MPLS/BGP VPNs

Olof Hagsand KTH CSC

Literature

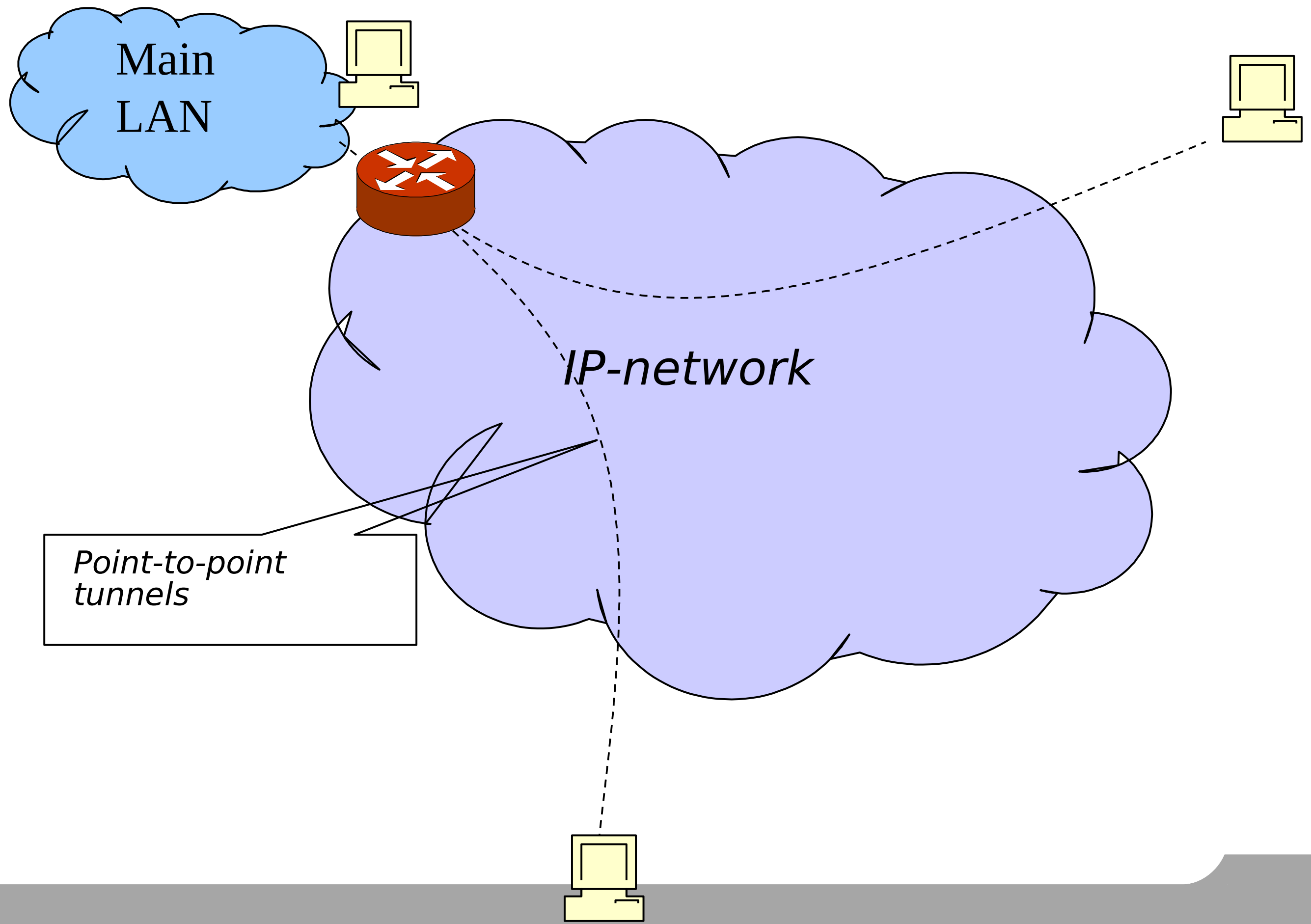
- Practical BGP: Chapter 10
- MPLS repetition,
  - see for example http://www.csc.kth.se/utbildning/kth/kurser/DD2490/ipro1-11/lectures/MPLS.pdf
- Reference:
  - JunOS Cookbook: Chapter 14 and 15
  - Junos software 10.1 VPNs Configuration Guide
  - draft-kompella-ppvpn-l2vpn-03.txt, Layer 2 VPN Over Tunnels
  - RFC 4364 bis (L3VPN)

Motivation to VPN

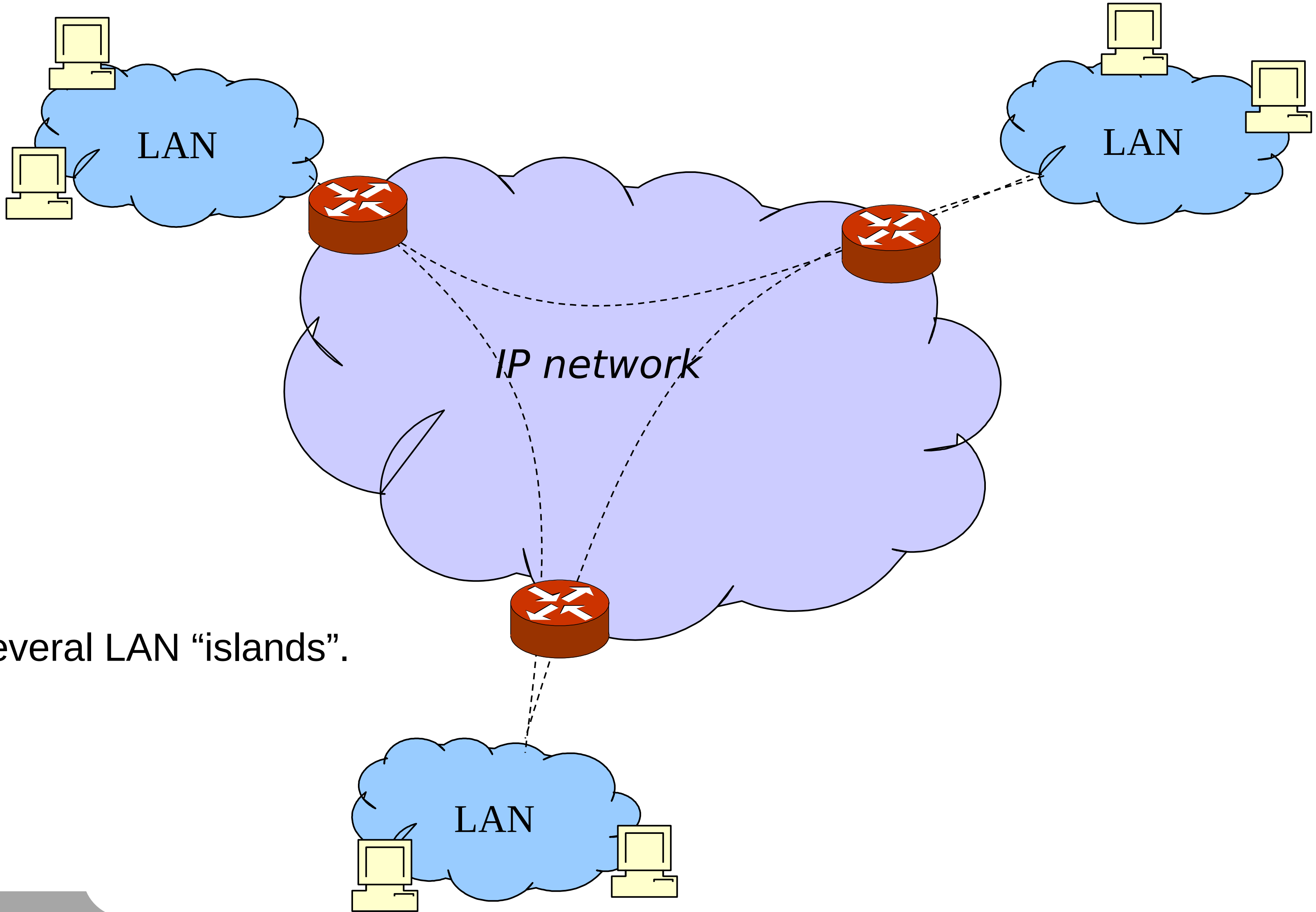
- Companies and organizations wish to connect their local offices, collect data in an isolated network, or have personel working from their home or while travelling.
- Leased lines are expensive, it makes sense to use IP and the Internet.
- The motivation for VPNs is therefore primary economical

VPN simple architecture

Connect hosts to central server/LAN.



Generic VPN Architecture



Connect several LAN “islands”.

Addressing and Security

- Public IP networks are public and have only one address domain.
- You may want to separate your private traffic from the global traffic (addressing)
- You may want to secure your traffic (encryption, authentication)
- Provider-based VPNs (peer)
  - You trust your provider
  - Guarantee resources
  - Provider adds service – more costly
  - One provider / set of providers only
- Customer-based VPNs (overlay)
  - Do it yourself using IPSEC tunneling
  - Cheap solution
  - Best effort
  - Internet

Provider-based VPNs using MPLS & BGP

There are several related variants including

- L2VPN – pseudowires
- VPLS – dynamic L2VPN
- L3VPN – RFC 4364

These solutions all use multiprotocol BGP, VRF (Virtual Routing and Forwarding), relays data with MPLS and have a BGP-free core.

In fact, when you have set up your MPLS+BGP core network, you can mix these VPNs. You can therefore *re-use* your infra-structure.

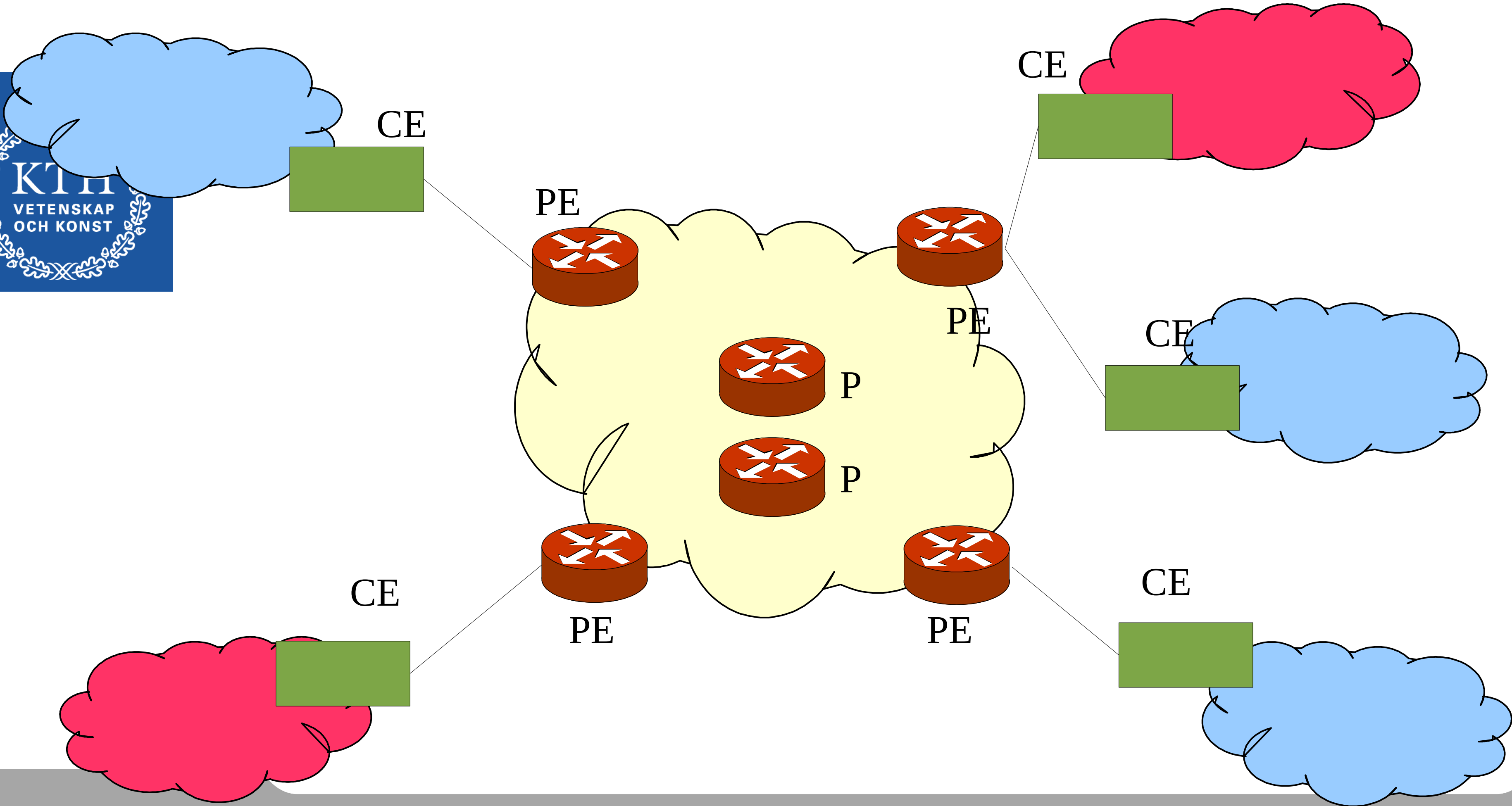
If you use RSVP you can also make traffic-engineering.

There is no 'security' in Provider-based VPNs.

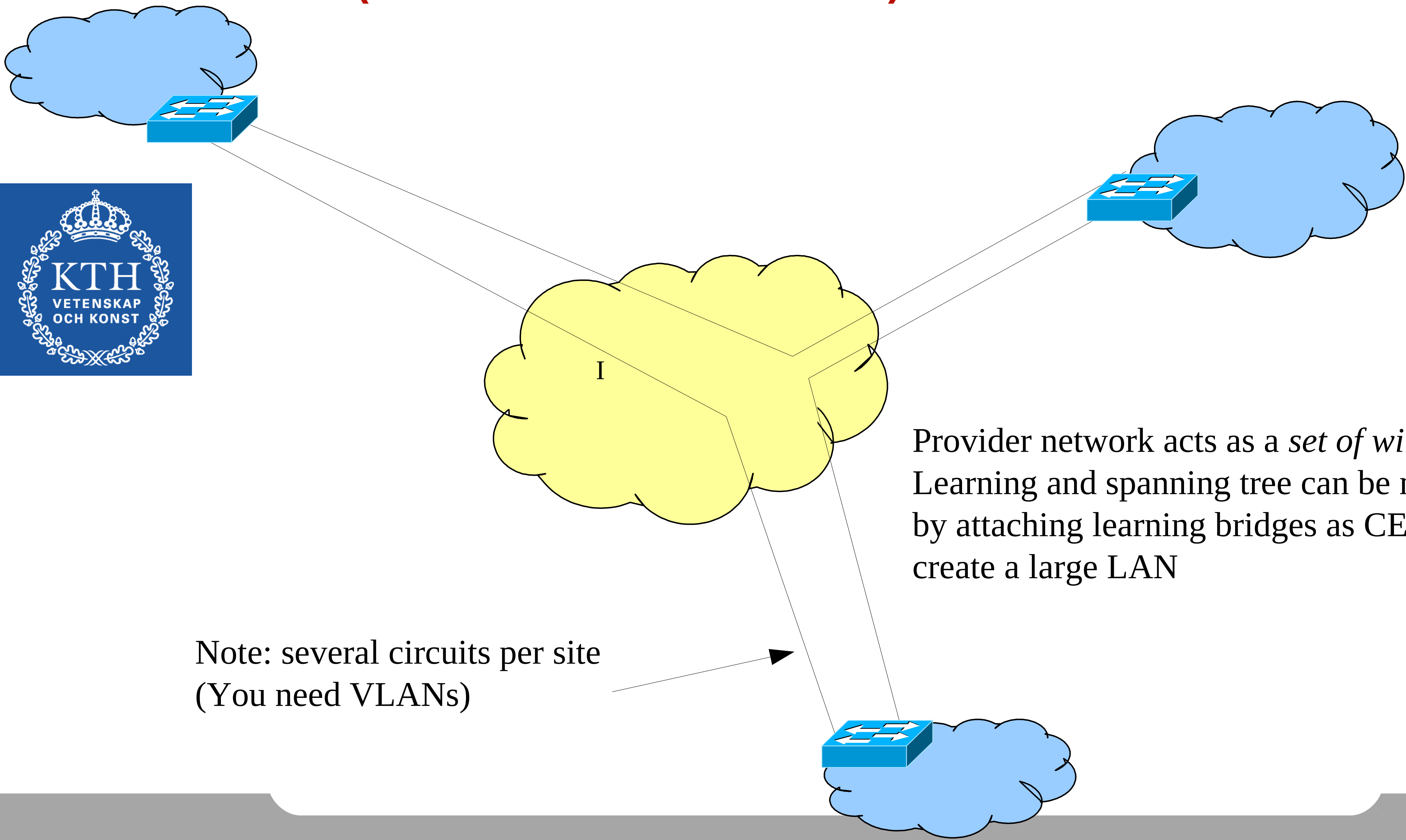
Provider-based VPNs

- CE - Customer Edge
- PE - Provider Edge (BGP)
- P - Provider (no BGP)

- More than one customer: red and blue
- More than two sites per customer
- CE is either router or L2 device



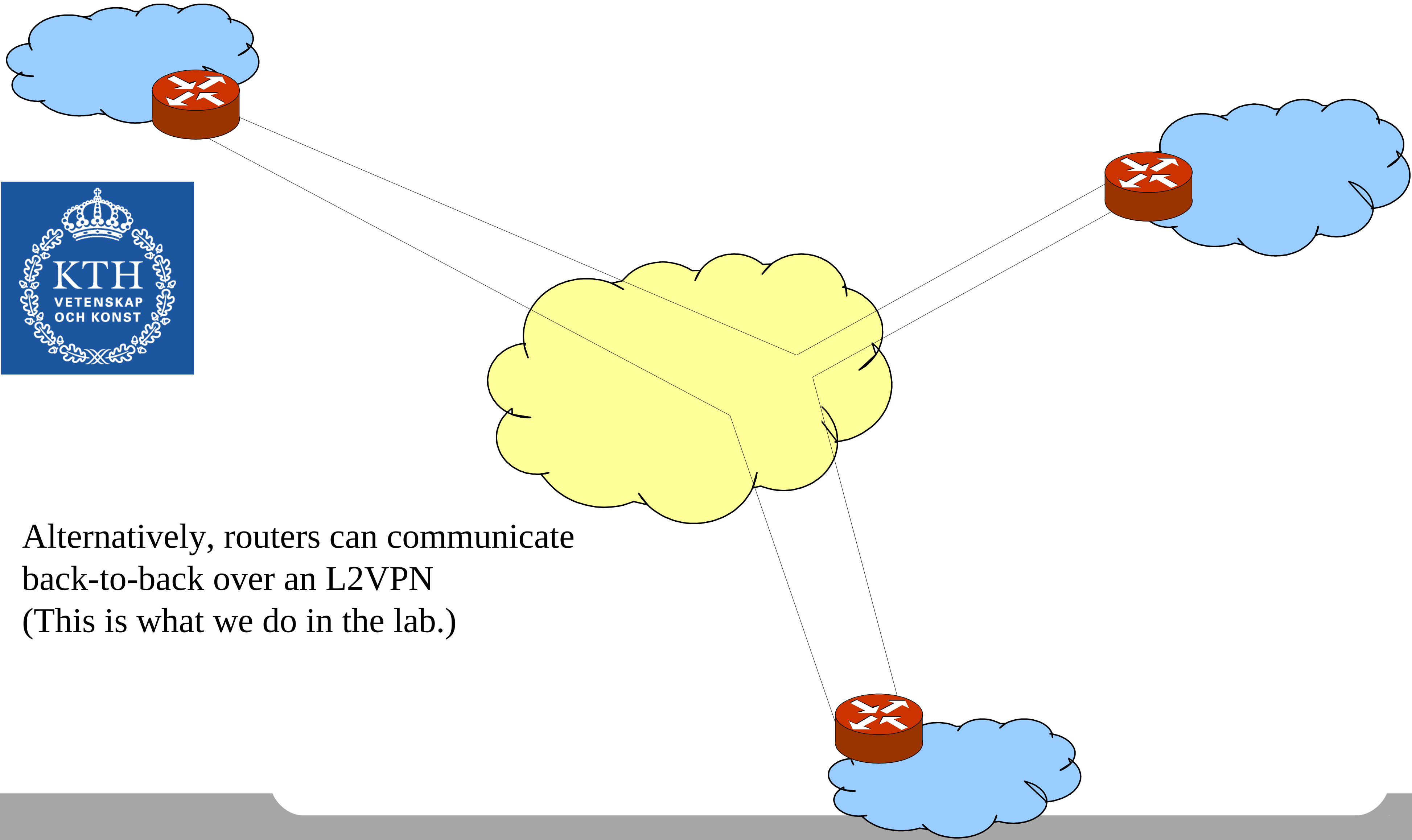
L2VPN Pseudowires (customer view)



Provider network acts as a *set of wires.* Learning and spanning tree can be made by attaching learning bridges as CE:s to create a large LAN

Note: several circuits per site
(You need VLANs)

L2VPN Pseduowires (customer view)



Alternatively, routers can communicate back-to-back over an L2VPN
(This is what we do in the lab.)

VPLS (Customer view)

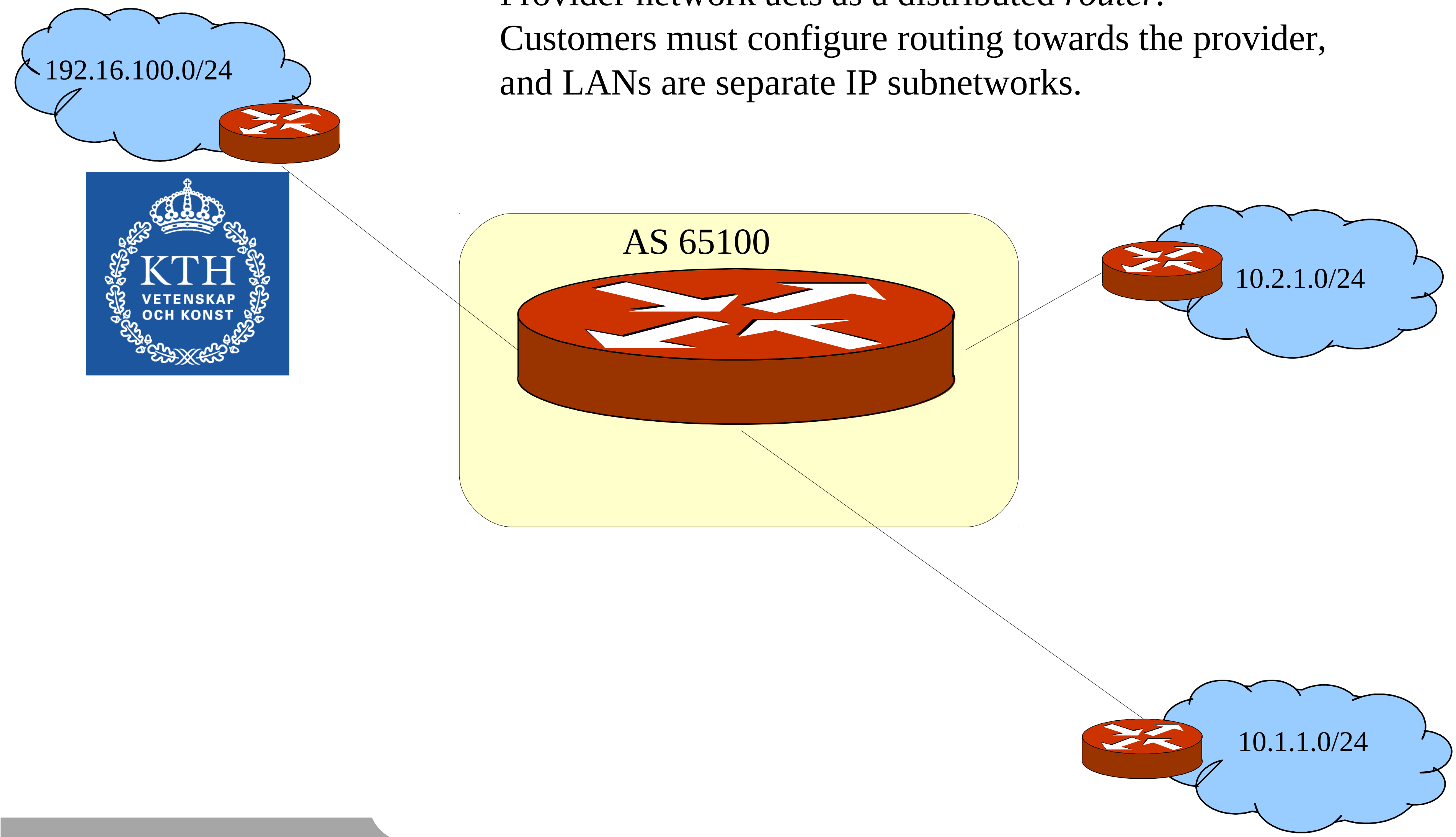
Provider network acts as a distributed *switch*
Provider network performs learning (and spaning-tree)

VPLS is *dynamic*
L2VPN is *static*

Note: only one circuit per site

L3VPN (Customer view)

Provider network acts as a distributed *router*.
Customers must configure routing towards the provider,
and LANs are separate IP subnetworks.

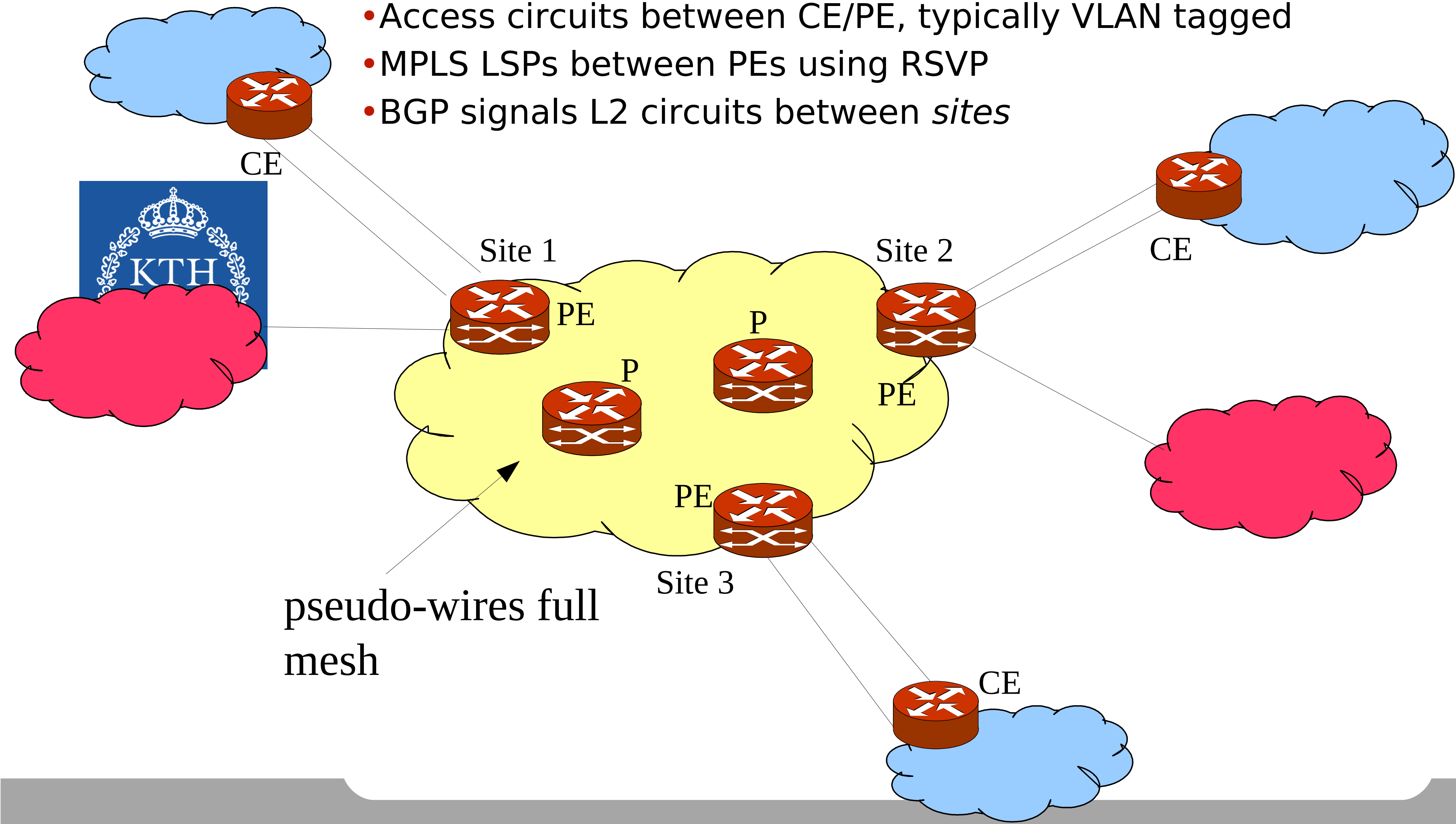


L2VPN pseudo-wire

- Static, multipoint ”overlay” solution
- Setup point-to-point L2 connections between every site in the VPN
  - Pseudo-wires
- L2 frames are encapsulated using IP and MPLS
- Requires homogenous link-layers (a wire) but can transform between some link-layers
- BGP is used as a signalling protocol to setup VPN connections between customer sites.
- RSVP (or LDP) is used to setup the MPLS paths
- MPLS multistacking is used to keep provider's network free of customer routing information
- Encryption by other means, security by trusting the provider

L2VPN provider view

- Access circuits between CE/PE, typically VLAN tagged
- MPLS LSPs between PEs using RSVP
- BGP signals L2 circuits between *sites*

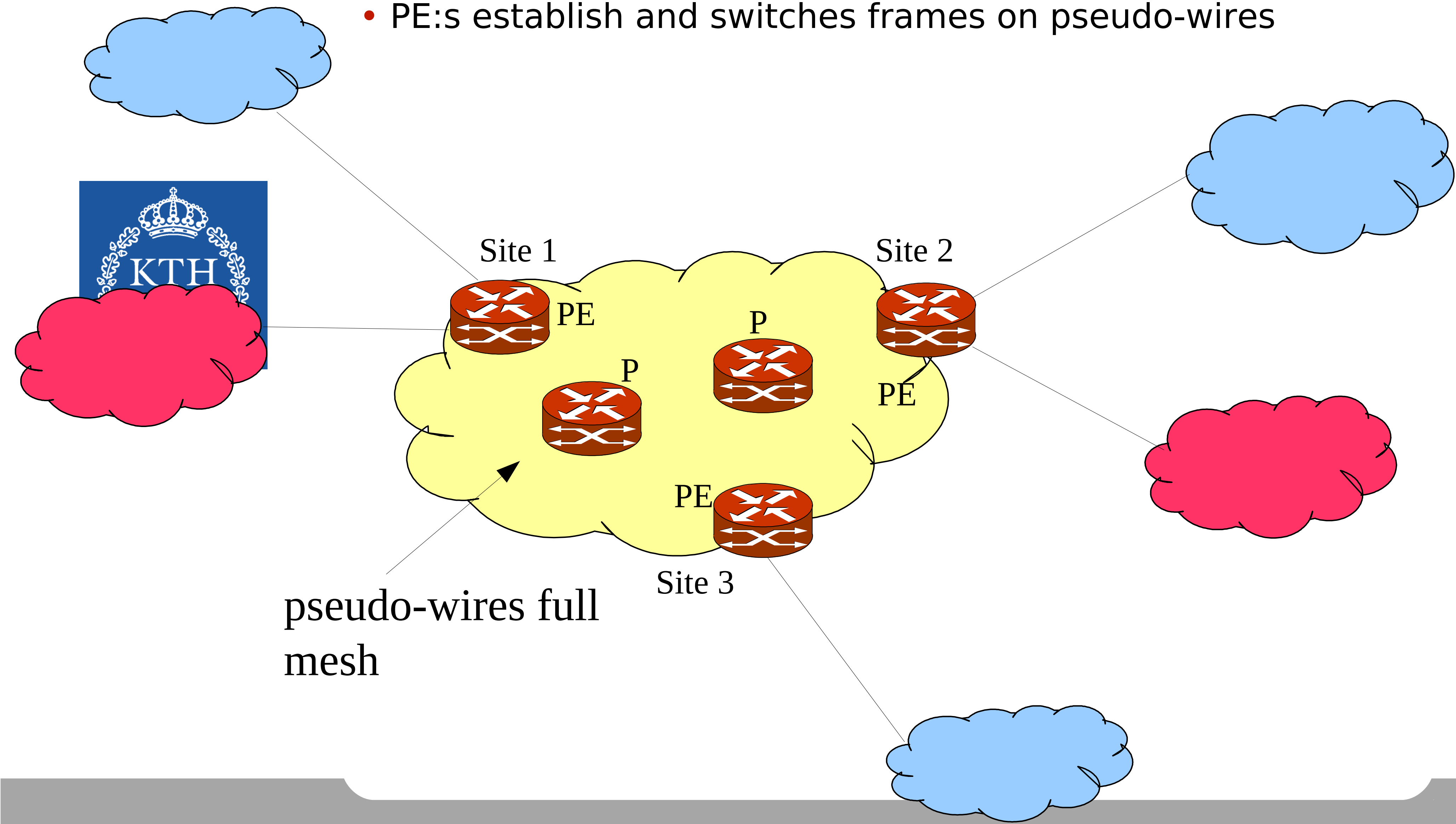


Virtual Private LAN Services (VPLS)

- Dynamic, multipoint ”peer” solution
- Backbone over IP
- Interconnects a switched L2 network
- MPLS is used together with BGP to create ”pseudo-wires” between the LAN islands.
- The PE:s dynamically establish pseudo-wires
  - Bridging (learning)
  - Spanning-Tree
- The PE:s actively chooses which pseudo-wire to send each frame on
- MP-BGP is used for distributing mac adress learning
- Disadvantage (similar to L3VPN)
  - Provider imports MAC learning tables into network

VPLS provider view

- PE:s establish and switches frames on pseudo-wires



CE-PE issues

- Since CE-PE communication needs to distinguish between different circuits, it is common to use virtual connections, as CE-PE circuits, such as VLANs. You assign one VLAN per ”wire”.
- There are many link-layers. You need to configure which encapsulation you use. We use 'ethernet-vlan', but it is possible to use other encapsulation types and translate between them using 'translational cross-connects'
- VPLS does not need VLANs, since only one connection is required, but there are still encapsulation issues

L2VPN CE-PE configuration

CE side:

```
fe-1/0/0 {
   vlan-tagging;
   unit 512 {
      Vlan-id 512; # vid and unit need not match
      family inet {
         address 10.10.11.1/30;
      }
   }
}
```

PE side: no IP address, configure encapsulation vlan-ccc

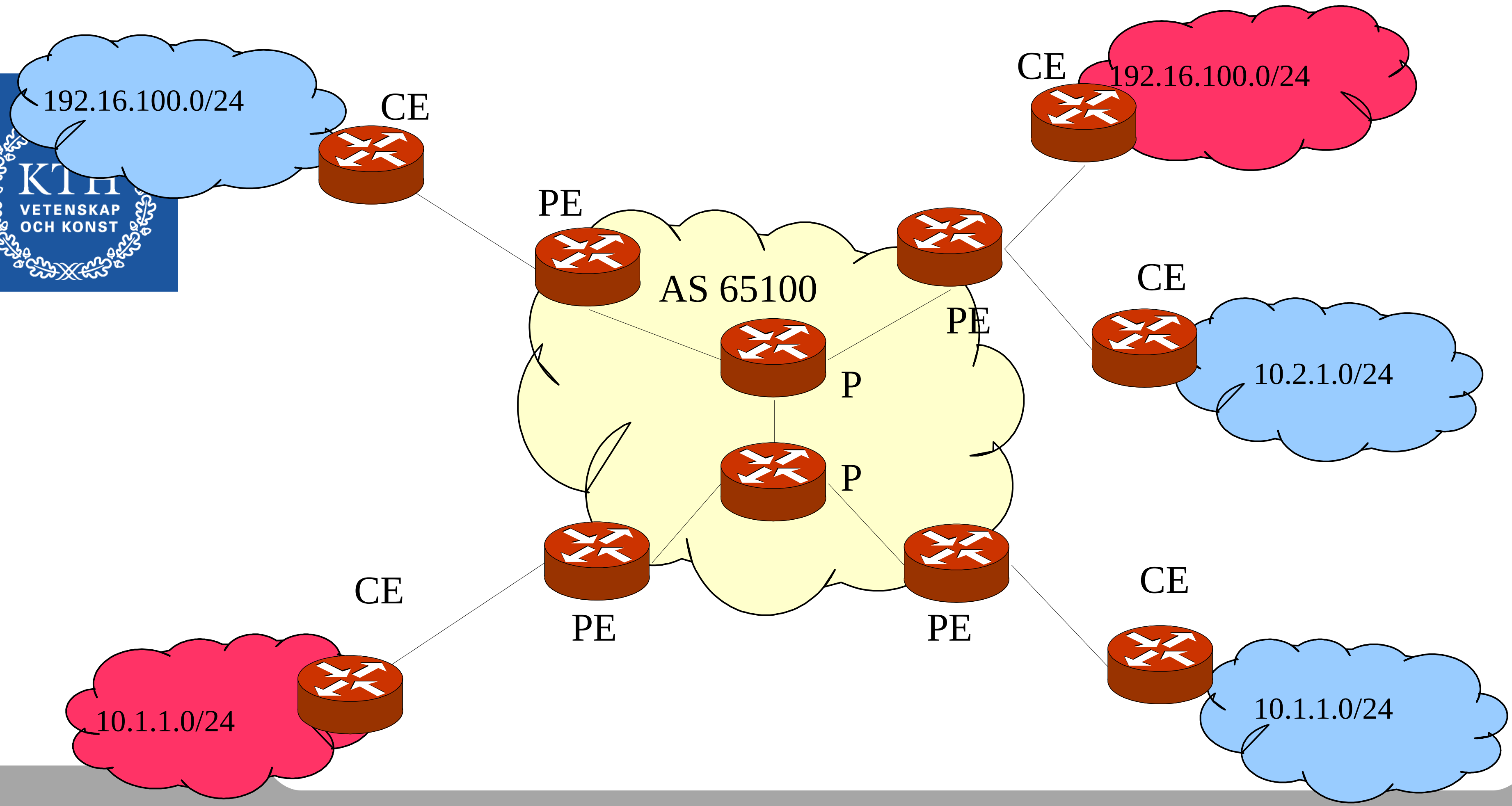
Constructing VPNs

- Before we go into details about configuring L2VPN, you need to understand some intrinsics about how VPNs are constructed.
- You need to understand:
  - Route distinguisher
  - VRFs
  - Route targets
- These are fundamental in all MPLS/BGP VPNs
- But these are most easily understood using L3VPN but are used in all VPNs

L3VPN

- L3VPN is a ”peer-type” and dynamic VPN using BGP and MPLS
- It connects IP-subnetworks belonging to the same private network.
- Each customer may use the same adress space, such as 1918 addresses
- Each customer site is modelled as a separate AS – customer interior routing runs independently at each site
- An address conversion scheme makes each customer VPN route unique within the provider's network
- Multiple routing and forwarding tables are supported on each PE separating different customer routing information
- BGP is used as a signalling protocol to setup VPN connections between customer sites.
- RSVP (or LDP) is used to setup the MPLS paths
- MPLS multistacking is used to keep provider's network free of customer routing information
- Disadvantage: Provider imports customer routing tables
- Encryption by other means, security by trusting the provider

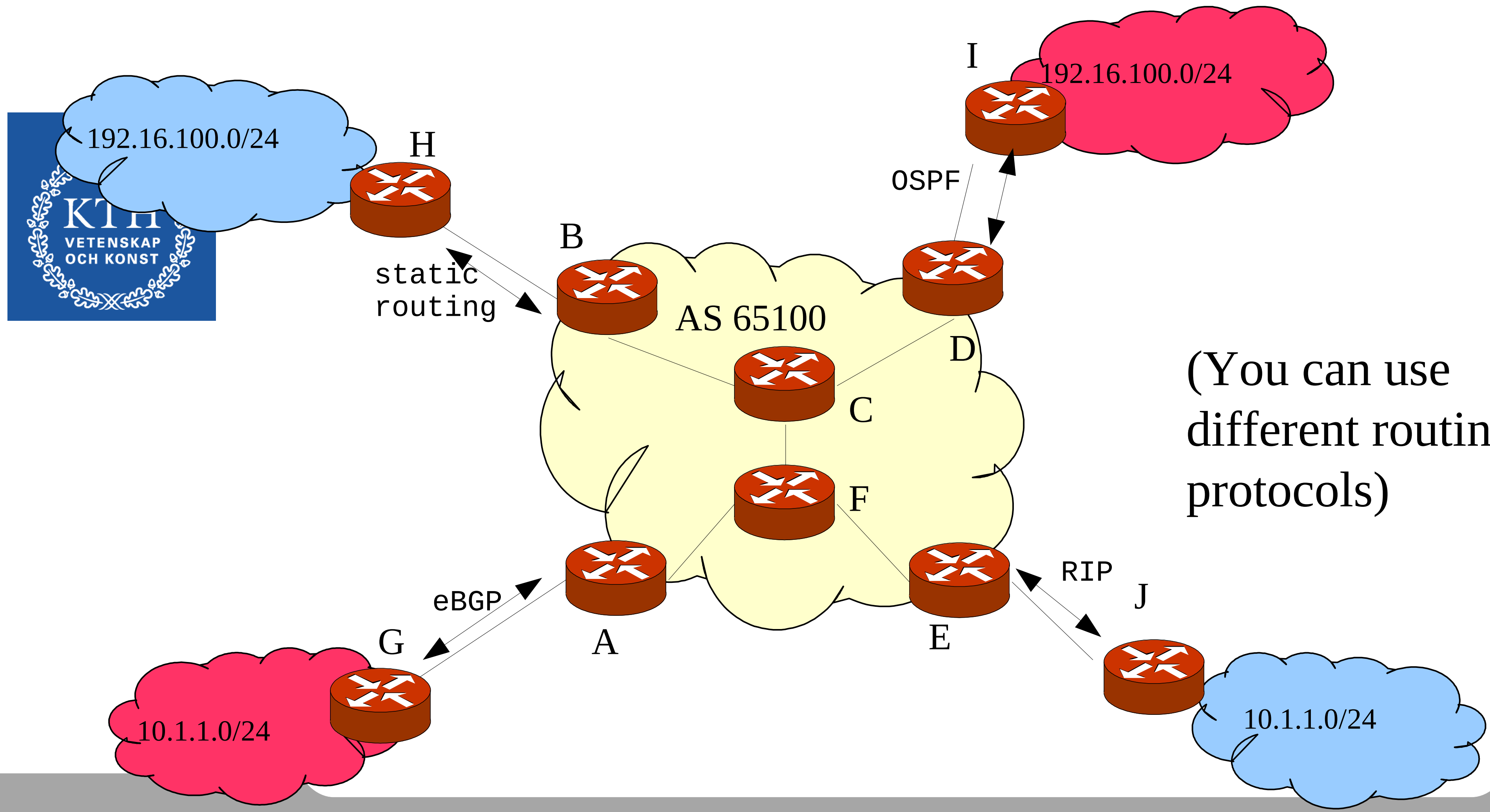
L3VPN example



CE to PE routing

- The local PE learns routes from the local customer CE
- Static routing, eBGP, RIP, or some other IGP
  - Customer should be able to decide
  - Often the customer wants a separate routing protocol for the CE-PE peering (eg. so OSPF link-state is not propagated to the provider)
- The PE router takes the routes and propagates them over the provider network to the remote PE:s
- The remote PE:s announce the client routes to matching remote CE sites
- The remote CE sites can then access the local CE

CE to PE routing (example)



(You can use different routing protocols)

The Route Distinguisher

Overlapping addresses: Route Distinguisher

- How does a provider keep different client prefixes unique?
  - Eg: Red and blue VPN both have 10.1.1.0/24
- A new address class is used, where a unique prefix is prepended to the VPN route
  - This unique prefix is called a *route distinguisher* (RD)
- A new (L3VPN) route is written:
  - *<route distinguisher>*::*<IPv4addr>/<prefixlen>*

8 bytes	4 bytes
Route Distinguisher	IPv4 address/site

Route Distinguisher format

I	T	Type[	Subtype]	Data

1 byte	1 byte	6-7 bytes

- The route distinguisher has the same format as the BGP extended community which is 8 bytes.
- Two variants Type 0 and Type 1
- Type 0
  - –Can be better to identify VPNs, or if many AS
- Type 1 used in the lab
  - –Easier to see the origin of the routes

8 bytes	4 bytes
Route Distinguisher	IPv4 address

Type 0:

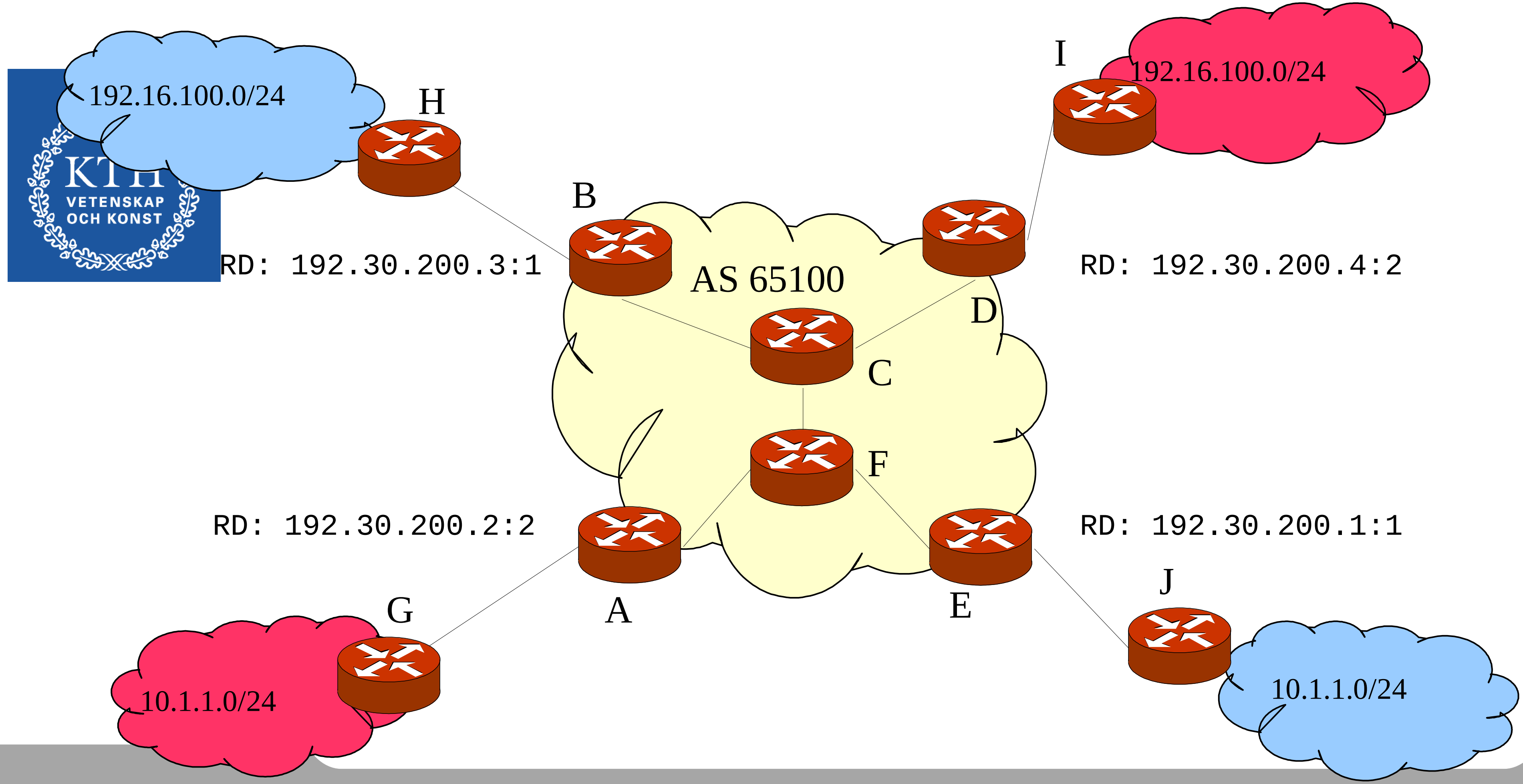
2 bytes	2 bytes	4 bytes	
Type/Subtype	AS#	Number	IPv4 address

Type 1:

2 bytes	4 bytes	2 bytes	
Type/Subtype	IP#	Number	IPv4 address

Route distinguisher type 1

- Example
  - 192.30.200.3:1::192.16.100.0/24 announced by B
- You can see where the routes come from
- And you can see which VPN they belong to (1=blue, 2=red)



Routing table example

Example: Routing table in a PE router (prefix + nexthop)

VPN-IPv4 address family (bgp.l3vpn in JunOS)

192.30.200.3:1::192.168.100.0/24	B
192.30.200.2:2::10.1.1.0/24	A
192.30.200.1:1::10.1.1.0/24	E
192.30.200.4:2::192.168.100.0/24	D

IPv4 address family:

192.30.200.3

192.30.200.2

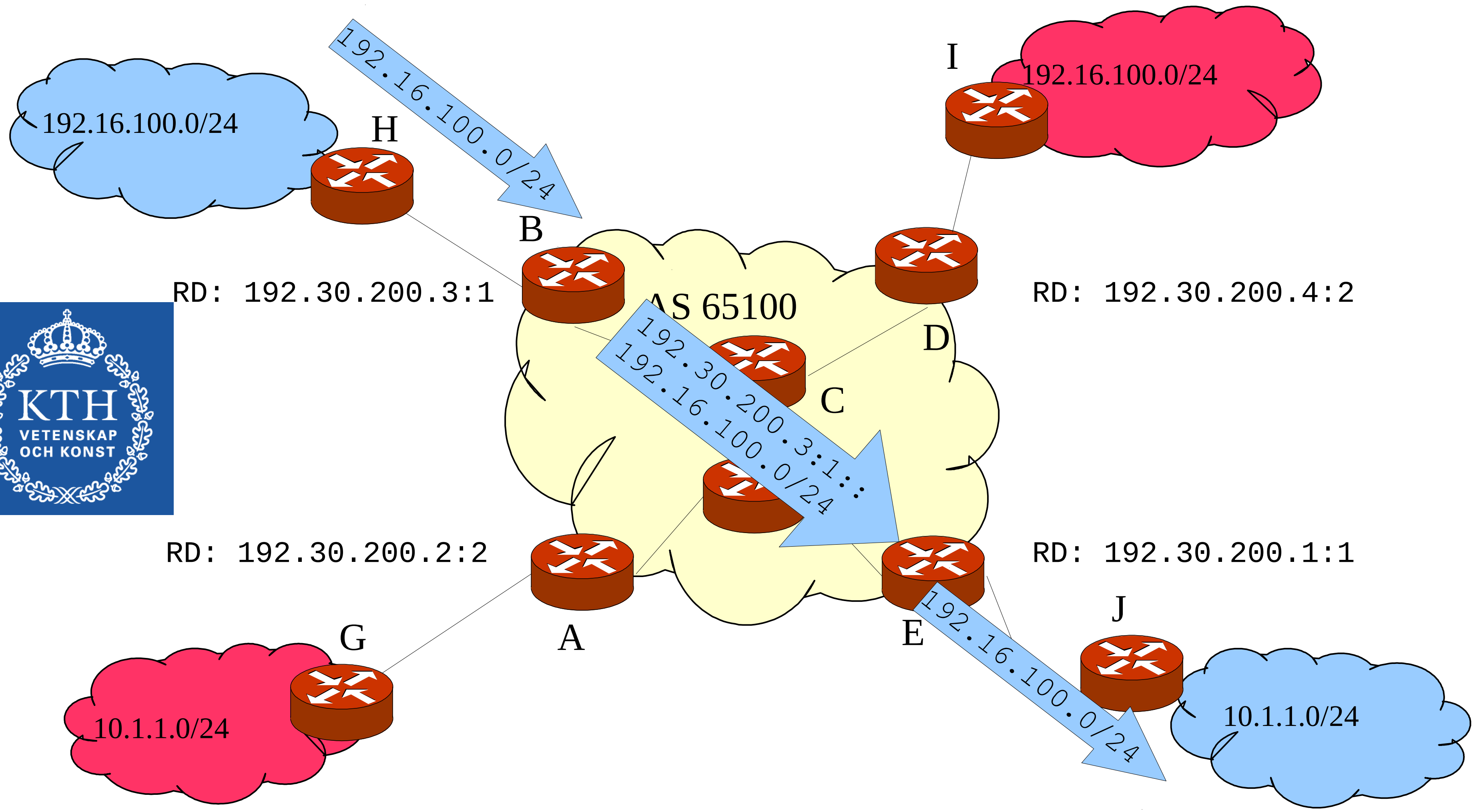
192.30.200.1

192.30.200.4

Operation

- A CE announces a prefix to a PE
  - Eg 192.168.100.0/24 to B by H
- The PE prepends the route distinguisher and announces it to the other PE:s
  - Eg 192.30.200.3:1::192.168.100.0/24
- The PEs receives the route, strips the route distinguisher and announces it to the local matching CE
  - Eg 192.168.100.0/24 to J by E
- The CE network can reach 192.168.100.0/24
- See figure on next slide

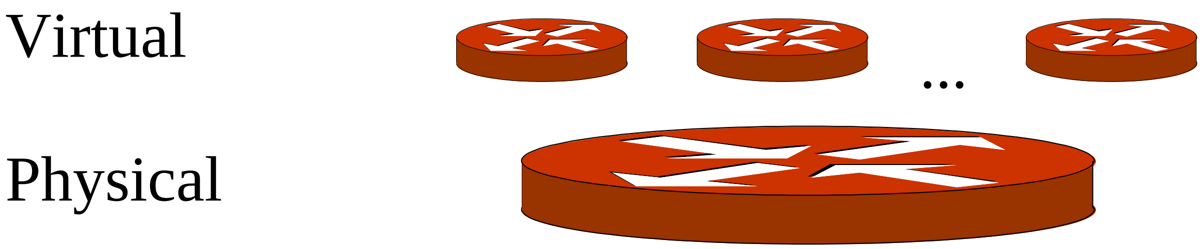
Operation: announcing prefixes



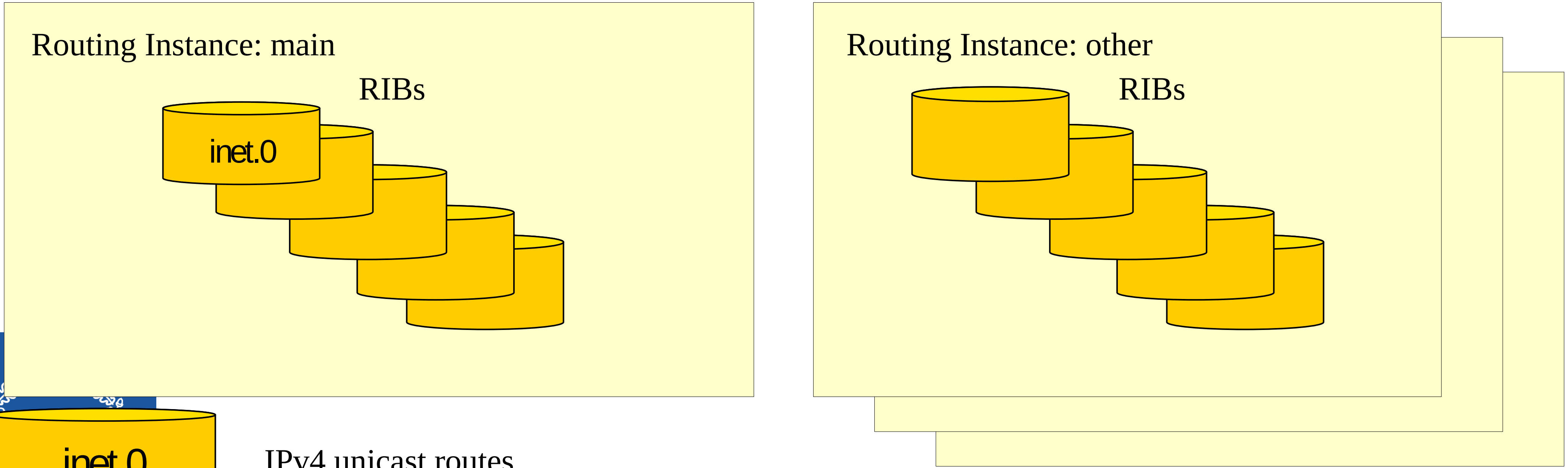
Virtual Routing and Forwarding

Virtual Routing and Forwarding - VRF

- A virtual router is a subset of a physical router.
- A virtual router has its own routing processes, routing tables, forwarding tables and its own interfaces,
- Typically interfaces of virtual routers are virtual (eg VLANs)
- The virtual routers are partitioned into several *disjoint* virtual routers.



Routing instances in JunOS



inet.0	IPv4 unicast routes
inet6.0	IPv6 unicast routes
inet.1	IPv4 multicast forwarding cache
inet.2	IPv4 multicast RPF table
inet.3	IPv4 routes learnt from MPLS-TE path exploration
bgp.l3vpn	VPN-IPv4 routes
mpls.0	MPLS label-switch table

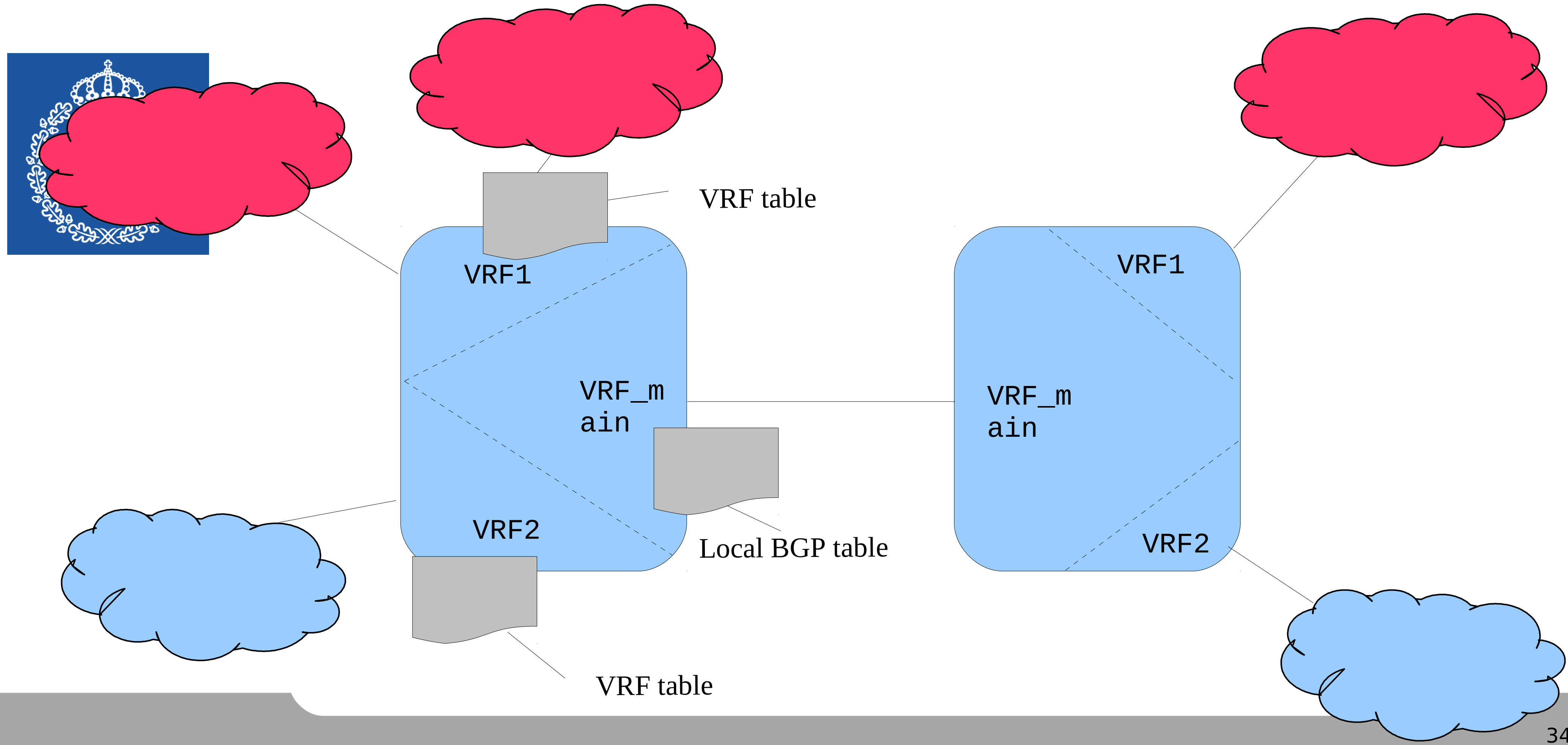
Example:

```
main.inet.0
 __juniper_private1__.inet.0
```

Logical routers, VPNs, virtual routers, etc, use routing instances.

VRF in a PE

Example: A router with two customers instances: VRF1 and VRF2.



Using MPLS and RSVP

Establish LSP:s between border routers

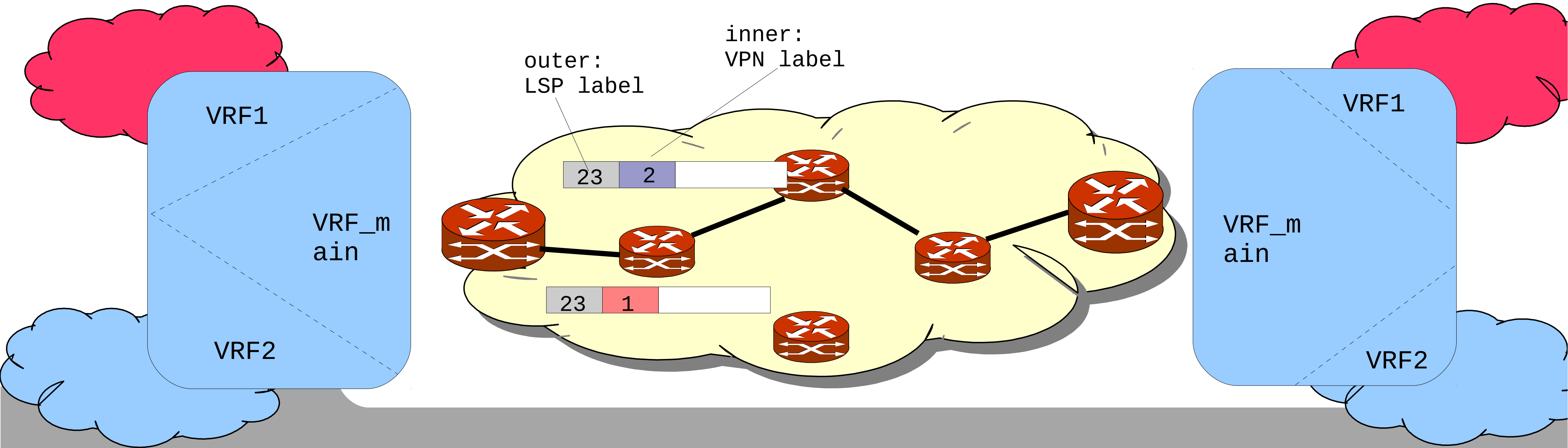
Use double stacking:

- outer tag: LSP PE<-->PE
- inner tag: VPN label

Internal nodes (P-nodes) are only aware of outer tags (PE to PE)

With RSVP you set up the *outer* tag

- and can also traffic engineer the LSP:s

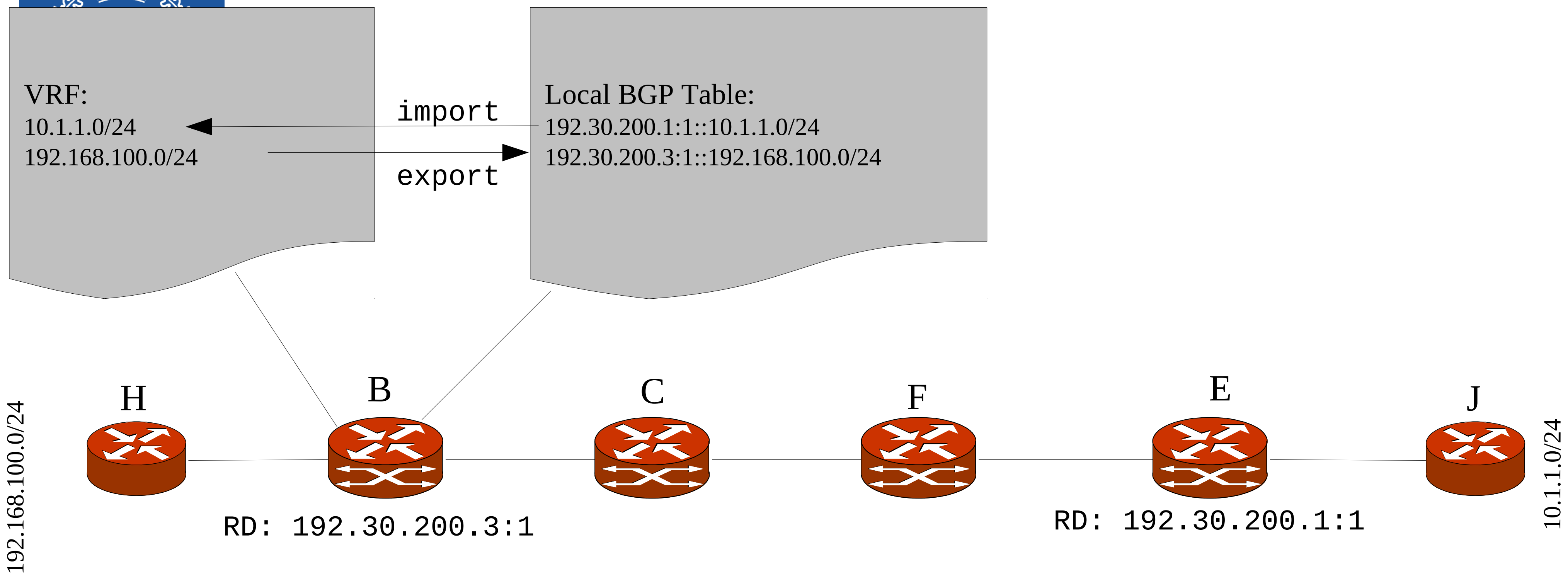


Route Target

VRF Importing and exporting

You export and import routes between the VRF and the global routing domain by adding or stripping the route-distinguisher using export and import rules.

The rules are expressed using route targets

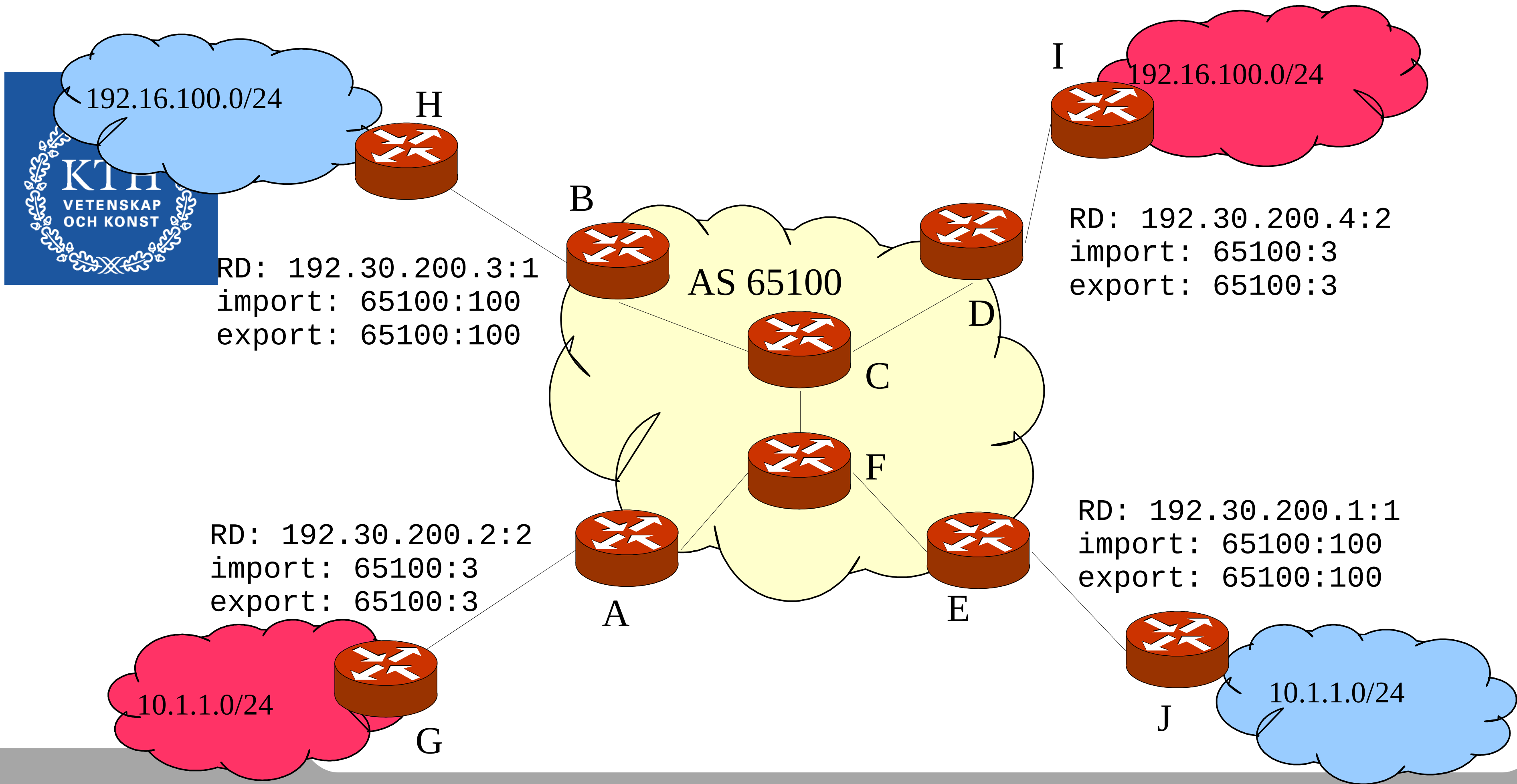


Route target

- The purpose of the *route target* (RT) extended community is to tag the VPN-IPv4 routes with VPN information
- Rules are then based on route targets
- The route target has the same format as the route-distinguisher
  - AS#:number (type 0) – Used in lab
  - IP#:number (type 1)
- The route target is used to *color* the routes
  - In our example red and blue
- Example:
  - RT 65100:100 - blue VPN
  - RT 65100:3 - red VPN
- Typically, every VRF has a set of import and export rules
- Every export rule corresponds to tagging the announced VPN-IPv4 route with a route target attribute
- Every import rule corresponds to matching targets with incoming route target attributes

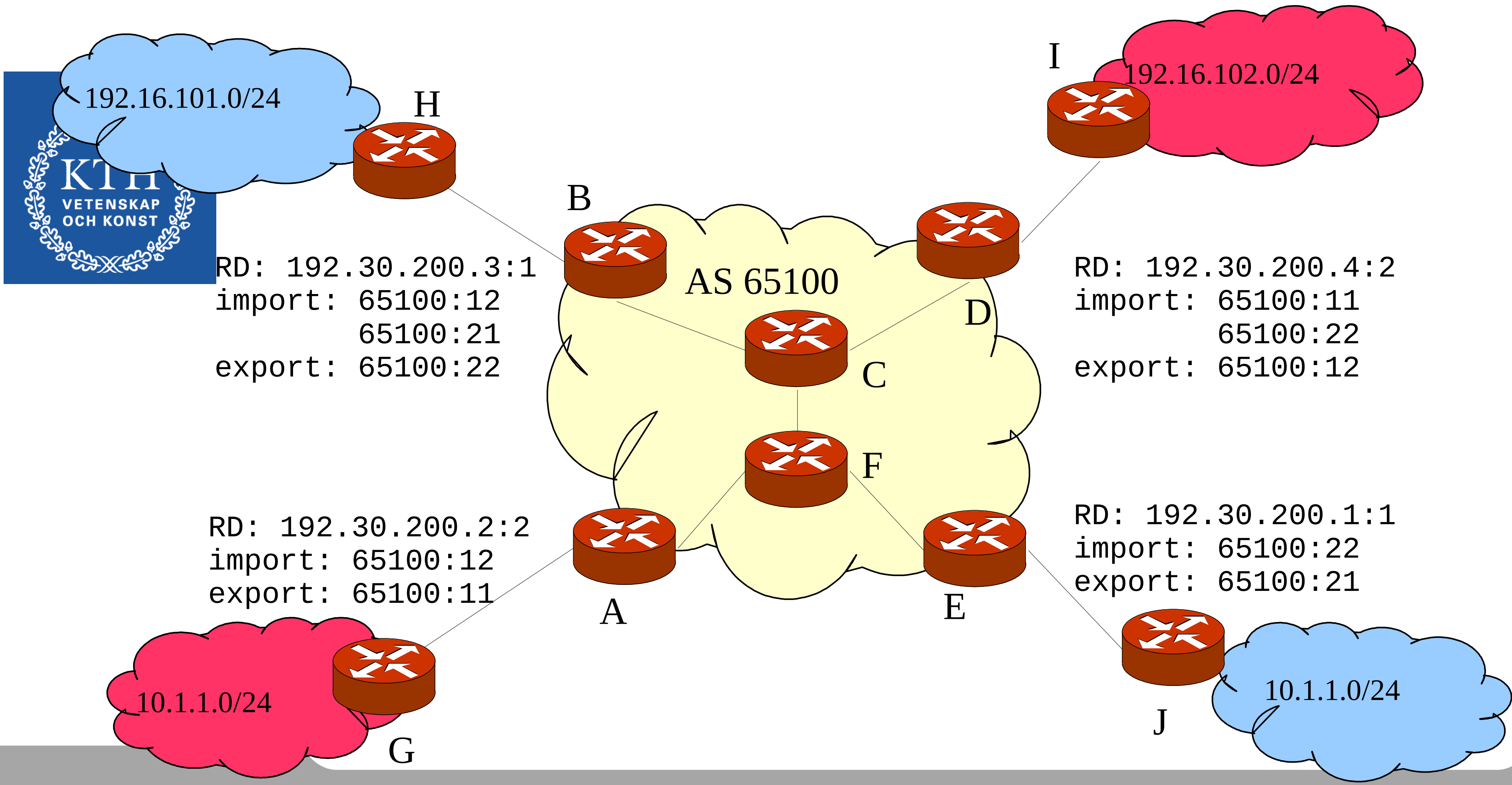
Route target example: full mesh

- Tag the routes when *exporting* to BGP
- *Import* routes matching the target community
- Full mesh is default policy and can be accomplished in JunOS simply with
  - set vrf-target target:<route target>



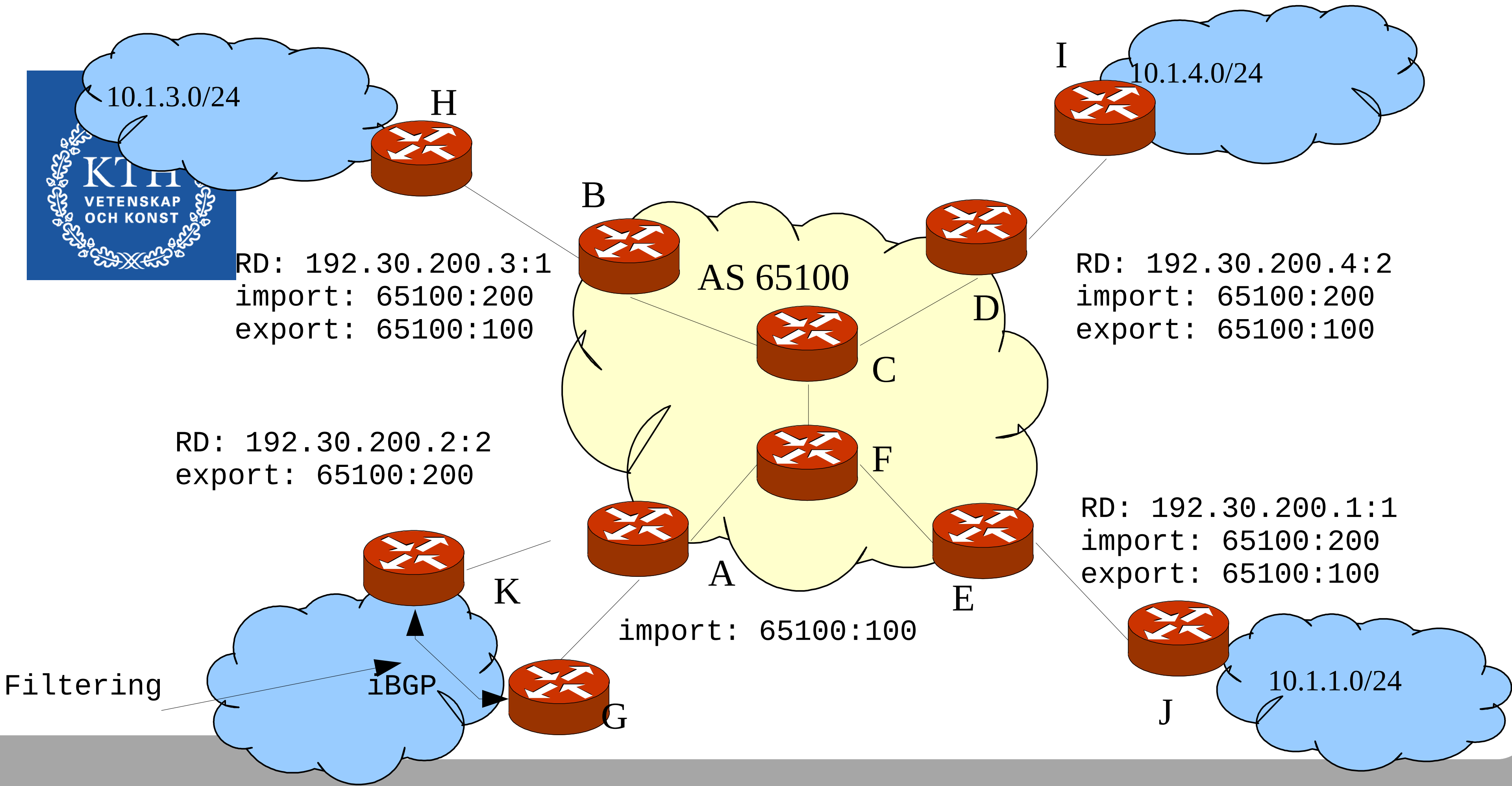
Extranet

- The Extranet is defined between the upper two customer sites
  - Note that the prefixes have been changed to be unique
  - And the route targets are unique per PE



Hub-and-spoke VPN

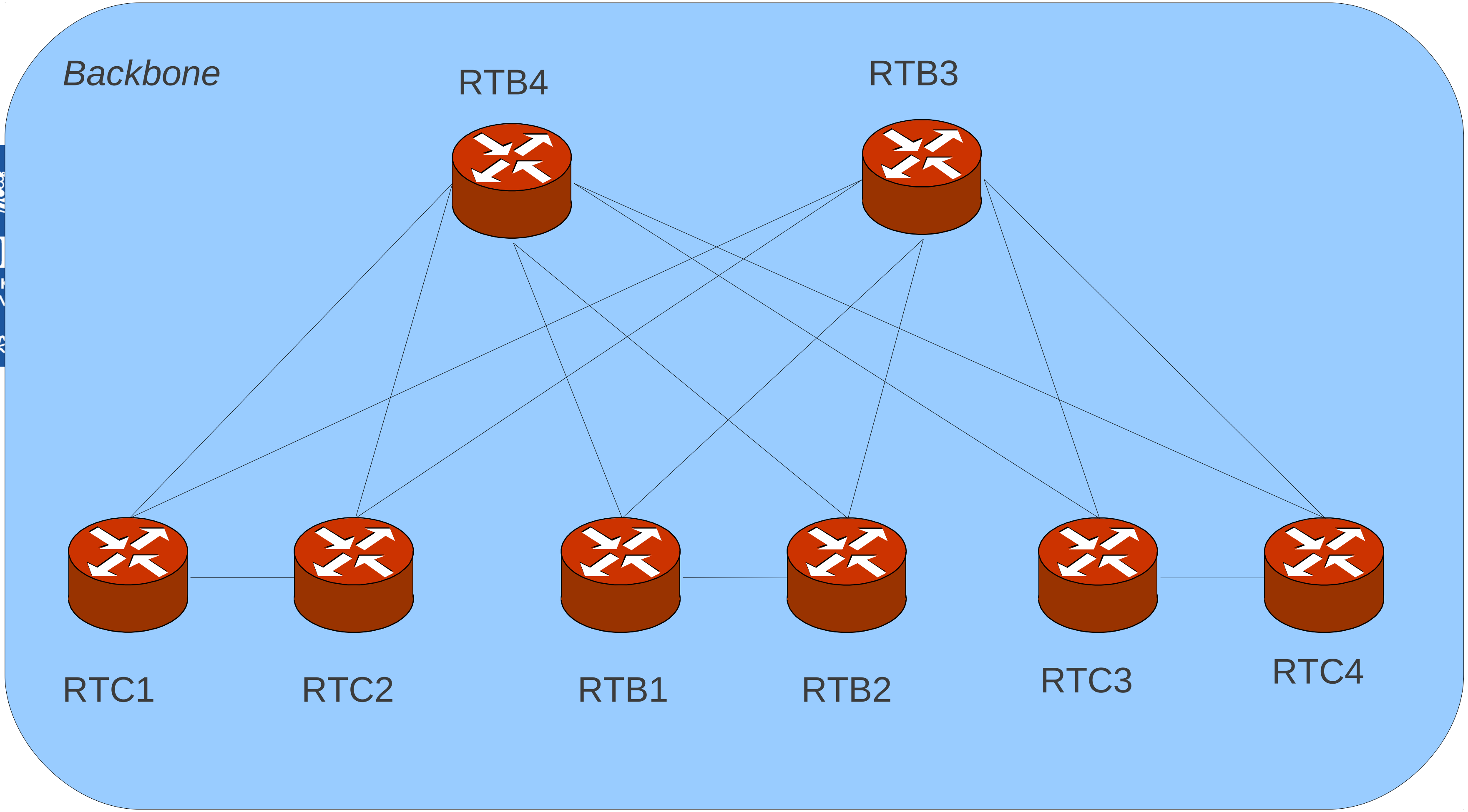
- All traffic passes via a HUB
- Filtering / security purposes
- Note the two peerings at A



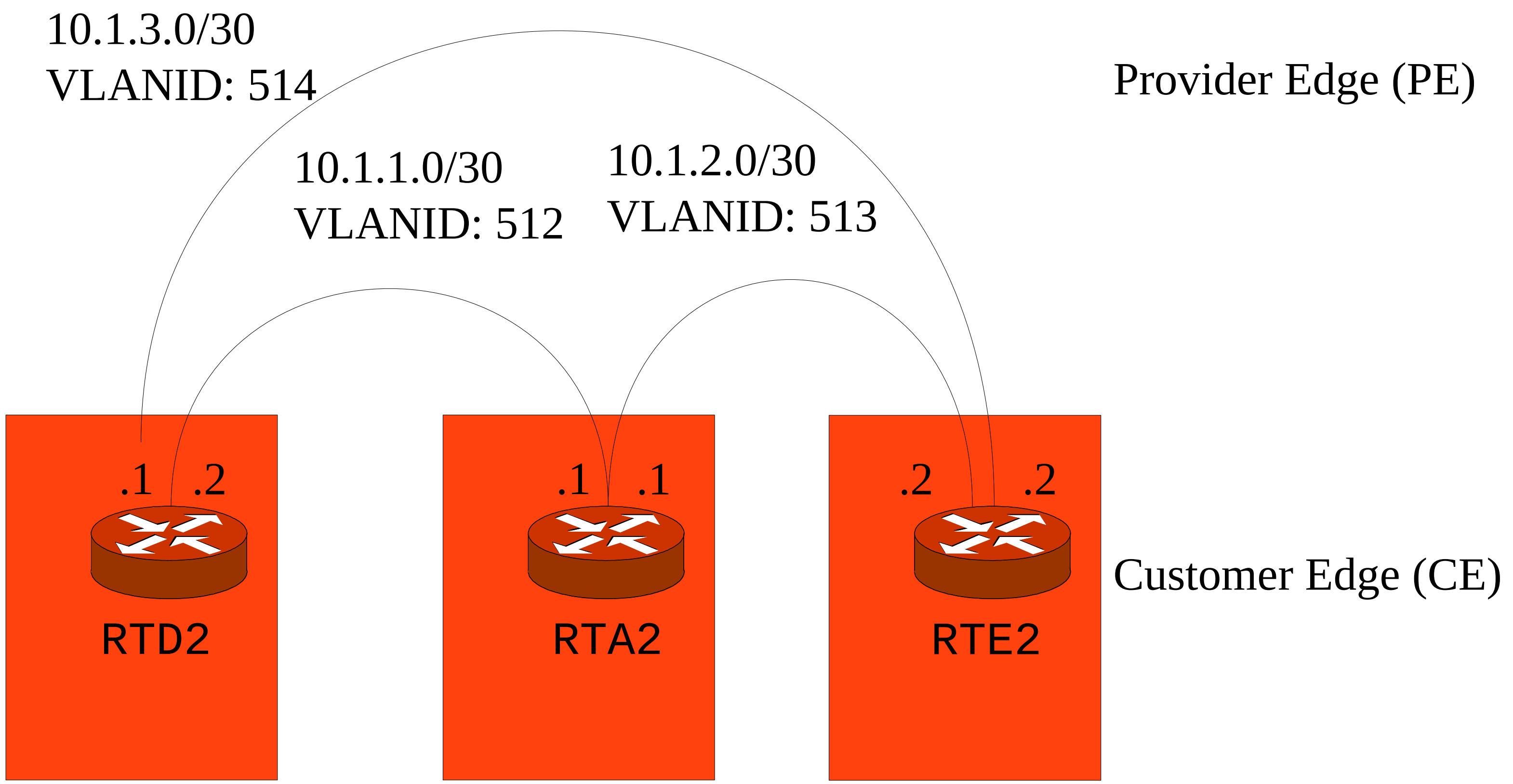
L2VPN and L3VPN lab

1) Build an MPLS backbone
2) Configure L2VPN
3) Configure L3VPN

MPLS backbone



L2VPN setup



10.1.3.0/30
VLANID: 514
10.1.1.0/30
VLANID: 512
10.1.2.0/30
VLANID: 513
Provider Edge (PE)
.1
.2
.1
.1
.2
.2
RTD2
RTA2
RTE2
Customer Edge (CE)

L2VPN configuration example

```
routing-instances {
    L2VPN {
        description "experimental L2VPN";
        instance-type l2vpn;
        interface fe-0/0/0.512;
        route-distinguisher 192.168.4.2:10;
        vrf-target target:65000:10;
        protocols {
            l2vpn {
                encapsulation-type ethernet-vlan;
                no-control-word;
                site RED {
                    site-identifier 1;
                    interface fe-0/0/0.512 {
                        remote-site-id 2;
                    }
                }
            }
        }
    }
}
```

L2VPN Junos show commands

- show l2vpn connections [extensive]
- show route protocol l2vpn
- show route protocol bgp
- show mpls lsp
- show bgp summary
- show route

```
193.10.255.5:10:1:1/96
        *[L2VPN/170/-101] 02:45:38, metric2 1
           Indirect
 193.10.255.6:10:2:1/96
        *[BGP/170] 01:36:41, localpref 100, from 193.10.255.6
           AS path: I
         > via so-0/1/0.0, label-switched-path btoc
193.10.255.13:10:3:1/96
        *[BGP/170] 01:38:11, localpref 100, from 193.10.255.13
           AS path: I
         > via so-0/1/0.0, label-switched-path btod
```

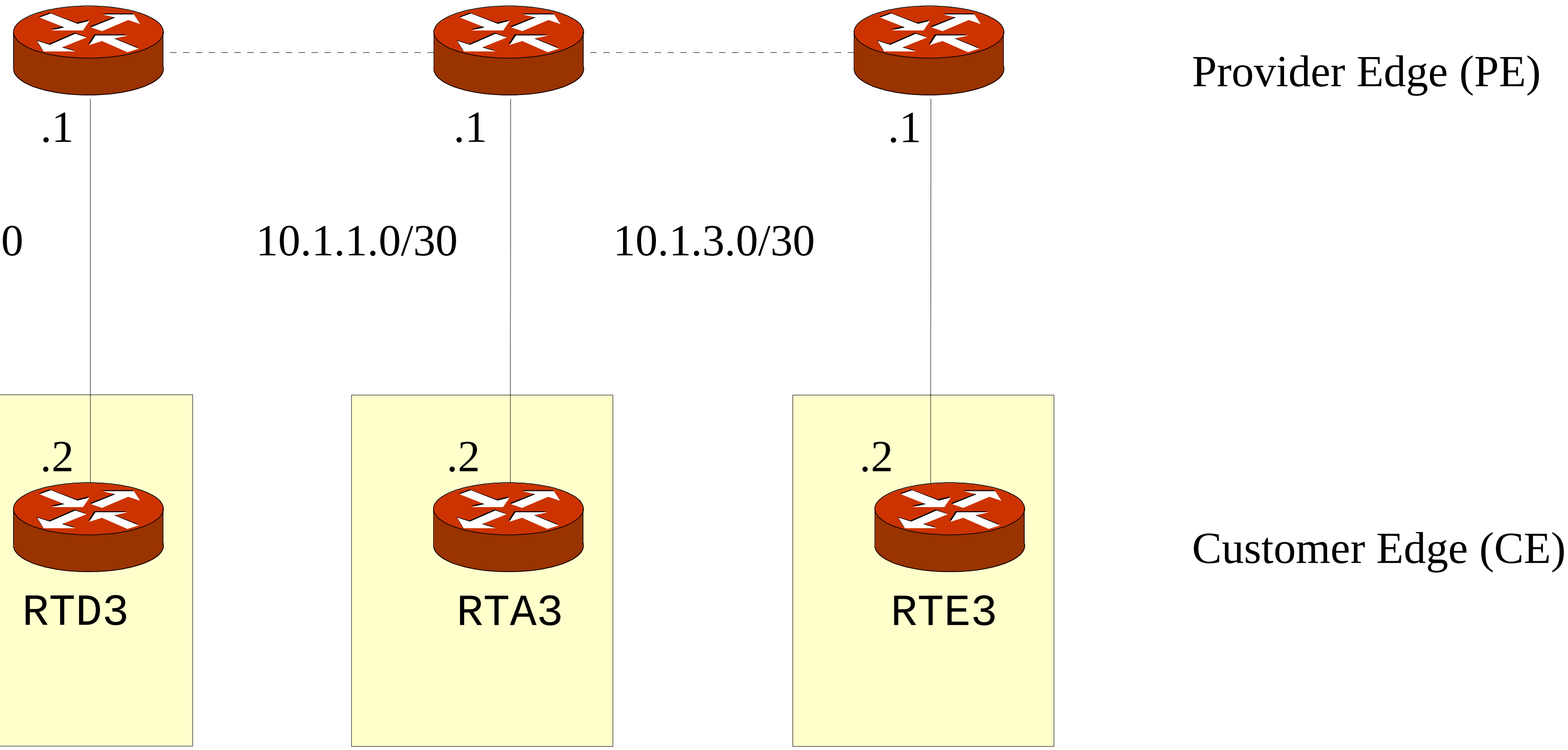
Configuring L2VPN

- Setup the backbone: ISIS, MPLS, RSVP, IBGP
  - Enable 'l2vpn signaling' as bgp protocol family
- Setup CE-PE circuits (VLANs)
  - Use Ethernet interface with units > 0
  - Use VIDs>=512 (or use 'flexible' services)
  - Set RFC1918 addresses on the VLANs
- Setup an l2vpn routing instance:
- Set route distinguisher
  - <PE loopback>:<vpnid>
- Setup sites and setup LSPs by connecting remote sites
  - Bind vlans to remote sites using vlanids
- Setup encapsulation
  - 'ethernet-vlan'
- Set no-control-word (used for other link-layers)
- Setup vpn import/export rules
  - use vrf-target
- L2VPN routes:
  - <RD>:<site>:1/96
  - Example: `193.10.255.5:10:3:1/96`

VPLS configuration example

```
routing-instances {
    VPLS {
        instance-type vpls;
        interface ge-3/0/1.512;
        route-distinguisher 192.168.4.2:10;
        vrf-target target:65000:10;
        protocols {
            vpls {
                no-tunnel-services;
                site RTA {
                    site-identifier 1;
                }
            }
        }
    }
}
```

L3VPN setup


.1
.1
.1
Provider Edge (PE)
10.1.2.0/30
10.1.1.0/30
10.1.3.0/30
.2
.2
.2
RTD3
RTA3
RTE3
Customer Edge (CE)

L3VPN configuration example

```
protocols {
   bgp {
      local-address 192.30.200.3;
      group internal {
        type internal;
        family inet-vpn unicast;
        neighbor 192.30.200.1;
     }
  }
}
routing-instances {
    VRF1_BLUE {
        instance-type vrf;
        interface fe-0/0/0.0;
        route-distinguisher 192.30.200.3:1;
        vrf-target target:65100:100;
        vrf-table-label;
        protocols {
            bgp {
                group siteB {
                    type external;
                    peer-as 1;
                    neighbor 192.16.100.1; # H
                }
            }
        }
    }
```

LAB overview