y módulos básicos que utiliza la implementación de RINA, podemos comenzar a instalar RINA.
Para instalar IRATI, comenzamos ejecutando:
'make menu config'
Dentro del directorio "linux" que se encuentra en una de las carpetas que fueron descargadas del grup. Para una instalación básica no es necesario realizar una modificación alguna, con esto comenzará a compilar el Kernel.
Para instalar tanto el kernel, como el espacio de usuario puedes realizarlo de dos formas:
1) Vía Script.
Install-from-scratch
2) Realizarlo de manera manual, siguiendo los siguientes pasos.
a) makeheaders-install
b) makebzImage modules
c) makemodules-installinstall

Apéndice C

Configuración del sistema 1

Configuración de VLAN Carga de los modulos del Kernel de RINA shim-eth-vlan normal-ipcp Editando el archivo de configuración (./ipcmanager .conf) Instalando el IPC Manager

Apéndice D

Configuración del sistema 2

Configuración de VLAN
Carga de los modulos del Kernel de RINA
shim-eth-vlan
normal-ipcp
Editando el archivo de configuración (./ipcmanager .conf)
Instalando el IPC Manager
Enrollment/Registro de Procesos IPC a una DIF Normal.