Prototipos de Redes Definidas por Software

Agenda

- Situación Actual de Internet
- SDN
- OpenFlow
- Mininet
- Prototipo usando switches habilitados
- Prototipo usando switches virtuales

Internet

- Éxito Increíble:
  - Proyecto de investigación -> infraestructura global
  - Basado en:
    - Red (entrega de mejor esfuerzo de paquetes) y
    - Hosts (aplicaciones arbitrarias)
- Innovación:
  - Web, P2P, VoIP, redes sociales, mundos virtuales
- Los cambios solo son posibles en los extremos.

Dentro de la “red”

- Equipamiento cerrado
  - Software incluido en el hardware
  - Interfaces específicas del vendedor
- (lenta) Estandarización de protocolos
- Poca innovación
  - Vendedores
  - (retardo) Introducción de nuevas características

Cambios provocados por los usuario

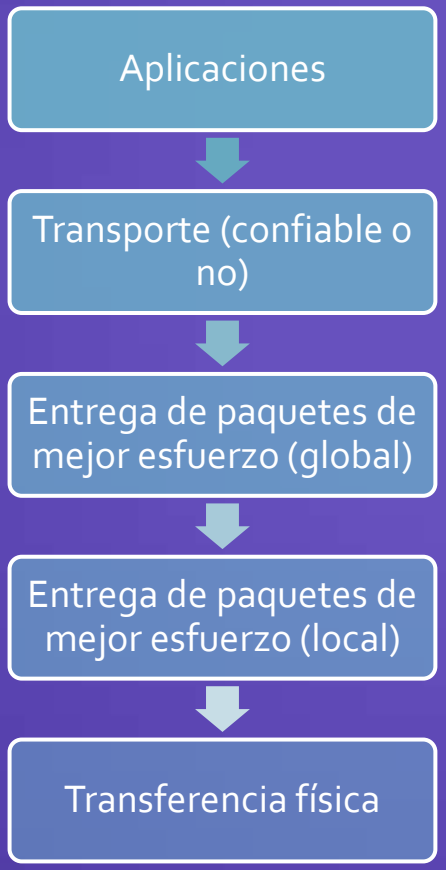
- Nuevos requerimientos:
  - Redes de gran escala
  - Movilidad
  - QoS
  - Migración de máquinas virtuales
  - Gran cantidad de datos (BigData)

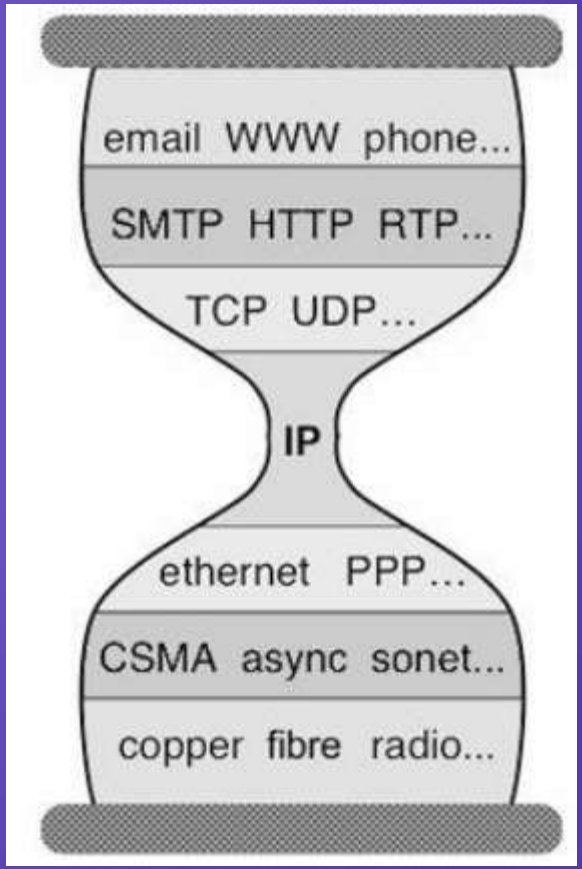
Los planos del "*networking*"

- Plano de Datos
  - Procesamiento y entrega de paquetes en base a políticas de reenvío:
    - Estado de reenvío + cabecera de paquete -> decisión de reenvió
- Plano de Control
  - Establece el estado de reenvío
    - Protocolos distribuidos
    - Configuración manual
    - Computación centralizada

Plano de Datos (Abstracciones)

- Capas:





Mecanismos del Plano de Control

- Diferentes Objetivos:
  - Enrutamiento
  - Aislamiento
  - Ingeniería de Tráfico
- No hay modularidad
- Funcionalidad limitada
  - Demasiados mecanismos sin abstracciones, lo que provoca una funcionalidad limitada.

El problema del Plano de Control

- El plano de control debe calcular el estado de reenvío.
  - Consistente con el hardware/software de bajo nivel.
  - Basarse en la topología completa de la red.
  - Realizarse en cada equipo de comunicaciones.

Plano de Control (Abstracciones)

- Ser compatible con hardware/software de bajo nivel
  - Abstracción para el modelo de reenvío general.
- Tomar decisiones basadas en la topología de red
  - Abstracción para el estado de la red.
- Configuración de cada dispositivo de red
  - Abstracción que simplifique el proceso de configuración.

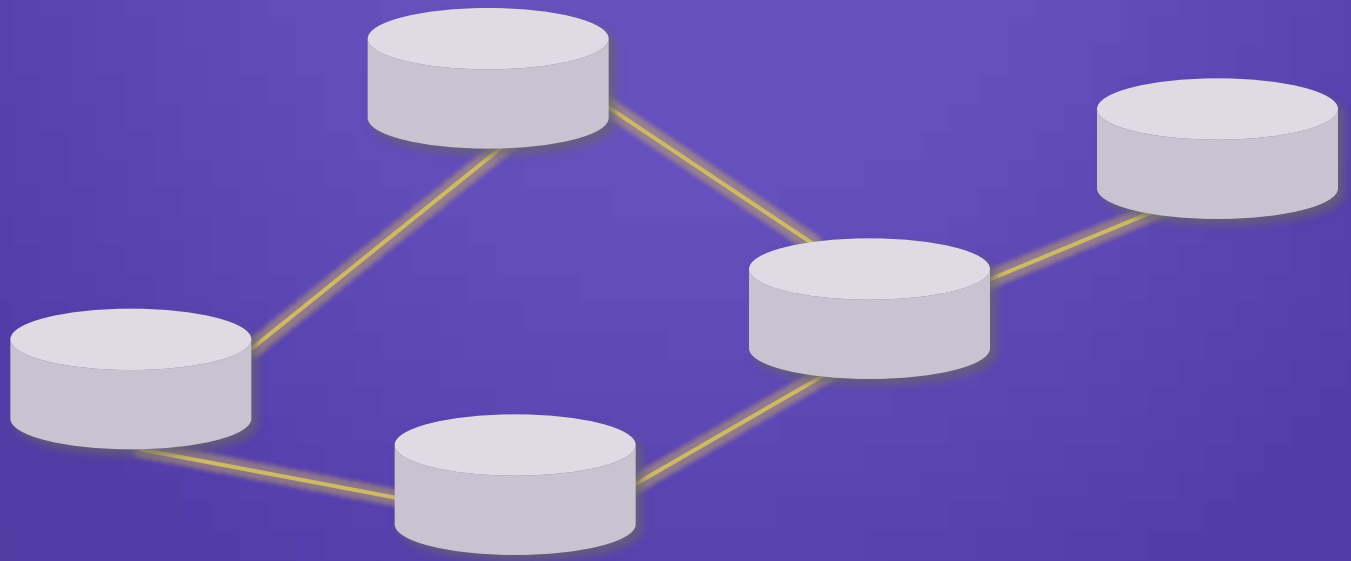
Abstracción del Reenvío

- Independiente de hardware/software.
- Propuesta actual: OpenFlow
  - Interfaz estandarizado para manipular el plano de control.
  - Configuración en términos de flujos
- Los detalles del diseño se basan en:
  - Coincidencia en cabeceras
  - Acciones

Abstracción del Estado de Red

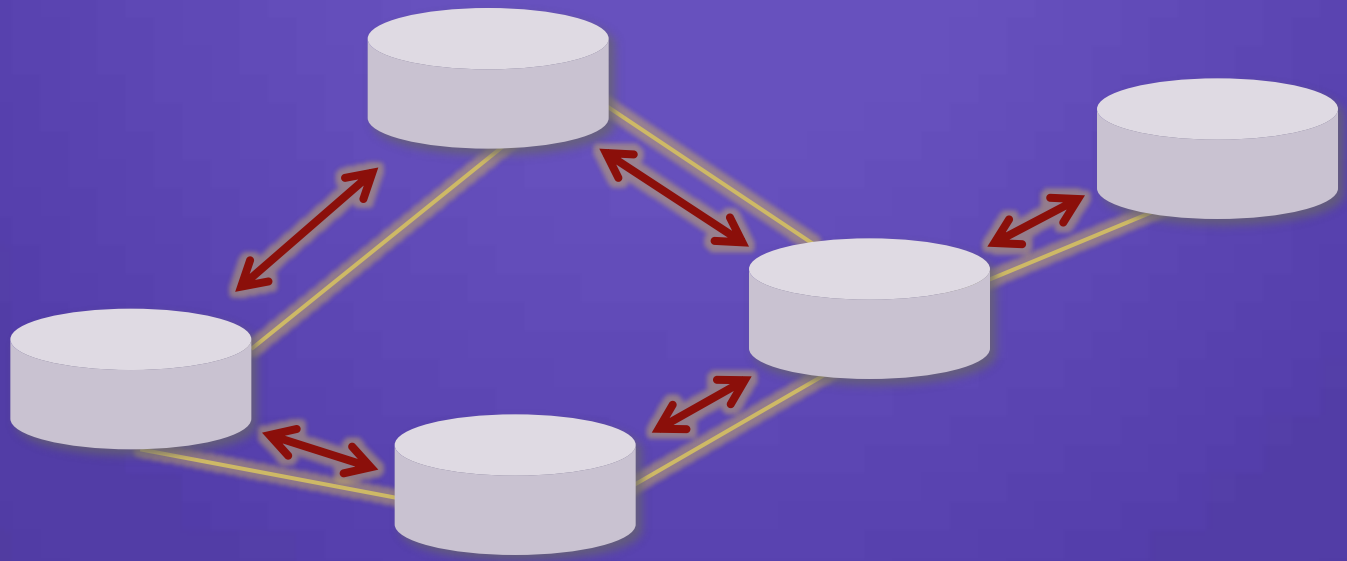
- Abstracción: Vista Global de la Red
- Implementación: Sistema Operativo de Red
  - Ejecución en varios servidores
    - replicación -> confiabilidad
- Información fluye en dos vías:
  - Información **desde** equipos de red para formar la vista
  - Configuración **hacia** equipos de red para controlar el reenvió

Red Tradicional

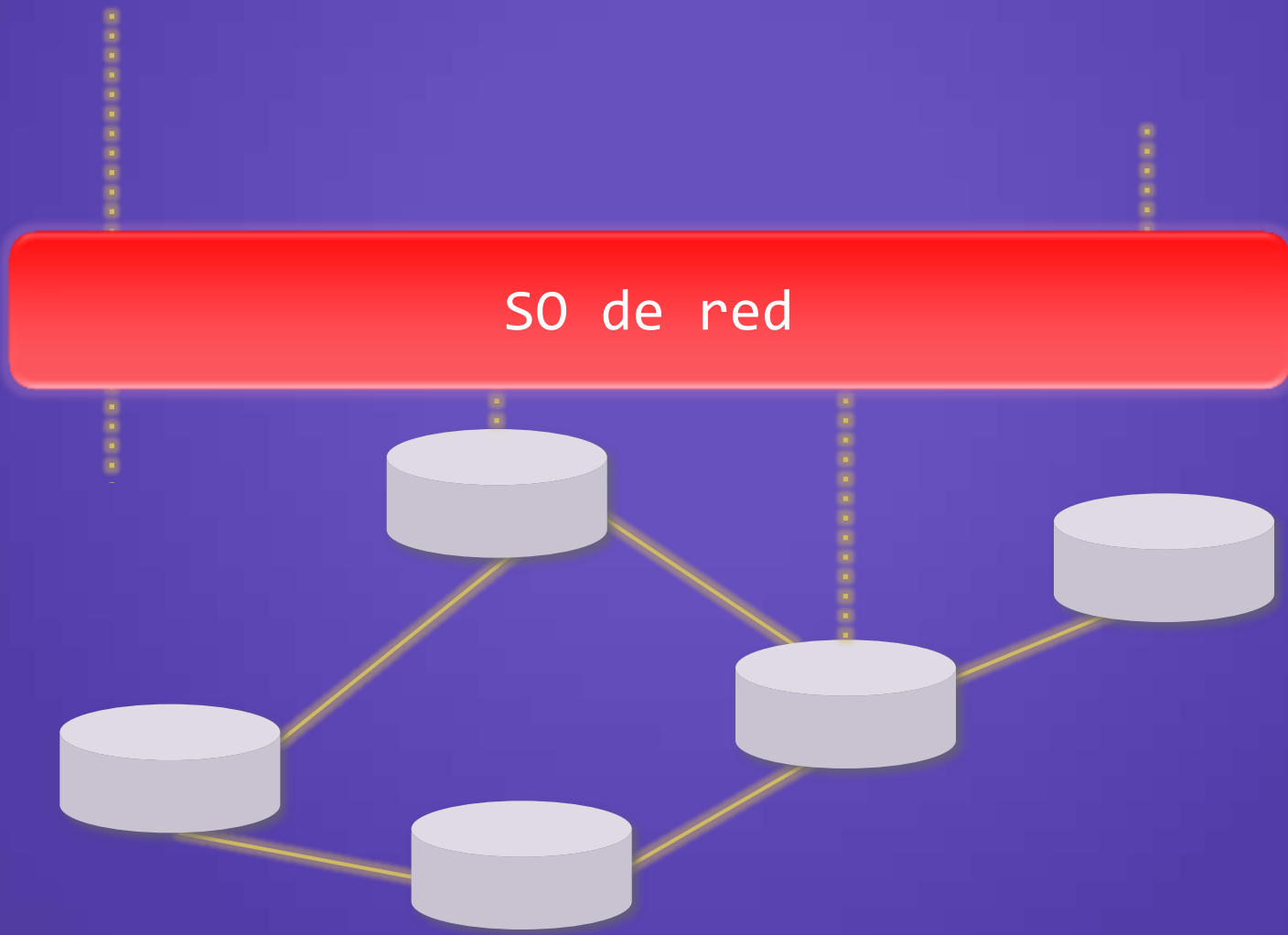


Mecanismos de Control Tradicional

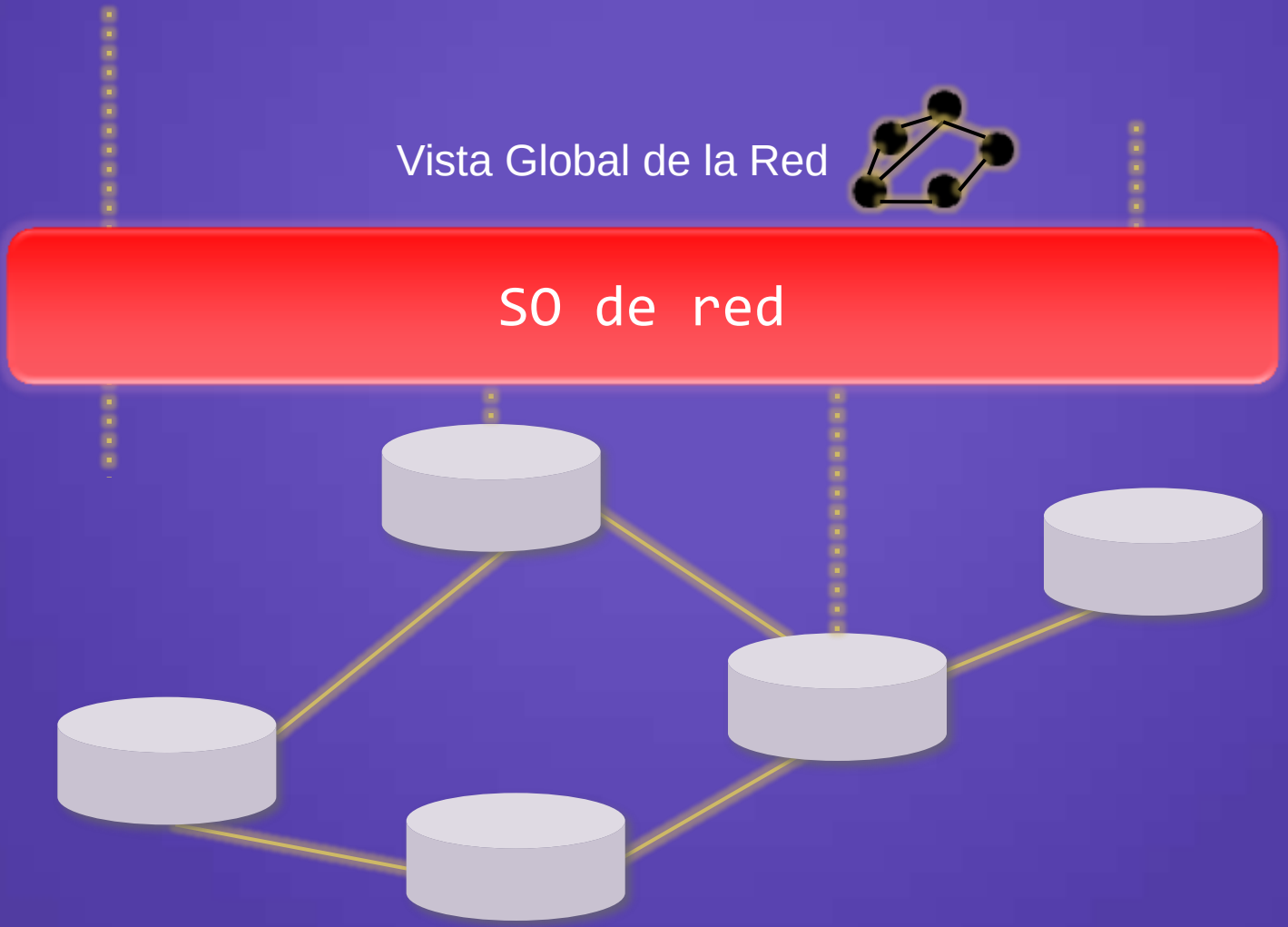
- Algoritmos distribuidos en los equipos (vecinos)



SDN



SDN



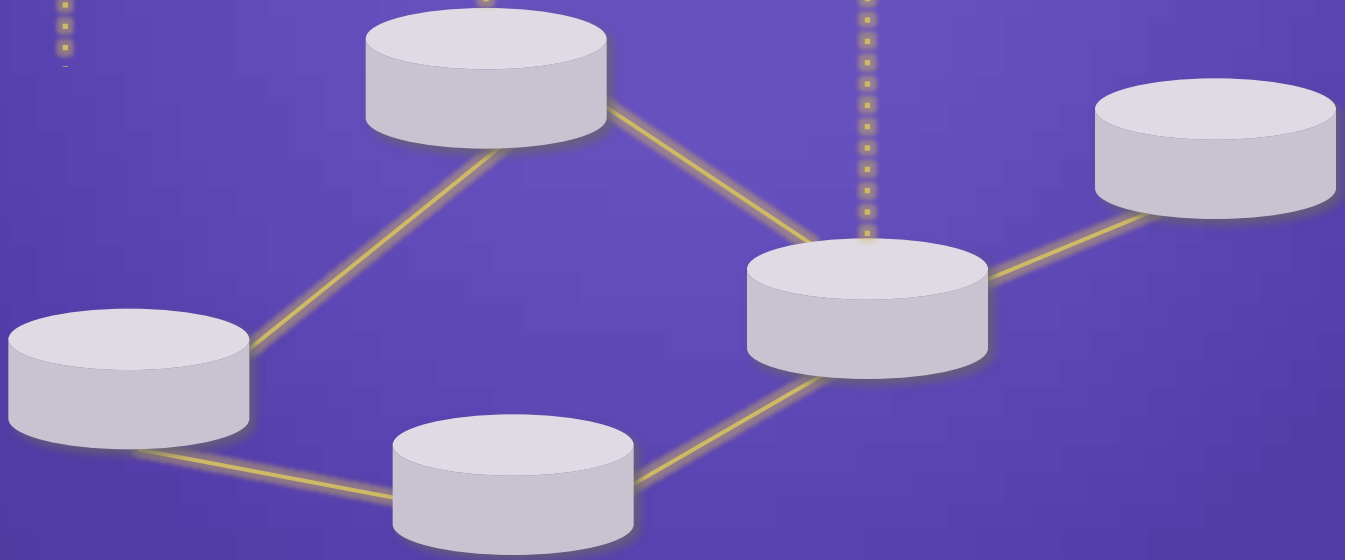
SDN

Enrutamiento, control de acceso, firewall, ...

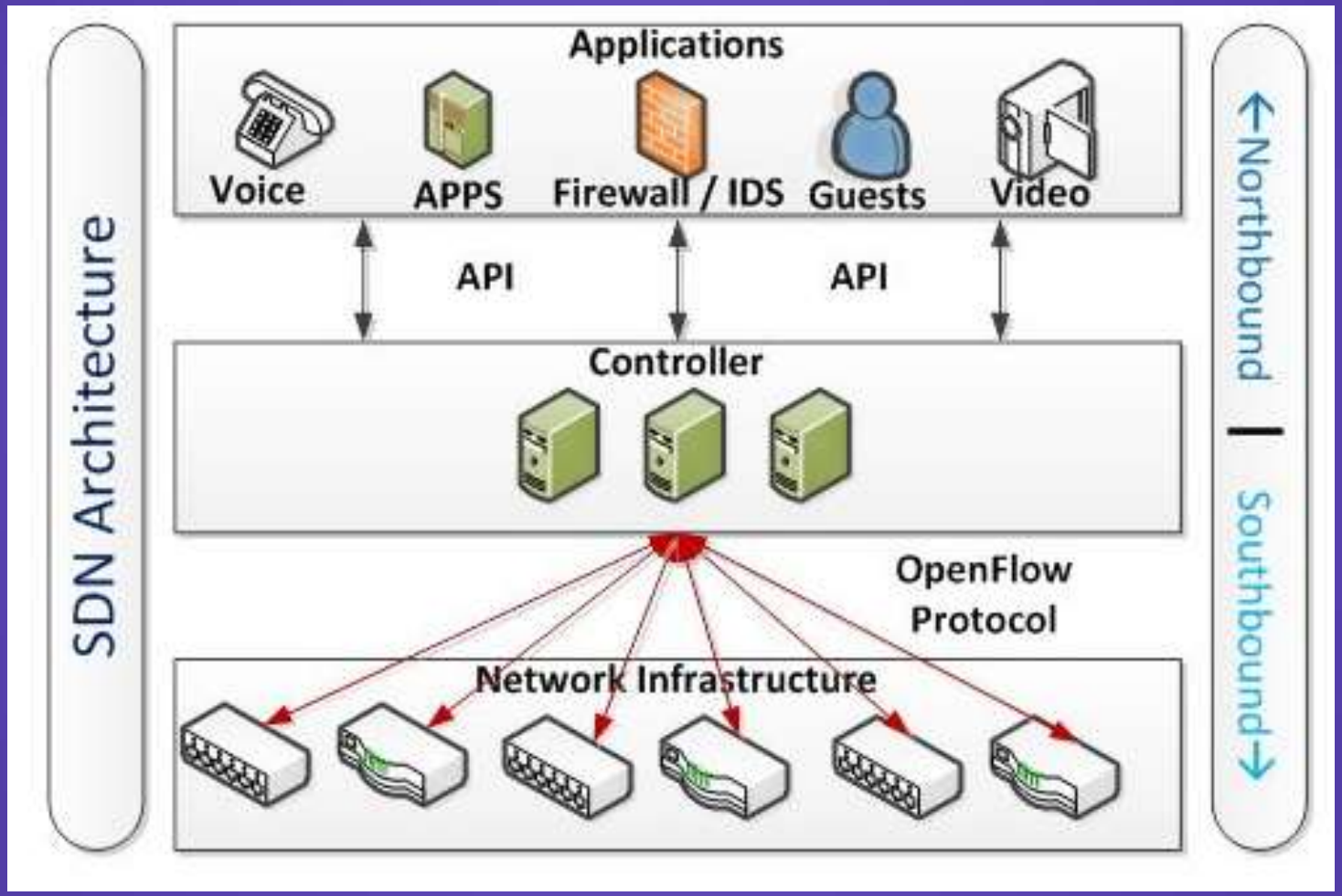
Aplicación de Control

Vista Global de la Red

SO de red



Arquitectura de SDN



Fuente: "SDN and OpenFlow for beginners" – Tiweari, V.

Arquitectura de SDN

- Infraestructura de Red:
  - Dispositivos de conectividad
    - Físicos o virtuales
    - Habilitados o dedicados
  - Denominados usualmente switches

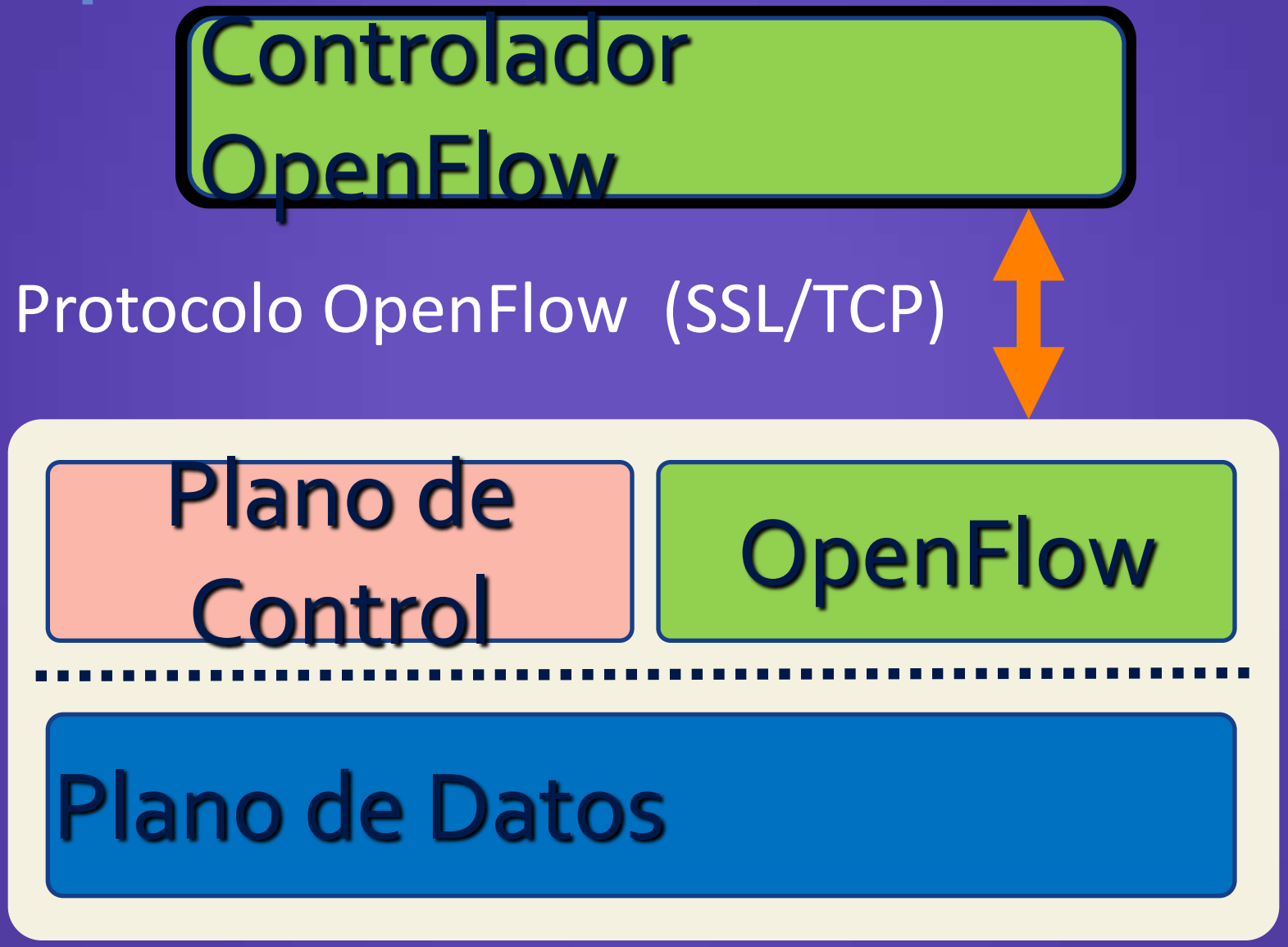
Arquitectura de SDN

- Controlador:
  - Software centralizado (lógicamente)
  - Interactúa con todos los dispositivos de conectividad.
  - Dispone de API abiertas.
  - Actúa como un sistema operativo de red
    - Visión general de la misma.
  - Las aplicaciones que se ejecutan en el controlador determinarán cómo se comportan los flujos.
  - Diferentes alternativas: Beacon, Floodlight, Trema, NOX/POX, Open DayLight, entre otros.

Arquitectura de SDN

- Aplicaciones:
  - Aplicaciones y servicios de red.
  - Interactúan con el controlador solicitándole ciertos requerimientos que la red debe cumplir.
- Protocolo OpenFlow:
  - Permite que el controlador (plano de control) se comunique con los dispositivos de conectividad (plano de datos).

OpenFlow


Controlador OpenFlow
Protocolo OpenFlow (SSL/TCP)
Plano de Control
OpenFlow
Plano de Datos

Funcionamiento de OpenFlow

Controlador

OpenFlow

Tabla de OpenFlow

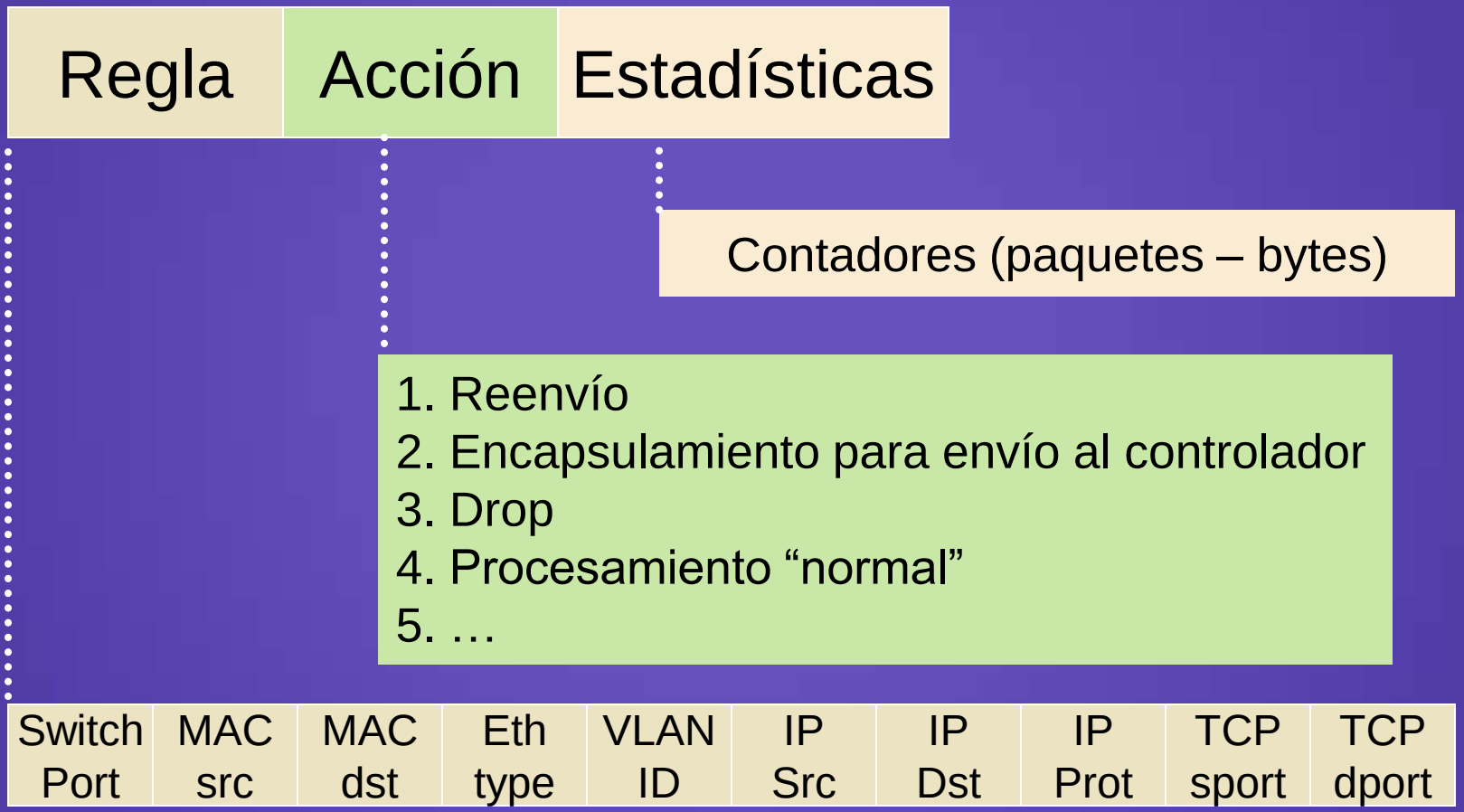
MAC src	MAC dst	IP Src	IP Dst	TCP sport	TCP dport	Acción
*	*	*	5.6.7.8	*	*	puerto 1

puerto 1 puerto 2 puerto 3 puerto 4

5.6.7.8

1.2.3.4

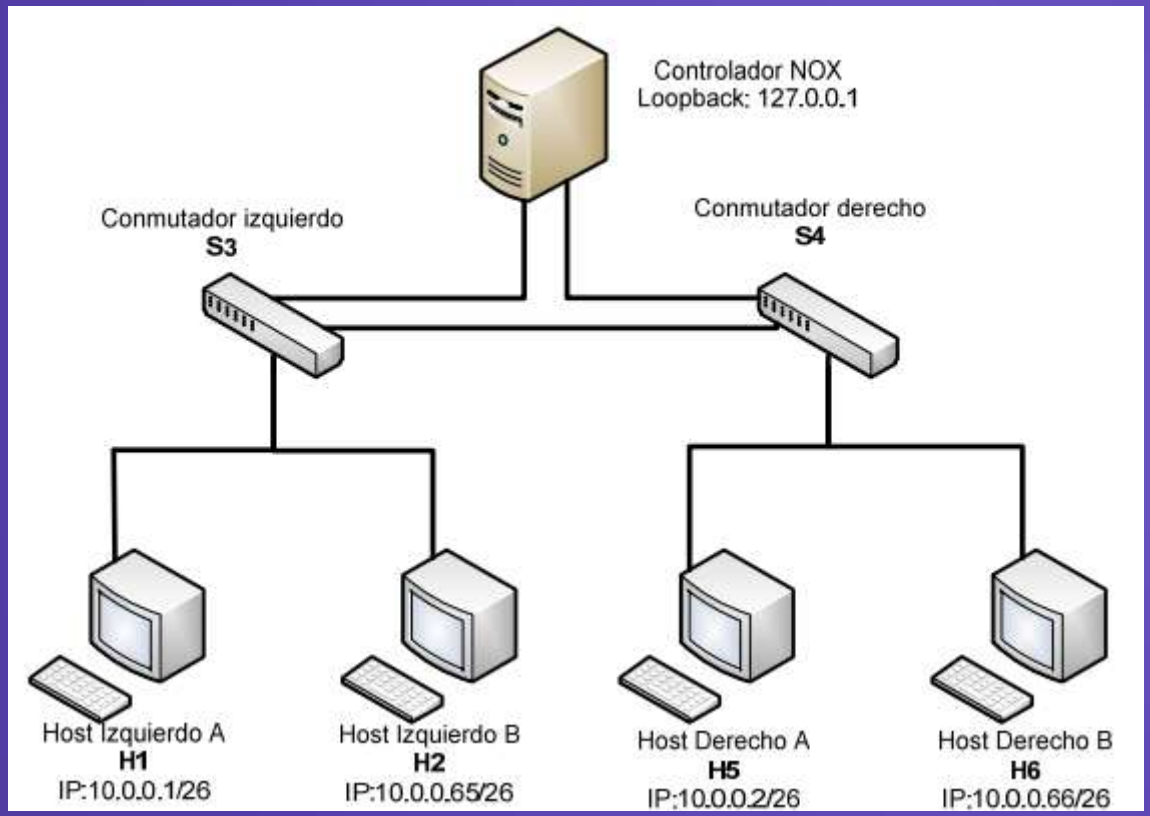
Registros de la Tabla



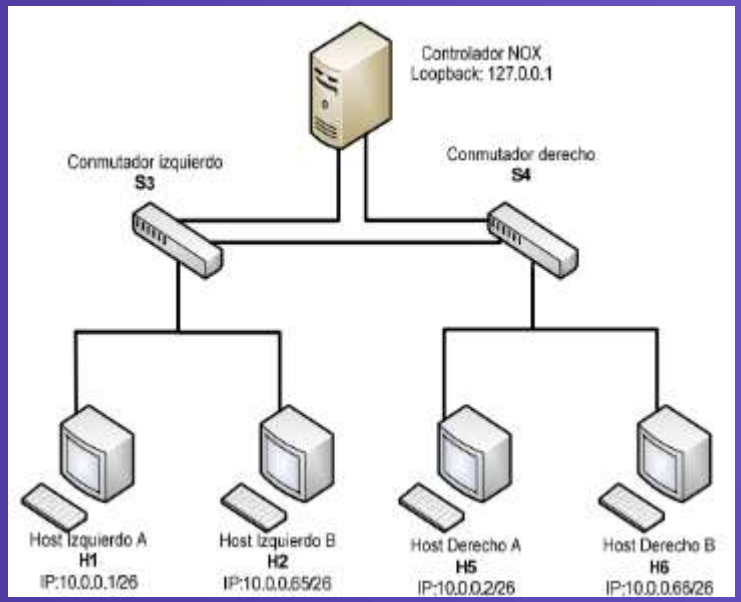
Mininet

- Simulador de SDN.
- Corre sobre Linux.
- Permite crear topologías personalizadas mediante scripts.
  - Python
- Para la simulación emula los diferentes enlaces, PC, switches y controladores
  - Emplea diferentes mecanismos de virtualización del sistema operativo Linux

Mininet


Controlador NOX
Loopback: 127.0.0.1
Conmutador izquierdo
S3
Conmutador derecho
S4
Host Izquierdo A
H1
IP:10.0.0.1/26
Host Izquierdo B
H2
IP:10.0.0.65/26
Host Derecho A
H5
IP:10.0.0.2/26
Host Derecho B
H6
IP:10.0.0.66/26

Mininet

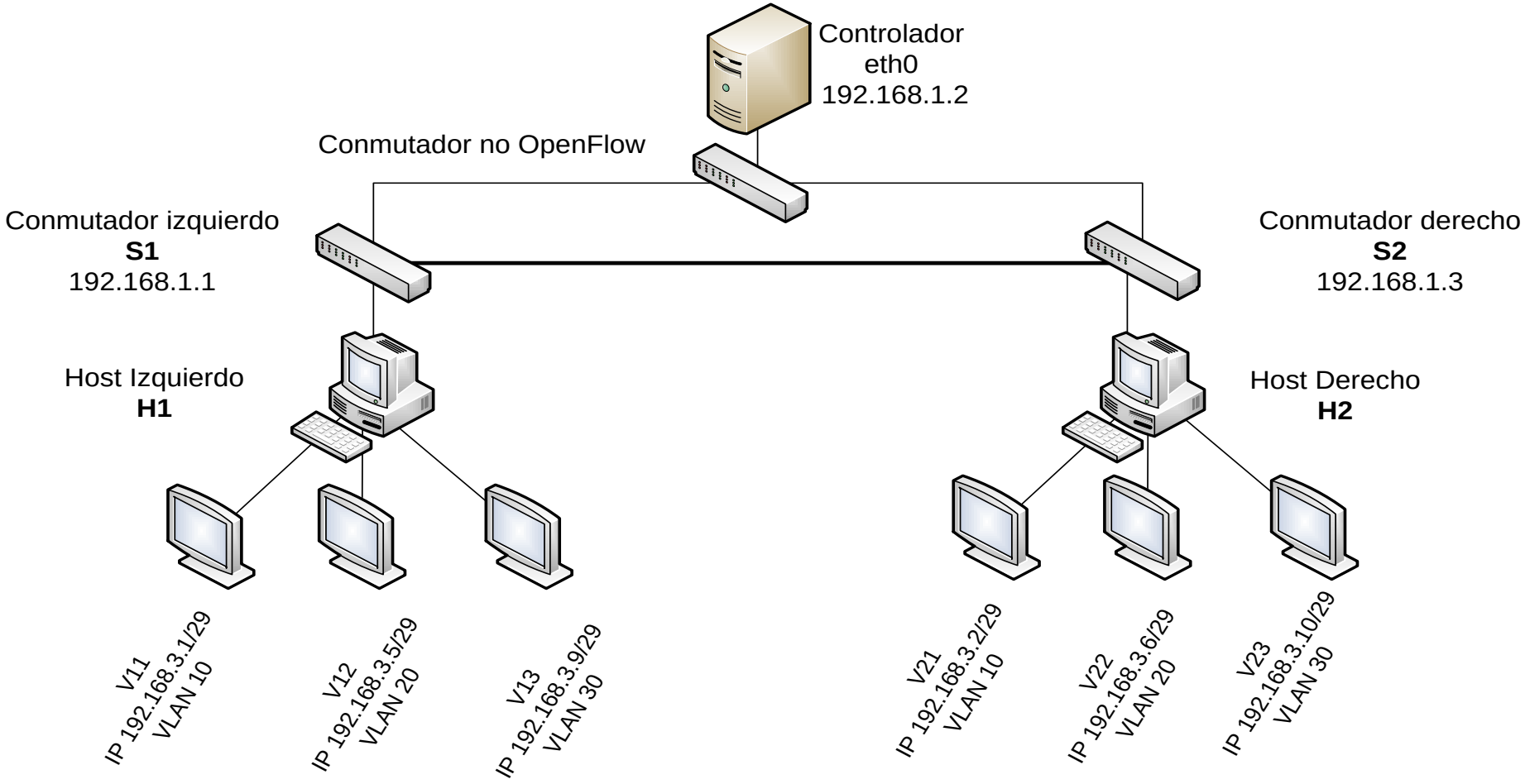



```
from mininet.topo import Topo, Node

class MiTopo( Topo ) :
  def __init__( self, enable_all = True ) :
    super( MiTopo.self ).__init__()

    hostIzquierdoA = 1
    hostIzquierdoB = 2
    switchIzquierdo = 3
    switchDerecho = 4
    hostDerechoA = 5
    hostDerechoA = 6

    self.add_node( switchIzquierdo, Node( is_switch = True ))
    self.add_node( switchDerecho, Node( is_switch = True ))
    self.add_node( hostIzquierdoA, Node( is_switch = False ))
    self.add_node( hostIzquierdoB, Node( is_switch = False ))
    self.add_node( hostDerechoA, Node( is_switch = False ))
    self.add_node( hostDerechoA, Node( is_switch = False ))

    self.add_edge( hostIzquierdoA, switchIzquierdo )
    self.add_edge( hostIzquierdoB, switchIzquierdo )
    self.add_edge( hostDerechoA, switchDerecho )
    self.add_edge( hostDerechoB, switchDerecho )
    self.add_edge( switchIzquierdo, switchDerecho )

    self.enable_all()

topos = { 'mitopo' : ( lambda : MiTopo() ) }
```

Prototipo de SDN basado en Switches Habilitados

- Prototipo de SDN empleando switches habilitados.
  - Switches de bajo costo con su firmware modificado
  - Linksys WRT54GL - OpenWRT
  - Soporte del protocolo OpenFlow
- Se emplearon cuatro de los principales controladores: NOX, POX, Beacon y Floodlight.
- Para cada controlador se desarrolló un componente de software
  - Python y Java

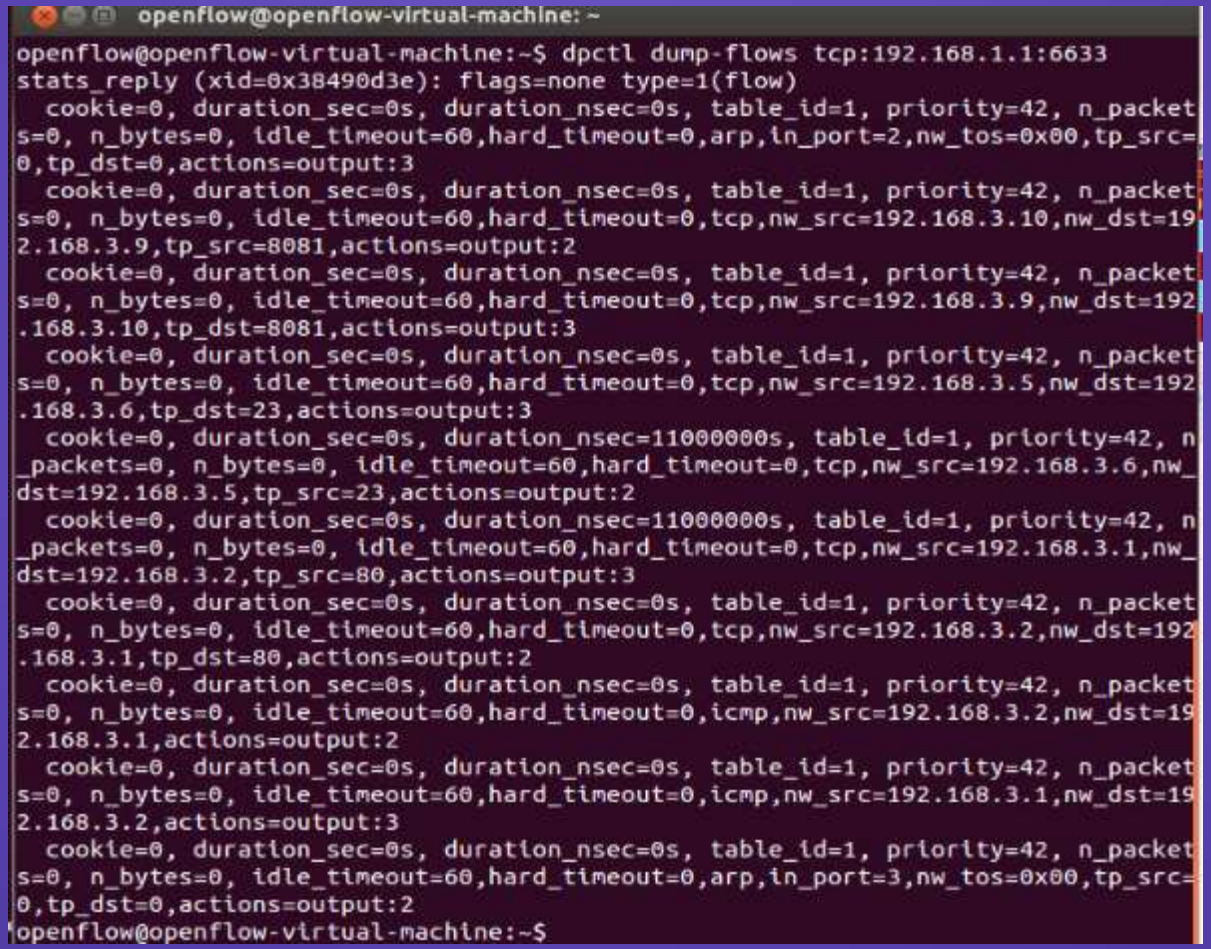
Prototipo de SDN basado en Switches Habilitados



Prototipo de SDN basado en Switches Habilitados

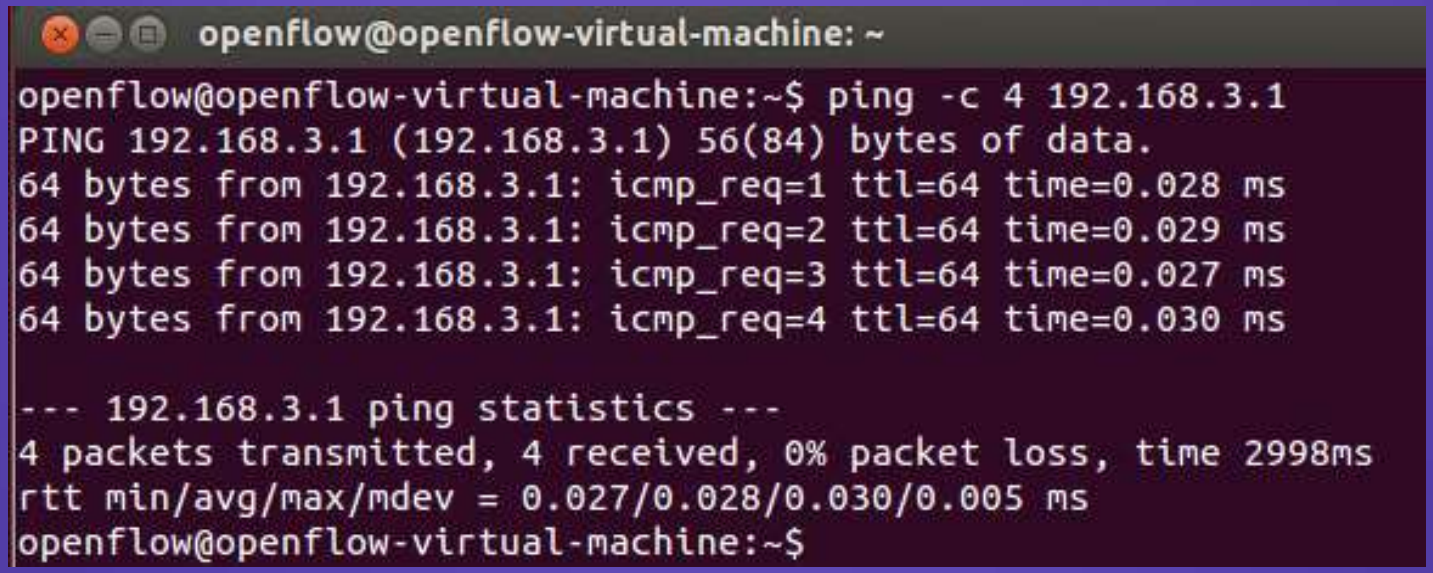
- Reglas:
  - ARP - intercambio de mensajes entre todos los hosts.
  - ICMP - intercambio de mensajes entre V11 y V21.
  - HTTP - intercambio de mensajes entre V11 (cliente– puerto dinámico) y V21 (servidor – puerto 80).
  - Telnet - intercambio de tráfico entre V12 (cliente – puerto dinámico) y V22 (servidor – puerto 23).
  - Video *streaming* - intercambio de mensajes entre V13 (cliente) y V23 (servidor - puerto 8081).

Prototipo de SDN basado en Switches Habilitados



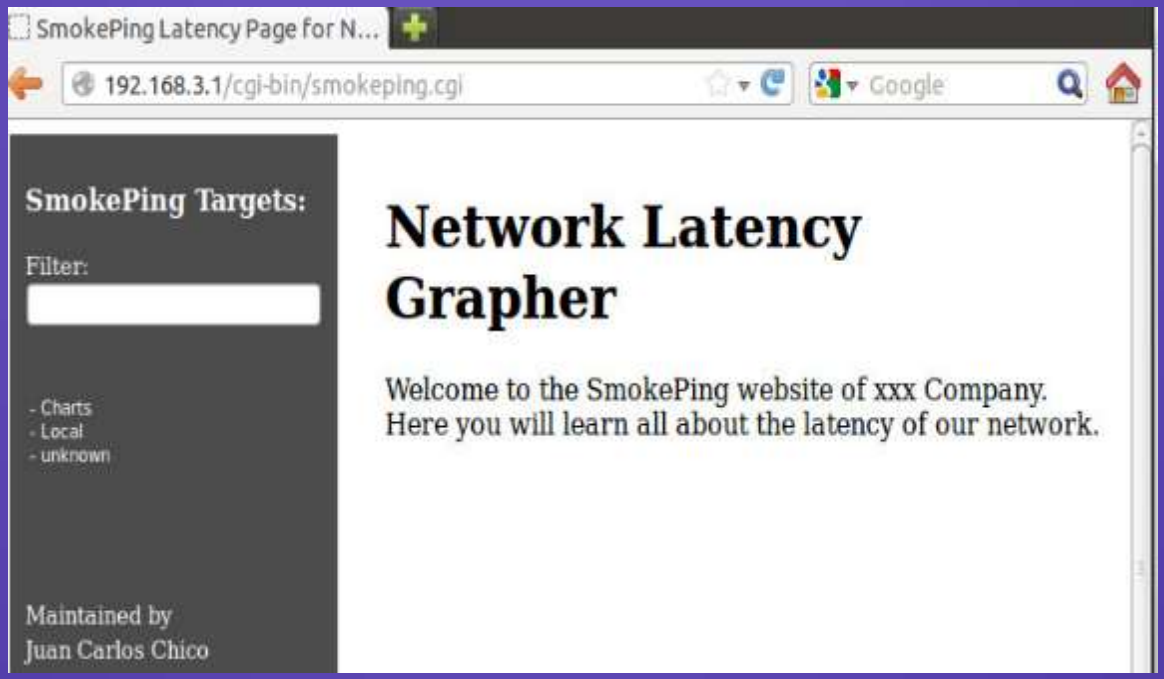
Prototipo de SDN basado en Switches Habilitados

- Pruebas:



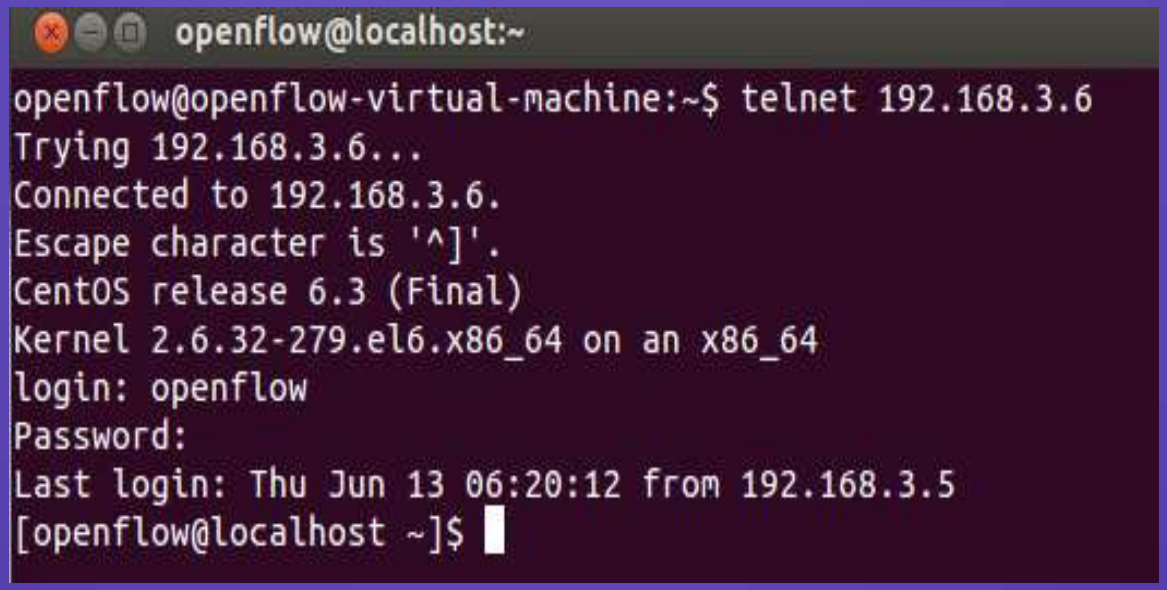
Prototipo de SDN basado en Switches Habilitados

- Pruebas:



Prototipo de SDN basado en Switches Habilitados

- Pruebas:



Prototipo de SDN basado en Switches Habilitados

- Pruebas:

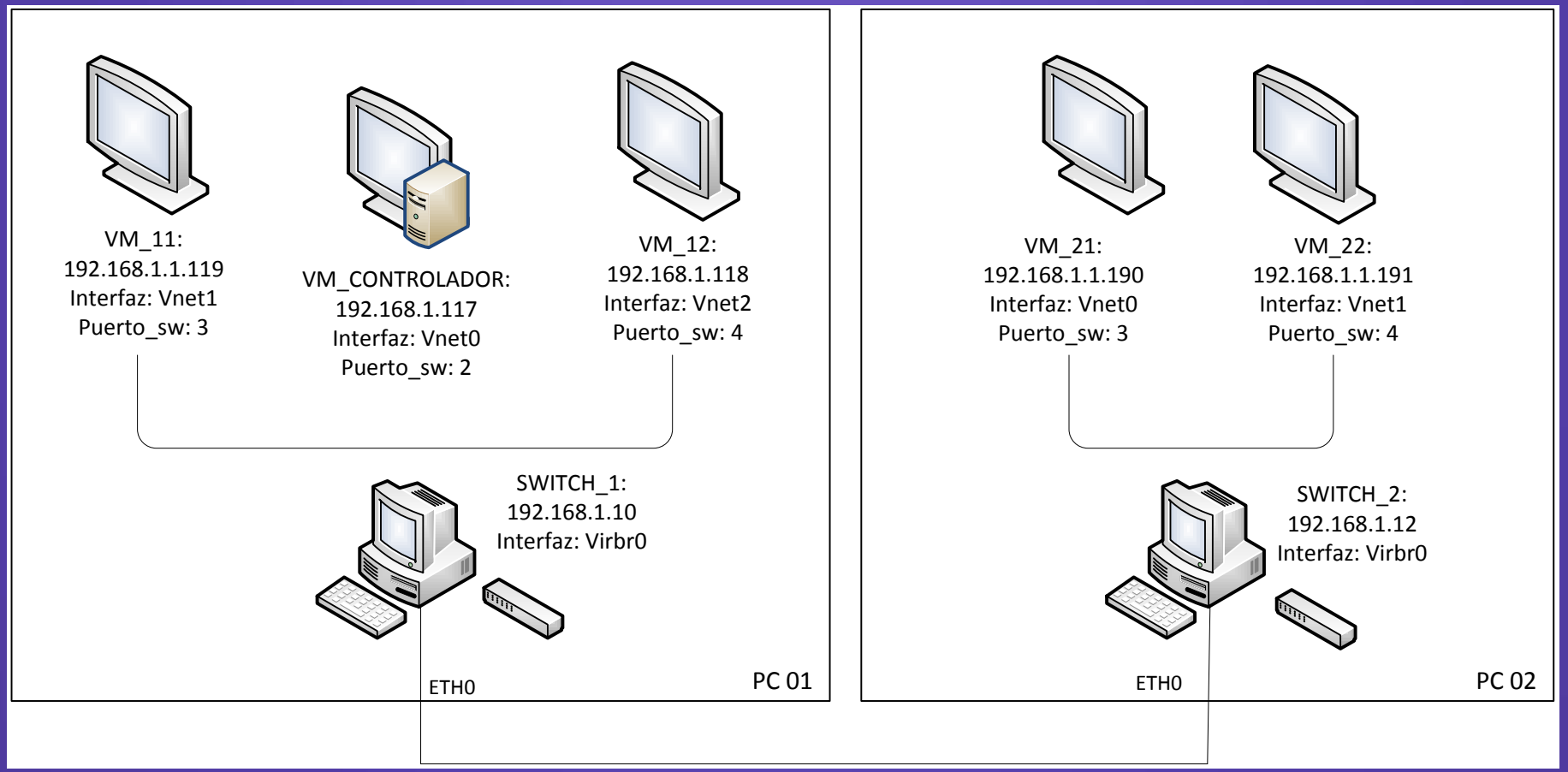
Prototipo de SDN basado en Switches Habilitados

- La programación de los tres componentes es similar.
  - Cambios debido a los nombres de métodos o variables.
- El administrador tiene el control completo sobre todo el tráfico que pasa por la red.

Prototipo de SDN basado en Switches Virtuales

- Prototipo de SDN empleando switches virtuales.
  - Switches implementados en software
  - Open vSwitch
- Se emplearon cuatro de los principales controladores: NOX, Beacon, Trema y Floodlight.
- Para cada controlador se hicieron pruebas para determinar en cual de ellos es más sencillo desarrollar una aplicación NAC
  - Python, Ruby y Java

Prototipo de SDN basado en Switches Virtuales



Prototipo de SDN basado en Switches Virtuales

- NAC (básico):
  - Permite control de acceso a la red.
  - Aplicativo cliente en cada PC.
    - Permitirá indicar si el PC está “saludable”.
  - Aplicación de control
    - Genera las reglas en función de la información del aplicativo cliente.
    - Deja pasar el tráfico o no permite el acceso del PC a la red.