Networks: Interfaces, Plural

i.g.batten@bham.ac.uk

20430737

Link-Agg

- If a switch or host supports it, you can put interfaces into a Link Aggregation group, controlled with LACP
- Appears to operating system as one “**logical**” interface, one IP address, one entry in routing table, etc.
- Modern solution for performance and availability
  - Failure detection much faster and simpler, as done at the link level
  - Fancy equipment lets you link-agg to multiple switches, or at least multiple independent cards in same switch

Benefit of LinkAgg

- Only one IP address
- Fast failover
- Efficient multiplexing and load spreading (header hashes, sometimes done by hardware)
    - This is better than round-robin as it maintains ordering in individual flows, and plays nicely with TCP offload.

Problem of LinkAgg

- Only goes as far as your nearest switch
- Doesn’t provide for (say) spreading over multiple internet connections

VLANs

- Way to get multiple networks delivered over single cable
- Ethernet frames have optional extra “tag” (0–1023 minimum, 0–4095 more common) identifying which network it is
- Untagged frames are assigned to a per-port default network, which in turn has a per-switch default of either 0 or 1 (usually).
- OS sees them as two separate networks

VLAN host-side

Untagged

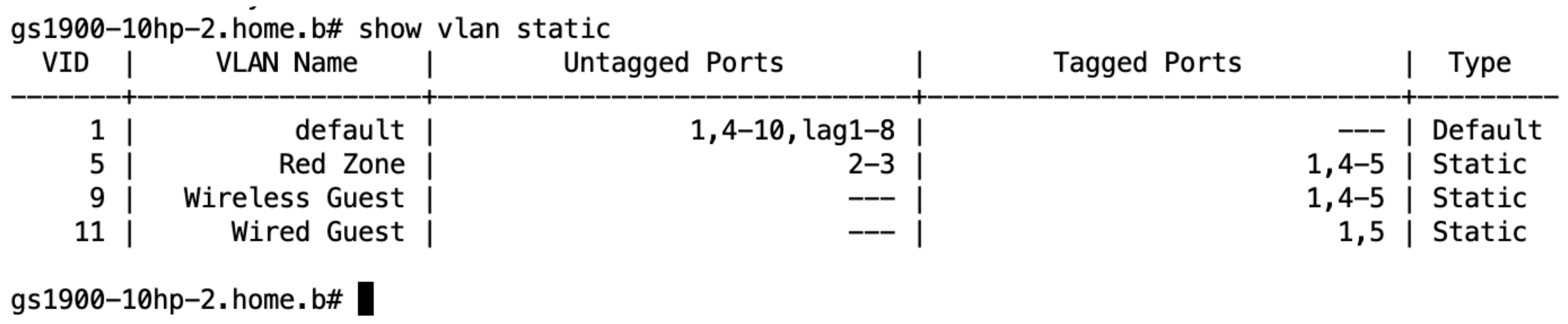
```
igb@pi-one:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr b8:27:eb:8c:0c:34
          inet addr:10.92.213.231  Bcast:10.92.213.255  Mask:255.255.255.0
          inet6 addr: 2001:8b0:129f:a90f:3141:5926:5359:1/64 Scope:Global
          inet6 addr: fe80::ba27:ebff:fe8c:c34/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:44938697 errors:0 dropped:0 overruns:0 frame:0
          TX packets:33296383 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:49825123 (47.5 MiB)  TX bytes:441607425 (421.1 MiB)

eth0.5    Link encap:Ethernet  HWaddr b8:27:eb:8c:0c:34
          inet addr:81.187.150.211  Bcast:81.187.150.223  Mask:255.255.255.240
          inet6 addr: 2001:8b0:129f:a90e:3141:5926:5359:1/64 Scope:Global
          inet6 addr: fe80::ba27:ebff:fe8c:c34/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:25759488 errors:0 dropped:0 overruns:0 frame:0
          TX packets:21680755 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2927238990 (2.7 GiB)  TX bytes:2669983251 (2.4 GiB)
```

VLAN tag = 5

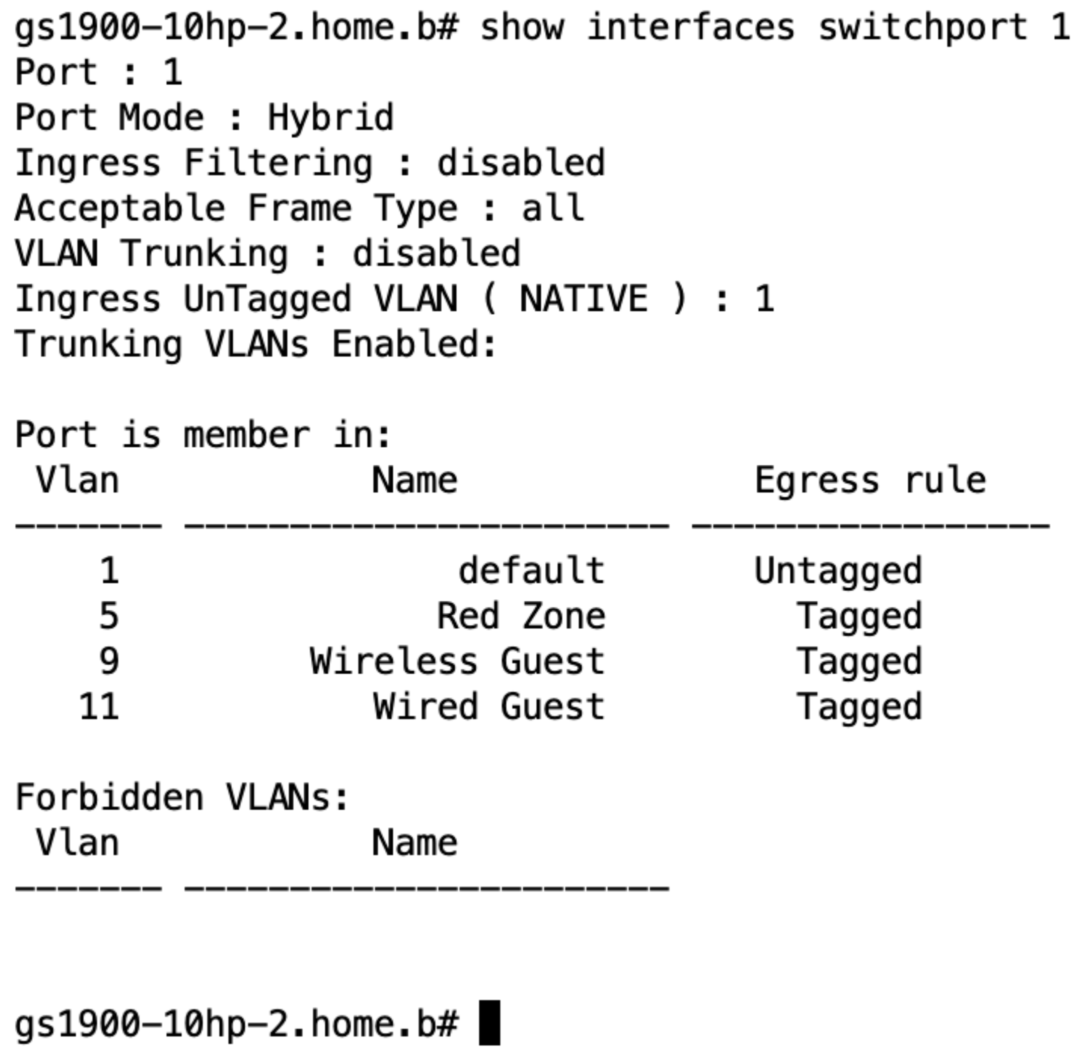
VLAN on switches

- Each VLAN has a tag
- Each port of a switch can output one or more VLANs, one of them optionally untagged (the PVID VLAN).
- Ports can be configured to accept combinations of tagged and untagged frames, the untagged frames if permitted, assumed to be in the PVID, tagged frames possibly filtered against an acceptable list.
- Flexible, and a good way to confuse yourself
- Ports outputting untagged frames sometimes called “access”, tagged frames “trunk”
  - (Warning: “ethernet trunk” or “ethernet trunking” can mean almost anything, from “send all packets, even VLANs that are not configured locally” to “LACP”, and when “trunk” is used of VLANs over LACP it gets even more confusing. Avoid the phrase, and specify what you mean in detail).



```
gs1900-10hp-2.home.b# show vlan static
  VID  |     VLAN Name     |         Untagged Ports         |         Tagged Ports           |  Type
-------+-------------------+--------------------------------+--------------------------------+---------
     1 |           default |                1,4-10,lag1-8 |                            --- | Default
     5 |          Red Zone |                          2-3 |                          1,4-5 | Static
     9 |    Wireless Guest |                          --- |                          1,4-5 | Static
    11 |       Wired Guest |                          --- |                            1,5 | Static

gs1900-10hp-2.home.b# 
```

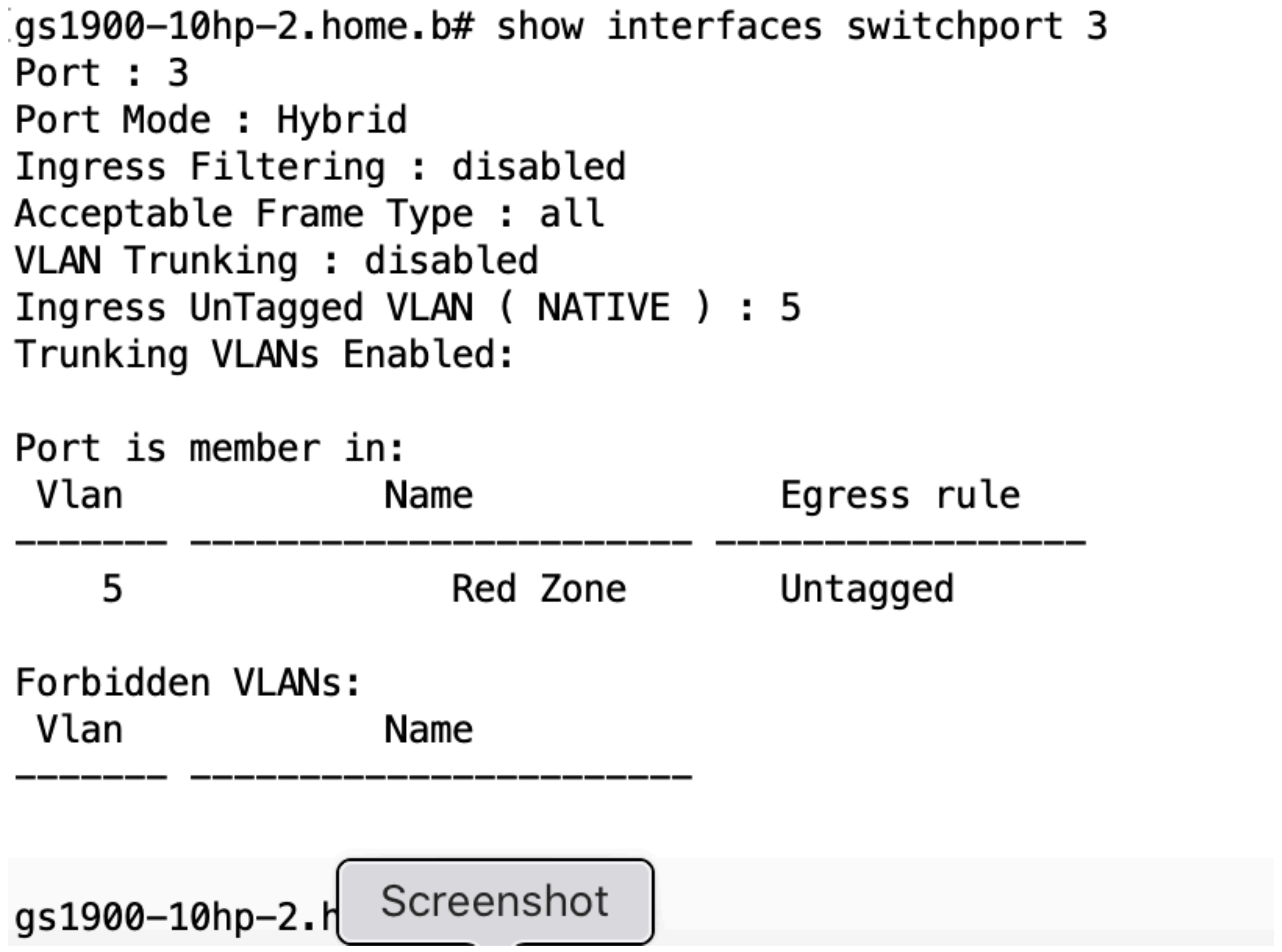


```
gs1900-10hp-2.home.b# show interfaces switchport 1
Port : 1
Port Mode : Hybrid
Ingress Filtering : disabled
Acceptable Frame Type : all
VLAN Trunking : disabled
Ingress UnTagged VLAN ( NATIVE ) : 1
Trunking VLANs Enabled:

Port is member in:
 Vlan            Name              Egress rule
------- ----------------------- -----------------
    1                  default      Untagged
    5                 Red Zone        Tagged
    9           Wireless Guest        Tagged
   11              Wired Guest        Tagged

Forbidden VLANs:
 Vlan            Name
------- -----------------------


gs1900-10hp-2.home.b# 
```



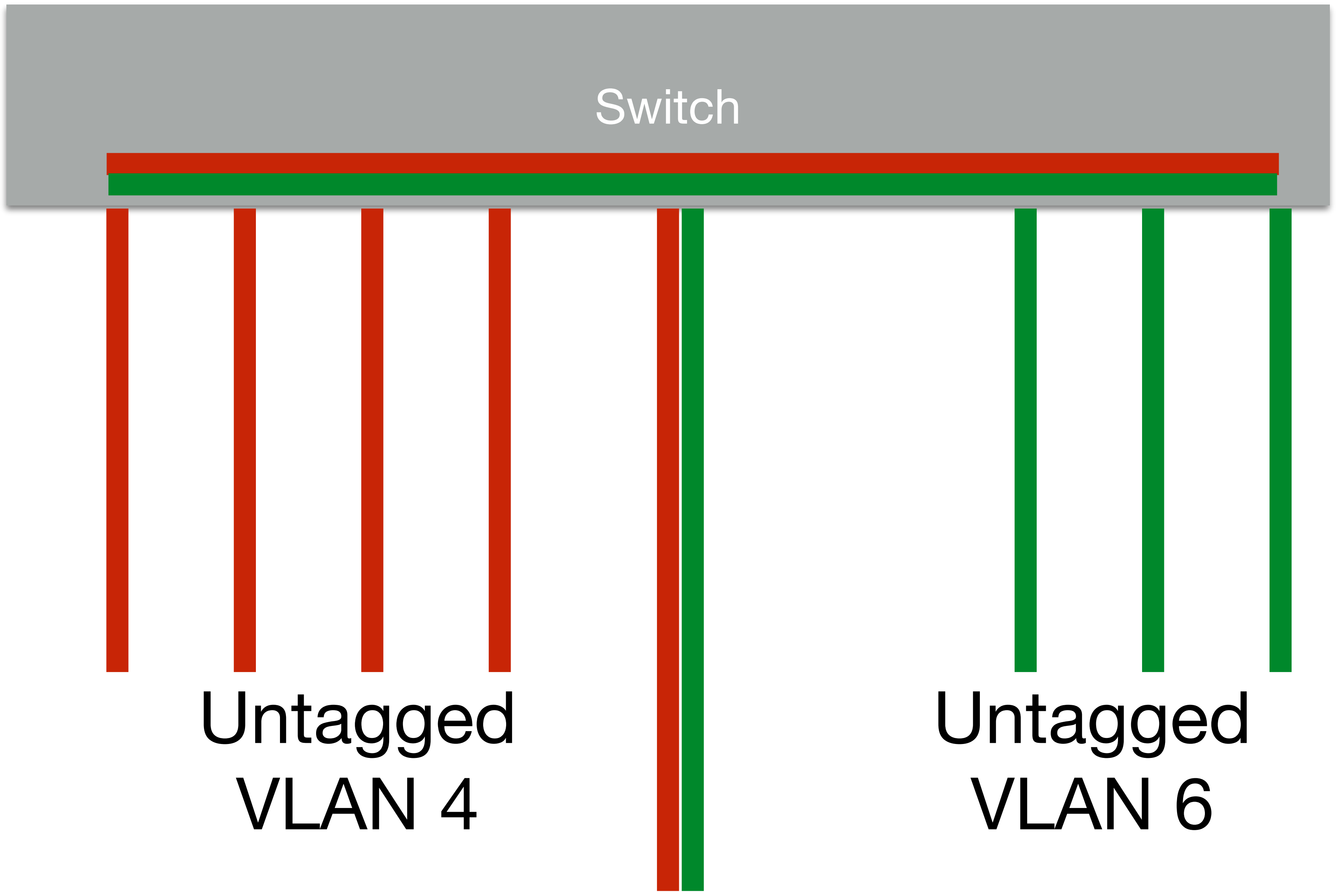
```
gs1900-10hp-2.home.b# show interfaces switchport 3
Port : 3
Port Mode : Hybrid
Ingress Filtering : disabled
Acceptable Frame Type : all
VLAN Trunking : disabled
Ingress UnTagged VLAN ( NATIVE ) : 5
Trunking VLANs Enabled:

Port is member in:
 Vlan            Name              Egress rule
------- ----------------------- -----------------
    5                 Red Zone      Untagged

Forbidden VLANs:
 Vlan            Name
------- -----------------------


gs1900-10hp-2.h
```

Logically divide switches



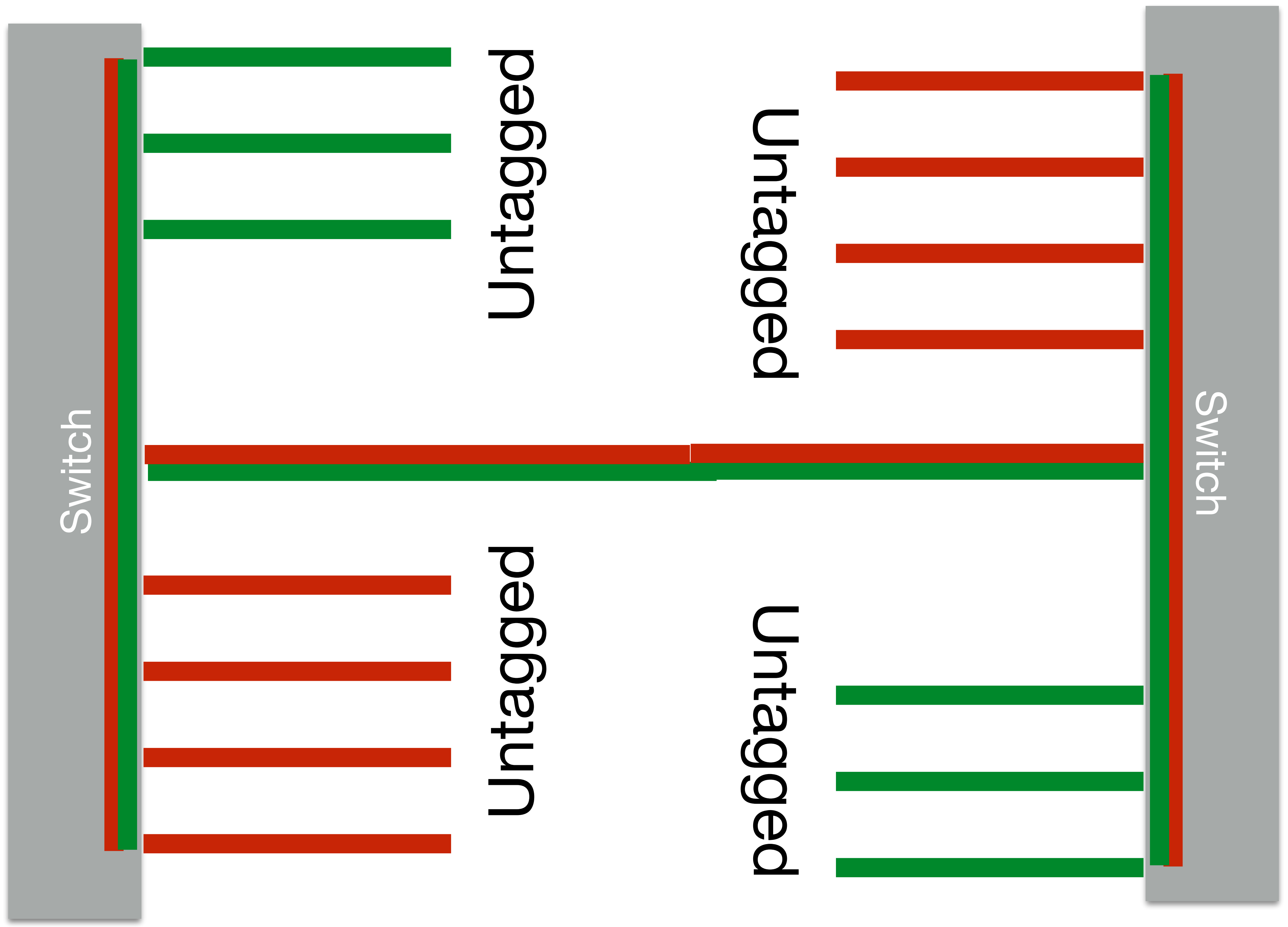
Untagged VLAN 4

Untagged VLAN 6

Tagged VLAN 4 and VLAN 6

“Router on a stick”

Linking Switches



Untagged ports

- AKA “access ports”
- Devices attached to them don’t have to understand VLANs
- Packets from one VLAN on the switch are transmitted with the tag stripped
- Packets arriving without a tag have a pre-defined tag added (PVID)
- Can also have tagged packets mixed in, to/from VLAN-aware devices
  - Freshly installing a machine to use a tagged interface is sometimes tricky, so attractive to use untagged for “primary” interface while still having access to other VLANs, cf. my home example

Tagged Ports

- All packets are transmitted with a tag, and only tagged packets are expected to be received
- Common for inter-switch links
- Tags can be filtered for security reasons (not very good security)
- Tags can also be re-written, although that way madness lies
    - Ditto “stacked tags” or “Q-in-Q”, which is best left to transmission people who need it.

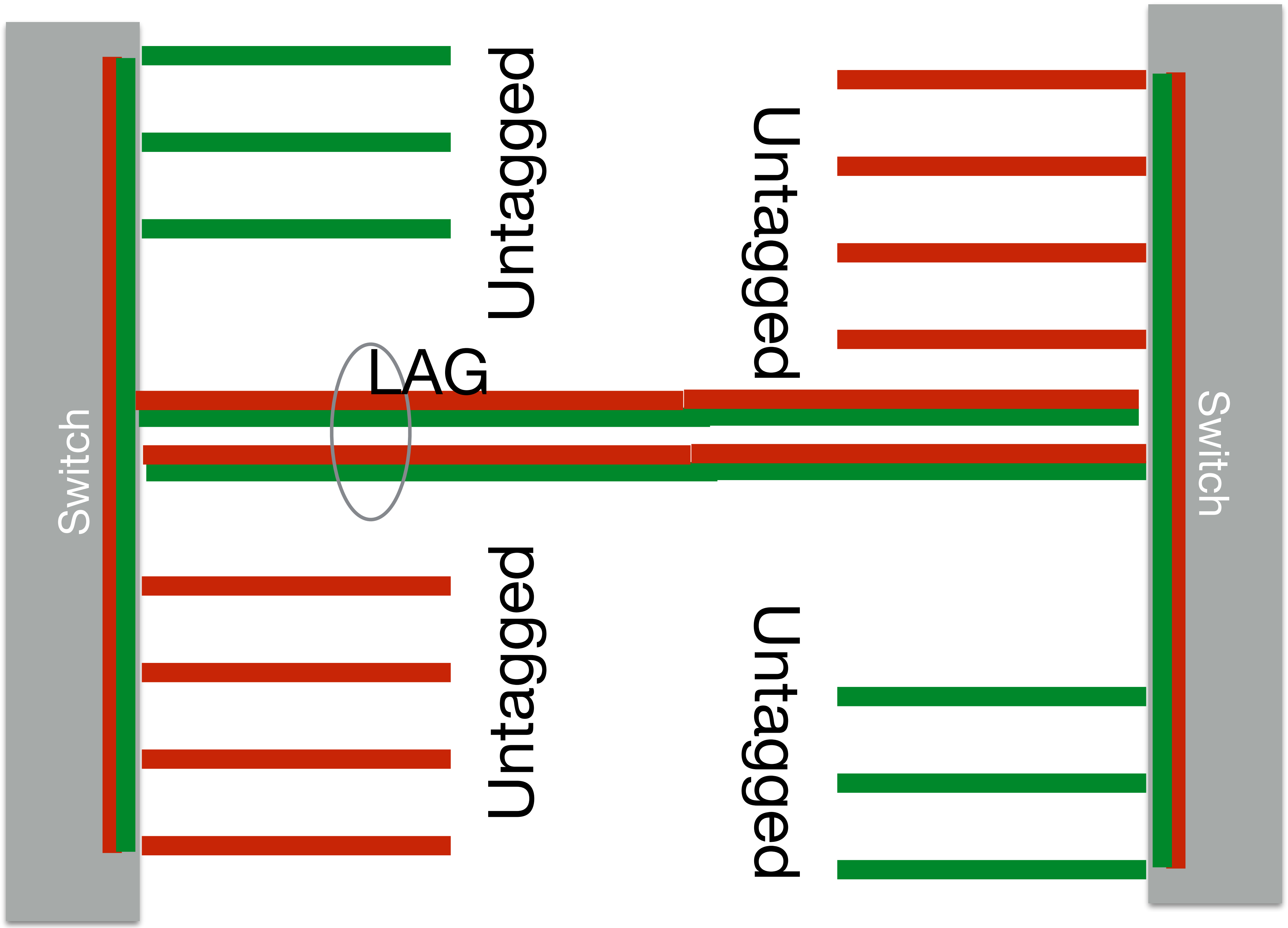
VLAN security

- Quite weak
- Akin to writing “private” on an envelope but not sticking the flap down
- Splitting a switch with untagged ports might be OK, as might inter-switch links, but in general using VLANs for anything involving hard separation will fail assurance
  - Anyone clever enough to use VLANs for this is clever enough to not use VLANs for this
  - All security fails in face of attacker who can reconfigure switch, and most switches aren’t very secure

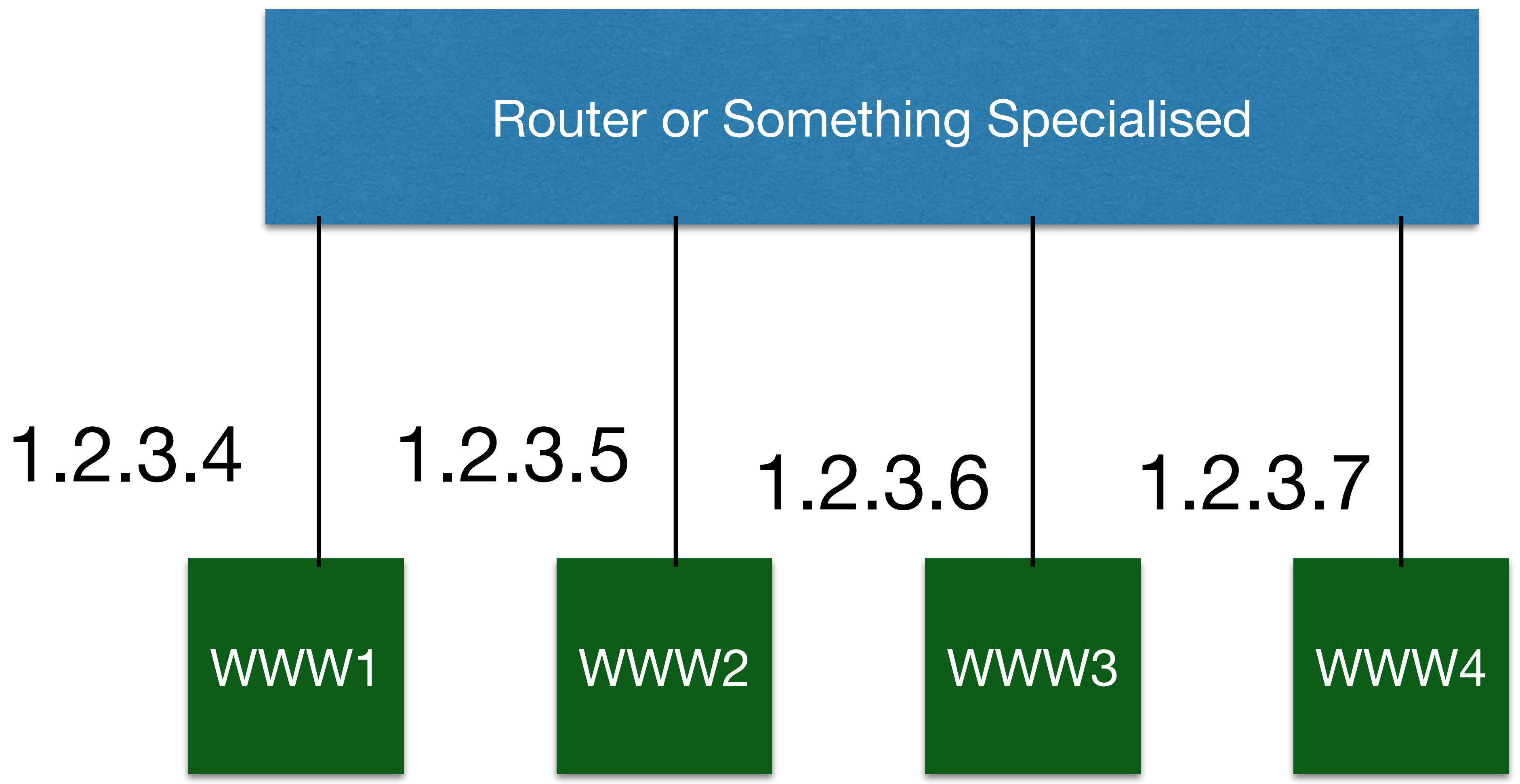
Link-Agg + VLANs

- You can deliver two or more networks over two or more cables, with full load balancing and failover
- Aggregate using Link Aggregation
- Then apply VLAN tagging for the separate networks running over the aggregated link
- Not enough people do this: it is very, very effective
- **The** design pattern for 2020s resilience

Linking Switches



Load Balancing



www.dom.ain -> four addresses

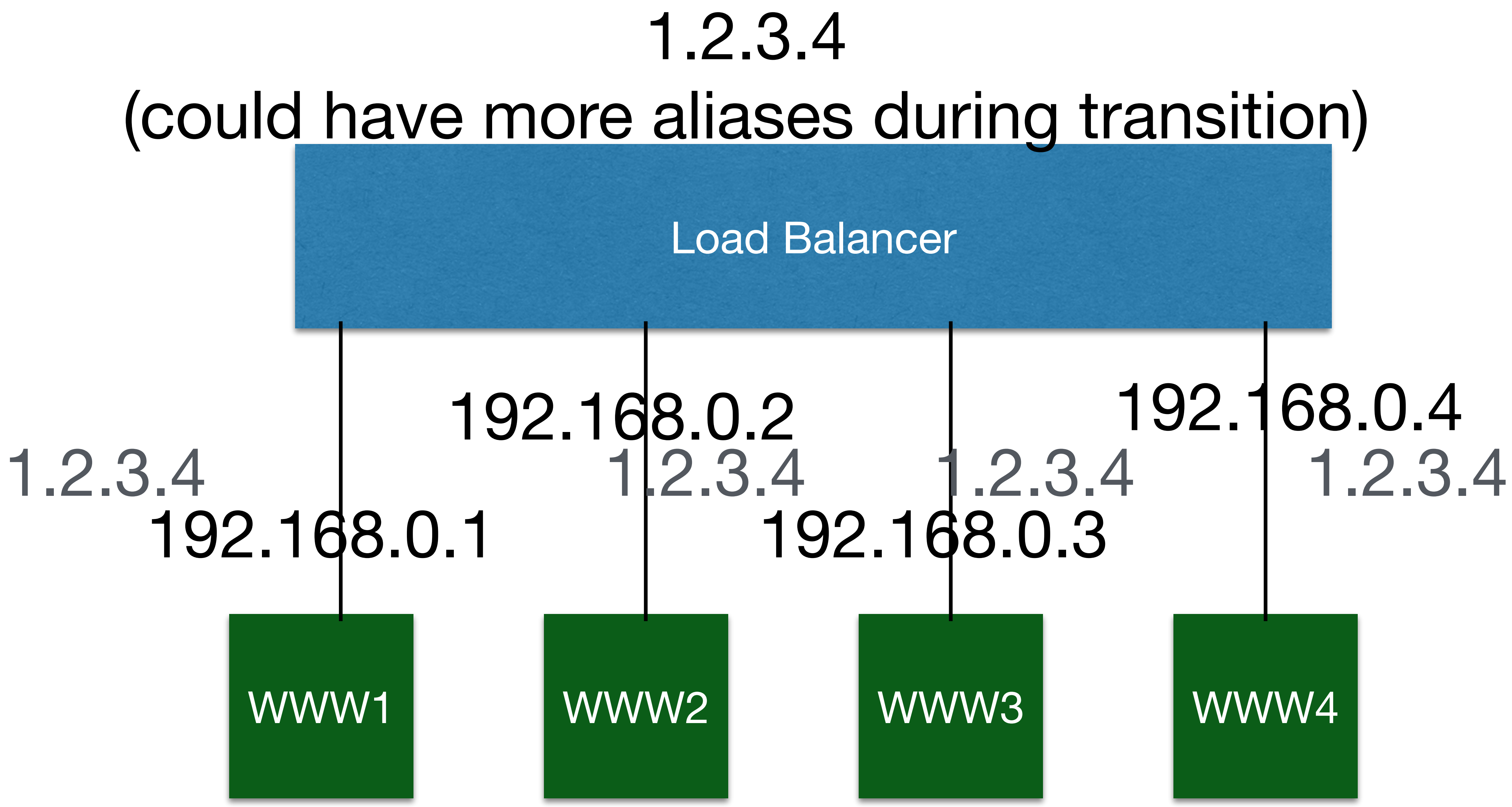
Naive Approach

- Treat as one machine with four addresses
- Rely on DNS round-robins to deal with balancing
  - Very rough balance
  - Poor tolerance of failure if clients don't try other addresses, slow tolerance of failure anyway

Load Balancers

- Inbound NAT
  - (Or simply routing, although that requires careful configuration: in 2021, rewriting headers is essentially free and much easier to manage and debug)
- Constantly ping devices and/or count live connections
- Sends incoming connection to least loaded, active machine
  - Can also use weighted balances, magic protocols to report load, measures of bandwidth, etc.
- Service lives on **one** IP number, balanced over multiple servers

Load Balance



www.dom.ain -> one address

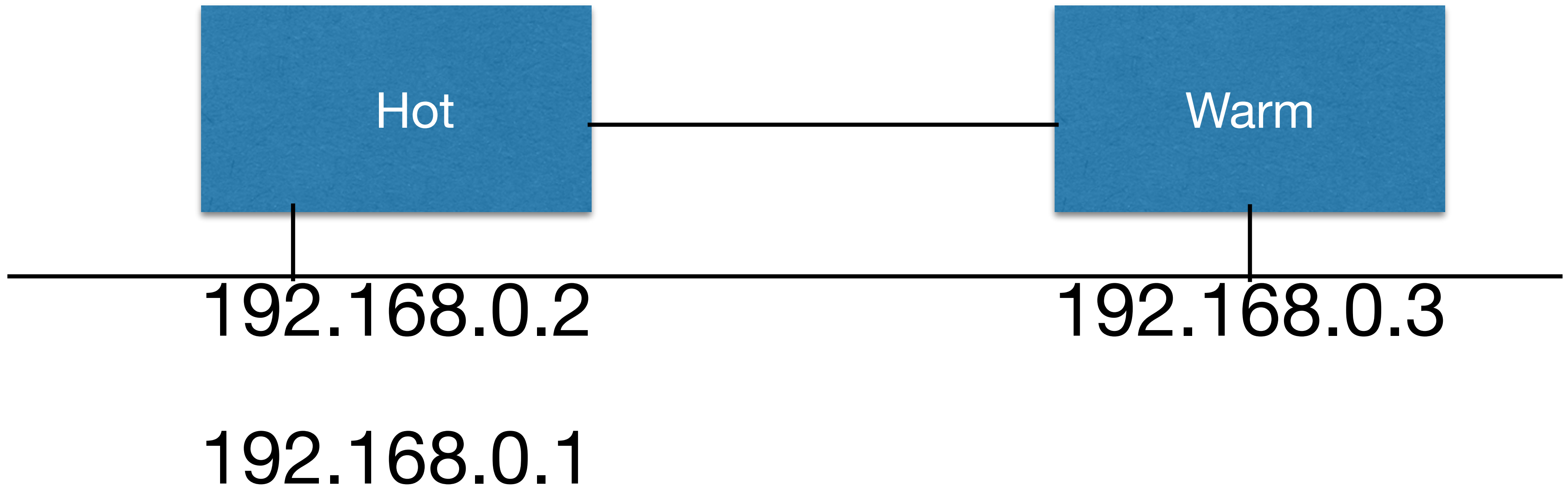
Load Balancing HTTP and HTTPS

- Sometimes needs slightly more than Inbound NAT, as connections to servers need to be slightly “sticky” — successive TCP connections from same source need to go to same destination because of session cookies and so on.
- Load balancer stores source of old sessions, uses same server for subsequent requests from same source.
  - Also a problem for load-spread web caches, but those are much less common today

VRRP

- What about routers, firewalls, etc which need to see all traffic?
- Can’t easily load balance, but can do failover

VRRP



VRRP

- Advertise shared IP number as real service address
    - Router for DHCP, DNS records for WWW, etc
- Mutual monitoring and takeover of shared address
- Ideally with multiple redundant networks between peers, as “split brain” situation potentially quite nasty (for example, database failover bad if both sides think they are alive)
- In simple cases, close to self configuring.

VRRP can be Simple

```
! Configuration File for keepalived
! https://raymii.org/s/tutorials/Keepalived-Simple-IP-failover-on-Ubuntu.html

vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 51
@pi-one    priority 150
@pi-three    priority 100
    virtual_ipaddress {
        81.187.150.215
    }
    ! see https://e-mc2.net/keepalived-documentation-nightmare
    ! virtual_ipaddress_excluded means "do not include this in the
    ! VRRP exchange, but bring it up when you are a master and
    ! take it down when you are a backup".  Using this to do mixed
    ! 4/6 is implied in the manual page
    virtual_ipaddress_excluded {
        2001:8b0:129f:a90e:3141:5926:5359:5
    }
}
```

Full Architectures

- A common design pattern is a pair of load balancers, operating active/passive with VRRP, managing a pool of back-end servers, operating active/active
- If you have time and a fascination with slow-motion car crashes, read Slaughter and May report into what went wrong at TSB in 2018: Chapter 14 (pp 129 et seq) is where the action is, and is precisely this design pattern going wrong.
- https://www.tsb.co.uk/news-releases/slaughter-and-may/

Any Casting

- I talked about BGP, by which you can advertise routes to a particular IP network to the Internet core.
- What happens if you advertise the same address twice? In different places?
- That doesn’t sound good!

Any Casting

- Google offer a DNS resolver on 8.8.8.8
  - Cloudfront on 1.1.1.1, IBM/etc on 9.9.9.9, hurry the day when GCHQ/NCSC manage to get permission for 25.25.25.25
- How does this work?

Any Casting

- Locate machines in data centres around the world.
- Each has a unique IP address, and can be reached via that address.
- Also have 8.8.8.8 (or whatever) on a VLAN, and advertise that address over BGP
- Core routers will pick one or more routes to 8.8.8.8
- Will tend to be (a) local and (b) up
- Load balancing and redundancy!

Any Casting

- Works well with UDP
- Works less well with TCP, particularly long-lived TCP, as if you get a new route midway it will obviously disconnect
- Ideal for DNS, static web content via CDN, email reception, etc.
- Has killed DNS load balancing in large-scale enterprises.

Summary

- VLAN: multiple logical networks on one set of cables and switches
- Link Agg: multiple cables (and possibly switches) acting as one network
- Load balancing: multiple systems sharing load directed to one IP address in one location
- VRRP: multiple systems in one location, one taking all the load
- Anycasting: multiple servers world-wide taking the load, with caveats