2-8-2019

Erick Ceja Fuentes

Jorge Alejandro Díaz Sanchez

Andres Alejandre Jimenez

Arturo Itzamna Cortez Sanchez

Luis Eduardo Hernandez Ayala

REDMOTE

Tabla de contenido

[Introducción 2](#_Toc15591910)

[Mininet 2](#_Toc15591911)

[¿Qué es mininet? 2](#_Toc15591912)

[Tipos de topología 2](#_Toc15591913)

[RYU 3](#_Toc15591914)

[¿Qué es RYU? 3](#_Toc15591915)

[Instalación de RYU 3](#_Toc15591916)

[Controladora 3](#_Toc15591917)

[Métodos utilizados en el proyecto 3](#_Toc15591918)

[GET 3](#_Toc15591919)

[SET 7](#_Toc15591920)

[DELETE 8](#_Toc15591921)

# Introducción

El proyecto se realizara utilizando RYU como controllador sobre Mininet

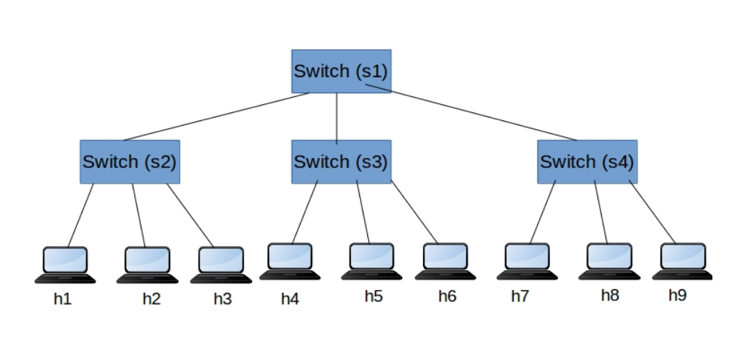
# Mininet

## ¿Qué es mininet?

Mininet crea una red virtual, con un kernel montado en una maquina virtual (VM, cloud) con un solo comando.

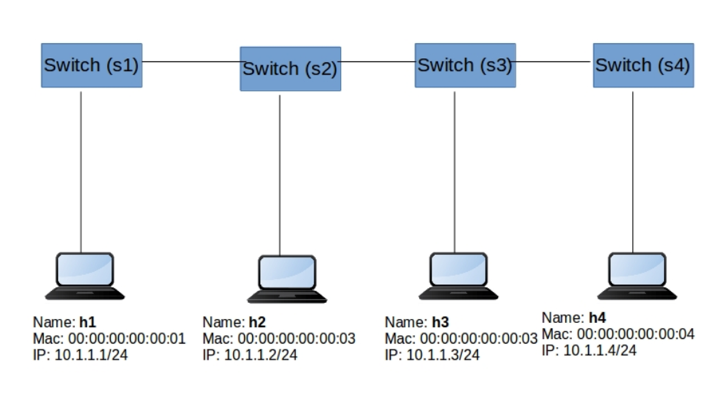
### Tipos de topología

#### Topología de árbol



sudo mn --controller= remote -x, --switch=ovsk,protocols=OpenFlow13 --topo=tree,depth=2,fanout=3

#### Topología linear



sudo mn --controller=remote -x --switch=ovsk,protocols=OpenFlow13 --topo=linear,4

# RYU

## ¿Qué es RYU?

Es una herramienta basada en componentes para SDN, contiene una API con la cual se pueden crear redes SDN.

## Instalación de RYU

En este caso se instaló RYU utilizando un script del usuario ajinkyakadam en github, el cual se encargaba de instalar varios paquetes necesarios para la utilización de este controlador.

Corriendo desde terminal:

wget https://gist.githubusercontent.com/ajinkyakadam/d6ef527a2ddbfb29bc53fdaf2270c228/raw/4cb0b694055e19b2aa5a166d6328070138c6a8b0/ryusetup.sh

Para descargar el script, para despues correr el script:

bash ryusetup.sh

# Controladora

Se creó una controladora en python desde la cual se puede tanto obtener información de los routers como configurar ciertos parametros.

Con fines demostrativos de nuestro proyecto se escriben a continuacion las funciones que se comunican con la API de ryu la cual se encarga de obtener y/o configurar los switches

## Métodos utilizados en el proyecto

### GET

#### get\_switch\_info()

##### Descripción general:

* Parametros: Ninguno
* Ejecución: Está función se encarga de consultar todos los switches que se encuentran en la topología, seguido de esto, consulta tanto la información de la descripción del switch como de su flujo, para finalmente mostrar esto
* Retorno: Regresa un diccionario con la información de todos los switches

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"id"**:1,  
            **"desc\_data"**:{    
                **"dp\_desc"**:"s1",  
                **"sw\_desc"**:"2.5.5",  
                **"hw\_desc"**:"Open vSwitch",  
                **"serial\_num"**:"None",  
                **"mfr\_desc"**:"Nicira, Inc."  
            },  
            **"flow\_data"**:[    
                {    
                    **"actions"**:[    
                        "OUTPUT:CONTROLLER"  
                    ],  
                    **"idle\_timeout"**:0,  
                    **"cookie"**:0,  
                    **"packet\_count"**:0,  
                    **"hard\_timeout"**:0,  
                    **"byte\_count"**:0,  
                    **"duration\_sec"**:242,  
                    **"duration\_nsec"**:306000000,  
                    **"priority"**:1,  
                    **"length"**:88,  
                    **"flags"**:0,  
                    **"table\_id"**:0,  
                    **"match"**:{    
                        **"dl\_type"**:2054  
                    }  
                },  
                {    
                    **"actions"**:[    
  
                    ],  
                    **"idle\_timeout"**:0,  
                    **"cookie"**:0,  
                    **"packet\_count"**:0,  
                    **"hard\_timeout"**:0,  
                    **"byte\_count"**:0,  
                    **"duration\_sec"**:242,  
                    **"duration\_nsec"**:306000000,  
                    **"priority"**:1,  
                    **"length"**:64,  
                    **"flags"**:0,  
                    **"table\_id"**:0,  
                    **"match"**:{    
                        **"dl\_type"**:2048  
                    }  
                },  
                {    
                    **"actions"**:[    
                        "OUTPUT:NORMAL"  
                    ],  
                    **"idle\_timeout"**:0,  
                    **"cookie"**:0,  
                    **"packet\_count"**:0,  
                    **"hard\_timeout"**:0,  
                    **"byte\_count"**:0,  
                    **"duration\_sec"**:242,  
                    **"duration\_nsec"**:306000000,  
                    **"priority"**:0,  
                    **"length"**:80,  
                    **"flags"**:0,  
                    **"table\_id"**:0,  
                    **"match"**:{    
  
                    }  
                }  
            ]  
        }  
    ]  
}

#### get\_switch\_layer\_two\_info()

##### Descripción general:

* Parametros: Ninguno
* Ejecución: Está función se encarga de consultar la información de capa dos del switch
* Retorno: Regresa un diccionario con la información de todos los switches

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"id"**:1,  
            **"port\_data"**:[    
                {    
                    **"hw\_addr"**:"de:61:a0:28:0a:47",  
                    **"curr"**:0,  
                    **"supported"**:0,  
                    **"max\_speed"**:0,  
                    **"advertised"**:0,  
                    **"peer"**:0,  
                    **"port\_no"**:"LOCAL",  
                    **"curr\_speed"**:0,  
                    **"name"**:"s1",  
                    **"state"**:1,  
                    **"config"**:1  
                },  
                {    
                    **"hw\_addr"**:"d2:af:e8:2f:16:bb",  
                    **"curr"**:2112,  
                    **"supported"**:0,  
                    **"max\_speed"**:0,  
                    **"advertised"**:0,  
                    **"peer"**:0,  
                    **"port\_no"**:1,  
                    **"curr\_speed"**:10000000,  
                    **"name"**:"s1-eth1",  
                    **"state"**:0,  
                    **"config"**:0  
                },  
                {    
                    **"hw\_addr"**:"fa:8f:b3:5a:78:4b",  
                    **"curr"**:2112,  
                    **"supported"**:0,  
                    **"max\_speed"**:0,  
                    **"advertised"**:0,  
                    **"peer"**:0,  
                    **"port\_no"**:2,  
                    **"curr\_speed"**:10000000,  
                    **"name"**:"s1-eth2",  
                    **"state"**:0,  
                    **"config"**:0  
                }  
            ]  
        }  
    ]  
}

#### get\_switch\_layer\_three\_info(router\_id)

##### Descripción general:

* Parametros:
  + router\_id (Opcional :El id del router a obtener información
* Ejecución: Está función se encarga de consultar la información de capa tres que se encuentre configurada hasta el momento en la red de mininet
* Retorno: Regresa un diccionario con la información de capa tres de los switches

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"internal\_network"**:[    
                {    
                    **"address"**:[    
                        {    
                            **"address\_id"**:3,  
                            **"address"**:"192.168.0.10/32"  
                        },  
                        {    
                            **"address\_id"**:1,  
                            **"address"**:"192.168.0.1/32"  
                        },  
                        {    
                            **"address\_id"**:2,  
                            **"address"**:"192.168.0.2/32"  
                        }  
                    ]  
                }  
            ],  
            **"switch\_id"**:"1"  
        },  
        {    
            **"internal\_network"**:[{}],  
            **"switch\_id"**:"2"  
        },  
        {    
            **"internal\_network"**:[{}],  
            **"switch\_id"**:"3"  
        },  
        {    
            **"internal\_network"**:[{}],  
            **"switch\_id"**:"4"  
        }  
    ]  
}

### SET

#### set\_address(switch\_id, address)

##### Descripción general:

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + address: la dirección ipv4 con mascara a asignar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"switch\_id"**:"1",  
            **"command\_result"**:[    
                {    
                    **"result"**:"success",  
                    **"details"**:"Add address [address\_id=1]"  
                }  
            ]  
        }  
    ]  
}

#### set\_\_default\_route(switch\_id, gateway)

##### Descripción general:

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + gateway: la dirección ipv4 con mascara a asignar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"switch\_id"**:"1",  
            **"command\_result"**:[    
                {    
                    **"result"**:"success",  
                    **"details"**:"Add route [route\_id=1]"  
                }  
            ]  
        }  
    ]  
}

#### set\_static\_route(switch\_id, destination, gateway)

##### Descripción general:

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + destination: la dirección ipv4 con mascara a asignar
  + gateway: la dirección ipv4 con mascara a asignar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"switch\_id"**:"1",  
            **"command\_result"**:[    
                {    
                    **"result"**:"success",  
                    **"details"**:"Add route [route\_id=2]"  
                }  
            ]  
        }  
    ]  
}

### DELETE

#### delete\_address(switch\_id, address\_id)

##### Descripción general:

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + address\_id: la dirección ipv4 con mascara a eliminar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"switch\_id"**:"1",  
            **"command\_result"**:[    
                {    
                    **"result"**:"success",  
                    **"details"**:"Delete route [route\_id=2]"  
                }  
            ]  
        }  
    ]  
}

#### delete\_static\_route(switch\_id, route\_id)

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + route\_id: la dirección ipv4 con mascara a eliminar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Descripción general:

* Parametros:
  + switch\_id: el id del switch a agregar dirección
  + destination: la dirección ipv4 con mascara a asignar
  + gateway: la dirección ipv4 con mascara a asignar
* Ejecución: Está función se encarga de agregar una dirección ipv4 al switch seleccionado
* Retorno: Regresa el estatus de la respuesta dependiendo de si fue exitosa, así como su id

##### Ejemplo de respuesta:

{    
    **"status\_code"**:200,  
    **"data"**:[    
        {    
            **"switch\_id"**:"1",  
            **"command\_result"**:[    
                {    
                    **"result"**:"success",  
                    **"details"**:"Delete address [address\_id=1]"  
                }  
            ]  
        }  
    ]  
}