UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Despliegue de infraestructura en RINA

Memoria

Nicolás Orellana Celis

Trabajo final de Grado

Coordinador: Davide Careglio

Tecnologías de la información

Curso Q1 2021-2022

Septiembre 2021

Abstracto

Resumen

En el año 2018, se lanzó el proyecto Experimenting with Real Application-specific QoS Guarantees in a Large-scale RINA Demonstrator, que para simplificar, se llamó ERASER.
Este proyecto nace de la creciente investigación nacida sobre RINA (Recursive InterNetwork Architecture) durante los últimos años.
Se ha visto que varios investigadores se han visto interesados en las posibilidades que nos ofrece este nuevo paradigma de arquitectura de red, siendo uno de los posibles reemplazos al modelo tradicional TCP/IP, modelo que lleva reinando sobre Internet por los últimos cincuenta años.

El objetivo de este proyecto era realizar un estudio sobre la arquitectura de RINA y sobre la herramienta que, pienso, mejor implementada está para poder desplegar una simulación de RINA: IRATI (Investigating RINA as an Alternative to TCP/IP). Una vez realizado este estudio, nuestro objetivo principal era replicar el proyecto ERASER hecho en remoto en los servidores de Fed4Fire, pero esta vez hacer una replica en un escenario virtual (mediante VirtualBox), para posteriormente poder realizar las pruebas en un escenario real en servidores comprados para su utilización en el CBA.

Durante el desarrollo de este proyecto descubrí distintas cosas que podrán ayudar a acabar el desarrollo del proyecto, ya que debido a diversos obstáculos, el mismo no pudo llegar a finalizarse en su totalidad. Sin embargo, pienso que las notas que he dejado y la simulación hecha tanto virtual como física será una base importante para que se pueda llegar a finalizar este proyecto en su escenario más complejo.

Como resultado hemos comprobado que, efectivamente, se puede realizar un despliegue de RINA en servidores físicos sin necesidad de tener que realizar una simulación mediante máquinas virtuales. Realizando todos los experimentos en máquinas reales, siendo estas mismas los servidores localizados en el CBA.

Resum

L'any 2018, es va llançar el projecte Experimenting with Real Application-specific QoS Guarantees in a Large-scale RINA Demonstrator, que per simplificar, es va anomenar ERASER.
Aquest projecte neix de la creixent investigació nascuda sobre RINA (Recursive InterNetwork Architecture) durant els darrers anys.
S'ha vist que molts investigadors s'han vist interessats en les possibilitats que ens ofereix aquest nou paradigma d'arquitectura de xarxa, essent un dels possibles reemplaçaments al model tradicional TCP/IP, model que ha regnat sobre Internet els darrers cinquanta anys.

L'objectiu d'aquest projecte era fer un estudi sobre l'arquitectura de RINA i sobre l'eina que, penso, està millor implementada per poder desplegar una simulació de RINA: IRATI (Investigating RINA as an Alternative to TCP/IP).
Un cop realitzat aquest estudi, el nostre objectiu principal era replicar el projecte ERASER fet en remot als servidors de Fed4Fire, però aquesta vegada fer una rèplica en un escenari virtual (mitjançant VirtualBox), per posteriorment poder realitzar les proves en un escenari real a servidors per a la seva utilització al CBA.

Durant el desenvolupament d'aquest projecte vaig descobrir diferents coses que podran ajudar a acabar el desenvolupament del projecte, ja que a causa de diversos obstacles, aquest no va poder arribar a finalitzar totalment. Així i tot, penso que les notes que he deixat i la simulació feta tant virtual com física serà una base important perquè es pugui arribar a finalitzar aquest projecte al seu escenari més complex.

Com a resultat hem comprovat que, efectivament, es pot fer un desplegament de RINA en servidors físics sense necessitat d'haver de fer una simulació mitjançant màquines virtuals. Realitzar tots els experiments en màquines reals, sent aquestes mateixes els servidors localitzats al CBA.

Abstract

In 2018, the Experimenting with Real Application-specific QoS Guarantees in a Large-scale RINA Demonstrator project was released, which for simplicity was called ERASER.
This project was born from the growing research about RINA (Recursive InterNetwork Architecture) during the last years.
It has been seen that several researchers have been interested in the possibilities offered by this new network architecture paradigm, being one of the possible replacements for the traditional TCP/IP model, a model that has reigned over the Internet for the last fifty years.

The objective of this project was to carry out a study on the architecture of RINA and on the tool that, I think, is the best implementation to this day to be able to deploy a simulation of RINA: IRATI (Investigating RINA as an Alternative to TCP/IP).
When this study was over, our main objective was to replicate the ERASER project which was done remotely on the Fed4Fire servers, but this time we wanted to replicate it in a virtual scenario (using VirtualBox), in order to be able to carry out the tests in a real scenario on servers purchased to be used in the CBA.

During the development of this project I discovered different things that could help to finish the development of the project, since due to obstacles, it could not be completed in its entirety. However, I think that the notes that I have left and the simulation made both virtual and physical will be an important basis for this project to be completed in its most complex scenario.

As a result, we have verified that, effectively, a RINA deployment can be carried out on physical servers without the need to carry out a simulation using virtual machines. Carrying out all the experiments on real machines, these being the servers located in the CBA.

Agradecimientos

Primero que nada, debo agradecer a mi director y codirector del proyecto, Davide Careglio y Jordi Perello. Sin su aprobación de este proyecto, nada de esto habría sido posible.
Especiales agradecimientos a Davide, que siempre supo tener un hueco para poder ayudarme en el desarrollo del proyecto y ofrecerme una mano cuando la necesitase.

También me gustaría agradecer a Albert Lopez, personal de CBA, cuya ayuda también fue de extrema importancia para poder desplegar el experimento en un entorno físico.
Agradecer también a la UPC por estos años de enseñanza, que a pesar de haber sido un camino largo y duro, también me ha servido para aprender a conllevar todo tipo de dificultades tanto profesionales como personales.

También agradecer a mi familia el apoyo que siempre me han brindado y la paciencia que han tenido para seguir brindándomelo.

Por último, pero no menos importante, también agradecer a mi círculo social más cercano y a mi pareja, Paula, ya que el apoyo moral y buenos momentos que me han dado han sido indispensables.

ÍNDICE

Índice de Tablas

Índice de figuras

Acrónimos

RINA Recursive InterNetwork Architecture

TCP Transmission Control Protocol

IP Internet Protocol

OSI Open Systems Interconnection

SCTP Stream Control Transmission Protocol

SHIM6 Site Multihoming by IPv6 Intermediation

TLS Transport Layer Security

IPSec Internet Protocol security

GTP GPRS Tunnelling Protocol

GPRS General Packet Radio Service

LISP Locator/Identifier Separation Protocol

PDU Protocol Data Unit

DAP Distributed Application Process

DAF Distributed Application Facility

CDAP Common Distributed Application Protocol

DTP Data Transfer Protocol

DTCP Data Transfer Control Protocol

DIF Distributed IPC Facility

ICP Inter-Process Communication

PDUFT PDU Forwarding Table

RMT Relaying and Multiplexing Task

1. Introducción y contexto

1.1 Contexto

Este proyecto, cuyo título es “Despliegue de infraestructura en RINA” es un trabajo de final de grado, para el grado de ingeniería informática otorgado por la Facultad de informática de Barcelona (FIB-UPC). En concreto, este trabajo se enmarca en la especialidad de tecnologías de la información.

Cabe destacar que este trabajo se hará en colaboración con la fundación i2CAT [1], fundación sin ánimo de lucro dedicada a la investigación y a la innovación en el ámbito de distintas ramas de la informática tales como arquitecturas de Internet, servicios de internet o aplicaciones .

1.2 Conceptos

Para este trabajo nos importa enfocarnos en el apartado de arquitecturas de Internet/arquitecturas de red, ya que el proyecto tratará sobre la arquitectura de RINA (Recursive InterNetwork Architecture).

Una arquitectura de red es un marco sobre el cual se definen los protocolos a utilizar

A pesar de que desde los años 70 se lleva utilizando la arquitectura TCP/IP [2], hay una historia detrás de esto. El modelo TCP/IP está totalmente basado en el modelo OSI, un modelo que se utilizó ampliamente como una referencia para poder diseñar las posibles suites de protocolos que se pensaba que se podían utilizar como modelo real en Internet.

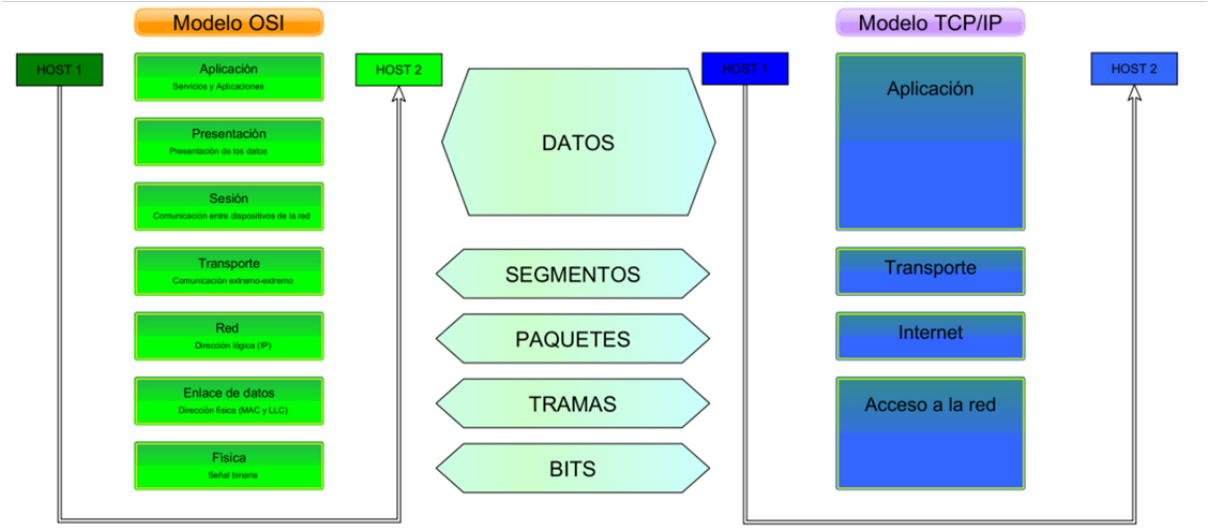


Figura 1 Modelo OSI y Modelo TCP/IP (Fuente: Artículo de universidad [3])

Como se puede observar en la Figura 1, el modelo TCP/IP es un modelo simplificado basado en el modelo OSI, el modelo TCP/IP absorbe las capas de aplicación, presentación y sesión para crear la capa de Aplicación de red y absorbe las capas Física y de enlace para crear la capa de enlace de datos. Pese a que este modelo se haya asentado como el modelo estandarizado utilizado en internet, esto no quiere decir que este haya sea el único modelo. Y es que en la década de los 70, el ingeniero John Day ya había comenzado a intuir que podrían haber problemas con el modelo TCP/IP, culminando todo en su libro de 2008 *Patterns in Network Architecture: A return to Fundamentals*[4], en el cual explica los problemas del modelo actual de Internet y ayuda a entender cómo podría la arquitectura RINA solucionar los problemas descritos.

1.3 Problema

Tal y como hemos mencionado anteriormente, actualmente Internet funciona mediante el modelo TCP/IP, aún cuando fue el modelo escogido para utilizar a través de Internet, esta suite de protocolos tiene problemas que con el paso del tiempo se están comenzando a hacer más y más evidentes. Podemos destacar los siguientes[5]:

● La necesidad de tener más capas: debido a la actual arquitectura de Internet, tenemos un número limitado de capas sobre la capa de enlace de datos y esto dificulta el poder añadir nuevos protocolos de red.
En caso de querer añadir nuevos protocolos de red, tendremos que añadir cierto overhead (p.e MPLS)

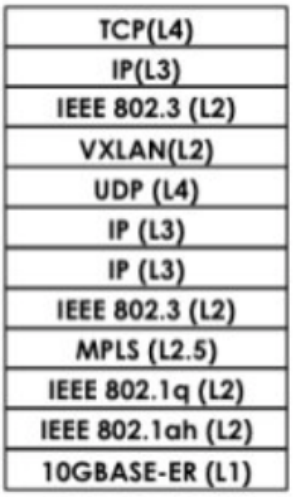


Figura 2 Problemas en el overlay de TCP/IP (Fuente: Director del proyecto)

● La complejidad trae problemas: siguiendo el punto anterior, el hecho de tener que añadir overheads cada vez que queramos aportar algo nuevo a Internet, hace que las PDU's se vuelvan cada vez más complejas, esto solo dificulta la comunicación y vuelve poco fiable la comunicación.

● Multihoming: el multihoming es la práctica en redes de Internet que permite que un equipo esté conectado a más de una red. El problema radica en que la red no puede saber que dos direcciones IP van al mismo dispositivo, para solucionar esto se han creado distintos protocolos como SCTP, SHIM6 o Multipath TCP, sin embargo esto como hemos indicado anteriormente, crea complejidad en los paquetes.

● Movilidad: el hecho de que en redes móviles nos tengamos que ir moviendo entre nodos implica un *handover,* término utilizado para definir el cambio de red troncal a la que está conectado un dispositivo móvil debido al desplazamiento hacía un nodo más cercano.
Las aplicaciones por su parte, necesitan una dirección que sea estable. Cuando un usuario necesita un handover, sin embargo, para que las redes sean escalables, su dirección IP debe cambiar ya que se conecta a diferentes redes. Una vez más, creando protocolos como pueden ser GTP o LISP se ha intentado solventar este problema, pero otra vez más caemos en el aumento de complejidad además de que algunos necesitan túneles, incrementando aún más el overhead en las PDU's.

- Seguridad: el simple hecho de añadir seguridad en las redes mediante protocolos como IPSec o TLS implica un overhead al tener que crear un túnel y tener que añadir una cabecera IP extra.

Tal y como hemos podido observar, la arquitectura TCP/IP trae consigo muchos problemas que acabarán afectando a la larga al desarrollo de nuevas tecnologías.

1.4 Actores implicados

En este apartado indicaremos qué actores se beneficiarán de este proyecto y porqué motivo.

Director y codirector del proyecto
El director de este proyecto es Davide Careglio y el codirector Jordi Perelló, será el encargado de supervisar y guiar el proyecto, asegurándose que se realice de forma correcta y continuada.

Autor del proyecto
El autor del proyecto es uno de los actores principales del proyecto debido a que será el encargado de llevar a cabo las pruebas/experimentos del proyecto y asegurarse que todos estos experimentos sean realizados correctamente, además de documentar todo el proceso.

i2cat
Se trata de un centro de investigación, donde uno de sus temas de investigación es RINA, por lo tanto también se pueden llegar a ver beneficiados de los resultados de este proyecto.

Usuarios de internet
En general, al tratarse de un proyecto enfocado en RINA, puede llegar a afectar a cualquier usuario que acceda a Internet, ya que el objetivo principal de RINA es reemplazar el modelo TCP/IP

2. Justificación

2.1 Estudios previos

La arquitectura de RINA se trata de un modelo de red relativamente nuevo y por lo tanto, aún está en proceso de investigación y prueba.
A pesar de esto, podemos encontrar trabajos e investigaciones sobre RINA, aunque gran parte de la información que hay sobre esta arquitectura es a un nivel teórico, podemos encontrar proyectos dignos de mención que pueden ayudarnos a entender mejor cómo funciona la implementación de RINA.
Mediante diversos estudios sobre RINA, también se ha llegado a comprobar lo siguiente:

- Solo requiere un protocolo de transferencia de datos genérico y un protocolo de plano de control genérico que se puede personalizar para múltiples casos de uso mediante políticas programables.

- Es asegurable por diseño [6], más confiable para transportar que TCP / IP [7].

- Soporta movilidad y multi-homing sin túneles ni protocolos especializados [8].

- Es compatible con un framework de QoS consistente desde la aplicación hasta el cable [9].

- Permite enrutamiento/reenvío personalizado minimizando el tamaño de la tabla de enrutamiento [10].

- Admite la renumeración dinámica de varias capas sin afectar a los servicios activos [11].

- Admite segmentación de redes sin protocolos especiales [12].

- Simplifica el concepto de virtualización de redes aplicando el paradigma de paravirtualización [13].

Para este proyecto sin embargo, nos basaremos en el proyecto ERASER, explicado a continuación

Proyecto ERASER

El proyecto ERASER (por sus siglas **E**xperimenting with **R**eal **A**pplication **s**pecific QoS Guarantees in a Large-scale **R**INA Demonstrator) fue un proyecto llevado a cabo en el año 2018 por la Universidad Politécnica de Cataluña (UPC) coordinado por el Dr. Jordi Perelló.
Este proyecto fue una prueba para evaluar el QoS (Quality of Service) que RINA puede garantizar en diferentes escenarios.
Específicamente, se crearon 2 escenarios diferentes en este proyecto:

- El primer escenario se trata de un escenario pequeño con solo 10 nodos donde se dispone de 1 cliente que accede a 1 servidor que está transmitiendo un video (VLC). De esta forma se le fue asignando diferentes prioridades al cliente para comprobar que se cumpliese el QoS que se asignaba al flujo de datos.

- El segundo escenario se trata de un escenario más complejo, el cual se observa mejor en la siguiente figura

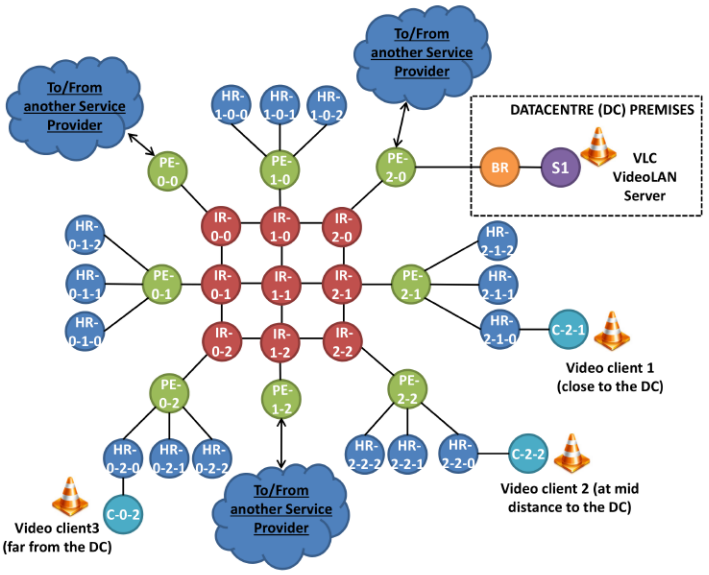


Figura 3 Escenario 2 ERASER (Fuente: Proyecto ERASER[14])

Como se puede observar, este escenario se trata de uno mucho más complejo, incluyendo una malla de routers en el centro. En este caso se intentó simular el acceso de clientes a corta distancia, media distancia y larga distancia.

El proyecto ERASER será la base de este proyecto, ya que ERASER fue desplegado mediante Fed4Fire[15], específicamente en el banco de pruebas de Virtual Wall en Ghent (Bélgica). Este proyecto se encargará de replicar las pruebas llevadas a cabo en este entorno virtual a un entorno local utilizando servidores ubicados en las instalaciones del grupo de investigación de banda ancha(CBA) de la Facultad de informática de Barcelona (FiB-UPC).

3. Alcance

3.1 Objetivos

El objetivo principal de este proyecto es poder desplegar la infraestructura llevada a cabo en el proyecto ERASER de forma local en los servidores localizados en el CBA. Tal y como hemos mencionado anteriormente en el apartado de Justificación, el proyecto ERASER fue desplegado en los servidores de Virtual Wall de Fed4Fire localizados en Bélgica, por lo que el objetivo principal será trasladar ese mismo proyecto a Barcelona.
Siendo este el objetivo principal, podemos igualmente segmentar el objetivo principal en los siguientes:

- Desplegar los dos escenarios presentados en el proyecto ERASER en un entorno simulado
- Desplegar los dos escenarios presentados en el proyecto ERASER en los servidores del CBA.
- Estudiar e investigar sobre la arquitectura de RINA
- Asegurarnos de que los resultados de replicar el experimento sean los correctos
- Investigar sobre la herramienta para poder implementar RINA: IRATI [16]
- Intentar continuar y experimentar escenarios de tráfico diferentes respecto a ERASER

3.2 Requisitos

En este apartado indicaremos los requisitos funcionales y no funcionales del proyecto

Funcionales

- En general, se debe poder replicar los experimentos realizados en ERASER
- Poder asignar una prioridad al flujo de datos
- Se debe poder instalar RINA en los servidores del CBA
- Se debe poder transmitir entre nodos RINA

No funcionales

- Disponibilidad: La plataforma tiene que estar transmitiendo para poder realizar las pruebas, por lo que necesitaremos un nivel mínimo de disponibilidad
- Escalabilidad: El proyecto tiene que quedar preparado para futuras ampliaciones para pruebas a escala más grande
- Usabilidad: Debe poder ser fácil desplegar un experimento
- Fiabilidad: Debe ser fiable de forma que un mismo experimento debe poder dar los mismos resultados siempre

3.3 Riesgos y obstáculos

A continuación detallaremos los riesgos y obstáculos que nos podremos encontrar a la hora de realizar

- Límite de tiempo: Al tratarse de un proyecto con un tiempo establecido, tendremos que tener cuidado con los inconvenientes que puedan surgir cuando se esté desarrollando el mismo.

- Desviación de la planificación: Puede que a medida que se realice el proyecto, tengamos que desviarnos de la planificación establecida en un inicio. Tendremos que contemplar estas posibilidades, ya que pueden significar tener que ocupar más tiempo en algún apartado más que en otro.

- Limitaciones de los servidores: Tendremos que tener en cuenta que ya no estamos tratando con los servidores de Virtual Wall, por lo que tendremos que considerar que tenemos un número limitado de servidores y estos a su vez, tienen ciertas limitaciones de hardware.

4. Metodología

En este proyecto utilizaremos la metodología en cascada, se trata de una metodología lineal donde dividiremos los procesos de desarrollo de este proyecto en partes:

- Empezaremos por estudiar la arquitectura de RINA y entenderla del todo para poder comenzar con el proyecto.
- Una vez el autor entienda cómo funciona la arquitectura de RINA, puede comenzar con las primeras pruebas en un entorno simulado.
- Cuando se tenga constancia de que se ha dominado este entorno simulado, podrá comenzar con las pruebas en el entorno real con los servidores físicos.

Para llevar a cabo las pruebas, se dispondrá de un repositorio en Github donde se podrá realizar un control de versiones además de poder guardar el código fuente. De esta manera también podemos asegurar la recuperación de versiones anteriores del proyecto.

También hay que tener en consideración que se tendrá una comunicación continua con los coordinadores del proyecto para asegurarse de que el proyecto se está desarrollando de forma correcta.

5. Descripción de las tareas

5.1 Organización y tareas

En este apartado detallaremos las tareas del proyecto, para cada una de ellas, daremos una breve descripción y se proporcionará una estimación de horas de cada tarea.
Hay que tener en cuenta que a pesar de que las tareas están detalladas en cuanto a su duración de la forma más precisa posible, pueden haber ciertas modificaciones para poder adaptar el proyecto en caso de necesidad.
Este proyecto tendrá una duración de un cuatrimestre, por lo que tendrá una duración de unos 112 días/490 horas aproximadamente, empezará en septiembre de 2021 y finalizará a mediados de febrero de 2022.
En cuanto a las horas dedicadas al proyecto semanalmente, se estima que se le dedique unas 30 horas semanales, aunque esto podría llegar a verse modificado como hemos indicado anteriormente. Esto puede deberse a causas externas de índole personal (enfermedad o defunción de un familiar, por ejemplo), debido a una mala gestión del tiempo del que se dispone, problema con los servidores que utilizaremos en el proyecto o una desviación en la planificación establecida.

5.1.1 T1. Gestión del proyecto

- **GdP1 - Reuniones**: Reuniones con el tutor para poder realizar el seguimiento del proyecto y analizar si el proyecto avanza correctamente. Las reuniones durarán aproximadamente 1.5 horas.
    - Duración aproximada: 25 horas

- **GdP2 - Alcance del proyecto**: Realizar un estudio del contexto del proyecto y del estado del arte, en este apartado además, justificamos el porqué del proyecto y también hablaremos aspectos como la metodología a seguir o los riesgos que pueden surgir al realizar el proyecto.
    - Duración aproximada: 25 horas

- **GdP3 - Planificación del proyecto**: Indicar la planificación del proyecto, indicando las distintas tareas que se llevarán a cabo en el mismo y mediante las cuales vamos a completar el proyecto, también indicando los requisitos que necesitaremos para concluir las tareas.
    - Duración aproximada: 15 horas

- **GdP4 - Presupuesto y sostenibilidad del proyecto**: Analizar el coste que puede conllevar llevar a cabo el proyecto, además de indicar su impacto en el mercado y la sostenibilidad del mismo
    - Duración aproximada: 30 horas

- **GdP5 - Documento final de la gestión del proyecto**: Finalmente tendremos que reunir el trabajo en uno y aprovechar esto para pulir y mejorar en todo los aspectos lo que se pueda.
    - Duración aproximada: 20 horas
    - Dependencias: PdP2,PdP3,PdP4

5.1.2 T2. Implementación

- **IdP1 - Estudio inicial sobre la arquitectura en RINA**: Investigar sobre la arquitectura RINA y sobre qué nos aporta en contra de lo que ya se ha establecido.
    - Duración aproximada: 30

- **IdP2 - Preparar entorno virtual para realizar las pruebas en RINA**: Se tendrán que desplegar máquinas virtuales donde se llevarán a cabo las simulaciones. Por lo que se tendrá que instalar un sistema operativo y clonar estas máquinas virtuales.
    - Duración aproximada: 10

- **IdP3 - Estudio sobre la herramienta IRATI**: Investigar sobre el stack de IRATI y las aplicaciones de pruebas incluidas para poder realizar los experimentos.
    - Duración aproximada: 20

- **IdP4 - Despliegue simple en entorno virtualizado**: Se realizará una implementación simple conectando primero dos máquinas virtuales entre sí y se ejecutarán aplicaciones de pruebas para validar el escenario
    - Duración aproximada: 50
    - Dependencia: IdP2, IdP3

5.1.3 T3. Validación

- **VdP1 - Implementación del escenario simplificado de ERASER:** Se desplegará RINA en 10 máquinas virtuales y se harán pruebas para comprobar la correcta configuración del escenario.
    - Duración aproximada: 60
    - Dependencia: IdP

- **VdP2 - Validación del escenario simplificado de ERASER:** Se establecerán diferentes flujos de datos bidireccionales entre aplicaciones con diferentes requisitos de Quality of Service (QoS) y se validarán las prestaciones comparando los resultados con los obtenidos en ERASER.
    - Duración aproximada: 20
    - Dependencia: VdP1

- **VdP3 - Implementación del escenario completo de ERASER:** Se realizará la implementación completa con 37 maquinas virtuales del escenario más complejo de ERASER usando los servidores del CBA (FIB).
    - Duración aproximada: 90
    - Dependencia: VdP1

- **VdP4 - Validación del escenario completo de ERASER:** Se establecerán diferentes flujos de datos bidireccionales entre aplicaciones con diferentes requisitos de Quality of Service (QoS) y se validarán las prestaciones comparando los resultados con los obtenidos en ERASER.
    - Duración aproximada: 30
    - Dependencia: VdP3

- **VdP5 - Validación de la transmisión de un video HD (High Definition):** Se establecerán 3 conexiones entre un servidor de video y 3 clientes para la transmisión de un video HD. Los 3 clientes estarán posicionas a diferentes distancias del servidor. El objetivo es comprobar que la calidad del video se mantiene con RINA mientras no es posible con los sistemas actuales.
    - Duración aproximada: 20
    - Dependencia: VdP3

- **VdP6 - Experimentación:** Una vez validado el escenario completo, se realizarán pruebas diferentes cambiando diferentes parámetros como: configuración de los nodos RINA, requisitos de QoS de las aplicaciones, características del tráfico, para validar las prestaciones de RINA.
    - Duración aproximada: 30
    - Dependencia: VdP4.

5.1.4 T4. Documentación

- **DdP1 - Documentación del proyecto**: redactar la documentación del proyecto:
    - Duración estimada: 35h

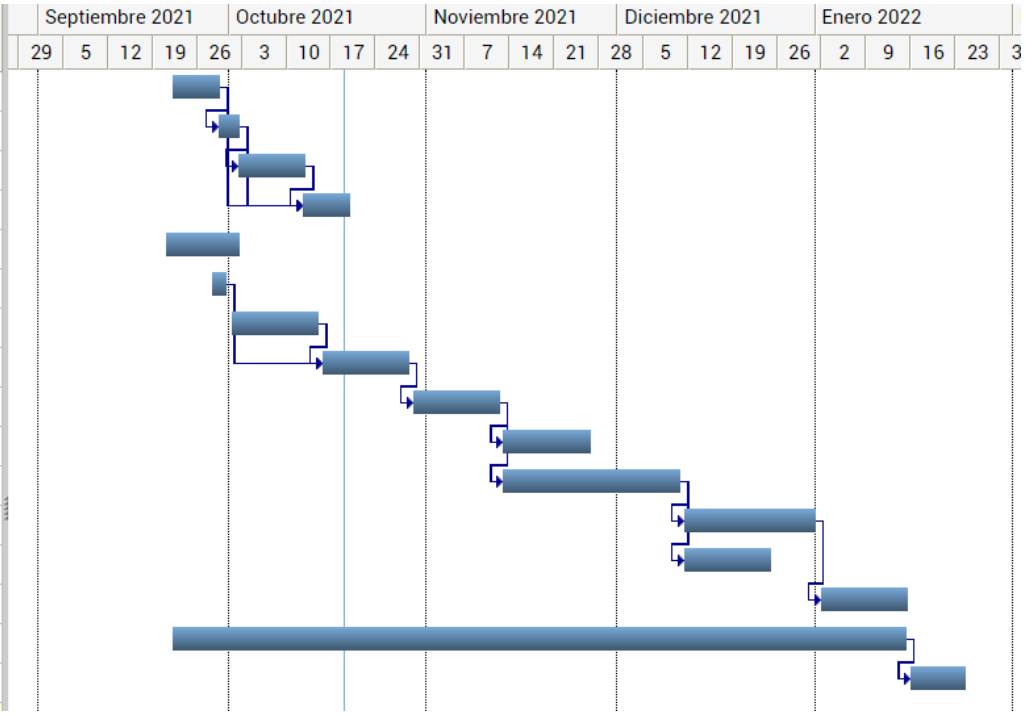
- **DdP2 - Revisión del documento**: Revisión de la documentación y posibles mejoras al documento final.
    - Duración estimada: 10h
    - Dependencias: DdP1

Tareas	Horas
T1. Gestión del proyecto	
GDP2 - Alcance del proyecto	25
GDP3 - Planificación del proyecto	25
GDP4 - Presupuesto y sostenibilidad del proyecto	15
GDP5 - Documento final de la gestión del proyecto	30
	115
T2. Implementación	
IdP1 - Estudio inicial sobre la arquitectura en RINA	30
IdP2 - Preparar entorno virtual para realizar las pruebas en RINA	10
IdP3 - Estudio sobre la herramienta IRATI	20
IdP4 - Despliegue simple en entorno virtualizado	50
	110
T3. Validación	
VdP1 - Implementación del escenario simplificado de ERASER	60
VdP2 - Validación del escenario simplificado de ERASER	20
VdP3 - Implementación del escenario completo de ERASER	90
VdP4 - Validación del escenario completo de ERASER	30
VdP5 - Validación de la transmisión de un video HD	20
VdP6 - Experimentación	30
	250
T4. Documentación	
DdP1 - Documentación del proyecto	35
DdP2 - Revisión del documento	10
	45
TOTAL	**490**

Tabla 1. Distribución de horas

6. Diagrama de Gantt

Nombre	Duración	Inicio	Fin
GDP2 - Alcance	6días	09/21/2021	09/28/2021
GDP3 - Planificación	4días	09/28/2021	10/01/2021
GDP4 - Presupuesto y sostenibilidad	7días	10/01/2021	10/11/2021
GDP5 - Documento final de la gestión del proyecto	6días	10/11/2021	10/18/2021
IdP1 - Estudio inicial sobre la arquitectura en RINA	10días	09/20/2021	10/01/2021
IdP2 - Preparar entorno virtual para realizar las pruebas en RINA	3días	09/27/2021	09/29/2021
IdP3 - Estudio sobre la herramienta IRATI	10días	09/30/2021	10/13/2021
IdP4 - Despliegue simple en entorno virtualizado	10días	10/14/2021	10/27/2021
VdP1 - Implementación del escenario simplificado de ERASER	10días	10/28/2021	11/10/2021
VdP2 - Validación del escenario simplificado de ERASER	10días	11/11/2021	11/24/2021
VdP3 - Implementación del escenario completo de ERASER	20días	11/11/2021	12/08/2021
VdP4 - Validación del escenario completo de ERASER	15días	12/09/2021	12/29/2021
VdP5 - Validación de la transmisión de un video HD	10días	12/09/2021	12/22/2021
VdP6 - Experimentación	10días	12/30/2021	01/12/2022
DdP1 - Documentación del proyecto	82días	09/21/2021	01/12/2022
DdP2 - Revisión del documento	7días	01/13/2022	01/21/2022



7. Gestión del riesgo

En este apartado indicaremos qué riesgos pueden afectar al correcto desarrollo del proyecto

Problemas en el entendimiento de la arquitectura de RINA
Debido a que se trata de una arquitectura que se encuentra en constante desarrollo, puede llegar a complicar

Hardware
Problemas debido a errores de hardware, pueden haber errores en los equipos que retrasen el proyecto.
Estos errores podrían ocurrir tanto en el equipo del autor como en los equipos localizados en el CBA. A pesar de esto, debido a que la información y los avances estarán guardados en la nube, las pérdidas no serían demasiado grandes.

- Probabilidad: Media
- Solución: En este caso, si se trata de un fallo que inutiliza por completo el equipo, no nos quedaría más remedio que esperar a que se solucione el fallo. Si se trata de un fallo que nos permita continuar con el proyecto, aunque con alguna dificultad, podríamos continuar mientras esperamos a que se solucione el fallo de hardware.

Desviación del planning original
Puede que nos desviemos del planning originalmente diseñado debido a una mala estimación en el tiempo que hemos asignado a cada tarea, esto puede hacer que el proyecto no se realice en los plazos establecidos.

- Probabilidad: Alta
- Solución: Realizar una reestructuración del planning original que se adapte a las complicaciones que surjan.

Bugs
RINA es una arquitectura que está en constante desarrollo, esto quiere decir que el código está en continuo cambio. Debido a esto, pueden surgir bugs que dependiendo de su gravedad, podrían llegar a dejar al proyecto congelado. En caso de tratarse de un bug debido al mal uso por parte del autor, se podría detectar y no conlleva una pérdida de tiempo muy grande, pero si se trata de un bug del stack de IRATI, resultaría en un problema más grande.

- Probabilidad: Media
- Solución: En caso de tratarse de un bug por la mala utilización del stack de IRATI se podría solucionar indagando en el fallo, en cambio, en caso de ser un fallo interno del código de IRATI, habría que contactar con los diseñadores para indicarles acerca del bug.

8. Presupuesto

8.1 Identificación de los costes

En este apartado detallaremos los costes asociados al proyecto. Ahora pasaremos a identificar y a detallar estos costes.

8.1.1 Costes de personal

Para empezar, definiremos el coste que tiene el personal asociado a las tareas que creamos en el diagrama de Gantt.
Los roles que hemos escogido para que desarrollen las tareas son los siguientes:

- Manager del proyecto: Es el encargado de supervisar y asegurarse que el proyecto se esté realizando de una forma adecuada al planning originalmente hecho.

- Desarrollador especializado en redes: Es el encargado de la implementación a nivel técnico del proyecto. Sus responsabilidades son desplegar la infraestructura que se pide en el proyecto, tanto la simple como la compleja.

- Tester: Es el responsable de realizar las pruebas pertinentes dentro del proyecto para garantizar la robustez del proyecto y confirmar el correcto funcionamiento del mismo.

Hay que tener en cuenta que hemos tenido que multiplicar por 1.3 el salario bruto para tener en cuenta la seguridad social de los puestos.

Puesto	Coste
Manager del proyecto	40000*1.3 €/año = 52000€ = 25 €/hora
Desarrollador especializado en redes	33000*1.3 €/año = 42900€ = 20,63 €/hora
Tester	28000*1.3 €/año = 36400€ = 17,50 €/hora

Tabla 2. Costes de los integrantes del proyecto [17] [18] [19]

8.1.2 Costes de material

Una vez descrito los puestos de trabajo necesarios para poder llevar a cabo el proyecto, tenemos que tener en cuenta que a la hora de realizar el mismo, necesitaremos de material para realizar el mismo.
De cara al cálculo, tomaremos en consideración el hecho de que el hardware tiene un coste de amortización. La fórmula de amortización es la siguiente:

$$Amortización = \frac{coste\ hardware}{4\ años\ x\ 220\ días\ laborables\ x\ 6\ horas\ de\ dedicación\ al\ día} x\ horas\ totales$$

También hay que considerar que el software que utilizaremos para desarrollar el proyecto es gratis, ya sea por tratarse de código abierto o por tratarse de software libre.

Material	Coste	Amortización
Hardware		
PC personal	1000	93€
Servidores CBA	3600 (4 servidores de 900 euros)	335€
Software		
IRATI stack	Gratis	-
Meet	Gratis	-
Github	Gratis	-
Virtualbox	Gratis	-
Software asociado a IRATI p.e librerías	Gratis	-
Google Meet	Gratis	-

Tabla 3. Costes de los materiales (Tabla propia)

8.1.3 Costes indirectos

Además de los costes de los materiales y software, también son relevantes los costes indirectos que provocará nuestro proyecto. Estos son los siguientes:

Gasto	Coste
Electricidad	80€/mes
Internet	50€/mes

Tabla 4 Precio medio de electricidad e internet [20][21]

8.1.4 Costes de contingencia e incidencias

Para finalizar, indicaremos los costes por los posibles problemas que puedan surgir en mitad del desarrollo del proyecto. Estas incidencias se pueden encontrar detalladas en el apartado *7. Gestión de riesgo*.

Para empezar este apartado, comenzaremos indicando que hemos dejado un 10% para un plan de contingencia sobre incidencias que no hayamos detectado. Teniendo en cuenta que el proyecto tiene un costo de 15892,60€, el presupuesto para el plan de contingencia sería de unos 1589,26€.

Riesgo	**Probabilidad**	**Tiempo**	**Coste estimado**	**Coste**
Hardware	10%	10h	206,30€	20.63€
Desviación del planning original	50%	10h	250€	125€
Bugs	30%	25h	515.75€	154.73€
TOTAL				**300,36€**

Tabla 5. *Costes de contingencia (Tabla propia)*

8.2 Estimación de costes

ID	Título	Duración	Responsable	Coste
GdP1	Reuniones	25h	Project Manager	625€
GdP2	Alcance del proyecto	25h	Project Manager	625€
GdP3	Planificación del proyecto	15h	Project Manager	375€
GdP4	Presupuesto y sostenibilidad del proyecto	30h	Project Manager	750€
GdP5	Documento final de la gestión del proyecto	20h	Project Manager	500€
IdP1	Estudio inicial sobre la arquitectura en RINA	30h	Desarrollador	618,90€
IdP2	Preparar entorno virtual para realizar las pruebas en RINA	10h	Desarrollador	206,30€
IdP3	Estudio sobre la herramienta IRATI	20h	Desarrollador	412,60€
IdP4	Despliegue simple en entorno virtualizado	20h	Desarrollador	412,60€
VdP1	Implementación del escenario simplificado de ERASER	60h	Desarrollador	1237,80€
VdP2	Validación del escenario simplificado de ERASER	20h	Tester	350€
VdP3	Implementación del escenario completo de ERASER	90h	Desarrollador	1856,70€
VdP4	Validación del escenario completo de ERASER	30h	Tester	403,80€
VdP5	Validación de la transmisión de un video HD	20h	Tester	525€
VdP6	Experimentación	30h	Desarrollador	618,90€
DdP1	Documentación del proyecto	35h	Project Manager	875€
DdP2	Revisión del documento	10h	Project Manager	250€
--------	PC personal	--------	--------	1000€
--------	Servidores CBA	--------	--------	3600€
--------	IRATI stack	--------	--------	--------
--------	Meet	--------	--------	--------
--------	Github	--------	--------	--------
--------	Virtualbox	--------	--------	--------
--------	Software asociado a IRATI p.e librerías	--------	--------	--------
--------	Google Meet	--------	--------	--------
--------	Electricidad	--------	--------	400€
--------	Internet	--------	--------	250€
--------	Contingencia	--------	Project Manager	1589,26€
--------	Hardware	--------	--------	20,63€
--------	Desviación del planning original	--------	--------	125€
--------	Bugs	--------	--------	154,73€
Total				**17782,22€**

Tabla 6. *Resumen de costes (Tabla propia)*

8.3 Control de gestión

En esta sección indicaremos cuáles serán las métricas que servirán para mantenernos dentro de los costes que hemos calculado y en caso de ser necesario, poder actuar si se está desviando algún coste.

Las métricas que utilizaremos serán las siguientes:

- Desvío de mano de obra en precio (desvío de coste por tarifa) = $(coste\ estimado\ -\ coste\ real)\ *\ consumo\ horas\ real$

- Desvío de mano de obra en consumo (desvío en eficiencia) = $(consumo\ horas\ estimadas\ -\ consumo\ horas\ reales)\ *\ coste\ estimado$

9. Sostenibilidad

9.1 Autoevaluación

Tomando en cuenta lo que he visto durante la carrera, normalmente siempre nos hemos enfocado en el aspecto ambiental y social de la sostenibilidad.
Puedo poner de ejemplo la asignatura de AC (Arquitectura de Computadores), donde tuvimos una charla muy interesante que hablaba sobre los residuos que provocamos al utilizar aparatos electrónicos.

Además de esto, también en la asignatura de ASO (Administración de Sistemas Operativos) tuvimos que asistir en la puesta en marcha de equipos que serían reciclados y enviados a países en vías de desarrollo.

Estas dos actividades me ayudaron a entender más en profundidad la sostenibilidad desde un punto de vista social y ambiental. Sin embargo, al realizar la encuesta propuesta de sostenibilidad me he dado cuenta que me faltan muchos conocimientos sobre la parte económica de la sostenibilidad, ya que es sobre la que menos he tenido la oportunidad de estudiar.

Como reflexión final, quiero indicar que como he comentado anteriormente, soy consciente del impacto social y ambiental de los proyectos, sin embargo, no tengo muchos conocimientos sobre el impacto económico de los mismos.
Por lo que creo que es muy enriquecedor entender que el impacto económico de un proyecto es igual de importante que el impacto social o ambiental. Esto se debe a que, al tratarse de uno de los pilares (económico, social y ambiental), si uno de estos falla, probablemente el proyecto no sea sostenible.

9.2 Dimensión Económica

¿Has estimado el coste de la realización del proyecto (recursos humanos y materiales)?
En el apartado *8. Presupuesto* hemos indicado el coste humano que tendrá nuestro proyecto, además de detallar el coste material e indicar un plan de contingencia en caso de haber desviaciones en cuanto al plan original.

¿Cómo se resuelve actualmente el problema que quieres abordar (estado del arte)? ¿En qué mejorará económicamente tu solución a las existentes?
El problema actualmente está resuelto utilizando las infraestructuras que proporciona Fed4Fire (Proyecto ERASER). El objetivo del proyecto es poder desplegar esta infraestructura de forma local en los servidores del CBA.
La mejora económica que dará nuestra solución es que ahora los servidores serán locales, por lo que no se tendrá que pagar los servicios de terceros para poder mantener el proyecto.

9.3 Dimensión ambiental

¿Has estimado el impacto ambiental que tendrá la realización del proyecto? ¿Te has planteado minimizar el impacto, por ejemplo, reutilizando recursos?
El impacto ambiental que tiene este proyecto es el que provocarán los servidores utilizados para realizar el mismo. Sin embargo, al principio del proyecto, se comentó con el director del proyecto la posibilidad de tener que comprar los servidores, en cambio, hemos podido reutilizar otros servidores.

¿Cómo se resuelve actualmente el problema que quieres abordar (estado del arte)? ¿En qué mejorará ambientalmente tu solución a las existentes?
Actualmente, el proyecto se resuelve mediante los servidores de Fed4Fire+ localizados en Bélgica. Ambientalmente, este proyecto mejorará respecto a su versión remota de forma que los recursos que utilizaremos son reutilizados (servidores).

9.4 Dimensión social

¿Qué crees que te va a aportar a nivel personal la realización de este proyecto?
A nivel personal, sé que este proyecto me aportará mucho en una gran cantidad de campos. De cara a un nivel tecnológico, me aportará el conocimiento de una tecnología que aún está en desarrollo como lo es RINA. También me aportará el conocimiento sobre cómo es trabajar en un proyecto de investigación, ya que nunca he hecho un proyecto tan grande como este.

¿Cómo se resuelve actualmente el problema que quieres abordar (estado del arte)? ¿En qué mejorará socialmente (calidad de vida) tu solución a las existentes?
Actualmente, el proyecto se resuelve mediante los servidores de Fed4Fire, localizados en Bélgica. A nivel social, será más fácil cualquier tipo de gestión del proyecto desde los servidores locales que si se continuase con los servidores remotos.

¿Existe una necesidad real del proyecto?
Como se ha comentado anteriormente, este proyecto se basa en un proyecto (ERASER) realizado en servidores externos, provistos por Fed4Fire. La necesidad del proyecto nace de la necesidad que se tiene de poder realizar más experimentos, estos se podrán llevar a cabo más fácilmente si se dispone de los servidores localizados en las mismas instalaciones del CBA.

10. Partes de la infraestructura de RINA, indagando más profundamente en RINA

Para empezar a indagar más en profundidad sobre RINA, tenemos que pensar en el trasfondo que tiene[4]. Y es que, a pesar de la guerra de protocolos que tuvo lugar entre los años 70 y 90, es un paradigma que aún se sigue investigando, por lo que ha sobrevivido al paso de los años.

RINA es el resultado de un esfuerzo por elaborar principios generales en redes informáticas que se aplican en todas las situaciones. RINA es la arquitectura específica, la implementación, la plataforma de prueba y el despliegue del modelo informalmente conocido como modelo IPC.

La entidad básica de RINA es el Proceso de aplicación distribuida o DAP(por sus siglas en inglés), que con frecuencia corresponde a un proceso en un host. Dos o más DAP constituyen una Facilidad de aplicación distribuida o DAF (Distributed Application Facility). Estos DAP se comunican utilizando el Protocolo de aplicación distribuida Común o CDAP(por sus siglas en inglés Common Distributed Application Protocol), intercambiando datos estructurados en forma de objetos.

Estos objetos están estructurados en una Base de Información de Recursos o RIB, que les proporciona un esquema de nombres y una organización lógica. CDAP proporciona seis operaciones básicas en los objetos de un DAP remoto: crear, eliminar, leer, escribir, iniciar y detener.

Para intercambiar información, los DAP necesitan una instalación subyacente que les proporcione servicios de comunicación. Esta instalación es otra DAF, denominada Instalación de IPC distribuida o DIF, cuya tarea es proporcionar servicios de IPC en un determinado ámbito. Los DAPs de un DIF se denominan Procesos IPC o IPCPs.

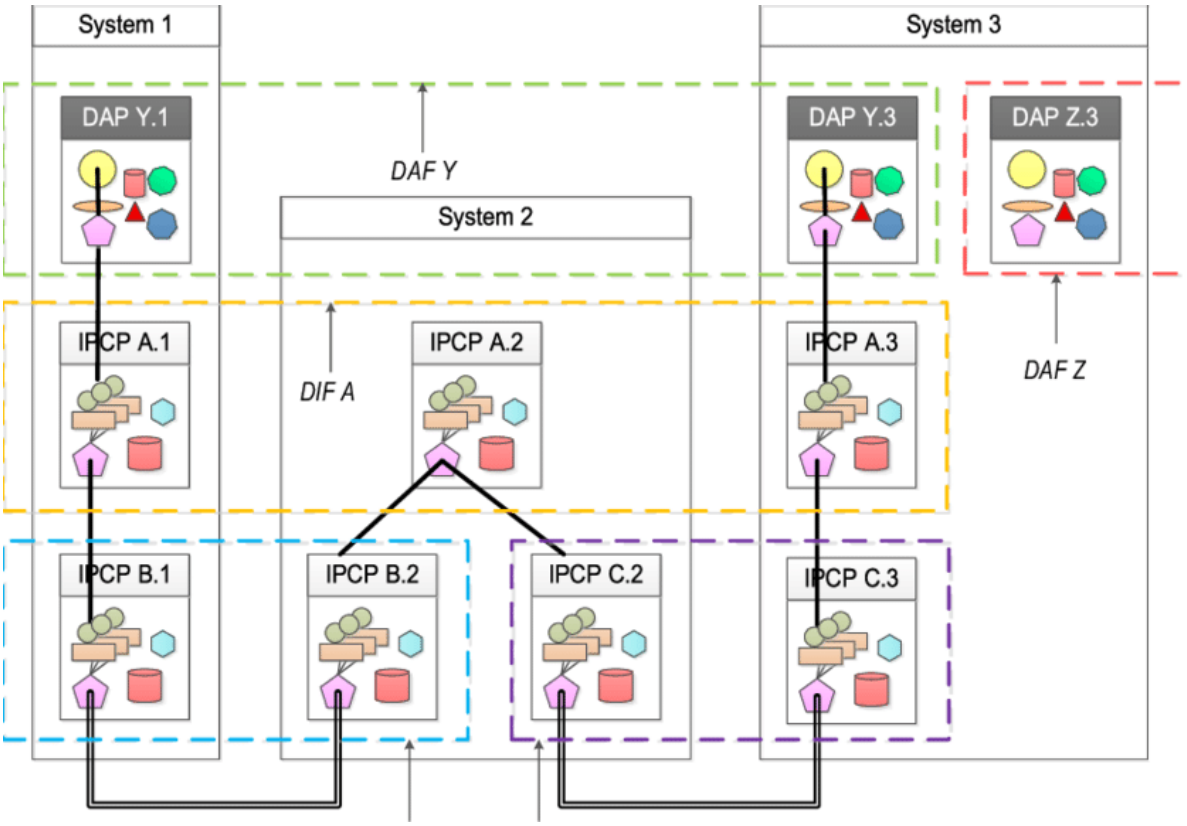


Figura 4 Esquema sobre el funcionamiento de los DAF/IPC/DAP. Fuente(Árticulo de Research Gate [22])

Los DIF, al ser DAF, a su vez utilizan otros DIF subyacentes, llegando hasta el DIF de la capa física que controla los cables y los conectores. Aquí es donde viene la recursividad de RINA. Como se muestra en la Figura 5, las redes RINA suelen estar estructuradas en DIF de alcance creciente. Extrapolando el caso del ejemplo de la figura, podríamos llegar a imaginar como podría funcionar la red mediante RINA: la capa más alta sería la más cercana a las aplicaciones, correspondiente al correo electrónico o sitios web por ejemplo. Las capas más bajas agregarían y multiplexarían el tráfico de las capas más altas, correspondientes a las redes troncales de los ISP. Los DIF de múltiples proveedores (como Internet público u otros) flotan sobre las capas del ISP. En este modelo se distinguen tres tipos de sistemas: hosts, que contienen DAPs; enrutadores interiores, internos a una capa; y enrutadores de borde, en los bordes de una capa, donde los paquetes suben o bajan una capa.

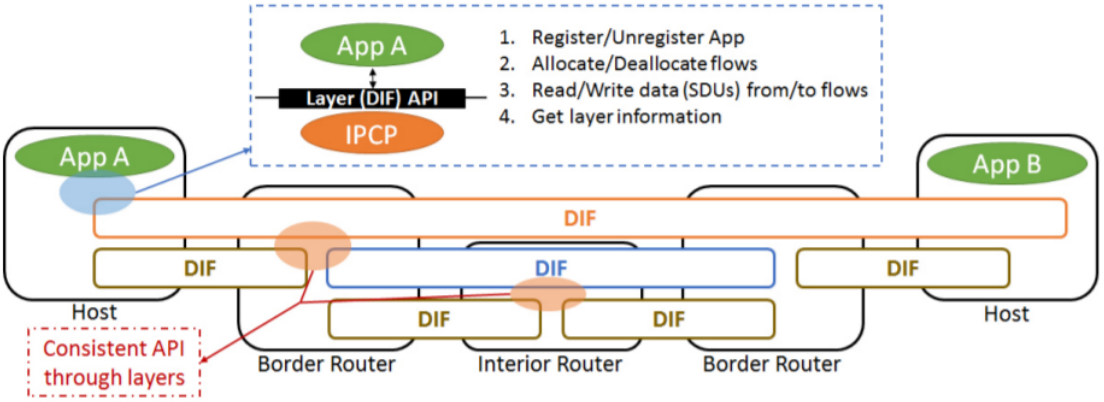


Figura 5 Ejemplo de modelo de sistema funcionando bajo RINA. Fuente: Proyecto ERASER

Un DIF permite que un DAP asigne flujos a uno o más DAP, simplemente proporcionando los nombres de los DAP específicos y los parámetros de QoS deseados, como límites en la pérdida de datos y latencia, entrega ordenada o fuera de orden, fiabilidad, etc...

Es posible que los DAP no confíen en el DIF que están utilizando y, por lo tanto, pueden proteger sus datos antes de escribirlos en el flujo a través de un módulo de protección SDU, por ejemplo, cifrándolos, este es uno de los puntos de estudio de IRATI también, la implementación de la seguridad en su sistema de simulación de RINA.

Todas las capas RINA tienen la misma estructura y componentes y proporcionan las mismas funciones; se diferencian únicamente en sus configuraciones o políticas. Esto refleja la separación de mecanismo y política en los sistemas operativos.

En resumen, RINA mantiene los conceptos de PDU y SDU, pero en lugar de poner capas por función, pone capas por alcance. En lugar de considerar que diferentes escalas tienen diferentes características y atributos, considera que toda comunicación tiene fundamentalmente el mismo comportamiento, solo que con diferentes parámetros.

Por lo tanto, RINA es un intento de conceptualizar y parametrizar todos los aspectos de la comunicación, eliminando así la necesidad de protocolos específicos.

10.1 Seguridad

Para dar cabida a la seguridad, RINA requiere que cada DIF/DAF especifique una política de seguridad, cuyas funciones se muestran en la Figura 6 . Esto permite asegurar no solo las aplicaciones, sino también las redes de backbones.
Una red pública es simplemente un caso especial donde la política de seguridad no hace nada. Esto puede generar sobrecarga para las redes más pequeñas, pero escala mejor con redes más grandes porque las capas no necesitan coordinar sus mecanismos de seguridad: se estima que Internet actual requiere alrededor de 5 veces más entidades de seguridad distintas que RINA.[22] Entre otras cosas, la política de seguridad también puede especificar un mecanismo de autenticación; esto hace obsoletos los cortafuegos y las listas negras porque un DAP o IPCP que no puede unirse a un DAF o DIF no puede transmitir ni recibir. Los DIF tampoco exponen sus direcciones IPCP a capas superiores, lo que evita una amplia clase de ataques de intermediarios.

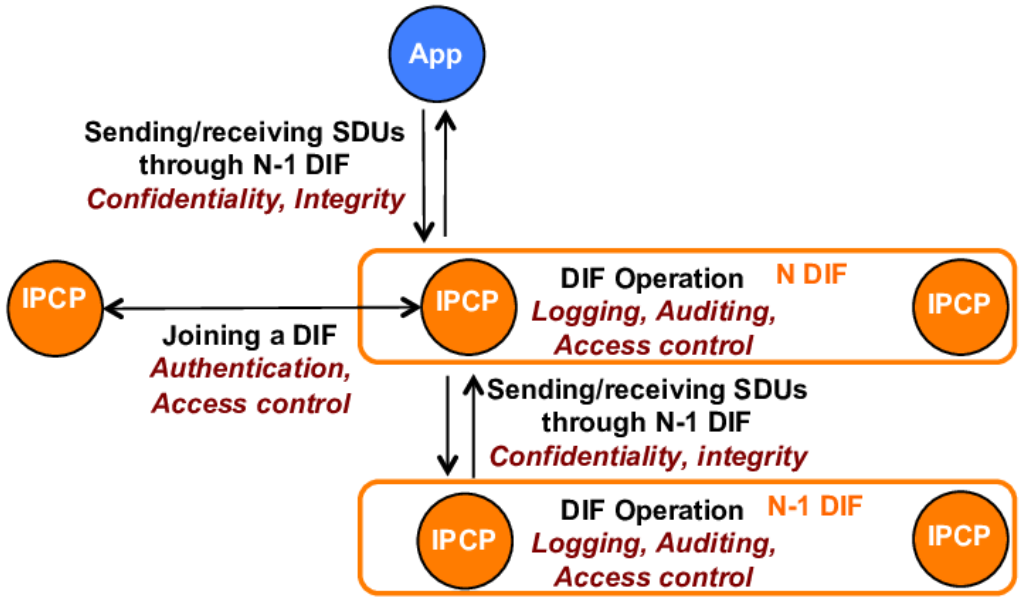


Figura 6 Ilustración de la seguridad en RINA. Fuente: Research Gate[23]

Por ejemplo, dado que RINA no tiene TWH(Three-way-handshake), no tiene mensajes de control correspondientes que se puedan falsificar o estados que se puedan usar incorrectamente, como en un flow del bit SYN.

11. IRATI

IRATI es la herramienta que utilizaremos para poder realizar la mayor parte del proyecto. Se trata de una implementación de la propia arquitectura de RINA hecha en código abierto para sistemas Linux.

Tal y como hemos hablado anteriormente en este proyecto, la necesidad de avanzar respecto a las limitaciones de TCP/IP nos lleva a la implementación de RINA en sistemas reales. IRATI(Investigating RINA as an Alternative to TCP/IP) funciona con distintos componentes.

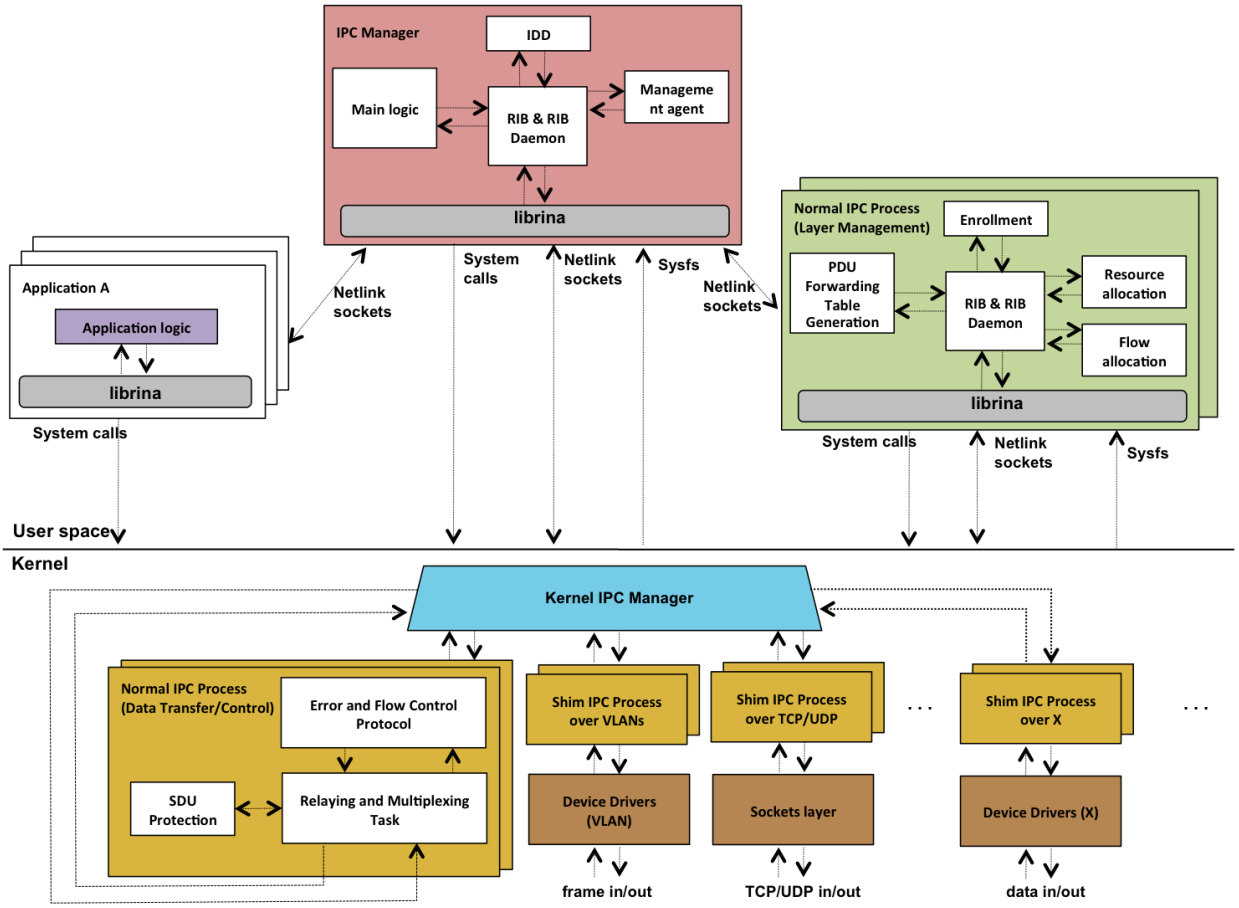


Figura 7 . Arquitectura de IRATI (Fuente IRATI stack[23])

11.1 Componentes principales de IRATI

Los componentes de IRATI[23] se podrían dividir en 4 paquetes principales:

Daemons (rinad): Paquete que contiene dos daemons cuya implementación está hecha en C++:

- IPC Manager Daemon (rinad/ src/ipcm): El IPC Manager Daemon es el componente principal de la gestión del IPC en el sistema. Este daemon actua tanto como un administrador de procesos IPC y como intermediario entre las aplicaciones y los procesos de IPC(cumpliendo funciones como por ejemplo mapeando la asignación de flujo o las solicitudes de registro de aplicaciones)

- IPC Process Daemon (rinad / src / ipcp): Este daemon es el encargado de implementar los componentes de gestión de capa de un proceso IPC (inscripción, asignación de flujo, etc..). Hay que destacar que hay uno de estos daemon por cada proceso IPC en ejecución en el sistema.

Librina (librina): El paquete librina contiene todas las librerias de IRATI que se han introducido para abstraer del usuario todas las interacciones del kernel (como las syscalls). Librina proporciona funcionalidades a los programas de RINA en el espacio de usuario a través de extensiones de lenguaje de scripting(bash/Perl por ejemplo) o bibliotecas enlazables estáticamente o dinámicamente (es decir, para programas C / C ++). Librina es más un framework / middleware que una biblioteca: tiene su propio modelo de memoria (explícito, sin garbage collection), su modelo de ejecución está controlado por eventos y usa mecánicas de concurrencia (tiene sus propios hilos) para hacer parte de su trabajo.

Componentes del kernel (linux / net / rina). El kernel contiene la implementación de los componentes de control de transferencia de datos / transferencia de datos de los procesos IPC normales, así como la implementación de shim DIF, que generalmente necesitan acceder a funcionalidades que solo están disponibles en el kernel. El Kernel IPC Manager (KIPCM) gestiona el tiempo de vida (creación, destrucción, monitoreo) de las otras instancias de componentes en el kernel, así como su configuración.
También proporciona coordinación en la frontera entre los diferentes procesos de IPC.

Aplicaciones de prueba y herramientas (rina-tools). Este paquete contiene aplicaciones de prueba y herramientas para probar y depurar el protocolo RINA. En este momento, el paquete rina-tools contiene la aplicación rina-echo-time, que puede funcionar tanto en modo "echo" (comportamiento similar al ping entre dos instancias de la aplicación) como en modo "performance" (comportamiento similar al iperf). Debido a problemas con las librerias, la aplicación de rina-echo-time no se encuentra disponible, para las pruebas utilizaremos la aplicación t-gen para gerar el tráfico.

A pesar de haber estos cuatro paquetes como hemos comentado anteriormente, nos centraremos en explicar los dos componentes que más hemos tenido que tener en cuenta para la realización del proyecto: **IPCM** e **IPCP**

11.2 IPC Manager Daemon

La clase rinad: IPCManager es la clase principal del programa, que representa el modelo de datos del IPC Manager y contiene todos los datos y funcionalidades que utilizan los subprocesos del IPC Manager.
Dado que solo hay un IPC Manager por sistema de procesamiento, solo se crea una instancia de rinad::IPCManager. La clase rinad::IPCManager expone un conjunto de métodos que pueden usarse para realizar operaciones de configuración en IRATI. Los métodos más importantes se utilizan para: creación de procesos IPC, la destrucción de procesos IPC, la asignación de procesos IPC a DIF y el registro de procesos IPC a DIF.

El primer paso que debe realizar el IPC Manager es analizar el archivo JSON que contiene la configuración del prototipo RINA para ese sistema. Esto se hace al comienzo de la ejecución del programa, antes de que se inicie el thread del script, el thread de la consola y el bucle de eventos. El propósito del módulo de análisis de configuración es llenar la instancia rinad::RinaConfiguration contenida en la instancia rinad::IPCManager, usando la información contenida en el archivo de configuración(normalmente y por defecto el ipcmmanager.conf).

IPC Manager implementa un CLI a través de un thread dedicado. Los comandos disponibles actualmente son los siguientes:

- help: muestra la lista de comandos disponibles o el uso de un comando específico.
- exit: sale de la consola.
- create-ipcp: crea un nuevo proceso de IPC.
- destroy-ipcp: destruye un proceso de IPC existente.
- list-ipcps: enumera los procesos de IPC existentes con información asociada.
- list-ipcp-types: enumera los tipos de procesos de IPC disponibles actualmente en el sistema.
- Assign-to-dif: Asignar un proceso de IPC a un DIF.
- register-at-dif: Registre un proceso de IPC dentro de un DIF.
- unregister-from-dif: anula el registro de un proceso de IPC de un DIF.
- enroll-to-dif: inscribe un proceso de IPC en un DIF.
- consulta-rib. Muestra la información de los objetos presentes en RIB de un proceso IPC.

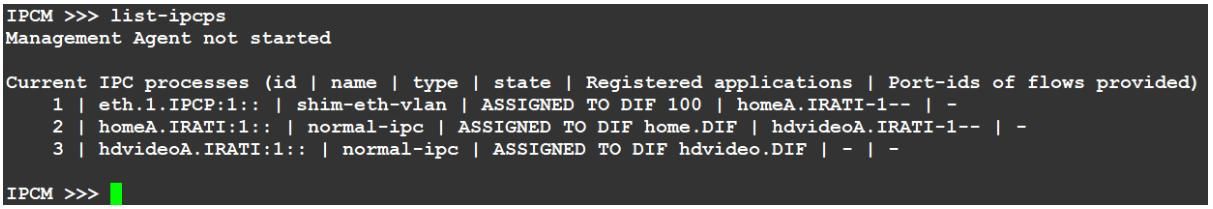


Figura 8 Ejemplo de ejecución de IPC Manager. Listamos los procesos IPC existentes.

11.3 IPCP Daemon

El IPCP Daemon es un contenedor de diferentes componentes que proporcionan funciones de ayuda en general (por ejemplo, codificación/decodificación de objetos, generación/análisis de mensajes CDAP[Common Distributed Application Protocol] o el RIB Daemon) o implementan funcionalidades de administración de capas (por ejemplo, inscripción, asignación de flujo, administración de espacio de nombres, asignación de recursos o gestión de seguridad).
El IPCP puede interactuar con el IPC Manager Daemon (IPCM) a través de la clase proxy *ExtendedIPCManager* de librina, que traduce las invocaciones de funciones a mensajes NetLink (NL) hacia el proceso IPCM. De manera similar, el IPCP Daemon interactúa con los componentes de transferencia de datos del proceso IPC a través de la clase de proxy *KernelIPCProcess* de librina, que traduce las invocaciones de funciones a mensajes NL o llamadas al sistema, dirigidas al kernel.

El IPCM crea el proceso del sistema IPCP bifurcando y ejecutando el ejecutable IPCP (es decir, ipcp). Tras la creación, el IPCP Daemon comprueba si tiene suficiente información para iniciarse, inicializa librina y crea una instancia de cada componente; tanto los componentes de utilidad (Encoder, CDAP Session Manager, RIB Daemon) como los que implementan funciones de gestión de capas (Enrollment, Resource Allocator, Namespace Manager, Flow Allocator y Security Manager). Una vez que se inicializan con éxito, la clase principal del proceso IPC invoca el método set_ipc_process en cada uno de ellos. El daemonIPCP tiene los siguientes componentes internos:

RIB Daemon: El IPCP aprovecha principalmente librina-rib para implementar su propio daemon RIB.

Tarea de enrollment: La clase rinad::EnrollmentTask es el punto de entrada a las funciones relacionadas con el neighbor discovery y el neighbor management del proceso IPC. La lógica de la secuencia de inscripción se implementa en dos máquinas de estado separadas por las clases rinad::EnrollerStateMachine y rinad::EnrolleeStateMachine.

Administrador de espacio de nombres: La clase rinad::NamespaceManager es el componente de proceso de IPC a cargo de administrar el espacio de nombres de direcciones del DIF y participa en el mantenimiento de la tabla de reenvío de directorios (DFT). El DFT es un mapa distribuido que hace coincidir los nombres de las aplicaciones (registradas con el DIF) con la dirección del proceso de IPC en el que están registrados. La implementación actual del DFT es un mapa completamente replicado cuyas actualizaciones se difunden a todos los miembros del DIF a través de inundaciones controladas.

Asignador de flujo: La clase rinad::FlowAllocator es el componente de IPC responsable de gestionar el ciclo de vida de los flujos proporcionados por el DIF. Las operaciones del ciclo de vida incluyen la creación, destrucción y monitoreo de parámetros de flujo clave para garantizar que el flujo proporcione un nivel de servicio aceptable (el prototipo actual implementa solo las operaciones de creación y destrucción).

Asignador de recursos: La clase rinad::ResourceAllocator es la encargada de proporcionar el punto de entrada a las funciones que realiza el ResourceAllocator. Hay una amplia gama de funciones que puede realizar este componente, pero actualmente IRATI solo proporciona la implementación de dos de ellas: la gestión de los flujos y la generación de la tabla de reenvío de PDU mediante enrutamiento.

La clase rinad::**NMinusOneFlowManager** gestiona la asignación y desasignación de los flujos N-1 utilizados por el proceso IPC, así como el registro del proceso ICP en uno o más DIF N-1.

El generador de tabla de reenvío de PDU es responsable de la generación de la tabla de reenvío de PDU (PDUFT), obtenida mediante el uso de una política (enrutamiento) específica. El PDUFT mapea las direcciones de destino y los identificadores de QoS con los identificadores de puerto de los flujos N-1. Estos flujos conectan el proceso IPC con sus vecinos en el DIF. La PDUFT es utilizada por el RMT, un componente de software que reside en el kernel-space, para multiplexar / demultiplexar las PDU. IRATI implementa un enfoque de enrutamiento basado en estado de enlace. Mantiene un gráfico que representa el conocimiento actual de la conectividad del DIF. Cada vértice del gráfico representa un proceso de IPC mientras que cada par de vértices representa un flujo N-1 que interconecta los correspondientes procesos de IPC. Se aplica un algoritmo (por ejemplo, la ruta más corta de Dijkstra) al gráfico para calcular las rutas desde un proceso de IPC de origen a todos los demás procesos de IPC en el DIF. Estas rutas se usan para llenar la PDUFT con entradas que mapean un par {dirección, QoS} a la lista de puertos N-1 que deben usarse para alcanzar el siguiente salto en la ruta hacia el destino.

Administrador de seguridad: El administrador de seguridad es el componente del proceso de IPC responsable de administrar todos los comportamientos relacionados con la seguridad (es decir, autenticación, control de acceso, protección SDU, administración de credenciales, auditoría). Dentro del alcance de IRATI, la implementación solo proporciona una implementación trivial de dos funciones relacionadas con el control de acceso. En este proyecto no trataremos la gestión de la seguridad, aunque creemos que una investigación relevante a este punto podría ser interesante.

11.4 Shim

La tarea del shim[23](un shim es una librería que intercepta las llamadas a las APIs y cambia los argumentos pasados, maneja la operación en sí o redirige la operación a otra parte
) del DIF es colocar una capa lo más pequeña posible sobre un protocolo heredado para permitir que un DIF de RINA lo use sin cambios. En otras palabras, debido a que el DIF asume que tiene una API RINA a continuación, Shim DIF permite que un DIF opere sobre una tecnología heredada o un medio físico(que en nuestro caso será sobre VLANs).
Los Shim DIF están completamente implementados en el kernel, ya que su funcionalidad es simple y generalmente interactúan con las API expuestas por los controladores de dispositivos o el código de la pila que solo está disponible en el kernel.

12 QoS Cube

12.1 Cubos de QoS

Definir cuáles son las características de los diferentes cubos de QoS[23] soportados por el DIF, así como sus políticas EFCP asociadas: conjunto de políticas DTP (conjunto de políticas de transferencia de datos) y conjunto de políticas DTCP (conjunto de políticas de transferencia de datos)

```
"qosCubes" : [ {
 "name" : "unreliablewithflowcontrol",
    "id" : 1,
    "partialDelivery" : false,
    "orderedDelivery" : true,
    "efcpPolicies" : {
        "dtpPolicySet" : {
          "name" : "default",
          "version" : "0"
        },
        "initialATimer" : 300,
        "dtcpPresent" : true,
        "dtcpConfiguration" : {
            "dtcpPolicySet" : {
              "name" : "default",
              "version" : "0"
            },
            "rtxControl" : false,
            "flowControl" : true,
            "flowControlConfig" : {
                "rateBased" : false,
                "windowBased" : true,
                "windowBasedConfig" : {
                  "maxClosedWindowQueueLength" : 50,
                  "initialCredit" : 50
                 }
            }
        }
    }
  }, {
  "name" : "reliablewithflowcontrol",
  ...
  } ]
```

- Name: el nombre del QoS cube (un string)
- id: el id del QoS cube (un unsigned integer)
- partialDelivery: verdadero/falso dependiendo de si se admite la entrega de SDU parciales
- orderedDelivery: verdadero/falso dependiendo de si en el pedido se requiere la entrega de SDU en orden
- initialATimer: valor inicial del temporizador A, en ms
- dtpPolicySet: nombre y versión del conjunto de políticas DTP asociado a este QoS cube
- dtcpPresent: verdadero si se requiere una instancia de DTCP para cada instancia de DTP, falso de lo contrario
- dtcpPolicySet: nombre y versión del conjunto de políticas DTP asociado a este QoS cube
- rtxControl: verdadero si DTCP realiza el control rtx, falso en caso contrario
- flowControl: verdadero si DTCP realiza control de flujo, falso en caso contrario
- windowBased: verdadero si DTCP realiza un control de flujo basado en ventanas, falso en caso contrario
- maxClosedWindowQueueLength: longitud máxima de la cola de ventana cerrada
- initialCredit: tamaño inicial de la ventana (solo si se usa el control de flujo basado en ventanas)

13. Explicación del funcionamiento del QTA-Mux

QTA mux[23] es una política de programación que facilita el intercambio de asignación de pérdida y delay entre PDUs de diferentes clases de QoS que desean acceder al mismo flujo de salida.

Cada PDU que llega al RMT pertenece a uno de los QoS-Cubes soportados por el DIF. QTAMux proporciona un mecanismo para intercambiar flujos de diferentes QoS-Cubes hacia un puerto en particular. El QTAMux tiene tres elementos principales, como se muestra en la figura a continuación:

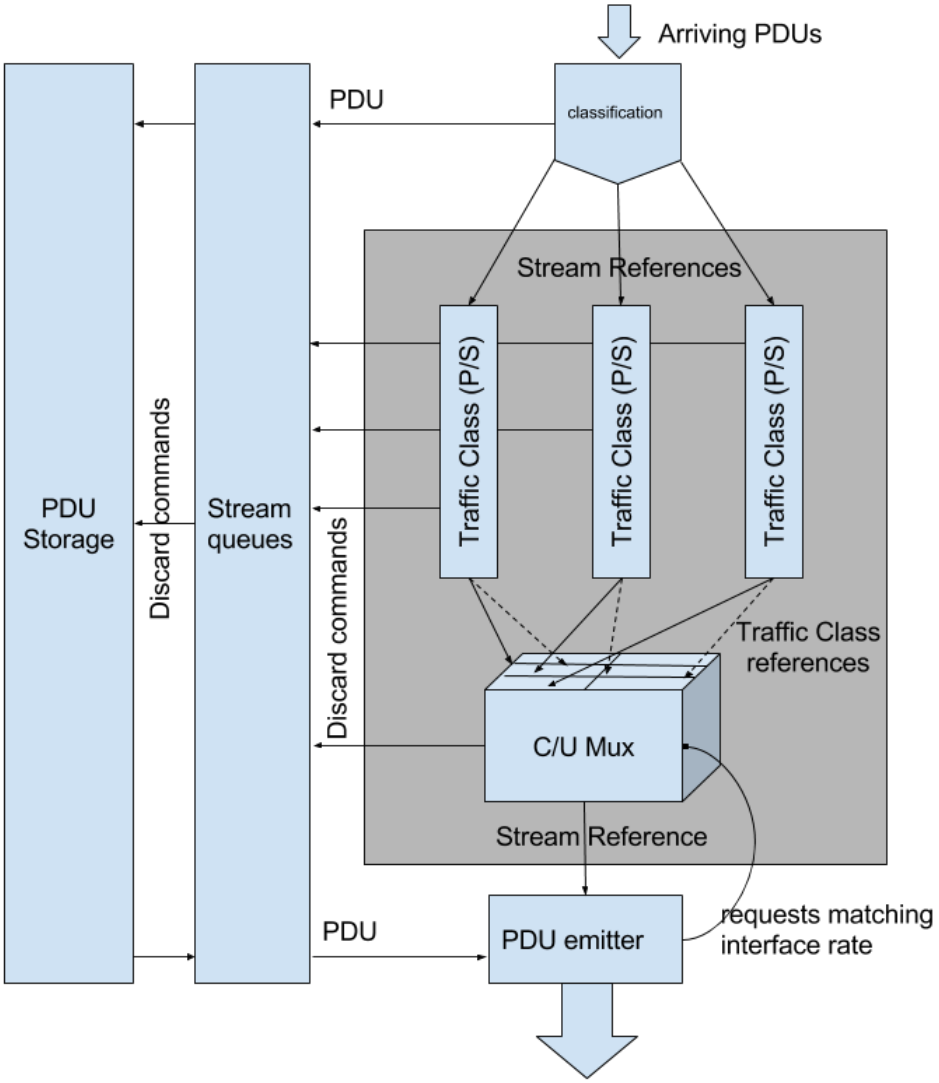


Figura 9. Estructura del QTA-Mux. Fuente: stack de IRATI[23]

El multiplexor Cherish-Urgency (C/U Mux): intercambia DeltaQ (delay y pérdida) entre el conjunto de flujos, utilizando la pérdida para mantener la capacidad.
Los policers/shapers (P/S): estos intercambian DeltaQ general contra el factor de carga, realizando contención dentro de la corriente para el tráfico en la misma clase de tratamiento.
La cola de transmisión: esto asegura la entrega en secuencia al tiempo que permite que el nivel de calidad entregado varíe dinámicamente.

13.1 C/U mux

Se puede considerar que el multiplexor Cherish-Urgency realiza una contención entre flujos. Su funcionamiento es muy sencillo de describir. Cada flujo de paquetes entrantes tiene niveles de urgencia, derivados de las asociaciones de PDU: estos corresponden a los requisitos de calidad del flujo en términos de pérdida y delay. La contención entre flujos para acceder al puerto de salida se gestiona mediante la admisión de paquetes en función de su nivel de clasificación de cherish y su servicio en función de su nivel de clasificación de urgencia. Así, el nivel de cherish de un paquete determina su probabilidad de pérdida, mientras que el nivel de urgencia determina su probable retraso. Dado que las clasificaciones de cherish y urgencia son independientes entre sí, la calidad asignada a los diferentes flujos no se limita a un esquema tradicional de ordenación de prioridades; en cambio, se logra una clasificación bidimensional, como se ilustra en la Figura 10.

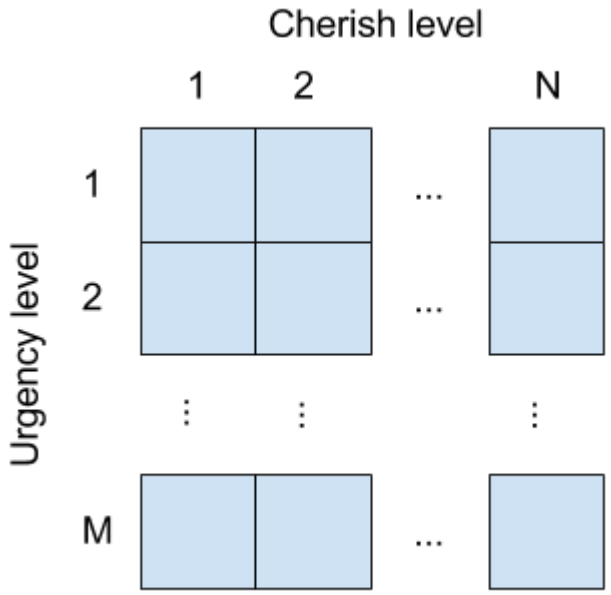


Figura 10. Estructura de los niveles de urgencia. Fuente: stack de IRATI[23]

14.Entorno para la realización del experimento en un entorno virtual

El software que utilizaremos para realizar la prueba en un entorno virtual será mediante la VirtualBox. Utilizaremos máquinas con Ubuntu 16.04. Para realizar el test simple utilizaremos 2 máquinas virtuales con entorno gráfico, mientras que el escenario simple se realizará con 10 nodos de máquinas virtuales sin entorno gráfico. Todas las máquinas fueron configuradas con 512 MB de ram. En cuanto a las interfaces de red, depende del escenario que estemos utilizando y que detallaremos más adelante.

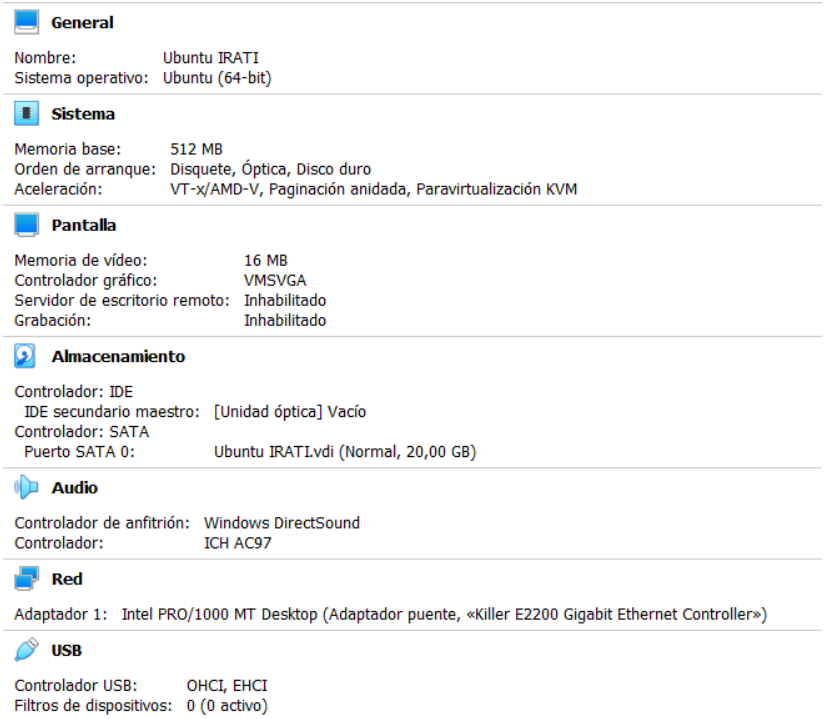


Figura 11 Ejemplo de máquina con entorno servidor.

Para la preparación de las máquinas siempre realizamo un apt-get update para poder tener los repositorios actualizados

15.Instalación del stack IRATI para la realización del experimento en el entorno virtual

El primer paso para poder realizar los experimentos fue instalar los paquetes necesarios para instalar IRATI, que en este caso son los siguientes:

- autoconf
- automake
- libtool
- pkg-config
- git
- g++
- libssl-dev
- protobuf-compiler
- libprotobuf-dev
- socat
- python
- linux-headers-$(uname -r)

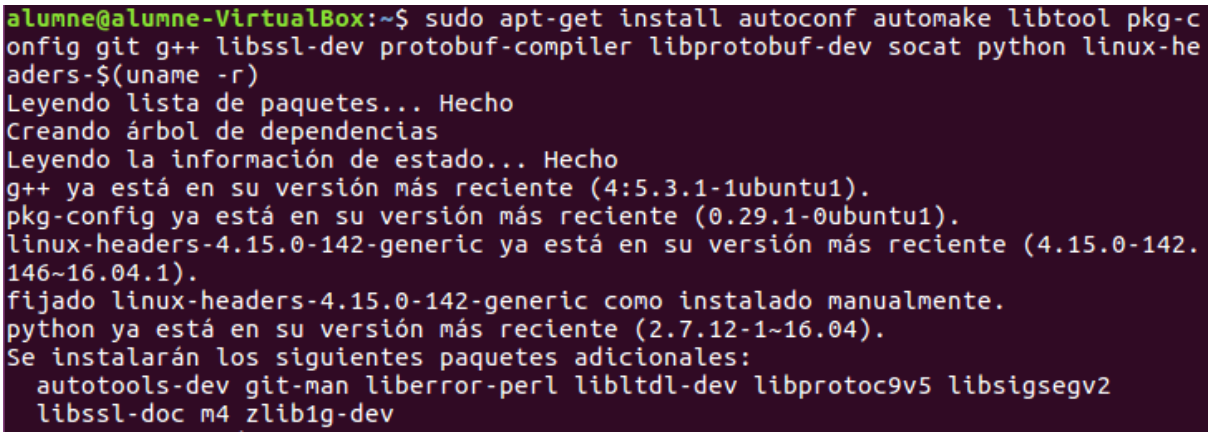


Figura 12 Instalación de los paquetes necesarios para poder instalar IRATI

Una vez realizada la instalación de los paquetes necesarios, procedemos a clonar el repositorio de GitHub

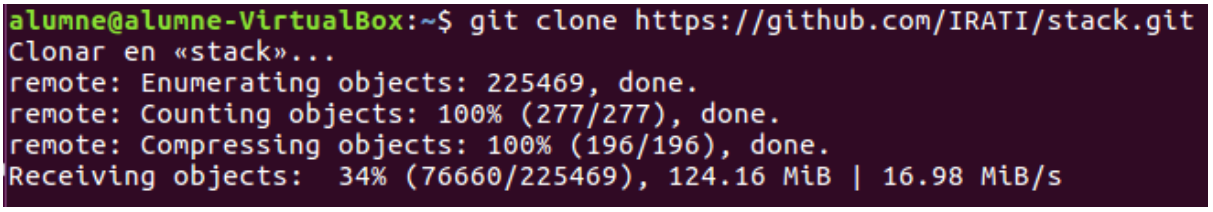


Figura 13 Clonación del repositorio de Github

Una vez tenemos clonado el stack en nuestro directorio, procedemos a realizar una instalación mediante autotools:

./configure --prefix <directorio_instalación>
make install

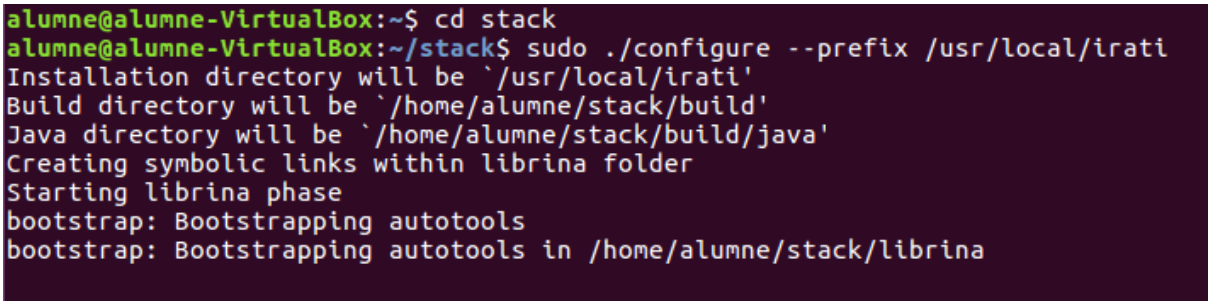


Figura 14 Primer paso de autotools

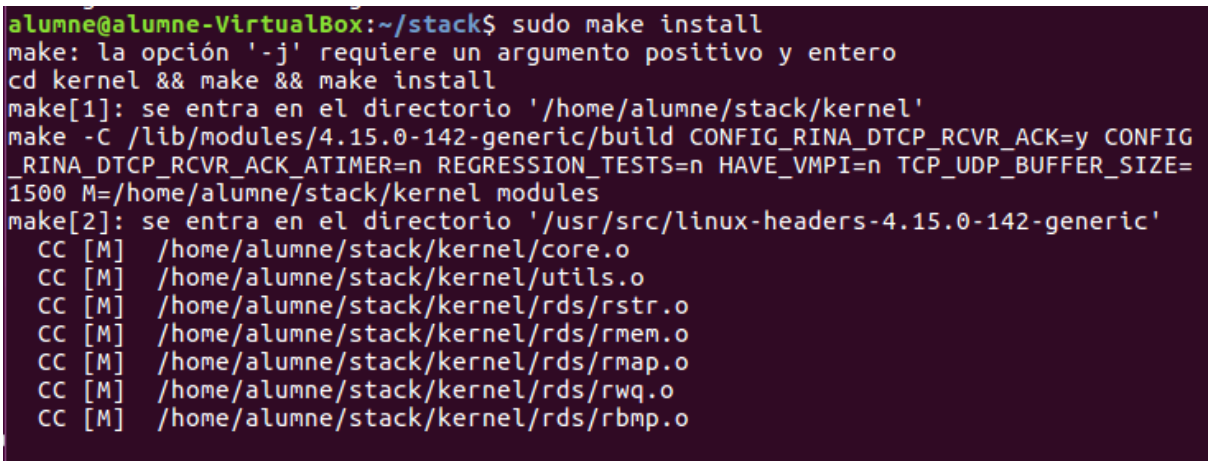


Figura 15 Segundo paso de autotools

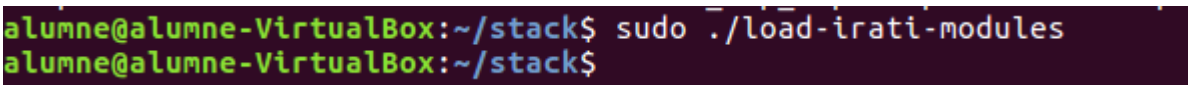


Figura 16 Ejecución del script para iniciar los modulos de IRATI.

En la figura anterior se comprueba la ejecución del script que realiza la iniciación de los modulos de IRATI. Internamente este módulo ejecuta los siguientes comandos:

```
modprobe rina-irati-core irati_verbosity=7
modprobe rina-default-plugin
modprobe normal-ipcp
modprobe shim-eth-vlan
modprobe shim-tcp-udp
```

Figura 17 Contenido del script load-irati-modules

De esta forma ya habríamos realizado la instalación del stack de IRATI en cualquier máquina.

16.Prueba con un escenario simple conectando dos equipos entre sí sobre una VLAN

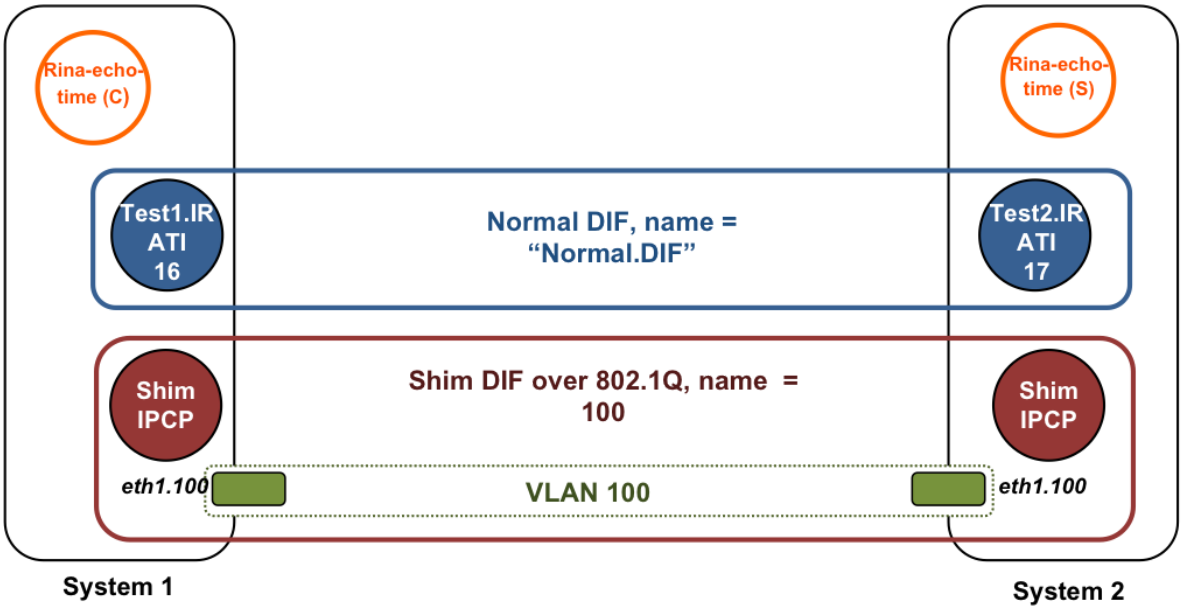


Figura 18 Escenario utilizado para realizar un test simple sobre IRATI. (Fuente stack de IRATI [23])

Lo primero para la realización del experimento es cargar los módulos de IRATI ejecutando el script.

```
alumne@alumne-VirtualBox:~/stack$ sudo ./load-irati-modules
[sudo] password for alumne:
alumne@alumne-VirtualBox:~/stack$ sudo ip link add link enp0s3 name enp0s3.100 t
ype vlan id 100
alumne@alumne-VirtualBox:~/stack$ sudo ip link set dev enp0s3 up
alumne@alumne-VirtualBox:~/stack$ sudo ip link set dev enp0s3.100 up
```

Figura 19 Ejecución del script para iniciar los modulos de IRATI.

Teniendo en cuenta que todos los ficheros estén bien configurados (ejemplos de ficheros en el Anexo), se ha de ejecutar el IPC Manager mediante el siguiente comando, indicando donde está el fichero ipcm.conf.

```
^Calumne@alumne-VirtualBox:/usr/local/irati/bin$ sudo ./ipcm -c ../etc/ipcm.conf
9758(1636678869)#ipcm (DBG): SIGSEGV handler installed successfully
9758(1636678869)#ipcm (DBG): SIGINT handler installed successfully
9758(1636678869)#ipcm (DBG): SIGQUIT handler installed successfully
9758(1636678869)#ipcm (DBG): SIGTERM handler installed successfully
9758(1636678869)#ipcm (DBG): SIGHUP handler installed successfully
9758(1636678869)#ipcm (DBG): SIGPIPE handler installed successfully
9758(1636678869)#ipcm (DBG): SIGCHLD handler installed successfully
9758(1636678869)#ipcm (DBG): Config file is: ../etc/ipcm.conf
9758(1636678869)#librina.timer (DBG): Timer with ID 12787376 started
9758(1636678869)#ipcm (DBG): Child IPC Process Daemon 9762 died, removed from pr
ocess table
9758(1636678869)#ipcm (DBG): Child IPC Process Daemon 9764 died, removed from pr
ocess table
9758(1636678869)#librina.logs (DBG): New log level: INFO
9770(1636678869)#ipcp[2] (DBG): SIGSEGV handler installed successfully
9770(1636678869)#ipcp[2] (DBG): SIGPIPE handler installed successfully
9770(1636678869)#ipcp[2] (DBG): SIGINT handler installed successfully
9770(1636678869)#librina.logs (DBG): New log level: INFO
```

Figura 20 Ejecución del IPCM Manager

Una vez hecho esto, tenemos que ejecutar el siguiente comando:

```
sudo socat - UNIX:/usr/local/irati/var/run/ipcm-console.sock
```

Figura 21 Ejecución del daemon de IPCM

Para luego poder ejecutar el daemon del IPCM.

```
IPCM >>> enroll-to-dif 2 Normal.DIF 100
DIF enrollment succesfully completed in 21 ms

IPCM >>>
```

Figura 22 enrollment del DIF del shim al normal DIF

El siguiente paso es realizar el enrollment de un proceso IPC a un DIF, el uso del comando es el siguiente:

```
enroll-to-dif <ipcp-id> <dif-name> <supporting-dif-name> <neighbor-process-name>
<neighbor-process-instance> # El último parametro es opcional, por defecto es 1
```

Figura 23 Usage del comando enroll-to-dif

En este caso estamos haciendo un enrollment del proceso IPC 2, que pertenece al DIF Normal

Una vez hecho esto ya podemos comprobar mediante la aplicación rina-echo-async incluída en el stack de IRATI que los dos DIF's tienen conectividad entre sí

Figura 24 Prueba de conectividad de cliente

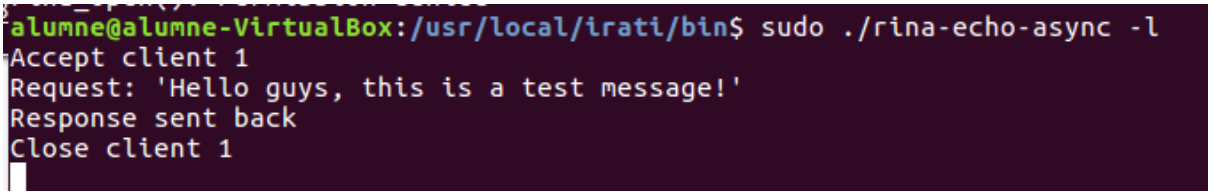


Figura 25 Prueba de conectividad del servidor

17.Escenario simple del proyecto ERASER

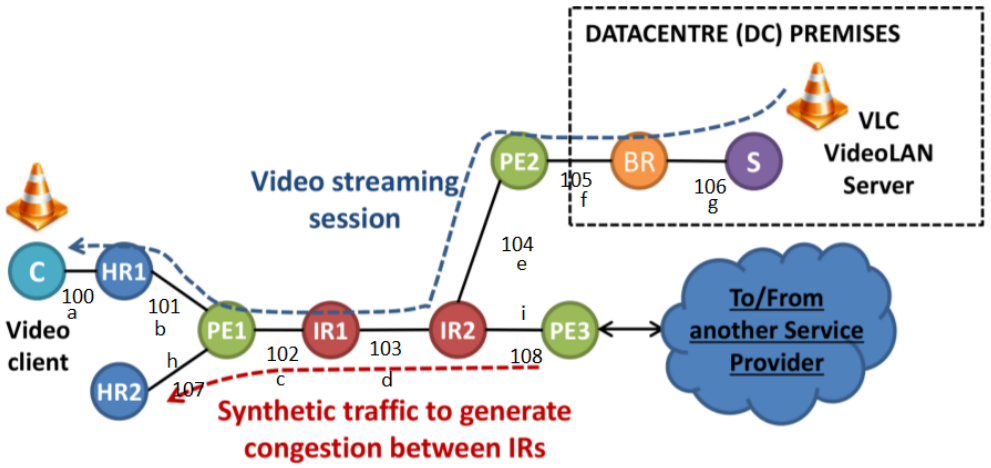


Figura 26 Estructura de conectividad del escenario simple. (Fuente: Proyecto ERASER)

Para la realización del escenario simple se ha empleado IRATI en un escenario inicial con 10 nodos como se muestra en la Figura . Para poder continuar con el escenario complejo, habría que seguir este como base, ya que sería replicar los mismos DIFS, pero con más nodos (37).
Este escenario dispone de 2 enrutadores internos (IR) interconectando 3 enrutadores Provider Edge (PE), uno de ellos dando conectividad a dos routers de casa (HR), otro donde se haya el servidor utilizado para la prueba de generación de tráfico de datos (en la experimentación final debería ser un servidor VLC VideoLAN) y un último para proporcionar conectividad a otro proveedor de servicio.

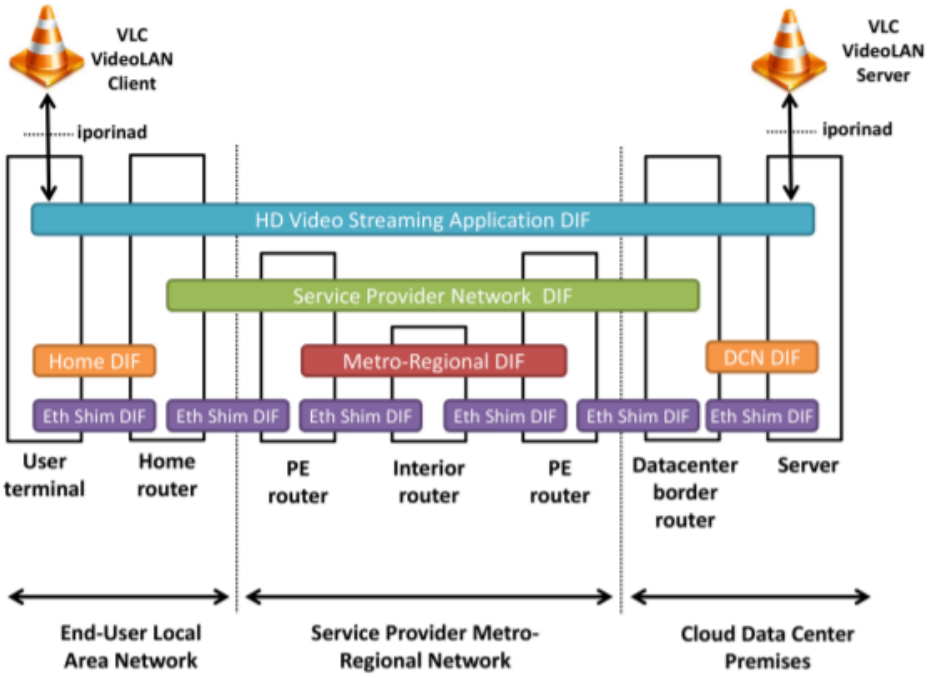


Figura 27 Ejemplo de configuración de los DIFs a crear (Fuente: Proyecto ERASER)

Para saber la configuración de cada VLAN hemos modificado la imagen original del proyecto ERASER para saber a que VLAN pertenece cada conexión, p.e: C pertenece a la VLAN 100 mientras que HR1 pertenece a la 100 y a la 101. Esto es debido a un motivo de simplificación y para ayudarnos a debugar y a encontrar más fácilmente si ha habido algún fallo en el nivel más bajo.

Para la creación de ficheros, nos basamos en la Figura 27. Mediante esta captura podemos hacernos una idea de los ficheros que hemos de crear.
En cada nodo tendremos que crear siempre el fichero IPCM.conf, que será mediante el cual iniciaremos el Manager de IPC. Por otro lado, dependiendo del nodo en el que estemos, este necesitará información sobre su DIF. De ahí que tengamos los siguientes ficheros repartidos por los nodos:

- hdvideo.dif
- spn.dif
- home.dif
- metro.dif

Cada uno de estos ficheros pertenece a uno de los DIF mediante el cual se realizará el escenario simple. En los anexos se ha adjuntado dos de estos ficheros de configuración.

Para finalizar en cuanto a la configuración añadir que también hay que tener en cuenta que hay un fichero shim.dif, que es mediante el cual se realizará la comunicación a bajo nivel mediante tecnología VLAN.
Como nota, cabe destacar que desde la última versión de IRATI, ya no es necesario utilizar VLAN's en las configuraciones y a partir de ahora se puede trabajar directamente con Ethernet.

18.Implementación del escenario simple

El escenario simple del proyecto ERASER con los 10 nodos se realizó de forma virtualizada utilizando VirtualBox.

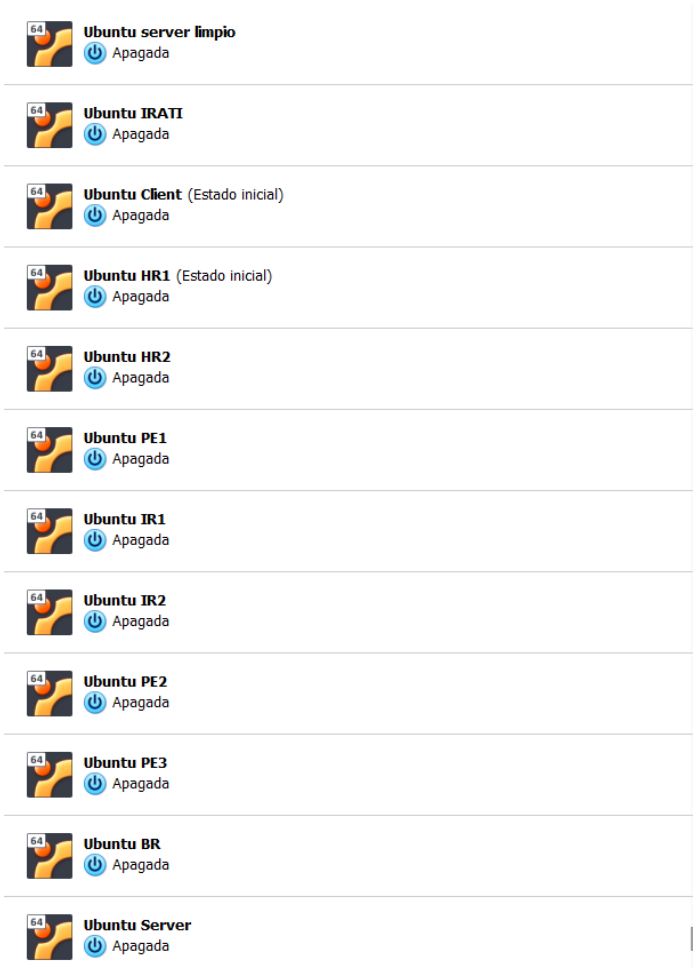


Figura 28 máquinas utilizadas para la simulación del entorno simple

Cada máquina disponía de 512MB de RAM y funcionaban bajo una imagen de Ubuntu Server 18.04.

```
IPCM >>> list-ipcps

Current IPC processes (id | name | type | state | Registered applications | Port-ids of flows provided)
   1 | eth.1.IPCP:1:: | shim-eth-vlan | ASSIGNED TO DIF 100 | homeA.IRATI-1-- | -
   2 | homeA.IRATI:1:: | normal-ipc | ASSIGNED TO DIF home.DIF | hdvideoA.IRATI-1-- | -
   3 | hdvideoA.IRATI:1:: | normal-ipc | ASSIGNED TO DIF hdvideo.DIF | - | -
```

Figura 29 ejemplo de volcado del comando list-ipcps sobre la máquina cliente

```
IPCM >>> enroll-to-dif 2 home.DIF 100 homeB.IRATI 1
DIF enrollment succesfully completed in 15 ms
```

Figura 30 ejemplo de enrollment del cliente

Una vez realizado el enrollment entre todas las máquinas podemos proceder a realizar la prueba de generación de tráfico o la prueba mediante rina-echo.

El cliente por su parte ha de ejecutar el comando tgen-client mientras que por otra parte el servidor ha de ejecutar tgen-server.

De esta forma podremos conseguir un volcado de la generación de tráfico punto a punto entre cliente y servidor.

```
sudo ./tg-server
Total :: 9867 Packets | 78.936 Mb
# Statistics per flow:
#       FlowID (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Rcv Bytes | Min
Latency | Avg Latency | Max Latency
       0 (0) | 9867 | 9867 | 0 % | 9867000 B |0.000529 | 0.000665841 | 0.001871
# Statistics per QoS ID:
#       (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Min Latency | Avg Latency |
Max Latency
       (0) | 9867 | 9867  | 0 % | 0.000529  | 0.000665841  | 0.001871
Total :: 9849 Packets | 78.792 Mb
# Statistics per flow:
#       FlowID (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Rcv Bytes | Min
Latency | Avg Latency | Max Latency
       0 (0) | 9849 | 9849 | 0 % | 9849000 B |0.000425 | 0.000551296 | 0.014635
# Statistics per QoS ID:
#       (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Min Latency | Avg Latency |
Max Latency
       (0) | 9849 | 9849  | 0 % | 0.000425  | 0.000551296  | 0.014635
Total :: 9810 Packets | 78.48 Mb
# Statistics per flow:
#       FlowID (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Rcv Bytes | Min
Latency | Avg Latency | Max Latency
       0 (0) | 9810 | 9810 | 0 % | 9810000 B |0.000181 | 0.000316067 | 0.001488
# Statistics per QoS ID:
#       (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Min Latency | Avg Latency |
Max Latency
       (0) | 9810 | 9810  | 0 % | 0.000181  | 0.000316067  | 0.001488
Total :: 9820 Packets | 78.56 Mb
# Statistics per flow:
#       FlowID (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Rcv Bytes | Min
Latency | Avg Latency | Max Latency
       0 (0) | 9820 | 9820 | 0 % | 9820000 B |0.000123 | 0.000262573 | 0.001381
# Statistics per QoS ID:
#       (QoSID) | Rcv Packets | Snd Packets | % Lost Packets | Min Latency | Avg Latency |
Max Latency
       (0) | 9820 | 9820  | 0 % | 0.000123  | 0.000262573  | 0.001381
```

Figura 31 Ejemplo de volcado del servidor cuando recibe paquetes

19.Implementación del escenario simple en entorno real

Explicaré la estructura hardware que tenemos para implementar en real este escenario y además explicaré el método que utilizamos para implementarlo.

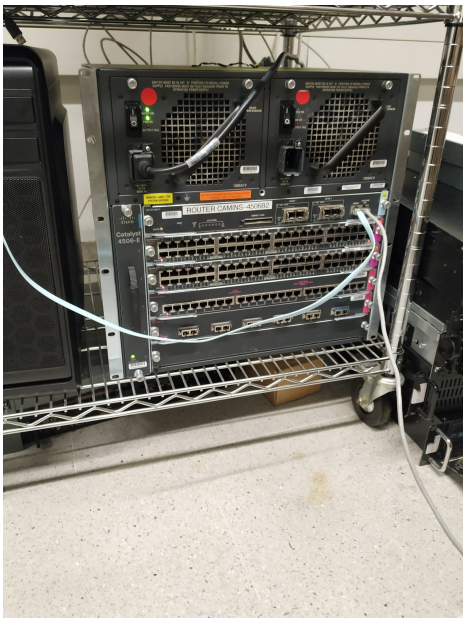


Figura 32 Switch Catalyst 4506-E, utilizado, a priori, para poder dar conexión a internet a los servidores.

Figura 33 tres de los diez de los servidores utilizados para hacer el deploy del escenario simple en un escenario real.

En cuanto a la configuración de los servidores, internamente se configuraron los mismos ficheros ipcm.conf y los mismos ficheros .dif que en el escenario virtualizado.
La única diferencia en el caso real es que tuvimos que realizar la clonación de las máquinas
Esto fue realizado mediante el software Clonezilla.

Los pasos a seguir fueron los siguientes:

- Realizar una instalación completa de Ubuntu e IRATI en un servidor.
- Clonar la imagen mediante Clonezilla (disk-to-image) en un disco duro externo.

También cabe destacar que en las máquinas reales implementamos scripts en cada máquina para poder realizar las configuraciones de red al iniciar el sistema.
Ejemplo de uno de los scripts:

```
#VLAN entre PE1 Y HR1
ip link add link eth1 name eth1.101 type vlan id 101
ip link set dev eth1 up
ip link set dev eth1.101 up



#VLAN entre PE1 y IR1
ip link add link eth2 name eth2.102 type vlan id 102
ip link set dev eth2 up
ip link set dev eth2.102 up

#VLAN entre PE1 y HR2
ip link add link eth3 name eth3.107 type vlan id 107
ip link set dev eth3 up
ip link set dev eth3.107 up
```

Figura 34 Ejemplo de script de inicialización de la red.

20.Resultados.

Al no poder realizar la prueba con el escenario complejo, no hemos podido realizar los experimentos con el flujo de vídeos.

Sin embargo, sí hemos podido comprobar la interconexión entre los nodos Cliente y Servidor mediante la herramienta ryna-echo-async.

Además de esto, también hemos podido generar tráfico sintético que se pudiese enviar desde el nodo Cliente al nodo Servidor mediante la aplicación tgen(traffic generator).

Como se puede observar en los volcados de tráfico, el tráfico es generado correctamente por el cliente y capturado por el servidor

21.Desviaciones y obstaculos

Como hemos comentado anteriormente, no conseguimos llegar al objetivo de simular el escenario complejo.
Esto fue debido a distintas circunstancias:

- Desconocimiento del proceso necesario para poder conectar un servidor a la red de la UPC. La red filtraba por MAC y hacía falta terminar de inventariar los equipos para poder conectarlos a la red y de esta forma poder conectarse mediante SSH

- De cara a la configuración del switch del escenario real, tuvimos ciertas complicaciones debido a que había que realizar un hard reset del mismo. Al realizar un hard reset, se perdió el fichero de imagen para realizar la instalación del switch. Entrando en modo rommon, donde, mediante una pequeña investigación, pudimos llegar a la conclusión que la forma sería crear un servidor TFTP (Trivial File Transfer Protocol) para poder realizar la transferencia entre el servidor y el switch.
  El principal problema que tuvimos fue que la imagen propiamente dicha, no es de código abierto y por lo tanto, hay que pagar por ella. De forma que se tuvo que crear una incidencia para que desde el servicio técnico de la UPC nos pudiesen proporcionar una imagen del switch. Esto provocó una gran ralentización en la realización del escenario real ya que no podíamos conectarnos mediante SSH a varias máquinas a la vez.

- No tener en cuenta en la planificación el tiempo que iba a conllevar la configuración de los servidores. Para llevar a cabo la configuración de los servidores, tuvimos que, además de la instalación del sistema operativo, trabajar con software que el alumno no había visto hasta el momento: Clonezilla. Este software fue utilizado para poder realizar un clon de un servidor en las otras 9 que hay disponibles. Investigando un poco, nos dimos cuenta que se podía lanzar una instalación en red, pero el hecho de investigar esta funcionalidad de Clonezilla llevaría tiempo, por lo que decidimos realizar la clonación de las máquinas una a una.

- Algo que también nos hizo perder algo de tiempo es que debido al tiempo que ha pasado desde que se lanzó el proyecto ERASER las librerías y código de IRATI se ha actualizado, dando lugar a pequeñas modificaciones en el código y en los ejecutables binarios de IRATI. Por ejemplo, la aplicación de rina-echo-time ya no existe en el stack y tuvimos que buscar la alternativa a la misma.

- Por otra parte, cuando estuvimos realizando la instalación del stack en los servidores, vimos que en la versión actual a día de hoy (20/01/2022) no funciona en distribuciones de Ubuntu. Por lo visto en este sistema operativo hay algún problema con un objeto, más concretamente el fichero: shim-eth-core.o, dando lugar a un fallo en la instalación. Para solucionar esto tuvimos que clonar el stack antiguo de otras máquinas de prueba (en este caso del primer experimento) para poder realizar el resto del proyecto.

22.Conclusiones y futuro del proyecto

En esta sección veremos las conclusiones del proyecto. En general, hablaremos sobre las conclusiones a las que hemos llegado y lo que nos ha aportado en general el trabajo. También hablaremos sobre el futuro del mismo.

Trabajar con el módulo de IRATI sobre RINA ha resultado muy interesante debido a que se trata de una tecnología aún en desarrollo y sobre la cual no hay demasiada documentación más allá de la que se encuentra en los repositorios y en el libro del Dr. John Day. Además de, por motivos de innovación, resultar difícil encontrar información muy específica sobre el protocolo.
A causa de esto, ha resultado una experiencia muy enriquecedora, tanto a nivel personal debido al nivel de exigencia por aprender como a nivel profesional, ya que ha dejado muchos conocimientos en cuanto a la arquitectura de RINA, además de demostrarme que no todo lo que se nos plantea tiene porque ser lo correcto (en referencia a la arquitectura TCP/IP).

Los resultados obtenidos del proyecto han sido satisfactorios en gran parte, ya que a excepción de la implementación del escenario complejo, se pudo realizar el estudio de la arquitectura RINA, el estudio sobre la herramienta IRATI, la virtualización de un escenario simple con 2 nodos y poder conectarlos entre sí mediante 1 DIF y finalmente la implementación del escenario simple.
Sin embargo, como hemos explicado en el apartado anterior, hubieron ciertas complicaciones que impidieron realizar el escenario complejo en el entorno real.

Por lo tanto, sabemos que en futuras implementaciones se ha de realizar el escenario complejo, cuyo desarrollo se centraría en la utilización de KVM debido a que el proyecto ERASER se desarrolló mediante esta plataforma y pensamos que sería más simple emularlo utilizando las mismas condiciones del proyecto y replicar el escenario simple en el escenario complejo utilizando cada servidor para almacenar un número N de nodos.
Otro punto a tener en cuenta también es comprobar el correcto funcionamiento del QTA-Mux mediante los tráficos establecidos.

Para concluir, indicar que a pesar de los problemas para llevar a cabo el proyecto, pienso que se ha avanzado mucho para poder realizar la experimentación necesaria en los servidores del CBA mediante el escenario simple.

Apendice

Ficheros de configuración

2 ficheros IPCM.conf

Fichero IPCM.conf del Cliente

```
{
  "configFileVersion": "1.4.1",
  "localConfiguration" : {
    "installationPath" : "/usr/local/irati/bin",
    "libraryPath" : "/usr/local/irati/lib",
    "logPath" : "/usr/local/irati/var/log",
    "consoleSocket": "/usr/local/irati/var/run/ipcm-console.sock",
    "pluginsPaths" : ["/usr/local/irati/lib/rinad/ipcp",
"/lib/modules/4.4.0-186-generic/extra/"]
  },
  "ipcProcessesToCreate" : [ {
    "type" : "shim-eth-vlan",
    "apName" : "eth.1.IPCP",
    "apInstance" : "1",
    "difName" : "100"
   }, {
    "type" : "normal-ipc",
    "apName" : "homeA.IRATI",
    "apInstance" : "1",
    "difName" : "home.DIF",
    "difsToRegisterAt" : ["100"]
   }, {
    "type": "normal-ipc",
    "apName" : "hdvideoA.IRATI",
    "apInstance" : "1",
    "difName" : "hdvideo.DIF",
    "difsToRegisterAt" : ["home.DIF"]
   } ],
```

```
  "difConfigurations" : [ {
    "name" : "100",
    "template" : "shimeth.c.100.dif"
  }, {
    "name" : "home.DIF",
    "template" : "home.dif"
  }, {
    "name" : "hdvideo.DIF",
    "template" : "hdvideo.dif"

  } ]
}
```

Fichero IPCM.conf de PE1

```
{
  "configFileVersion": "1.4.1",
  "localConfiguration" : {
    "installationPath" : "/usr/local/irati/bin",
    "libraryPath" : "/usr/local/irati/lib",
    "logPath" : "/usr/local/irati/var/log",
    "consoleSocket": "/usr/local/irati/var/run/ipcm-console.sock",
    "pluginsPaths" : ["/usr/local/irati/lib/rinad/ipcp",
"/lib/modules/4.4.0-186-generic/extra/"]
  },
  "ipcProcessesToCreate" : [ {
    "type" : "shim-eth-vlan",
    "apName" : "eth.1.IPCP",
    "apInstance" : "1",
    "difName" : "101"
   }, {
    "type" : "shim-eth-vlan",
    "apName" : "eth.2.IPCP",
    "apInstance" : "1",
    "difName" : "102"
   } ,{
    "type" : "shim-eth-vlan",
    "apName" : "eth.3.IPCP",
    "apInstance" : "1",
    "difName" : "107"
   } ,
```

```
{
    "type" : "normal-ipc",
    "apName" : "metroA.IRATI",
    "apInstance" : "1",
    "difName" : "metro.DIF",
    "difsToRegisterAt" : ["101","102","107"]
   },
{
    "type": "normal-ipc",
    "apName" : "spnC.IRATI",
    "apInstance" : "1",
    "difName" : "spn.DIF",
    "difsToRegisterAt" : ["metro.DIF"]
 } ],
  "difConfigurations" : [ {
    "name" : "101",
    "template" : "shimeth.pe1.101.dif"
  }, {
    "name" : "102",
    "template" : "shimeth.pe1.102.dif"
  }, {
    "name" : "107",
    "template" : "shimeth.pe1.107.dif"
  } ,{
    "name" : "spn.DIF",
    "template" : "spn.dif"
  }, {
    "name" : "metro.DIF",
    "template" : "metro.dif"
  } ]
}
```

Home DIF

```
{
   "difType" : "normal-ipc",
   "dataTransferConstants" : {
         "addressLength" : 2,
         "cepIdLength" : 2,
         "lengthLength" : 2,
         "portIdLength" : 2,
         "qosIdLength" : 2,
         "sequenceNumberLength" : 4,
         "maxPduSize" : 10000,
         "maxPduLifetime" : 60000,
       "ctrlSequenceNumberLength": 4,
       "frameLength": 4,
       "rateLength": 4
   },
   "qosCubes" : [ {
          "name" : "unreliablewithflowcontrol",
        "id" : 1,
        "partialDelivery" : false,
        "orderedDelivery" : true,
        "efcpPolicies" : {
            "dtpPolicySet" : {
              "name" : "default",
              "version" : "0"
            },
            "initialATimer" : 300,
            "dtcpPresent" : true,
            "dtcpConfiguration" : {
                "dtcpPolicySet" : {
                  "name" : "default",
                  "version" : "0"
                },
                "rtxControl" : false,
                "flowControl" : true,
                "flowControlConfig" : {
                    "rateBased" : false,
                    "windowBased" : true,
                    "windowBasedConfig" : {
                      "maxClosedWindowQueueLength" : 20,
                      "initialCredit" : 35
                     }
                }
```

```
        }
      }
    }, {
       "name" : "reliablewithflowcontrol",
     "id" : 2,
     "partialDelivery" : false,
     "orderedDelivery" : true,
     "maxAllowableGap": 0,
     "efcpPolicies" : {
        "dtpPolicySet" : {
          "name" : "default",
          "version" : "0"
        },
        "initialATimer" : 300,
        "dtcpPresent" : true,
        "dtcpConfiguration" : {
           "dtcpPolicySet" : {
             "name" : "default",
             "version" : "0"
           },
           "rtxControl" : true,
           "rtxControlConfig" : {
              "dataRxmsNmax" : 5,
              "initialRtxTime" : 1000
           },
           "flowControl" : true,
           "flowControlConfig" : {
              "rateBased" : false,
              "windowBased" : true,
              "windowBasedConfig" : {
                "maxClosedWindowQueueLength" : 20,
                "initialCredit" : 35
                        }
                     }
              }
     }
  } ],
  "knownIPCProcessAddresses" : [ {
       "apName" : "metroA.IRATI",
       "apInstance" : "1",
       "address" : 2
        }, {
       "apName" : "metroB.IRATI",
       "apInstance" : "1",
```

```
        "address" : 3
        }, {
      "apName" : "metroC.IRATI",
        "apInstance" : "1",
        "address" : 4
        }, {
      "apName" : "metroD.IRATI",
        "apInstance" : "1",
        "address" : 5
      }, {
      "apName" : "metroE.IRATI",
        "apInstance" : "1",
        "address" : 6
      }],
        "addressPrefixes" : [ {
        "addressPrefix" : 0,
        "organization" : "N.Bourbaki"
         }, {
        "addressPrefix" : 16,
        "organization" : "IRATI"
  } ],
  "rmtConfiguration" : {
    "pftConfiguration" : {
      "policySet" : {
        "name" : "default",
        "version" : "0"
      }
    },
    "policySet" : {
      "name" : "default",
      "version" : "1"
    }
  },
  "enrollmentTaskConfiguration" : {
    "policySet" : {
      "name" : "default",
      "version" : "1",
      "parameters" : [{
          "name"  : "enrollTimeoutInMs",
          "value" : "10000"
        },{
          "name"  : "watchdogPeriodInMs",
          "value" : "30000"
        },{
```

```
            "name"  : "declaredDeadIntervalInMs",
            "value" : "120000"
          },{
            "name"  : "neighborsEnrollerPeriodInMs",
            "value" : "30000"
          },{
            "name"  : "maxEnrollmentRetries",
            "value" : "3"
          }]
      }
    },
    "flowAllocatorConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
       }
    },
    "namespaceManagerConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
        }
    },
      "securityManagerConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
        }
    },
    "resourceAllocatorConfiguration" : {
       "pduftgConfiguration" : {
        "policySet" : {
          "name" : "default",
          "version" : "0"
        }
       }
    },
    "routingConfiguration" : {
       "policySet" : {
        "name" : "link-state",
        "version" : "1",
        "parameters" : [{
          "name"  : "objectMaximumAge",
          "value" : "10000"
```

```
        },{
          "name"  : "waitUntilReadCDAP",
          "value" : "5001"
        },{
          "name"  : "waitUntilError",
          "value" : "5001"
        },{
          "name"  : "waitUntilPDUFTComputation",
          "value" : "103"
        },{
          "name"  : "waitUntilFSODBPropagation",
          "value" : "101"
        },{
          "name"  : "waitUntilAgeIncrement",
          "value" : "997"
        },{
          "name"  : "routingAlgorithm",
          "value" : "Dijkstra"
        }]
    }
  }
}
```

HD Video DIF

```
{
   "difType" : "normal-ipc",
   "dataTransferConstants" : {
         "addressLength" : 2,
         "cepIdLength" : 2,
         "lengthLength" : 2,
         "portIdLength" : 2,
         "qosIdLength" : 2,
         "sequenceNumberLength" : 4,
         "maxPduSize" : 10000,
         "maxPduLifetime" : 60000,
       "ctrlSequenceNumberLength": 4,
       "frameLength": 4,
       "rateLength": 4
   },
   "qosCubes" : [ {
          "name" : "unreliablewithflowcontrol",
        "id" : 1,
        "partialDelivery" : false,
        "orderedDelivery" : true,
        "efcpPolicies" : {
            "dtpPolicySet" : {
              "name" : "default",
              "version" : "0"
            },
            "initialATimer" : 300,
            "dtcpPresent" : true,
            "dtcpConfiguration" : {
                "dtcpPolicySet" : {
                  "name" : "default",
                  "version" : "0"
                },
                "rtxControl" : false,
                "flowControl" : true,
                "flowControlConfig" : {
                    "rateBased" : false,
                    "windowBased" : true,
                    "windowBasedConfig" : {
                      "maxClosedWindowQueueLength" : 20,
                      "initialCredit" : 35
                     }
                }
```

```
        }
      }
    }, {
       "name" : "reliablewithflowcontrol",
     "id" : 2,
     "partialDelivery" : false,
     "orderedDelivery" : true,
     "maxAllowableGap": 0,
     "efcpPolicies" : {
        "dtpPolicySet" : {
          "name" : "default",
          "version" : "0"
        },
        "initialATimer" : 300,
        "dtcpPresent" : true,
        "dtcpConfiguration" : {
           "dtcpPolicySet" : {
             "name" : "default",
             "version" : "0"
           },
           "rtxControl" : true,
           "rtxControlConfig" : {
              "dataRxmsNmax" : 5,
              "initialRtxTime" : 1000
           },
           "flowControl" : true,
           "flowControlConfig" : {
              "rateBased" : false,
              "windowBased" : true,
              "windowBasedConfig" : {
                "maxClosedWindowQueueLength" : 20,
                "initialCredit" : 35
                        }
                    }
              }
     }
  } ],
  "knownIPCProcessAddresses" : [ {
       "apName" : "hdvideoA.IRATI",
       "apInstance" : "1",
       "address" : 2
        }, {
       "apName" : "hdvideoB.IRATI",
       "apInstance" : "1",
```

```
        "address" : 3
        }, {
      "apName" : "hdvideoC.IRATI",
        "apInstance" : "1",
        "address" : 4
        }, {
      "apName" : "hdvideoD.IRATI",
        "apInstance" : "1",
        "address" : 5
      } ],
        "addressPrefixes" : [ {
        "addressPrefix" : 0,
        "organization" : "N.Bourbaki"
         }, {
        "addressPrefix" : 16,
        "organization" : "IRATI"
   } ],
  "rmtConfiguration" : {
    "pftConfiguration" : {
      "policySet" : {
        "name" : "default",
        "version" : "0"
      }
    },
    "policySet" : {
      "name" : "default",
      "version" : "1"
    }
  },
  "enrollmentTaskConfiguration" : {
    "policySet" : {
      "name" : "default",
      "version" : "1",
      "parameters" : [{
          "name"  : "enrollTimeoutInMs",
          "value" : "10000"
        },{
          "name"  : "watchdogPeriodInMs",
          "value" : "30000"
        },{
          "name"  : "declaredDeadIntervalInMs",
          "value" : "120000"
        },{
          "name"  : "neighborsEnrollerPeriodInMs",
```

```
            "value" : "30000"
          },{
            "name"  : "maxEnrollmentRetries",
            "value" : "3"
          }]
      }
    },
    "flowAllocatorConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
       }
    },
    "namespaceManagerConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
        }
    },
      "securityManagerConfiguration" : {
       "policySet" : {
        "name" : "default",
        "version" : "1"
        }
    },
    "resourceAllocatorConfiguration" : {
       "pduftgConfiguration" : {
        "policySet" : {
          "name" : "default",
          "version" : "0"
        }
       }
    },
    "routingConfiguration" : {
       "policySet" : {
        "name" : "link-state",
        "version" : "1",
        "parameters" : [{
          "name"  : "objectMaximumAge",
          "value" : "10000"
        },{
          "name"  : "waitUntilReadCDAP",
          "value" : "5001"
        },{
```

```
          "name"  : "waitUntilError",
          "value" : "5001"
        },{
          "name"  : "waitUntilPDUFTComputation",
          "value" : "103"
        },{
          "name"  : "waitUntilFSODBPropagation",
          "value" : "101"
        },{
          "name"  : "waitUntilAgeIncrement",
          "value" : "997"
        },{
          "name"  : "routingAlgorithm",
          "value" : "Dijkstra"
        }]
    }
  }
}
```

Referencias

[1] i2CAT - The Internet Research Center. I2CAT [Online] [Citado el: 28 de 09 de 2021] https://i2cat.net/

[2] Cerf, Vinton G. & Cain, Edward , "The DoD Internet Architecture Model", Computer Networks, 7, North-Holland, 1983

[3] Universidad Autónoma del Estado de Hidalgo [Online] [Citado el: 28 de 09 de 2021] https://www.uaeh.edu.mx/scige/boletin/huejutla/n10/r1.html

[4] Day, John. Patterns in Network Architecture: A Return to Fundamentals. Pearson Education, Inc. 2008.

[5] Wikipedia contributors. [Online] [Citado el: 28 de 09 de 2021] https://en.wikipedia.org/wiki/Recursive_Internetwork_Architecture

[6] E. Grasa, O. Rysavy, O. Lichtner, H. Asgari, J. Day, L. Chitkushev, “From protecting protocols to protecting layers: designing, implementing and experimenting with security policies in RINA”, IEEE ICC 2016, Kuala Lumpur, Malaysia, July 2016.

[7] G. Boddapati, J. Day, I. Matta, L. Chitkushev, “Assessing the security of a clean-slate Internet architecture”, IEEE ICNP 2012, Austin, TX, USA, October 2012.

[8] E. Grasa et al., “Mobility management in RINA networks: experimental validation of architectural properties”, IEEE WCNC 2018, Barcelona, Spain, April 2018.

[9] S. Le.n et al., “Assuring QoS guarantees for heterogeneous services in RINA networks with ΔQ”, IEEE CloudCom 2016, Luxembourg City, Luxembourg, December 2016.

[10] S. Le.n, J. Perell., D. Careglio, E. Grasa, D. Lopez, P. A. Aranda. “Scalable topological forwarding and routing policies in RINA-enabled programmable Data Centres”, Transactions on Emerging Telecommunications Technologies, vol. 28, no. 12, December 2017.

[11] E. Grasa, L. Bergesio, M. Tarzan, D. Lopez, J. Day, L. Chitkushev, “Seamless Network Renumbering in RINA: Automate Address Changes Without Breaking Flows!”, EUCNC 2017, Oulu, Finland, June 2017.

[12] E. Grasa, M. Ponce de Leon, S. Van der Meer, D. Lopez, M. Tarzan, “Open multi-access edge computing and distributed mobility management with RINA”, IEEE NFV-SDN 2017, Berlin, German, Dec. 2017.

[13] S. Vrijders et al., “Reducing the complexity of virtual machine networking”, IEEE Communication Magazine, vol. 54, no. 4, Apr. 2016.

[14] J. Perelló, “Experimenting with Real Application specific QoS Guarantees in a Large-scale RINA Demonstrator (ERASER)”, Universitat Politècnica de Catalunya (UPC), September 2018.

[15] *Fed4Fire* FED4FIRE+. [Online] [Citado el: 28 de 09 de 2021] https://www.fed4fire.eu/

[16] “IRATI Project” http://irati.eu/, September 2021

[17]. Glassdoor. glassdoor.es [Online] [Citado el: 09 de 10 de 2021] https://www.glassdoor.es/Sueldos/barcelona-project-manager-sueldo-SRCH_IL.0,9_IM1015_KO10,25.htm?clickSource=searchBtn

[18]. Glassdoor. glassdoor.es [Online] [Citado el: 09 de 10 de 2021] https://www.glassdoor.es/Sueldos/barcelona-tester-sueldo-SRCH_IL.0,9_IM1015_KO10,16.htm?clickSource=searchBtn

[19]. Glassdoor. glassdoor.es [Online] [Citado el: 09 de 10 de 2021] https://www.glassdoor.es/Sueldos/administrador-de-redes-sueldo-SRCH_KO0,22.htm

[20]. Montalbán, E. larazon.es [Online] 31 de 08 de 2021 [Citado el: 09 de 10 de 2021] https://www.larazon.es/economia/20210831/gbm3kbuv3vfv5dke3klv2xcvc4.html

[21]. Vicente, A. selectra.es [Online] 14 de 09 de 2021 [Citado el: 09 de 10 de 2021] https://selectra.es/internet-telefono/internet

[22].Vladimír Veselý. RINASim. [Online] (2019, mayo) [Citado el: 12 de 12 de 2021]. Research Gate. https://www.researchgate.net/figure/DIF-DAF-DAP-and-IPCP-illustration_fig1_333306381

[23] I. GitHub - IRATI/stack: RINA implementation for OS/Linux. GitHub. [Citado el 20/12/2021 (2015)] https://github.com/IRATI/stack