FAUCET stacking

The whole is greater than the sum of its parts

Brad Cowie

Faucet

Faucet control modes

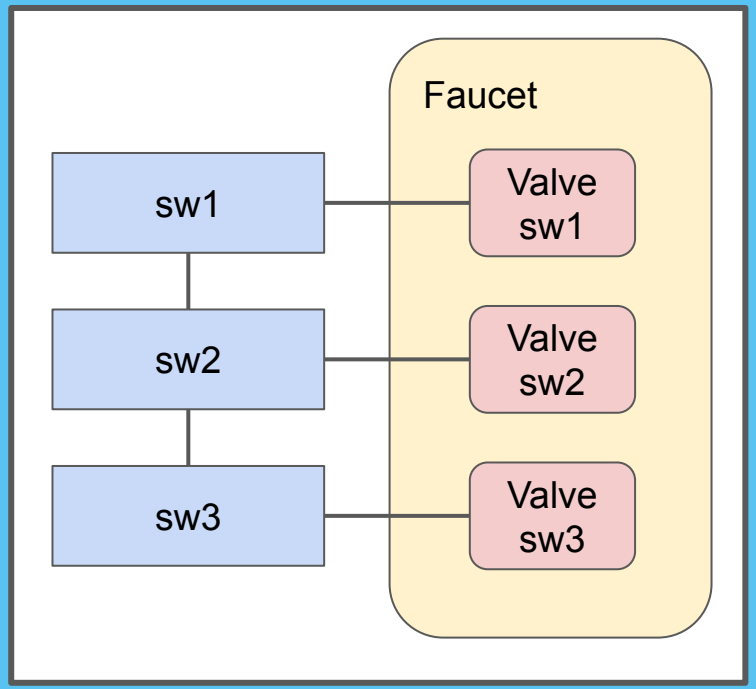
- Standalone mode
  - a.k.a independant, traditional, etc
- Stacked mode
  - a.k.a faucet fabric, distributed, etc

Standalone mode

- Default mode for faucet
- Learning/Routing happens per individual switch
- Pros
  - Simple
  - Expected behaviour for a switch
- Cons
  - Doesn’t utilise full network information controller has
  - Building a loop free network is more difficult (STP-like protocol required)

Faucet internals: standalone mode

- Each switch is mapped to a ***Valve***
- A ***Valve*** instance implements network behaviour
  - Learning
  - Routing
  - ACLs
  - etc
- In standalone mode, no coordination

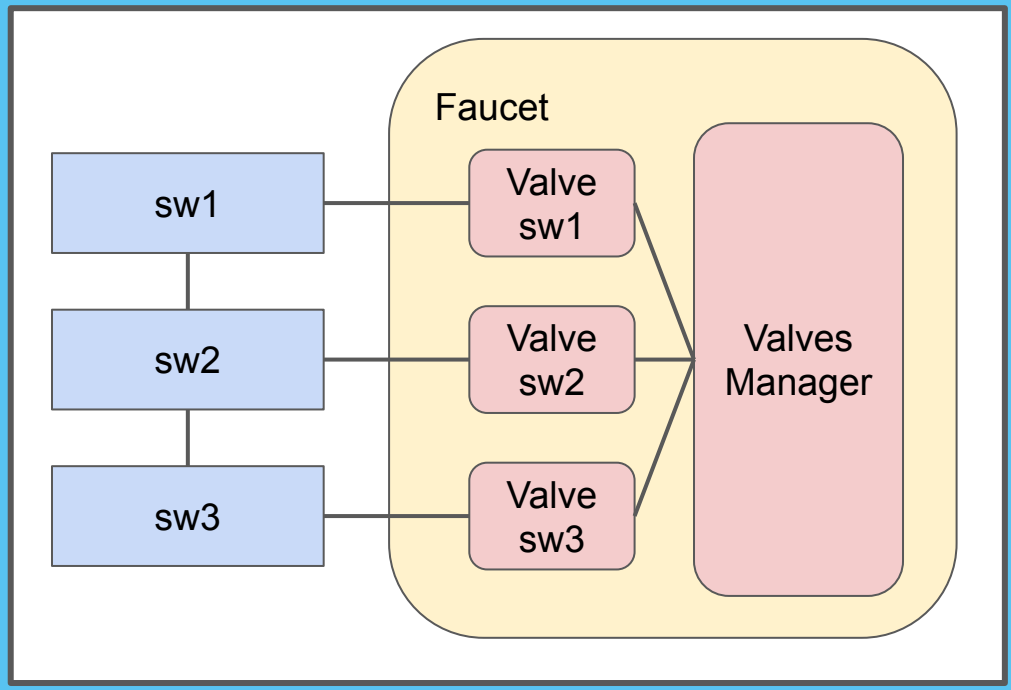


Stacked mode

- New mode for faucet
- The whole network acts as one big switch/router
  - Made up by individual devices that can come and go
- Faucet tracks network topology and makes decisions based on that
- Pros
  - Easily build a loop-free topology
  - Easily recover from failure scenarios
  - Less configuration required
- Cons
  - More complexity in controller
  - Not as obvious when you look at individual switch's flow table what is going on

Faucet internals: stacked mode

- Co-ordination between ***Valves***



How does it work?

- Use networkx to construct graph
  - root datapath (root)
  - datapaths (nodes)
  - stack links (edges)
- The graph can then be used to calculate shortest path
- Shortest path allows stacking to provide multiple redundant paths between switches without looping broadcast traffic

Stack algorithms

- Stacking code is built to be modular
- For different scenarios/topologies we can use different algorithms

Stack flooding algorithms

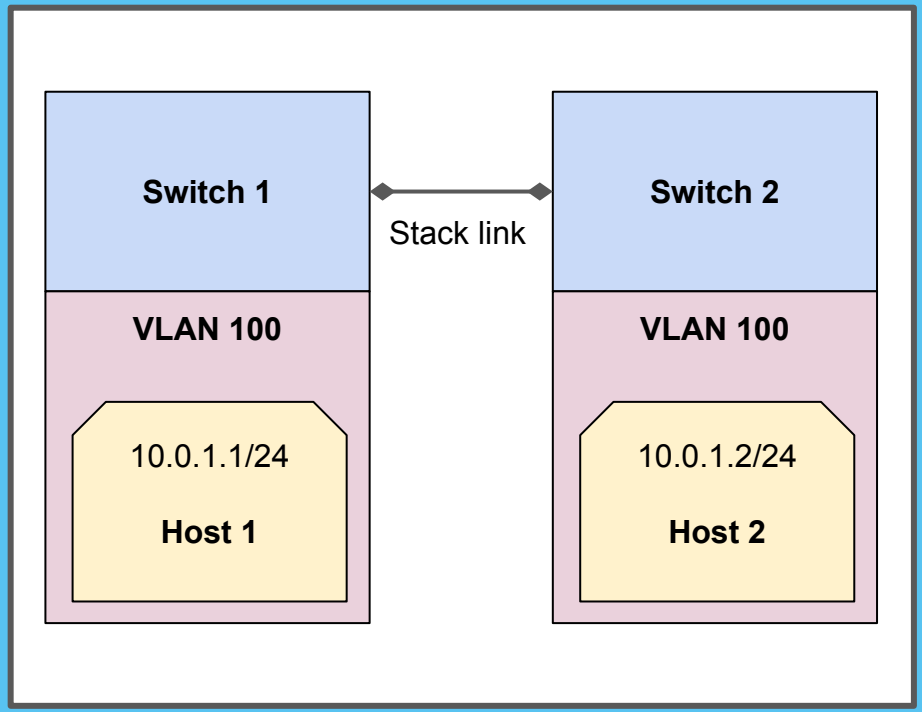
- ***ValveFloodStackManagerNoReflection***
  - For stacks of size 2 (all switches directly connected to root)
  - Non-root switches simply flood to the root
  - Root switch simply floods to all other switches
- ***ValveFloodStackManagerReflection***
  - For stacks of size > 2 (reflect floods off of root)
  - The root switch reflects incoming floods back out
  - Non-root switches flood only to the root
  - Non-root switches flood packets from the root locally and to downstream switches
- More optimal algorithms to come

Additional features of stacking

- Cable verification & link testing
  - Is my network physically wired the same as my config?
  - Do my stack links work?
- Automatic VLAN expression
  - Faucet automatically configures stack links to have the VLANs you need on them
- Failover
  - Redundant links
  - Redundant root
- Tunneling
  - Can automatically stitch up tunnel over network

Simplest stack topology

- Two switches
- Single stack link



Faucet

Simplest stack topology

Use switch1 as root of stack

Define stack links between switches

```
dps:
    switch1:
        dp_id: 0x1
        hardware: "Open vSwitch"
        stack:
            priority: 1
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        dp_id: 0x2
        hardware: "Open vSwitch"
        interfaces:
            1:
                description: "host2"
                native_vlan: 100
            2:
                stack:
                    dp: "switch1"
                    port: 2
```

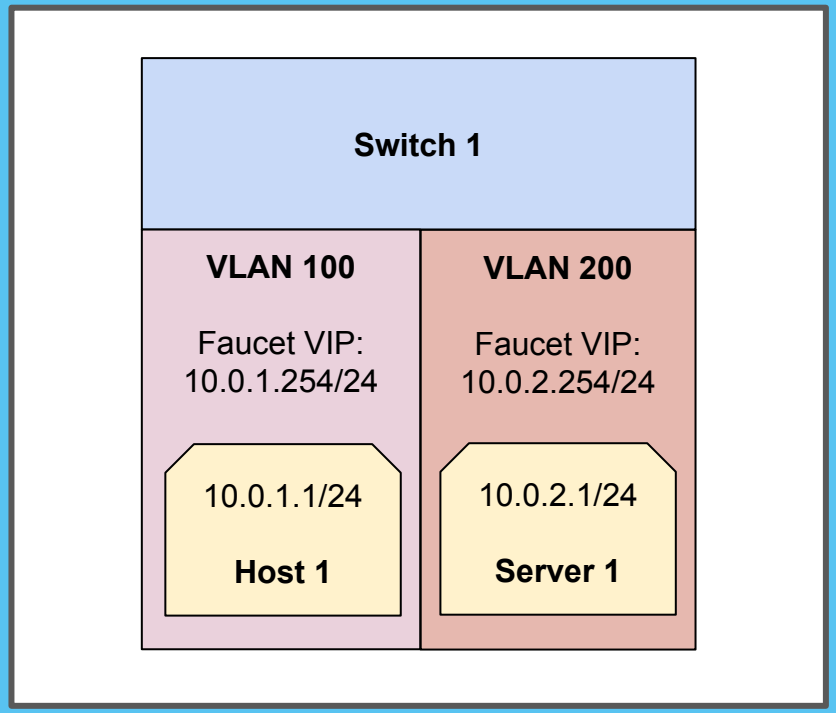
Faucet

More complicated topologies

- Inter-VLAN routing
- Tunneling
- Resilient network designs

Inter-VLAN routing

- Allow hosts on different VLANs to route between each other



Faucet

Inter-VLAN routing

Define virtual IPs on VLANs with routing enabled

```
vlans:
    hosts:
        vid: 100
        faucet_vips: ['10.0.1.254/24']
        faucet_mac: "00:00:00:00:00:11"
    servers:
        vid: 200
        faucet_vips: ['10.0.2.254/24']
        faucet_mac: "00:00:00:00:00:22"
routers:
    hosts-servers:
        vlans: ['hosts', 'servers']
dps:
    switch1:
        dp_id: 0x1
        hardware: "Open vSwitch"
        interfaces:
            1:
                description: "host1"
                native_vlan: "hosts"
            2:
                description: "server1"
                native_vlan: "servers"
```

Faucet

Inter-VLAN routing

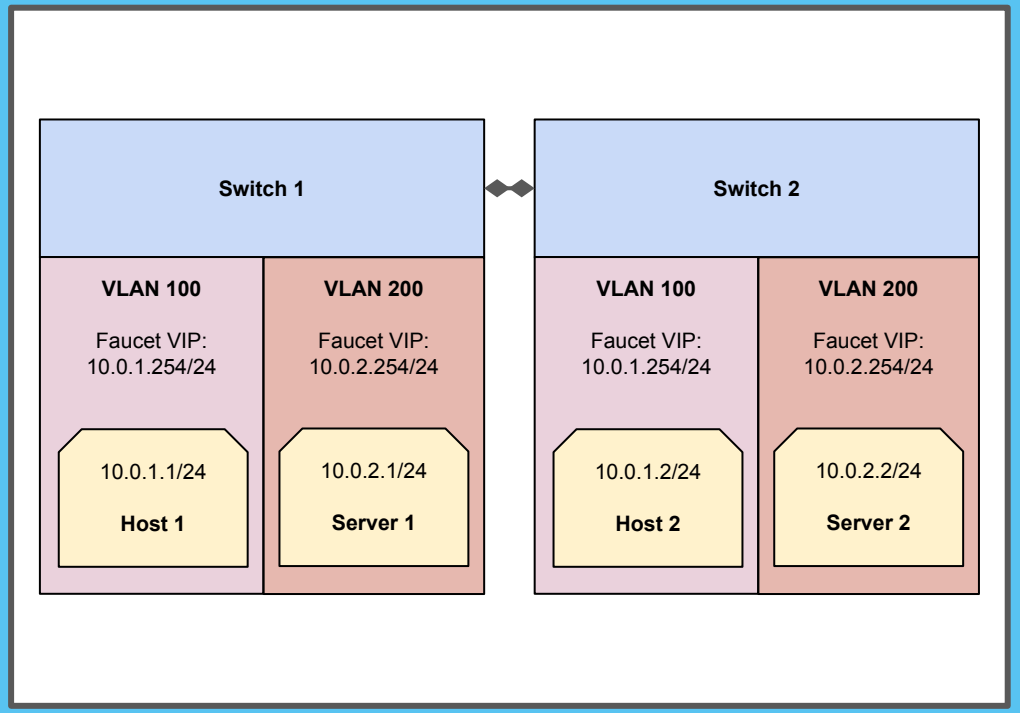
Define a VLAN router

```
vlans:
    hosts:
        vid: 100
        faucet_vips: ['10.0.1.254/24']
        faucet_mac: "00:00:00:00:00:11"
    servers:
        vid: 200
        faucet_vips: ['10.0.2.254/24']
        faucet_mac: "00:00:00:00:00:22"
routers:
    hosts-servers:
        vlans: ['hosts', 'servers']
dps:
    switch1:
        dp_id: 0x1
        hardware: "Open vSwitch"
        interfaces:
            1:
                description: "host1"
                native_vlan: "hosts"
            2:
                description: "server1"
                native_vlan: "servers"
```

Faucet

Multi-datapath inter-VLAN routing

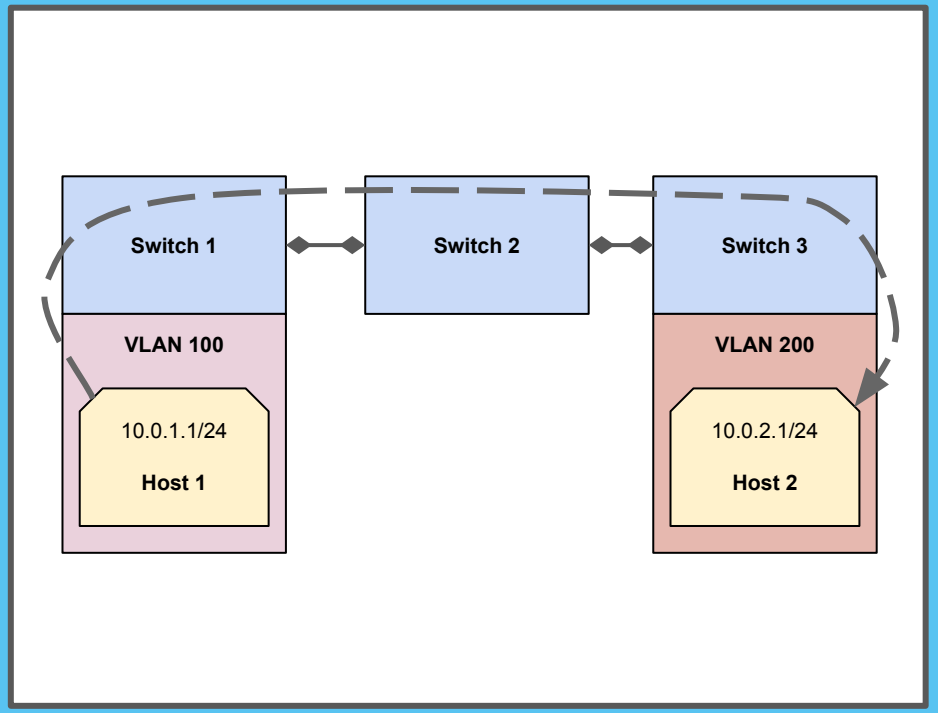
- Allow hosts on different VLANs to route between each other even if they are on different switches
- Automatically enabled when stack ports present and VLAN ***router*** statement is present



Faucet

Tunneling

- Automatic tunnel stitching over stack topology
- Use faucet ACLs to decide what flows to put inside tunnel
- When stack topology changes tunnel gets automatically rerouted



Faucet

Tunneling

ACL doesn't have specific match so will match all packets

Apply as a port ACL to match everything on that port

```
acls:
   tunnel-to-host2:
       - rule:
           actions:
               output:
                   tunnel:
                       type: "vlan"
                       tunnel_id: 902
                       dp: "switch3"
                       port: 1
dps:
    switch1:
        ...
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
                acls_in: ["tunnel-to-host2"]
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        ...
    switch3:
        ...
```

Faucet

Tunneling

New “tunnel” output action

```
acls:
   tunnel-to-host2:
       - rule:
           actions:
               output:
                   tunnel:
                       type: "vlan"
                       tunnel_id: 902
                       dp: "switch3"
                       port: 1
dps:
    switch1:
        ...
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
                acls_in: ["tunnel-to-host2"]
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        ...
    switch3:
        ...
```

Faucet

Tunneling

Type of tunnel, only VLAN tunnels supported right now

```
acls:
   tunnel-to-host2:
       - rule:
           actions:
               output:
                   tunnel:
                       type: "vlan"
                       tunnel_id: 902
                       dp: "switch3"
                       port: 1
dps:
    switch1:
        ...
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
                acls_in: ["tunnel-to-host2"]
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        ...
    switch3:
        ...
```

Faucet

Tunneling

Which VLAN ID to use

```
acls:
   tunnel-to-host2:
       - rule:
           actions:
               output:
                   tunnel:
                       type: "vlan"
                       tunnel_id: 902
                       dp: "switch3"
                       port: 1
dps:
    switch1:
        ...
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
                acls_in: ["tunnel-to-host2"]
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        ...
    switch3:
        ...
```

Faucet

Tunneling

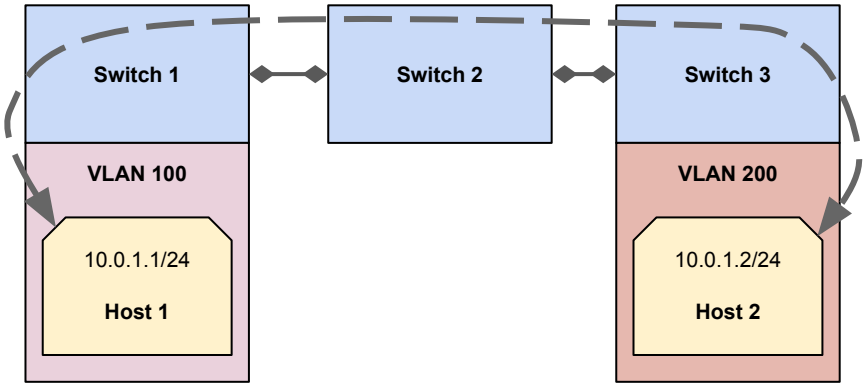
Where should packets matching this ACL go?

```
acls:
   tunnel-to-host2:
       - rule:
           actions:
               output:
                   tunnel:
                       type: "vlan"
                       tunnel_id: 902
                       dp: "switch3"
                       port: 1
dps:
    switch1:
        ...
        interfaces:
            1:
                description: "host1"
                native_vlan: 100
                acls_in: ["tunnel-to-host2"]
            2:
                stack:
                    dp: "switch2"
                    port: 2
    switch2:
        ...
    switch3:
        ...
```

Faucet

Tunneling - both directions

```
acls:
    tunnel-to-host1:
        - rule:
            actions:
                output:
                    tunnel:
                        Type: "vlan"
                        tunnel_id: 901
                        dp: "switch1"
                        port: 1
    tunnel-to-host2:
        - rule:
            actions:
                output:
                    tunnel:
                        type: "vlan"
                        tunnel_id: 902
                        dp: "switch3"
                        port: 1
```



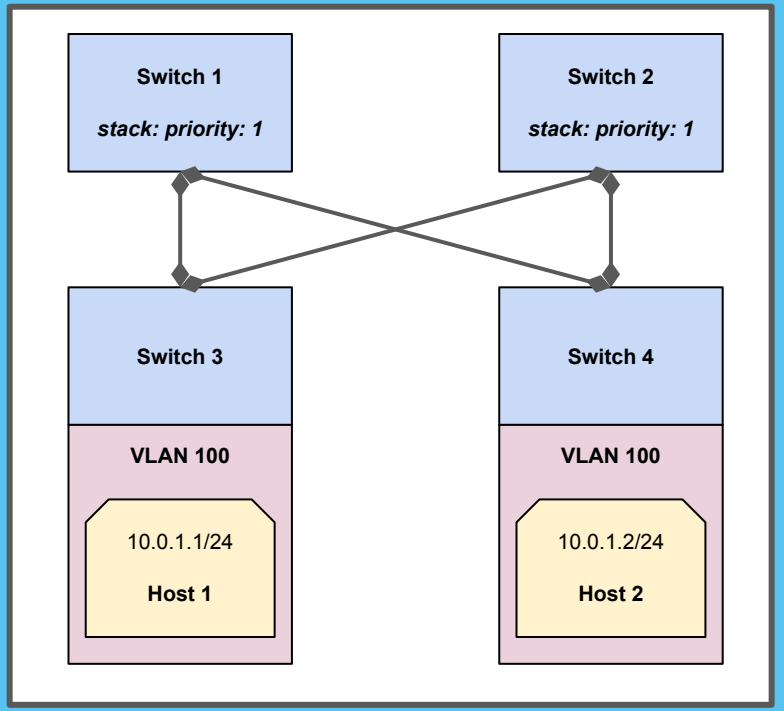
Faucet

Resiliency

- Stacking solves one resiliency problem at the cost of introducing another
  - Network graph + shortest path allows us to introduce loops
  - Network graph needs a switch to be the root
- One more feature is required to allow automatic recovery from hardware failure
  - Multi-root stack

Bulletproof faucet

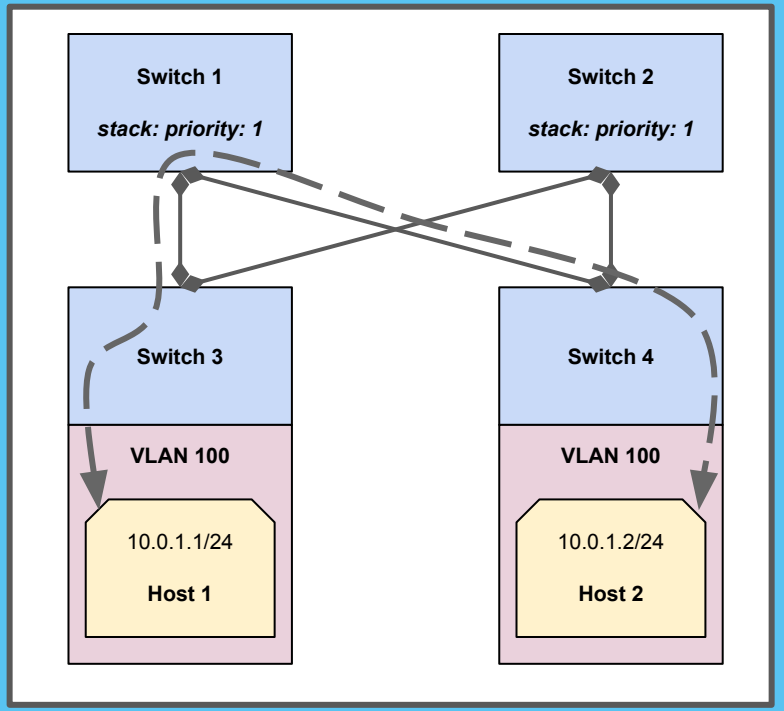
- Adding ***stack priority*** values to multiple switches allows us to have multiple root candidates
- When a root fails another can take over



Faucet

Bulletproof faucet

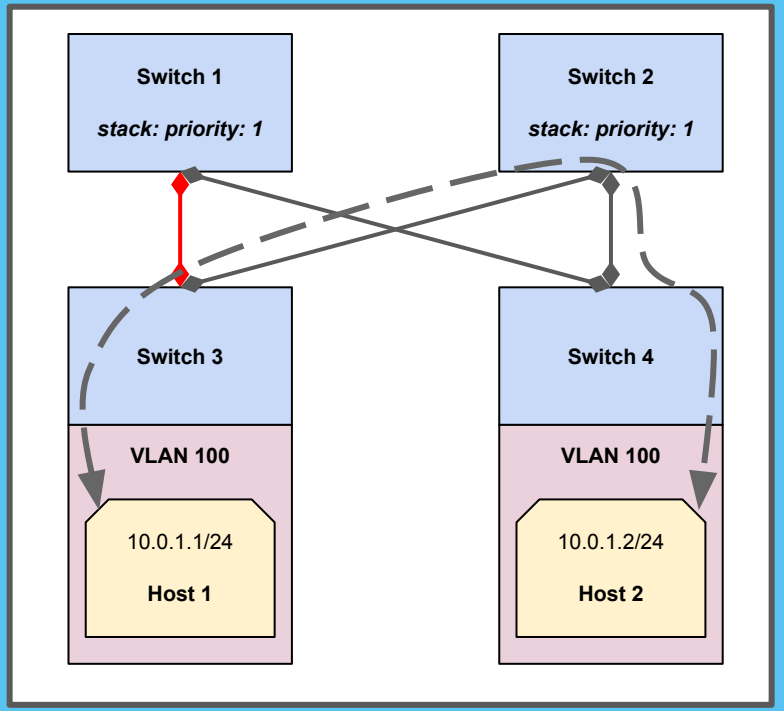
- Adding ***stack priority*** values to multiple switches allows us to have multiple root candidates
- When a root fails another can take over



Faucet

Bulletproof faucet

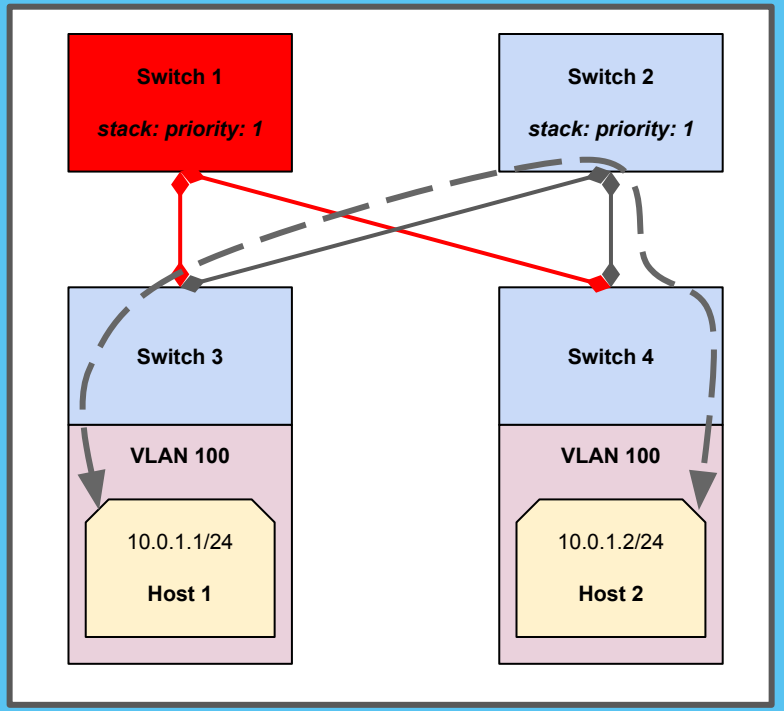
- Can survive a cable failure



Faucet

Bulletproof faucet

- Can survive a switch failure



Faucet

Want to try stacking for yourself?

- Complete the tutorial series on our website
- In an hour you will be able to configure everything we talked about today
  - Basic stacking
  - Inter-VLAN routing with stacking
  - Tunneling over a stack
  - Redundant stack links
  - Multi-root stack

https://docs.faucet.nz/en/latest/tutorials/stacking.html

Thanks

- Josh Bailey
- Mark Bishop (a.k.a mab68)

Questions?

https://faucet.nz

@faucetsdn

brad@waikato.ac.nz

@nzgizmoguy