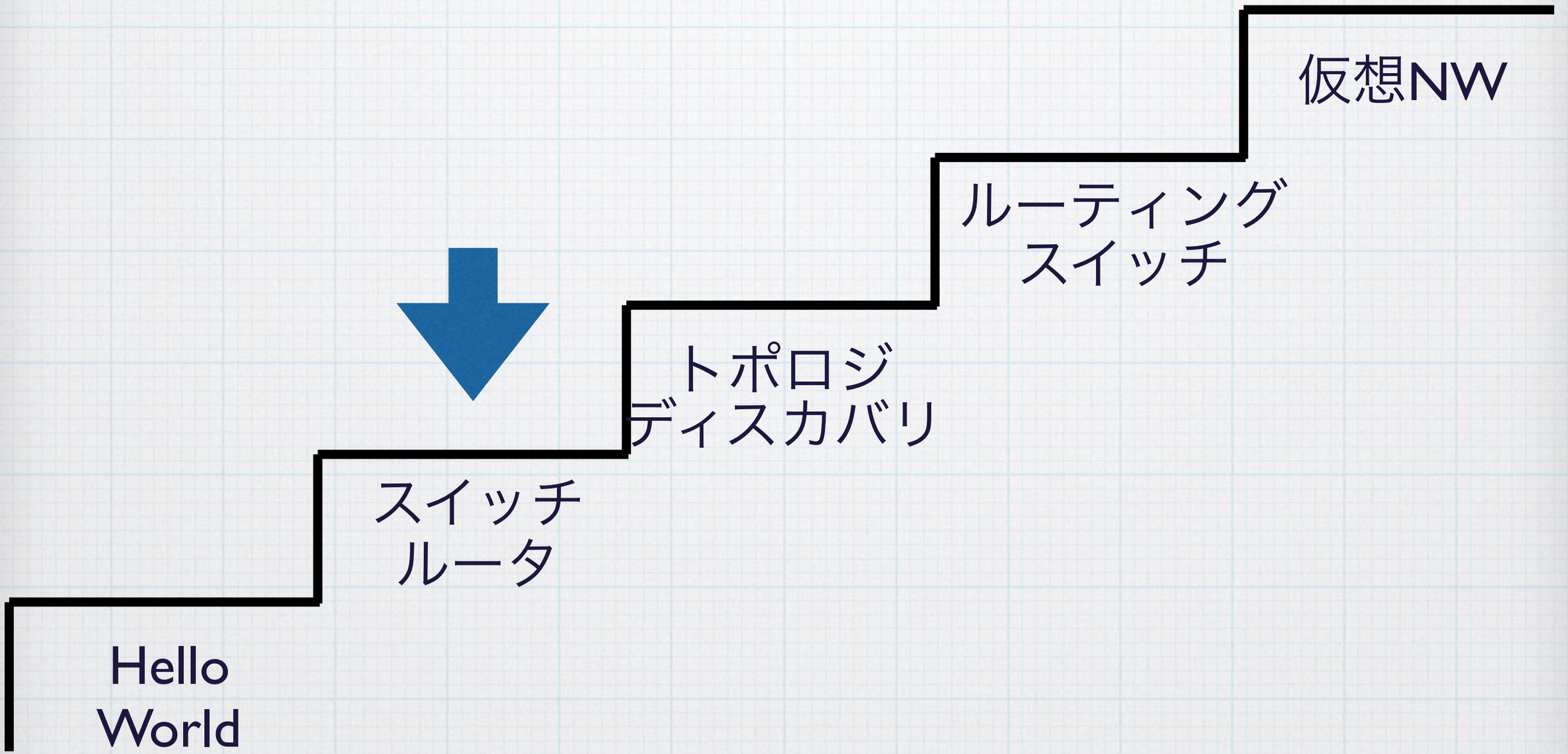
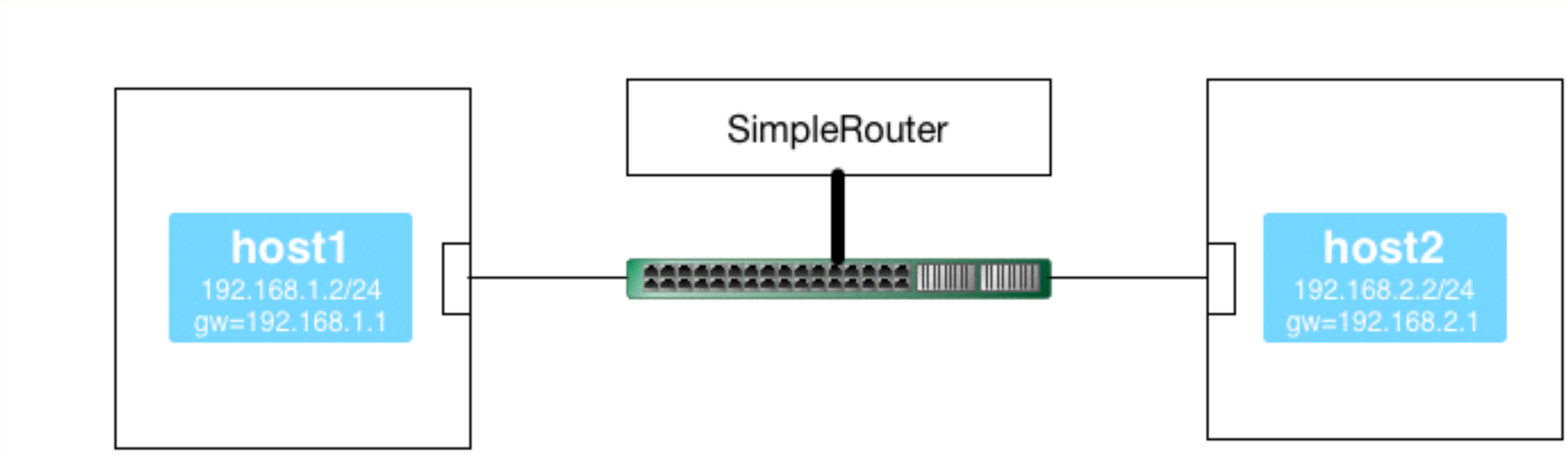
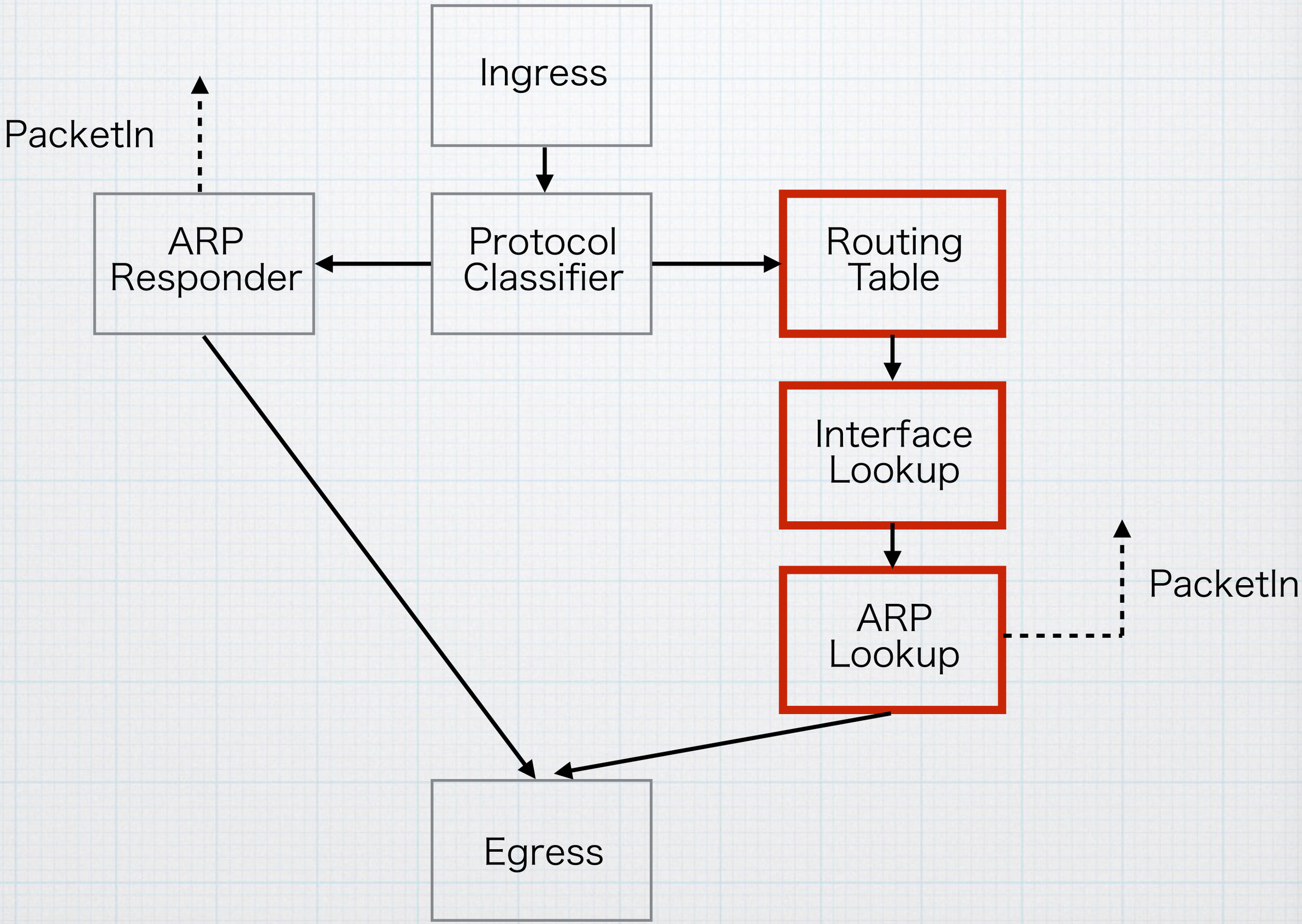
OpenFlow1.3を
使いこなそう
(ルータを作ろう 後編)


仮想NW
ルーティング
スイッチ
トポロジ
ディスカバリ
スイッチ
ルータ
Hello
World

レポート(グループ課題)

- ルータをマルチプルテーブル化
  - テーブル構成とソースコードの説明
  - 動作確認 (ARP, ping, etc.)




Ingress
PacketIn
ARP
Responder
Protocol
Classifier
Routing
Table
Interface
Lookup
ARP
Lookup
PacketIn
Egress

大ヒント

第6回 (11/18)

1. グループ課題: ルータをマルチプルテーブルで実装しよう

今回の課題には新しいTremaが必要です。次の手順でTremaをアップデートしてください。

```
$ bundle update
```

課題の大ヒント

```
$ sudo ovs-ofctl dump-flows br0x1 --protocols=OpenFlow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=12.684s, table=0, n_packets=11, n_bytes=858, priority=0 actions=goto_tabl
 cookie=0x0, duration=12.684s, table=1, n_packets=0, n_bytes=0, priority=0,arp actions=goto_tab
 cookie=0x0, duration=12.684s, table=1, n_packets=0, n_bytes=0, priority=0,ip actions=goto_tabl
 cookie=0x0, duration=12.674s, table=2, n_packets=0, n_bytes=0, priority=0,arp,in_port=1,arp_tp
 cookie=0x0, duration=12.665s, table=2, n_packets=0, n_bytes=0, priority=0,arp,in_port=1,arp_tp
 cookie=0x0, duration=12.632s, table=2, n_packets=0, n_bytes=0, priority=0,arp,in_port=2,arp_tp
 cookie=0x0, duration=12.623s, table=2, n_packets=0, n_bytes=0, priority=0,arp,in_port=2,arp_tp
 cookie=0x0, duration=12.654s, table=2, n_packets=0, n_bytes=0, priority=0,arp,reg1=0x1 actions
 cookie=0x0, duration=12.612s, table=2, n_packets=0, n_bytes=0, priority=0,arp,reg1=0x2 actions
 cookie=0x0, duration=12.594s, table=3, n_packets=0, n_bytes=0, priority=40024,ip,nw_dst=192.16
 cookie=0x0, duration=12.585s, table=3, n_packets=0, n_bytes=0, priority=40024,ip,nw_dst=192.16
 cookie=0x0, duration=12.603s, table=3, n_packets=0, n_bytes=0, priority=0,ip actions=load:0xc0
 cookie=0x0, duration=12.577s, table=4, n_packets=0, n_bytes=0, priority=0,reg0=0xc0a80100/0xff
 cookie=0x0, duration=12.569s, table=4, n_packets=0, n_bytes=0, priority=0,reg0=0xc0a80200/0xff
 cookie=0x0, duration=12.538s, table=5, n_packets=0, n_bytes=0, priority=2,ip,reg0=0xc0a80101 a
 cookie=0x0, duration=12.530s, table=5, n_packets=0, n_bytes=0, priority=2,ip,reg0=0xc0a80201 a
 cookie=0x0, duration=12.525s, table=5, n_packets=0, n_bytes=0, priority=1,ip actions=CONTROLLE
 cookie=0x0, duration=12.520s, table=6, n_packets=0, n_bytes=0, priority=0 actions=output:NXM_N
```

必要なアクション

- NiciraRegLoad.new(value, フィールド|レジスタ)
  - フィールドやレジスタに即値を入れる
- SendOutPort.new(:table)
  - テーブル0番にもう一回入れる
  - ARP未解決なパケットをコントローラに溜めておいて、解決したタイミングで入れる

```
ip_addr = IPv4Address.new('192.168.0.1')
# FlowModやPacketOutのアクションでreg1にIPアドレスを代入
Apply.new(NiciraRegLoad.new(ip_addr.to_i, :reg0))

# セットしたレジスタはMatchにかけることも可能(マスク指定も)
match: Match.new(reg0: network_address.to_i,
                 reg0_mask: netmask.to_i),
```

- レジスタ: 各パケットに付けられる変数
- テーブル間での値の引き渡し (宛先IPアドレス等)
- reg0〜reg7 の 8 個が使用可能

注意点

- ping応答はコントローラでやってください
  - Open vSwitchがICMPの書き換えに未対応
- TTL減算はやらなくていいです
- APIで不明な点はすぐ聞いてください
  - 今回の課題は実装量が多めです

レポート(再掲)

- ルータをマルチプルテーブル化
  - テーブル構成とソースコードの説明
  - 動作確認 (ARP, ping, etc.)