


# VISUALIZACIÓN EN 3D DE LA TOPOLOGÍA DE REDES SDN UTILIZANDO A-FRAME

Autor: Alberto Abades Hoyas  
Tutor: Dr. Pedro de las Heras Quirós  
Curso: 2020/2021



# Índice

- Introducción
  - Objetivos
  - Tecnologías
  - Diseño e implementación
  - Resultado
  - Conclusiones
  - Referencias
- 

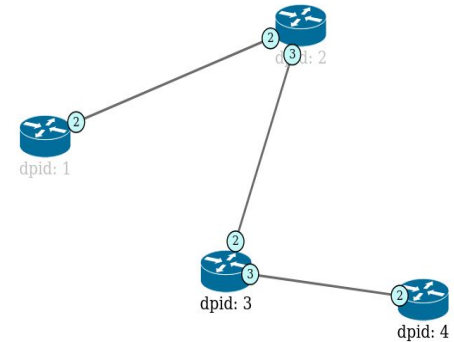
# Introducción

# Introducción

Realidad virtual y visualización de topología de redes



## Ryu Topology Viewer



# Objetivos

# Objetivo General

Representar en 3D en el navegador la topología de redes SDN gestionadas con Mininet utilizando la tecnología A-Frame

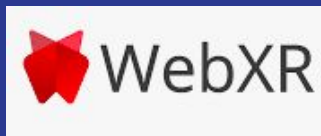
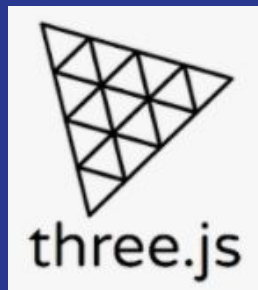
# Objetivos Específicos

- Visualizar en 3D en el navegador diferentes topologías
- Aprendizaje de A-Frame y Three.js para generar escenas
- Utilizar A-Frame para generar objetos en 3D
- Las topologías se deben visualizar de manera dinámica
- Se integrará con Mininet a través del controlador RYU



# Tecnologías

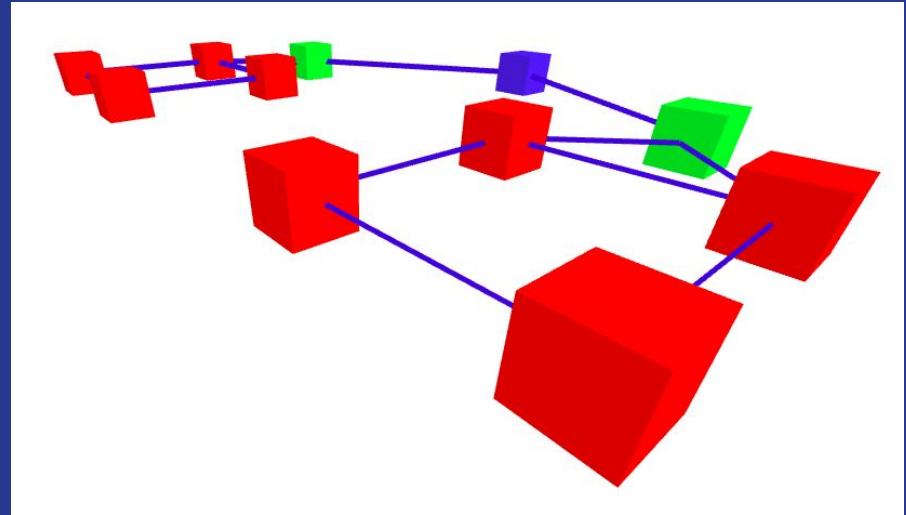
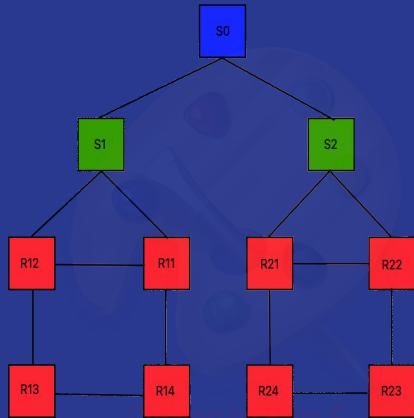




# Diseño e implementación

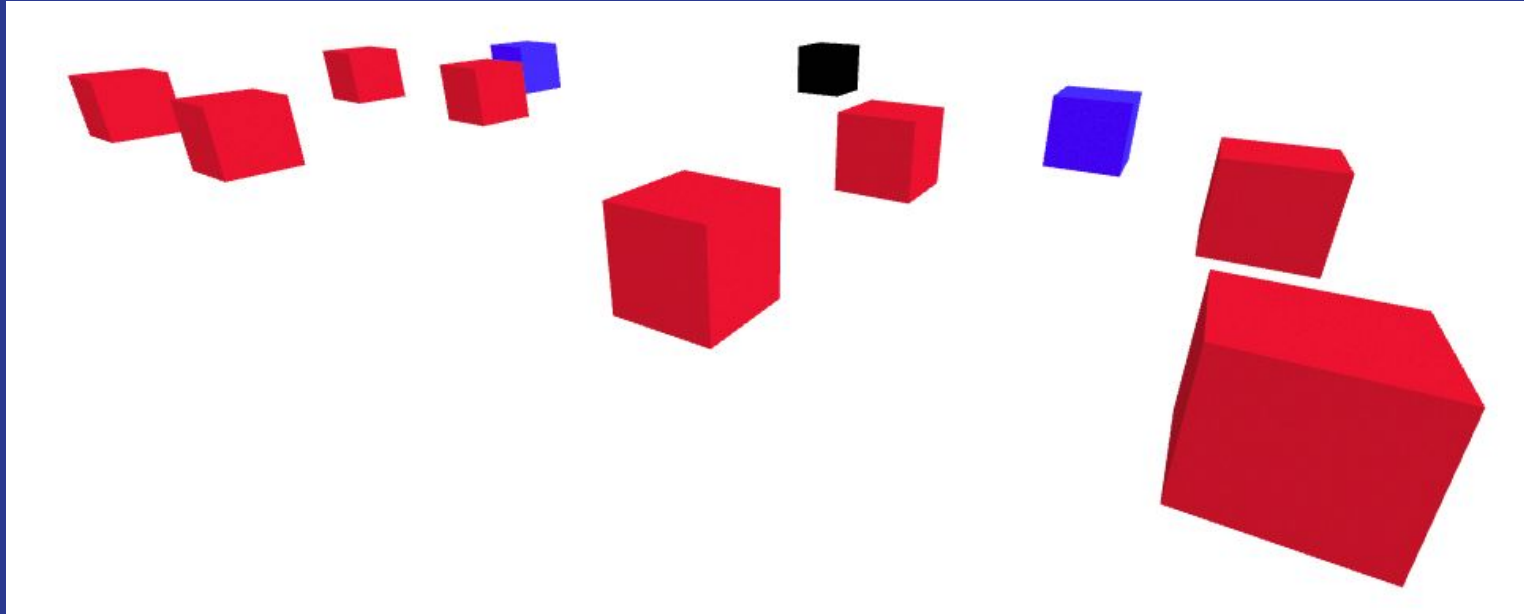
# Visualización de una escena

Generación de una escena, sin ninguna lógica, similar al objetivo final que se quiere conseguir.



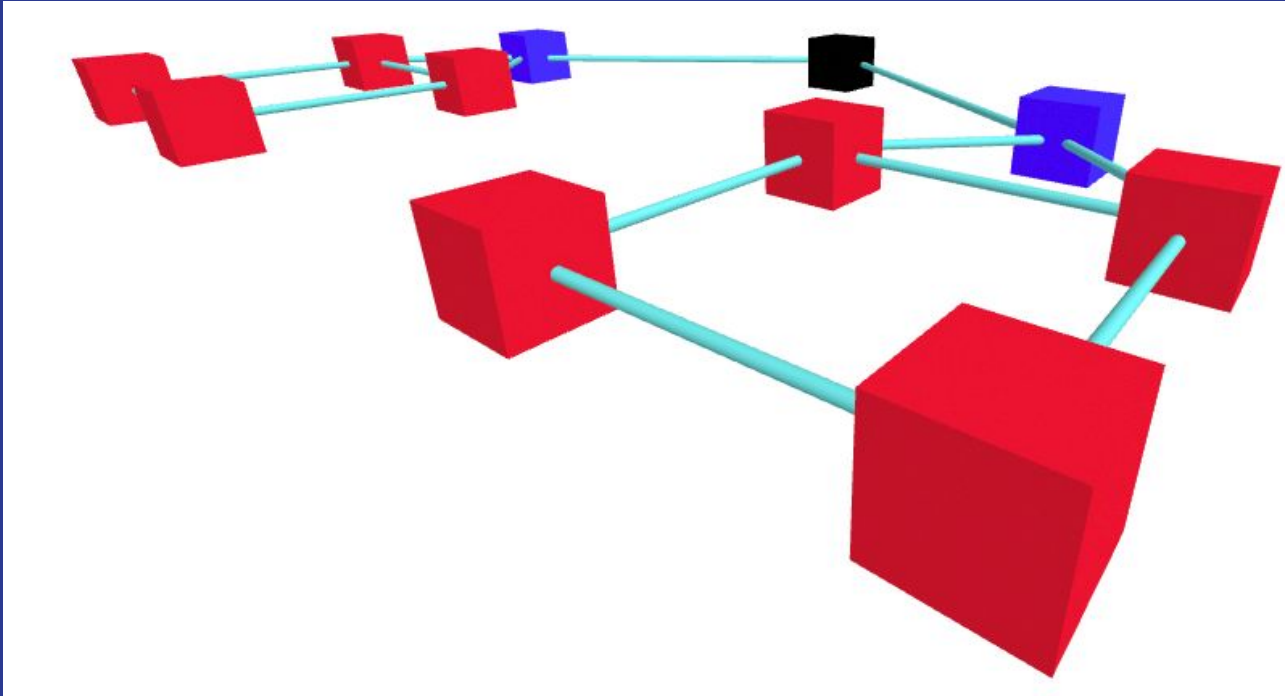
# Generación de entidades desde JavaScript

Generación de objetos desde un archivo JavaScript en lugar de hacerlo desde el archivo HTML. Primero incluimos sólo las cajas.



# Incluir enlaces entre las cajas

Incluimos los enlaces entre las cajas desde un archivo JavaScript



# Lectura de un archivo JSON

Obtenemos la información de la red a través de una llamada para recuperar un archivo JSON y se escribe en la consola.

```
[
  {
    "src": {"dpid": "0000000000000002", "port_no": "00000002", "hw_addr": "ea:54:6d:2b:64:da", "name": "s2-eth2"},
    "dst": {"dpid": "0000000000000001", "port_no": "00000002", "hw_addr": "06:31:ad:c6:15:87", "name": "s1-eth2"}
  },
  {
    "src": {"dpid": "0000000000000002", "port_no": "00000003", "hw_addr": "5e:18:5f:86:c3:7c", "name": "s2-eth3"},
    "dst": {"dpid": "0000000000000003", "port_no": "00000002", "hw_addr": "36:86:f6:3f:c8:a8", "name": "s3-eth2"}
  }
]
```

```
▼ Array [ {...}, {...} ]
  ▼ 0: Object { src: {...}, dst: {...} }
    ▶ dst: Object { dpid: "0000000000000001", port_no: "00000002", hw_addr: "06:31:ad:c6:15:87", ... }
    ▶ src: Object { dpid: "0000000000000002", port_no: "00000002", hw_addr: "ea:54:6d:2b:64:da", ... }
    ▶ <prototype>: Object { ... }
  ▼ 1: Object { src: {...}, dst: {...} }
    ▶ dst: Object { dpid: "0000000000000003", port_no: "00000002", hw_addr: "36:86:f6:3f:c8:a8", ... }
    ▶ src: Object { dpid: "0000000000000002", port_no: "00000003", hw_addr: "5e:18:5f:86:c3:7c", ... }
    ▶ <prototype>: Object { ... }
    length: 2
    ▶ <prototype>: Array []
```

# Arrancar una red SDN en Mininet

- Ejecutamos los pasos para arrancar una red SDN en Mininet y el controlador RYU para poder acceder a la información de la red se acaba de arrancar.
- Utilizamos una llamada GET para recuperar el archivo JSON del controlador y así poder analizarlo para quedarnos con la información principal

# Análisis de los archivos JSON de las SDN

Analizamos los archivos JSON que nos hemos descargado para encontrar una lógica genérica para poder obtener los datos de nuestra red sea cual sea su topología.

Vemos que los campos principales son:

- src.dpid
- src.port\_no
- src.hw\_addr
- src.name
- dst.dpid
- dst.port\_no
- dst.hw\_addr
- dst.name



# Generación de una escena a partir de los archivos JSON descargados

- Guardar la información de los switches y los enlaces
- Generar las coordenadas de cada switch
- Generar los switches e incluirlos en la escena
- Generar los enlaces e incluirlos en la escena

# Guardar la información de los switches y los enlaces

Guardamos en arrays la información de cuántos switches tenemos y cómo están conectados entre sí.

```
for (let i = 0; i < links.length; i++) {  
  let origen = links[i].src.dpid;  
  let destino = links[i].dst.dpid;  
  let origen_destino = [];  
  origen_destino.push(origen);  
  origen_destino.push(destino);  
  array_links.push(origen_destino);  
  
  const existe_origen = array_switches.includes(origen);  
  if (existe_origen == false){  
    array_switches.push(origen);  
  }  
  
  const existe_destino = array_switches.includes(destino);  
  if (existe_destino == false){  
    array_switches.push(destino);  
  }  
}
```

# Generación de una escena a partir de los archivos JSON descargados

- Guardar la información de los switches y los enlaces
- Generar las coordenadas de cada switch
- Generar los switches e incluirlos en la escena
- Generar los enlaces e incluirlos en la escena

# Generar las coordenadas de cada switch

Generamos coordenadas para los switches de manera aleatoria, poniendo como referencia el número de switches que tenemos

```
for(let i = 0; i < array_switches.length; i++){  
  let switch_coordenadas = [];  
  switch_coordenadas.push(array_switches[i]);  
  switch_coordenadas.push(Math.floor(Math.random() * ((array_switches.length+3) - (-3)) + (-3)));  
  switch_coordenadas.push(0.5);  
  switch_coordenadas.push(Math.floor(Math.random() * ((array_switches.length+3) - (-3)) + (-3)));  
  array_switches_final.push(switch_coordenadas);  
}
```

# Generación de una escena a partir de los archivos JSON descargados

- Guardar la información de los switches y los enlaces
- Generar las coordenadas de cada switch
- Generar los switches e incluirlos en la escena
- Generar los enlaces e incluirlos en la escena

# Generar los switches e incluirlos en la escena

Obtenemos las coordenadas que hemos generado y con ellas configuramos los switches para incluirlos en la escena

```
for (let i = 0; i < array_switches_final.length; i++) {  
  var scene = document.querySelector('a-scene');  
  var router = document.createElement('a-entity');  
  var_x = array_switches_final[i][1];  
  var_y = array_switches_final[i][2];  
  var_z = array_switches_final[i][3];  
  router.setAttribute('ID', array_switches_final[i][0]);  
  router.setAttribute('geometry', {primitive: 'box'});  
  router.setAttribute('material', 'color', 'blue');  
  router.setAttribute('position', {x: var_x, y: var_y, z: var_z});  
  router.setAttribute('scale', {x: 0.5, y: 0.5, z: 0.5});  
  scene.appendChild(router);  
}
```

# Generación de una escena a partir de los archivos JSON descargados

- Guardar la información de los switches y los enlaces
- Generar las coordenadas de cada switch
- Generar los switches e incluirlos en la escena
- Generar los enlaces e incluirlos en la escena

# Generar los enlaces e incluirlos en la escena

Obtenemos las coordenadas que hemos generado y con ellas configuramos los enlaces para incluirlos en la escena

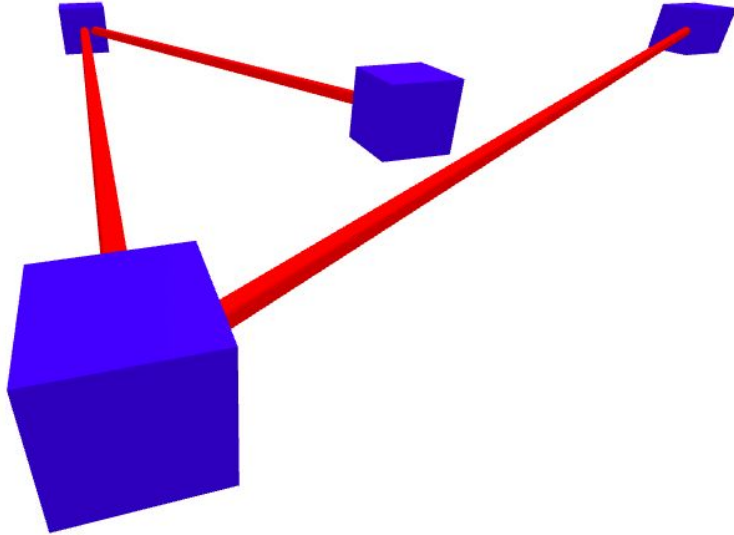
```
for(let i = 0; i < array_links.length; i++){
    let switch_origen = array_links[i][0];
    let switch_destino = array_links[i][1];
    for(let i = 0; i < array_switches_final.length; i++){
        if(switch_origen == array_switches_final[i][0]){
            coordXOrigen = array_switches_final[i][1];
            coordYOrigen = array_switches_final[i][2];
            coordZOrigen = array_switches_final[i][3];
        }
        if(switch_destino == array_switches_final[i][0]){
            coordXDestino = array_switches_final[i][1];
            coordYDestino = array_switches_final[i][2];
            coordZDestino = array_switches_final[i][3];
        }
    }
    coordOrigen = String(coordXOrigen) + " " + String(coordYOrigen) + " " + String(coordZOrigen);
    coordDestino = String(coordXDestino) + " " + String(coordYDestino) + " " + String(coordZDestino);
    var enlace = document.createElement('a-entity');
    var scene = document.querySelector('a-scene');
    enlace.setAttribute('tube', {'path': [coordOrigen, coordDestino], 'radius': '0.05'});
    enlace.setAttribute('material', 'color', 'red');
    enlace.setAttribute('material', 'opacity', '1');
    scene.appendChild(enlace);
}
```



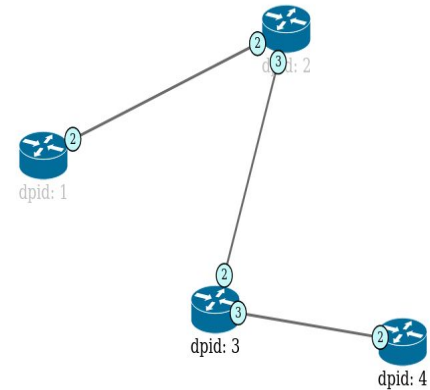


# Resultado

# Comparación entre la visualización en RYU y la visualización en 3D de una topología

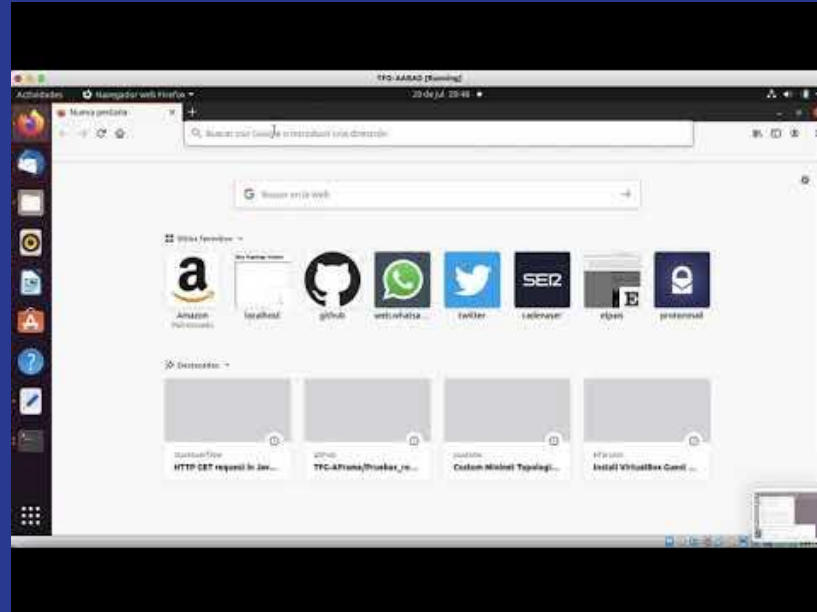


## Ryu Topology Viewer



# Ejecución sobre la máquina virtual

En el video se puede observar el proceso que hay que seguir para comprobar el funcionamiento de nuestro proyecto en la máquina virtual



# Conclusiones

# Consecución de objetivos

En este proyecto se han alcanzado los siguientes objetivos propuestos:

- Aprendizaje de la tecnología A-Frame
- Representación de topologías en 3D
- Integración con Mininet a través del controlador RYU

# Trabajos futuros

En el futuro se podrían hacer mejoras como:

- Mejorar la manera de visualizar la red incluyendo imágenes GLTF
- Recuperar información de intercambio de paquetes en la red y representarlo
- Generar lógica para poder arrancar o parar switches desde el navegador

# Esfuerzo realizado

Realizar este proyecto me ha llevado aproximadamente cuatro meses y medio.

	marzo				abril				mayo				junio				julio			
	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
Documentación																				
Escena con HTML																				
Escena con HTML + JavaScript																				
Lectura JSON + Integración																				
Memoria																				

# Referencias



## **REPOSITORIO EVOLUTIVO**

<https://github.com/albertoabades/TFG-AFrame>

## **ARCHIVOS DEL PROYECTO FINAL**

JavaScript

[https://github.com/albertoabades/TFG-AFrame/blob/main/Generar\\_Red.js](https://github.com/albertoabades/TFG-AFrame/blob/main/Generar_Red.js)

<https://github.com/albertoabades/TFG-AFrame/blob/main/aframe.min.js>

<https://github.com/albertoabades/TFG-AFrame/blob/main/aframe-extras.min.js>

HTML

[https://github.com/albertoabades/TFG-AFrame/blob/main/red\\_final.html](https://github.com/albertoabades/TFG-AFrame/blob/main/red_final.html)

## **MEMORIA**

[https://github.com/albertoabades/TFG-AFrame/blob/main/MemoriaTFG\\_AbadesHoyasAlberto\\_SDN\\_A-Frame.pdf](https://github.com/albertoabades/TFG-AFrame/blob/main/MemoriaTFG_AbadesHoyasAlberto_SDN_A-Frame.pdf)

## **DEMO**

[https://youtu.be/0WFDDuPH\\_Ac](https://youtu.be/0WFDDuPH_Ac)

## **PRESENTACIÓN**

[https://github.com/albertoabades/TFG-AFrame/blob/main/PresentacionTFG\\_AbadesHoyasAlberto\\_SDN\\_A-Frame.pdf](https://github.com/albertoabades/TFG-AFrame/blob/main/PresentacionTFG_AbadesHoyasAlberto_SDN_A-Frame.pdf)



Muchas gracias