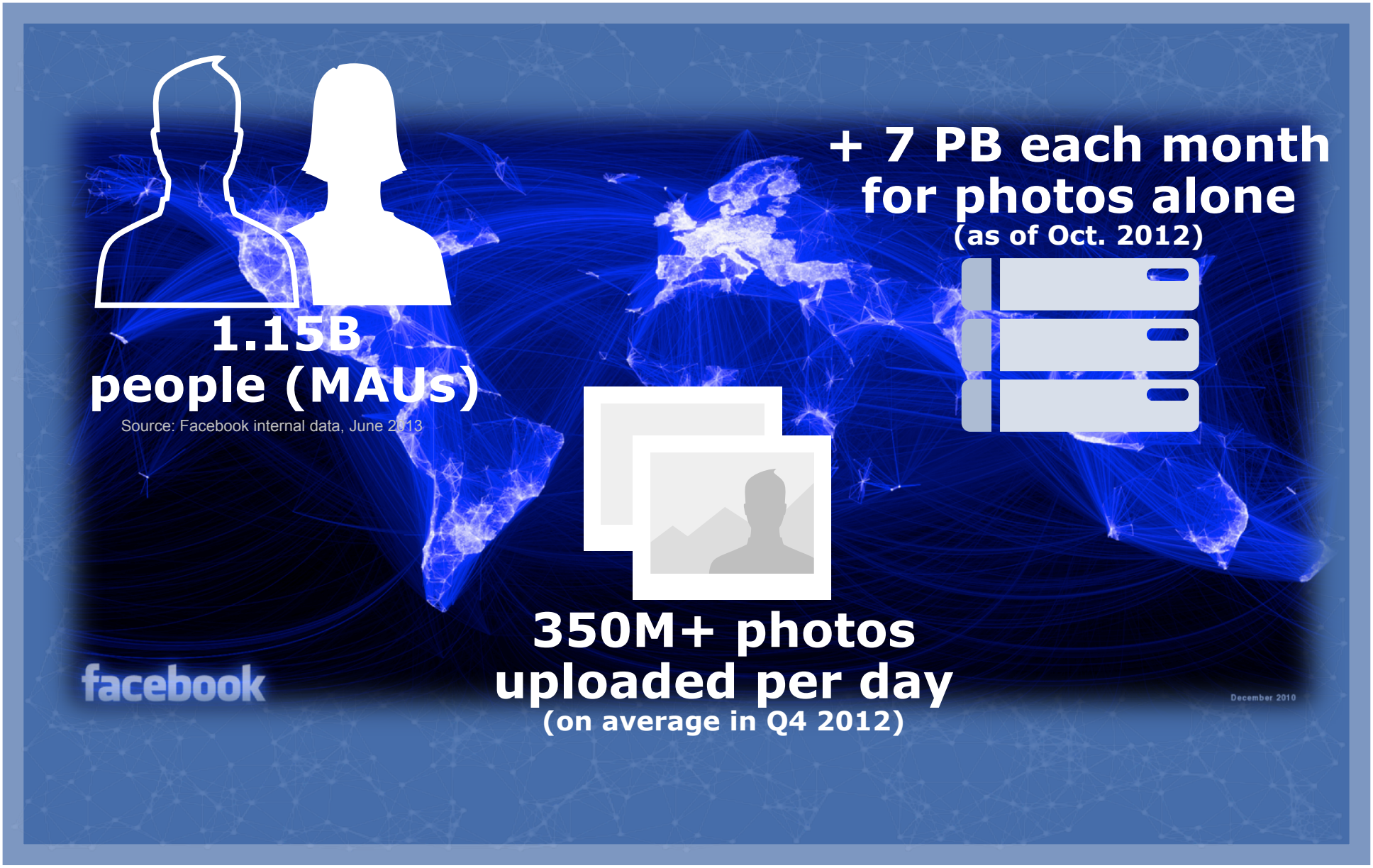
facebook

datacenter networking

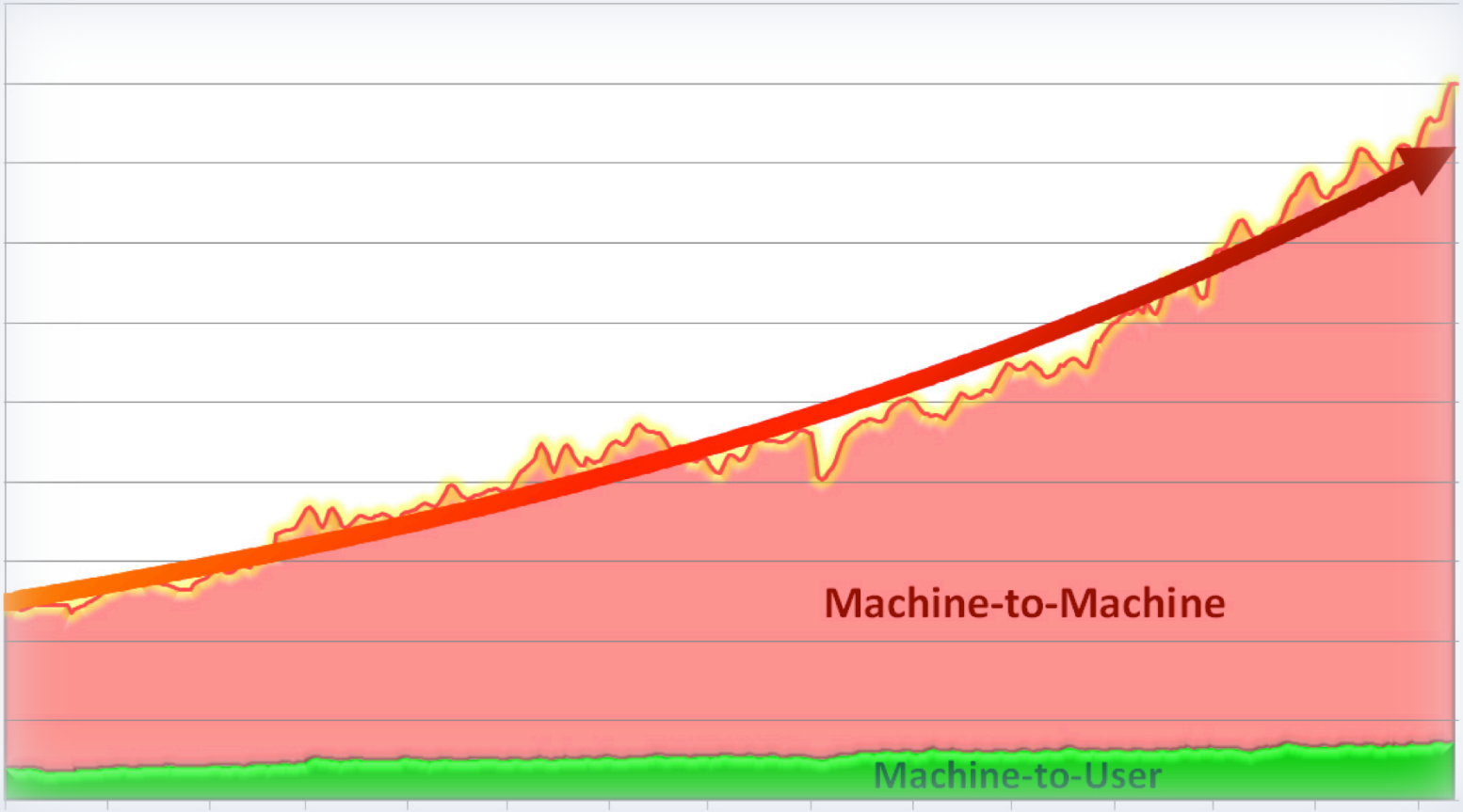
8-OCT-2013 – NANOG 59

david swafford
network engineer

dswafford@fb.com


1.15B
people (MAUs)
Source: Facebook internal data, June 2013
+ 7 PB each month
for photos alone
(as of Oct. 2012)
350M+ photos
uploaded per day
(on average in Q4 2012)
facebook
December 2010

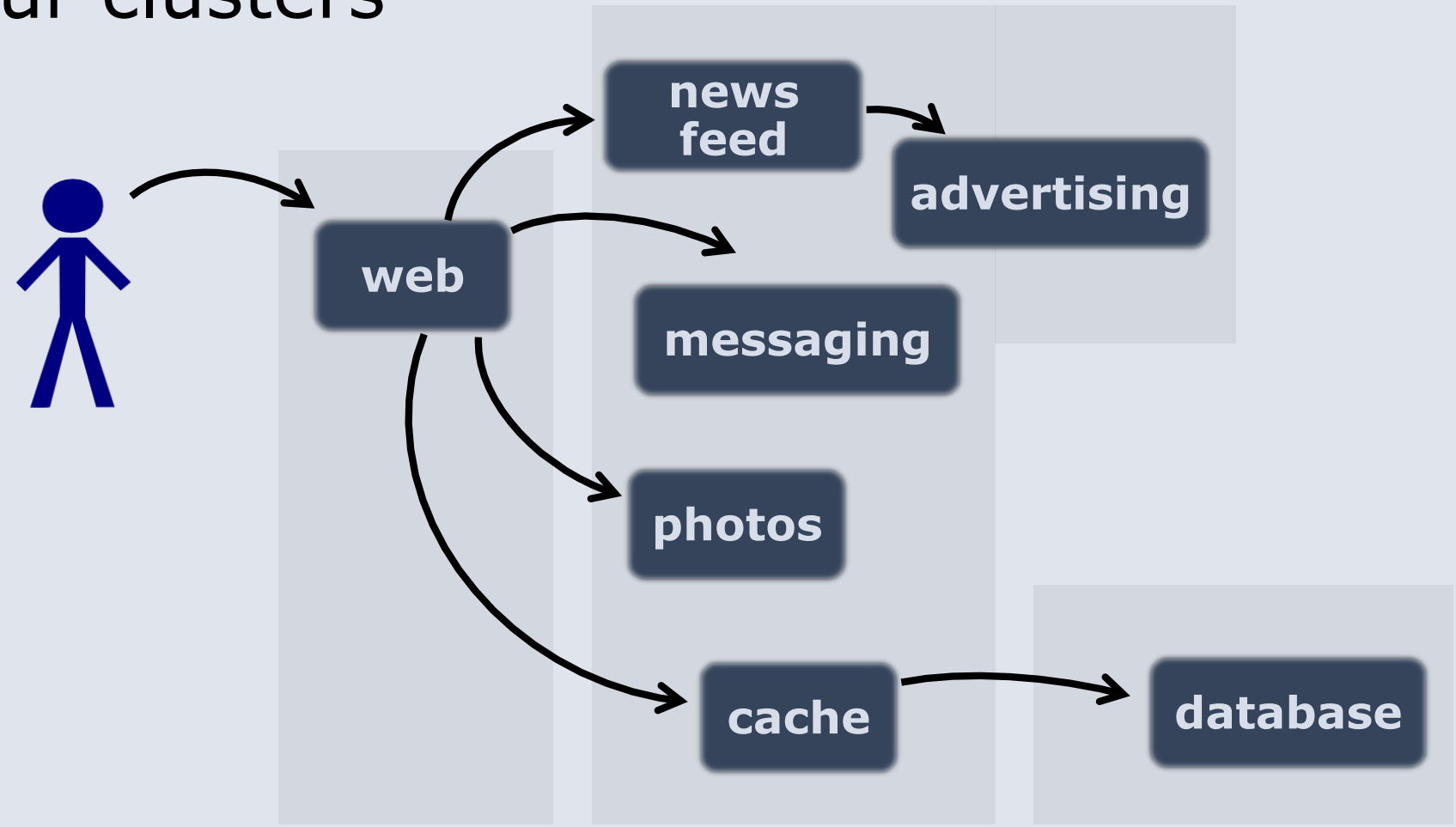
traffic growth



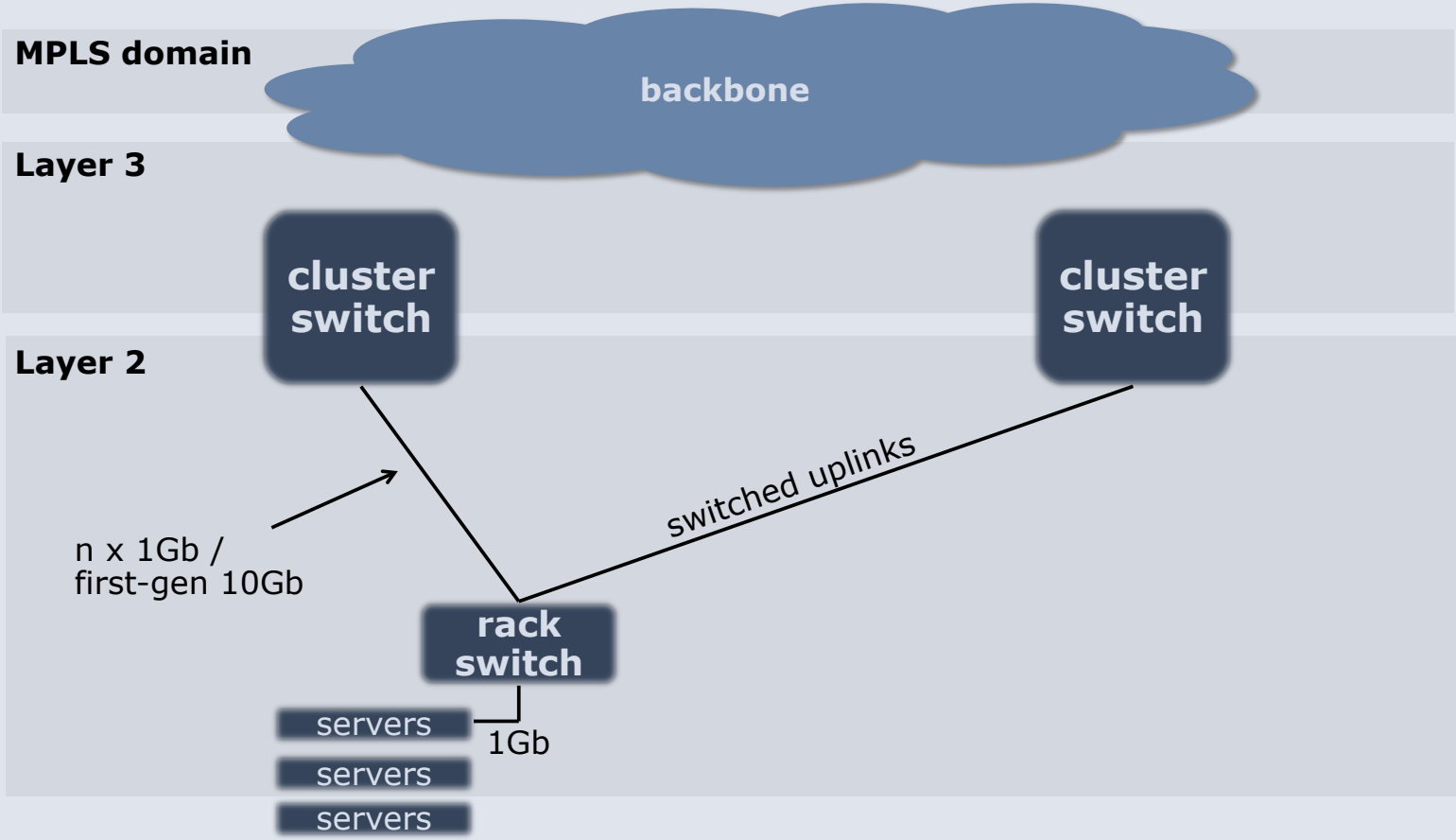
clusters

a unit of compute

our clusters



1st generation clusters

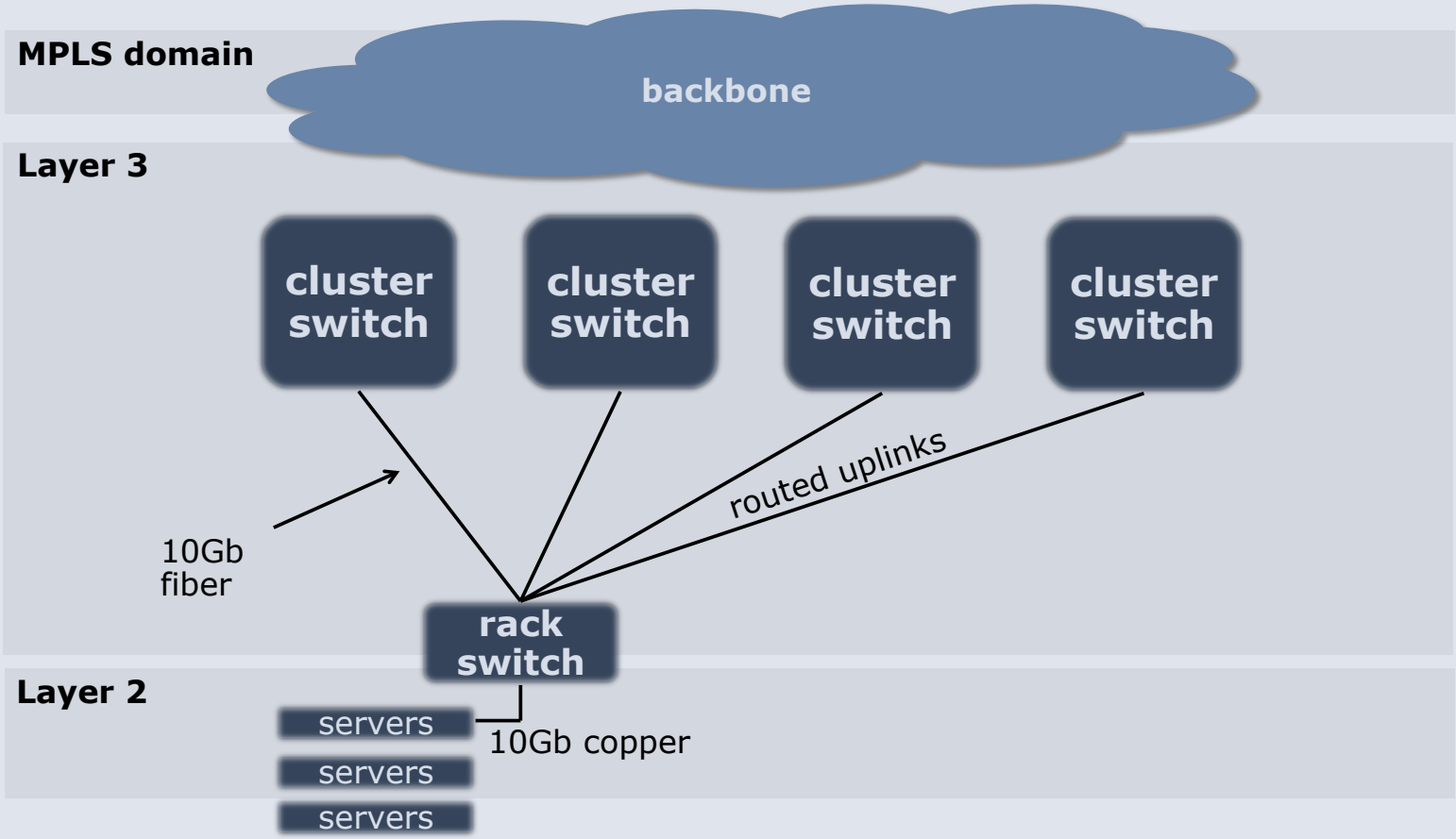


early challenges

usable capacity

Layer 2 scaling

2nd generation



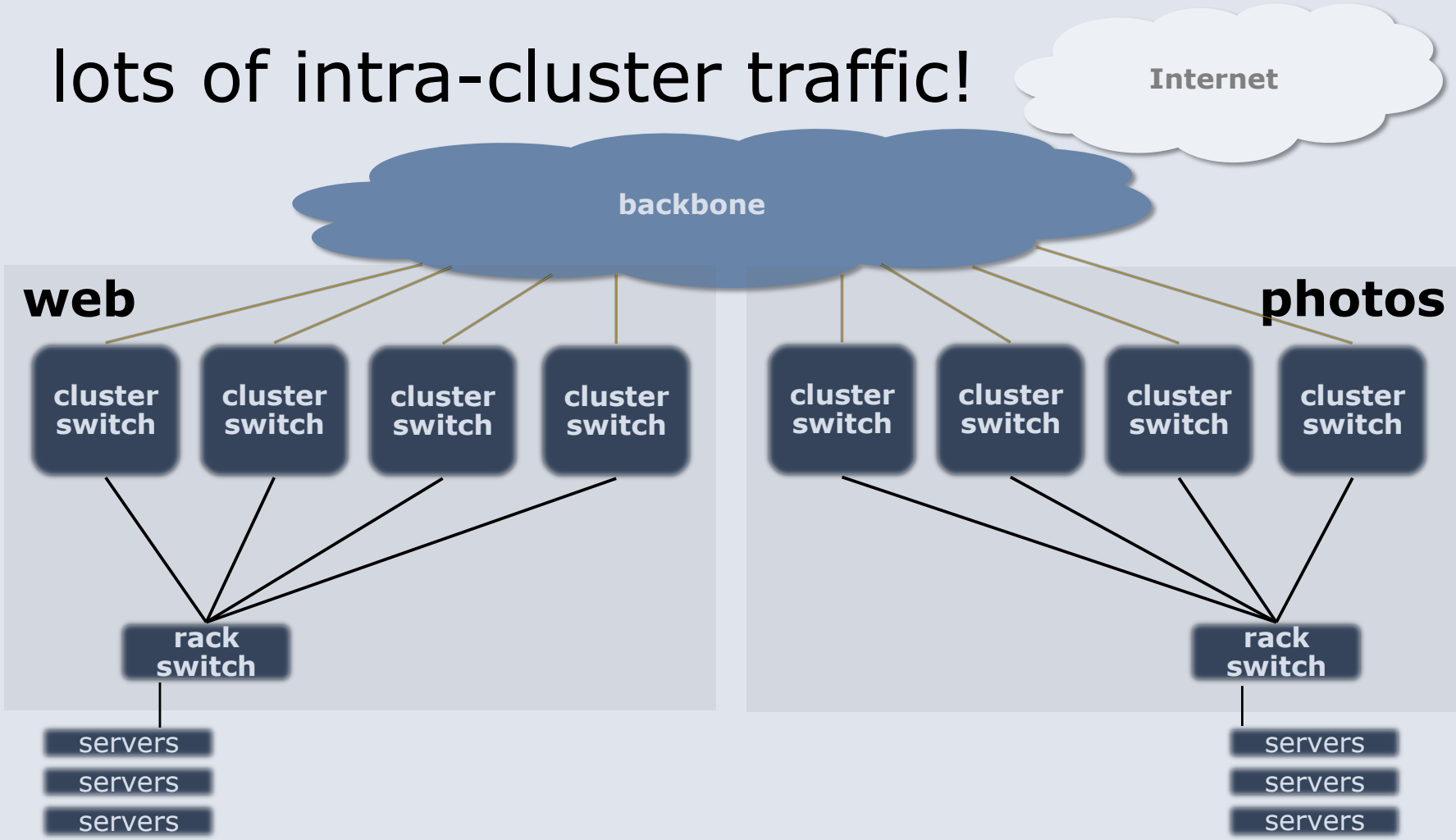
BGP to the rack

control

scale

no IGP

lots of intra-cluster traffic!



the primary role of our backbone

connects datacenters

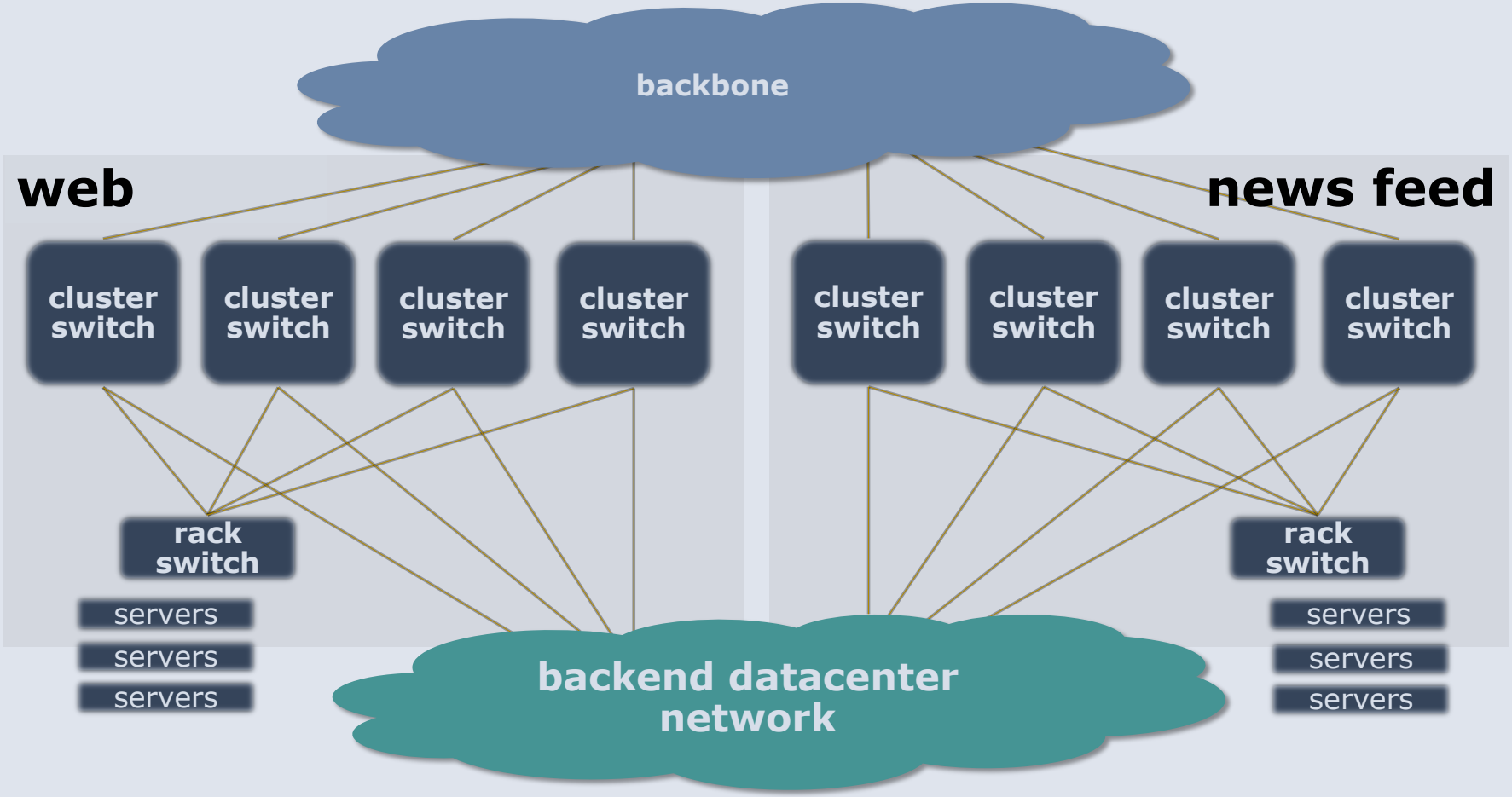
private / transit peering

growing pains

backbone devices are too powerful for intra-DC needs

looking for a better way to scale…

evolving


backbone
web
news feed
cluster switch
cluster switch
cluster switch
cluster switch
cluster switch
cluster switch
cluster switch
cluster switch
rack switch
rack switch
servers
servers
servers
servers
servers
servers
backend datacenter network

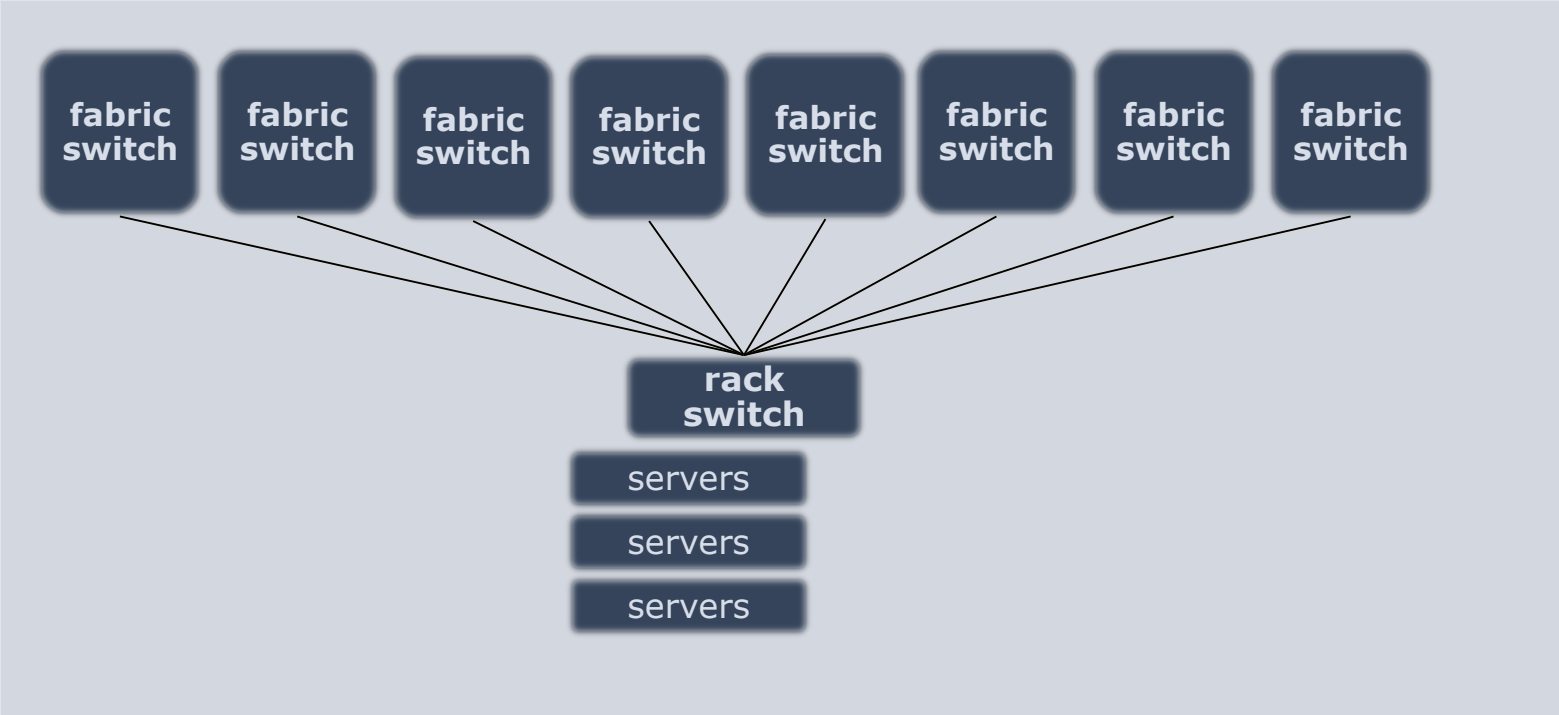
it works, why change?

chassis break in obscure ways

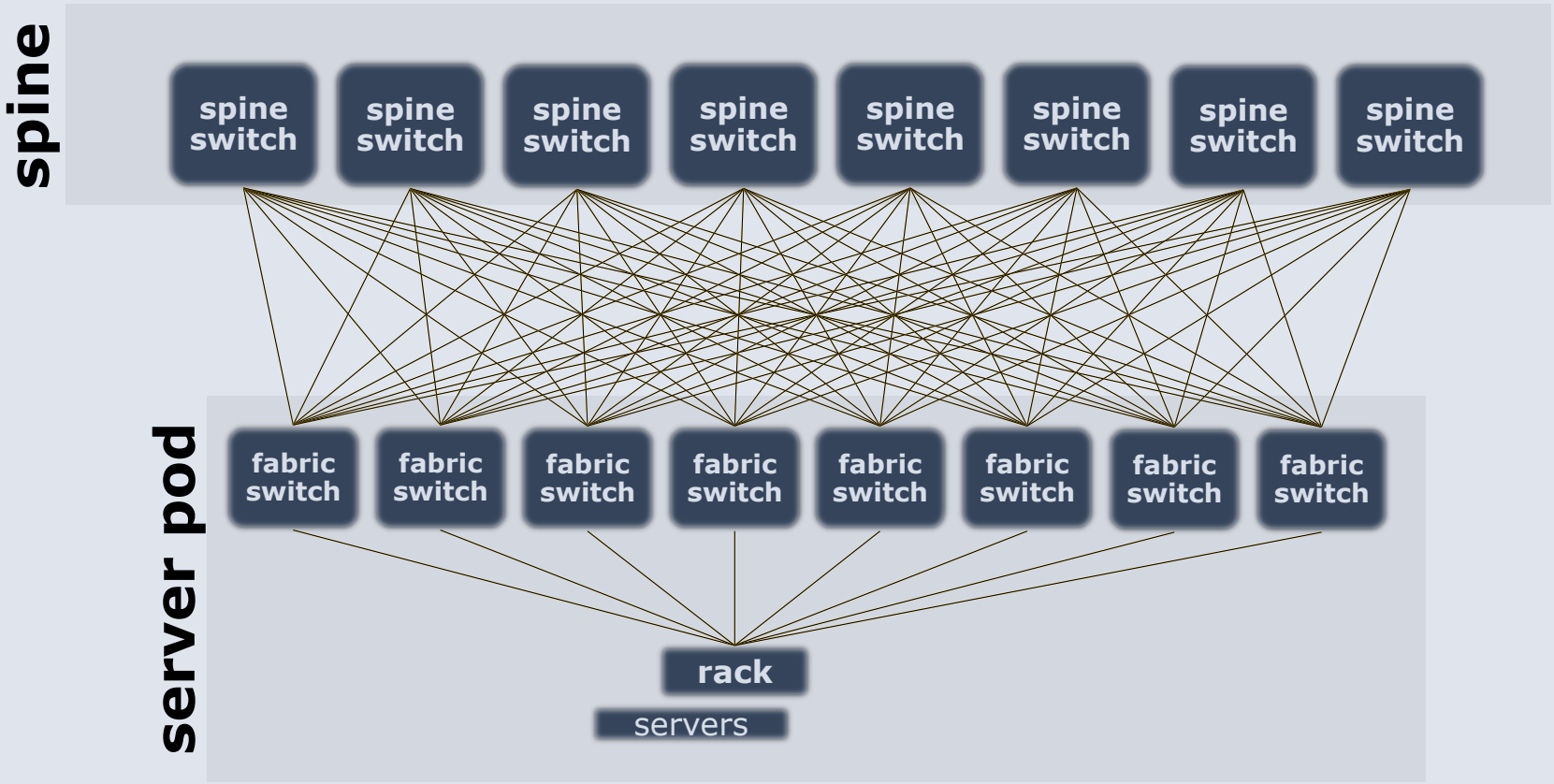
efficiency

3rd generation

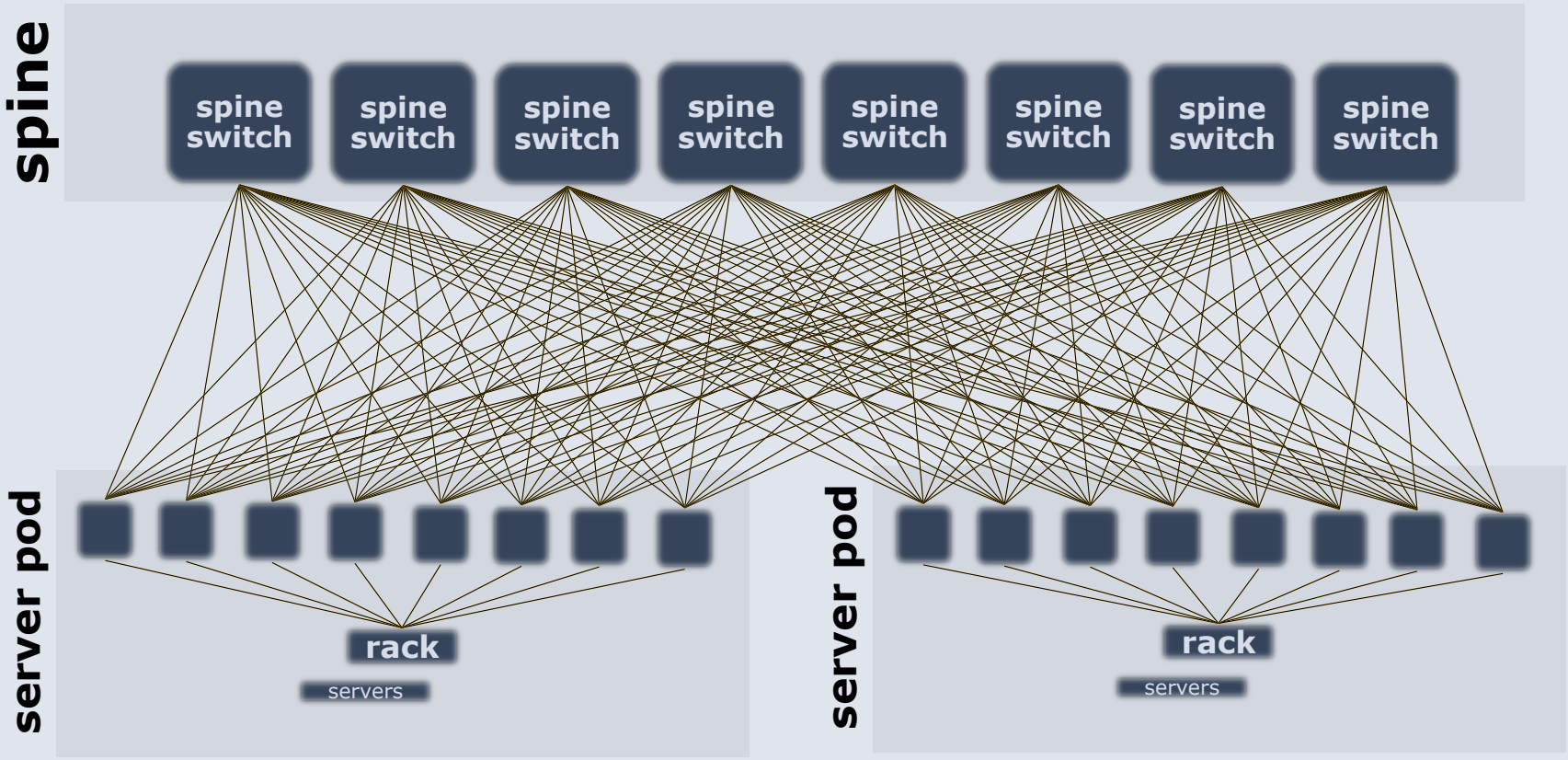
Folded Clos datacenter-wide fabric



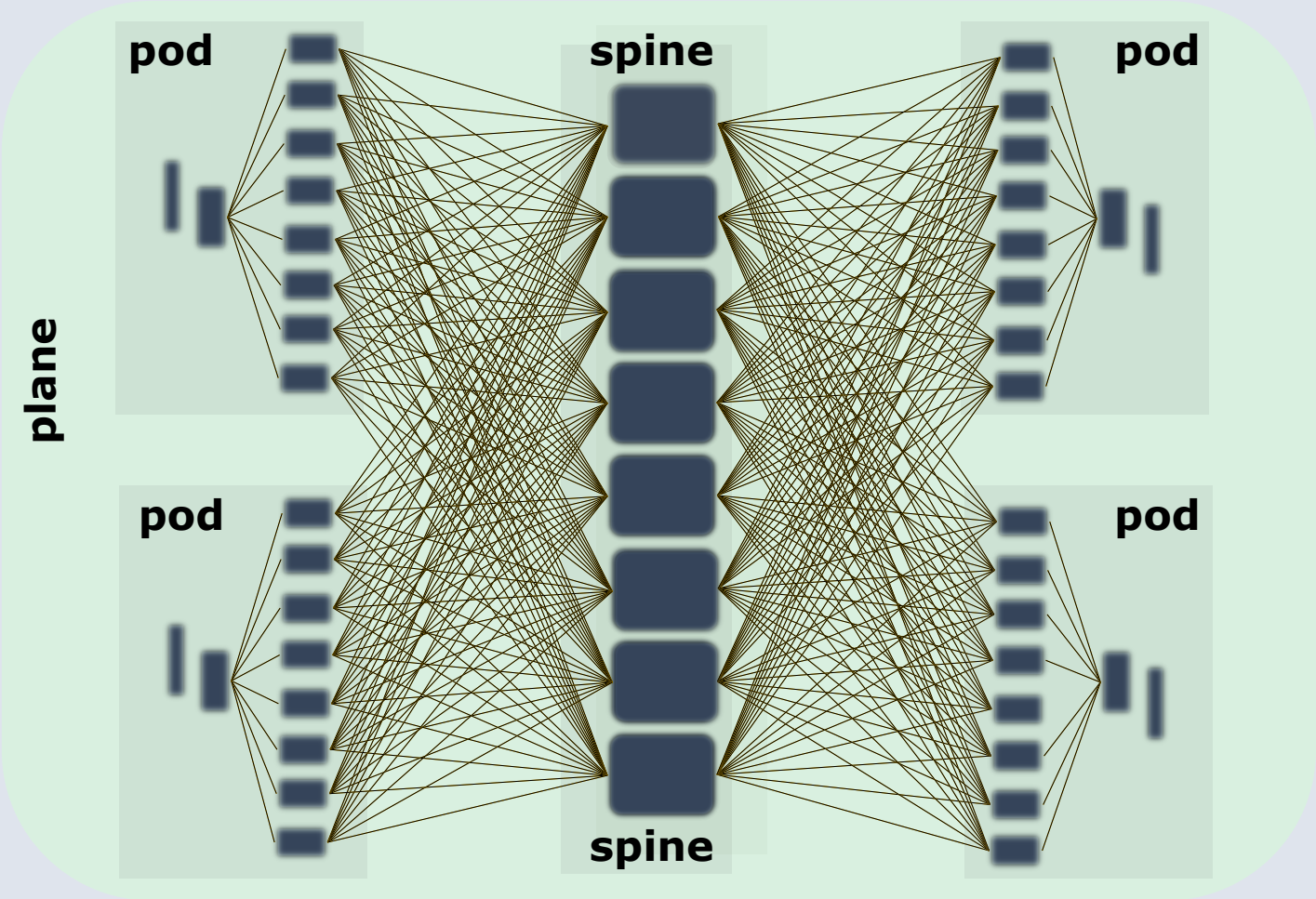
3rd generation



3rd generation

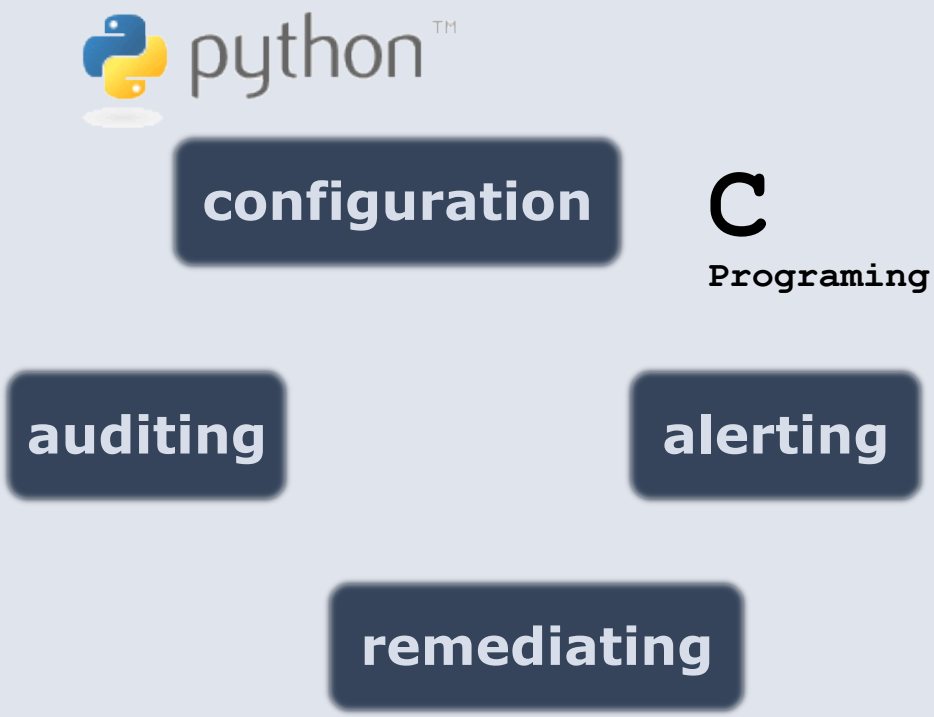


scaling to a full datacenter

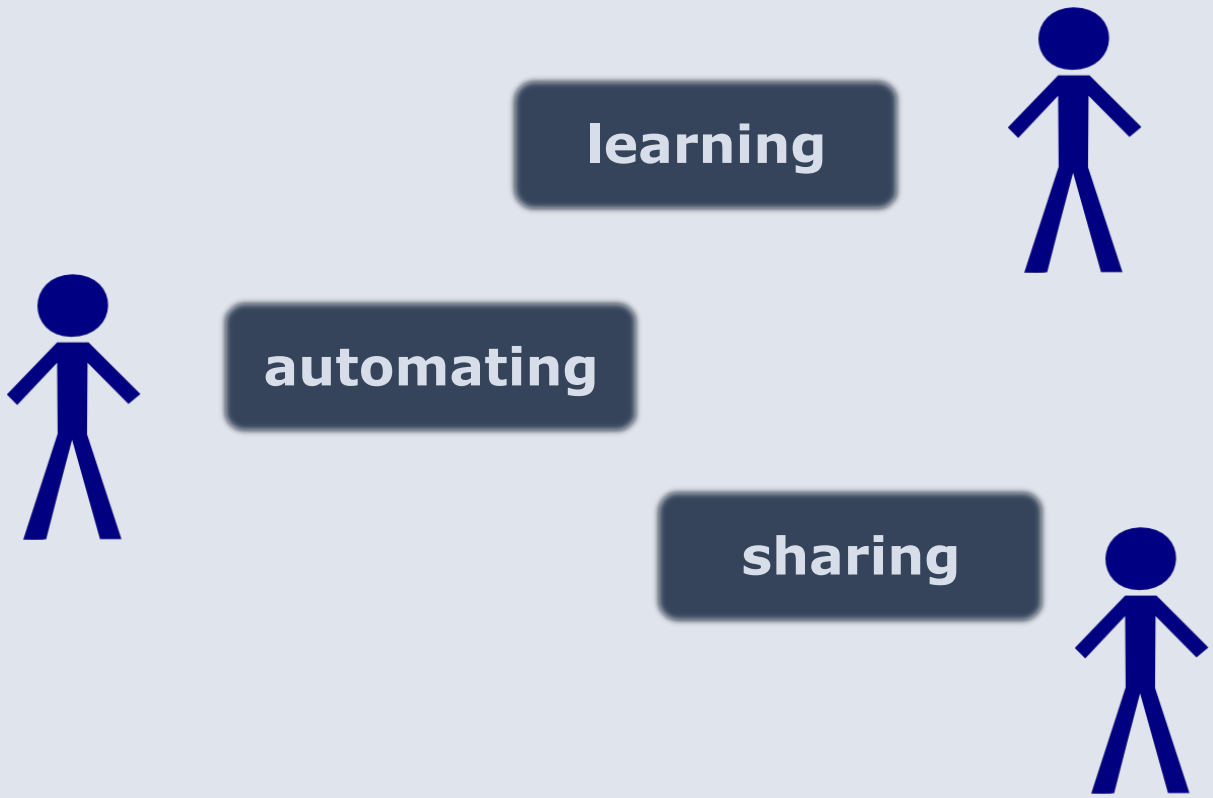


managing everything

approaching networking from a software mindset



frees up engineers to make greater impact



deploying a cluster switch

step	engineer	computer
planning the port map		✓
physical installation	✓	
generating configuration		✓
applying		✓
validating		✓
enabling for live traffic	✓	

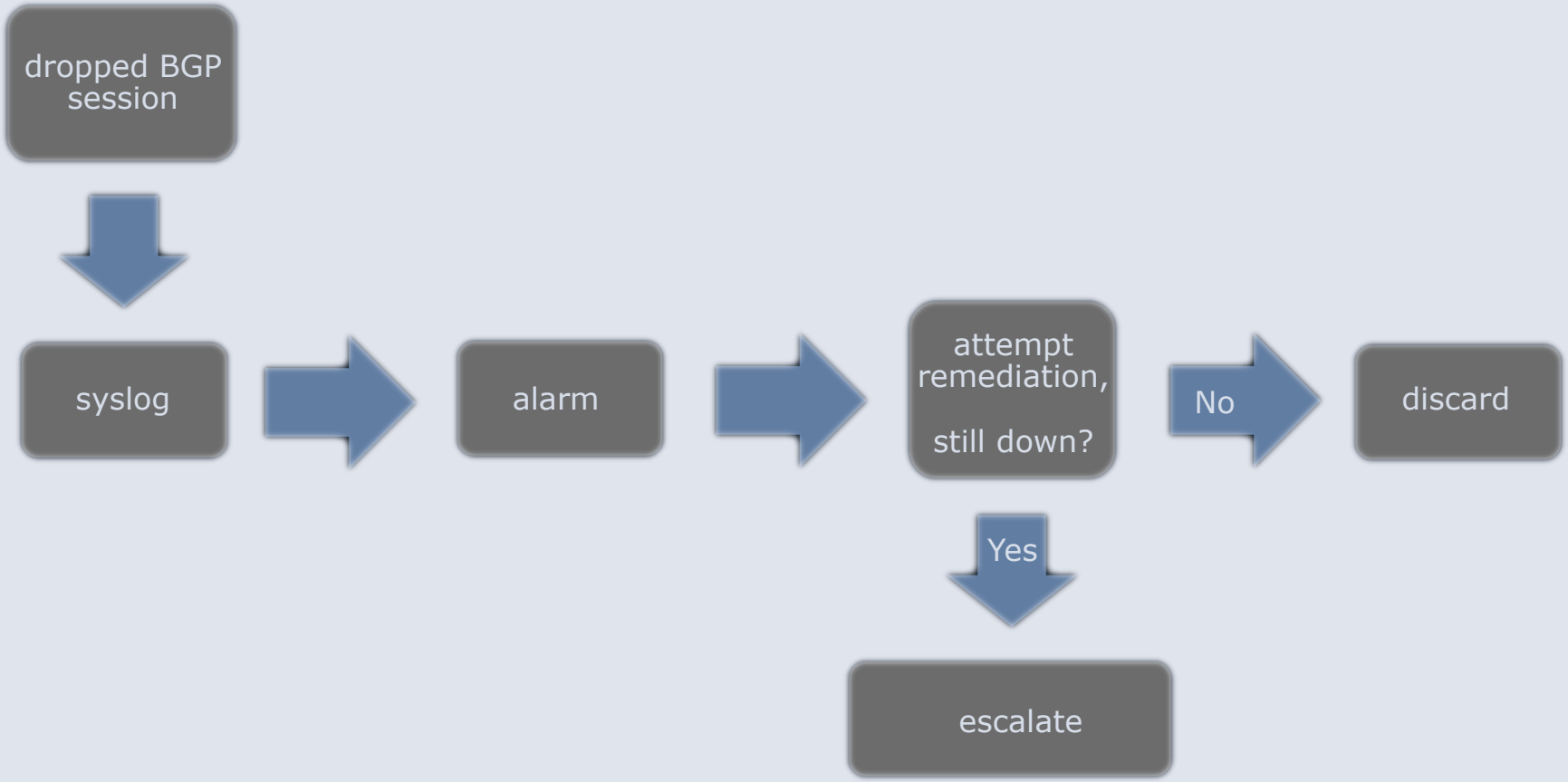
FBAR

engineers create audits and remediation scripts

audits trigger alarms

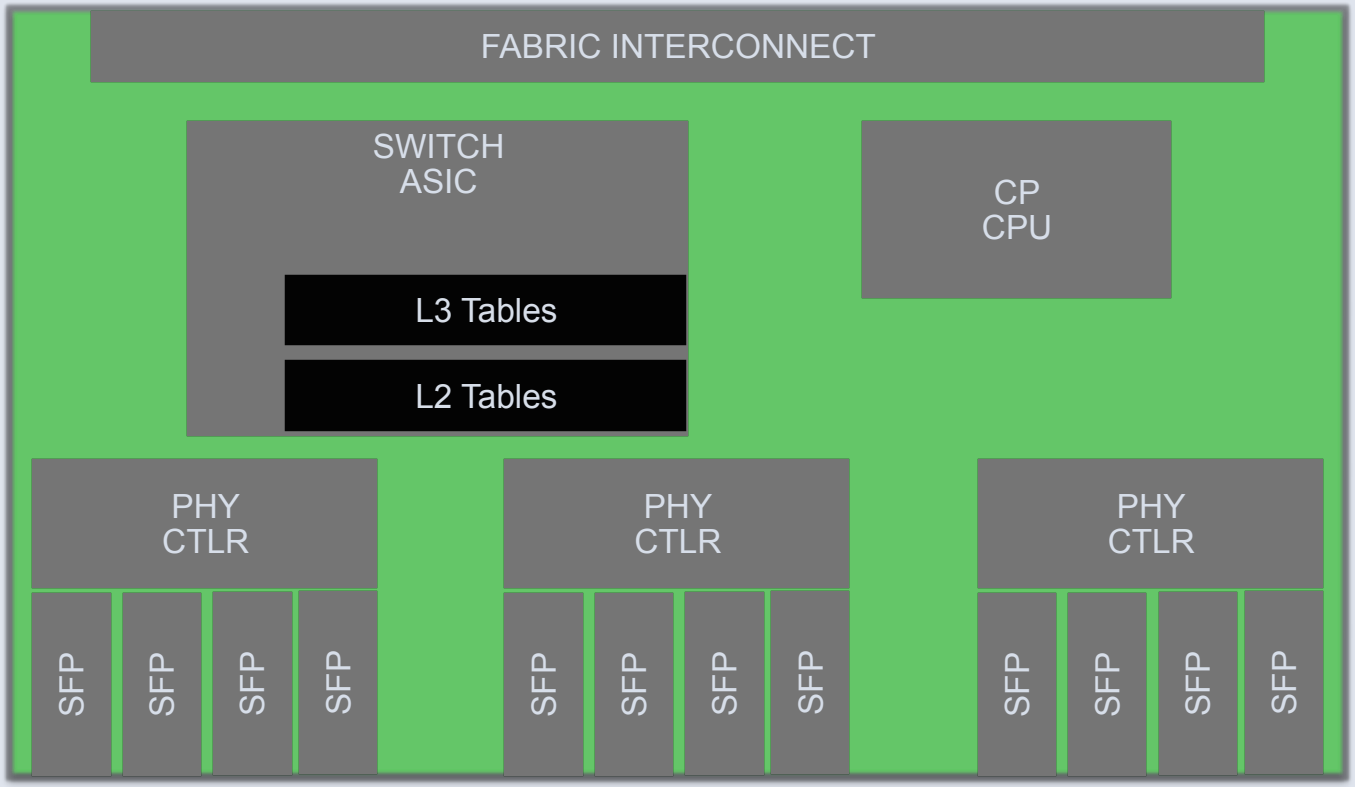
FBAR reacts to alarms

IX peering



For more information, see “Making Facebook Self-Healing”
https://www.facebook.com/note.php?note_id=10150275248698920

understanding the black box

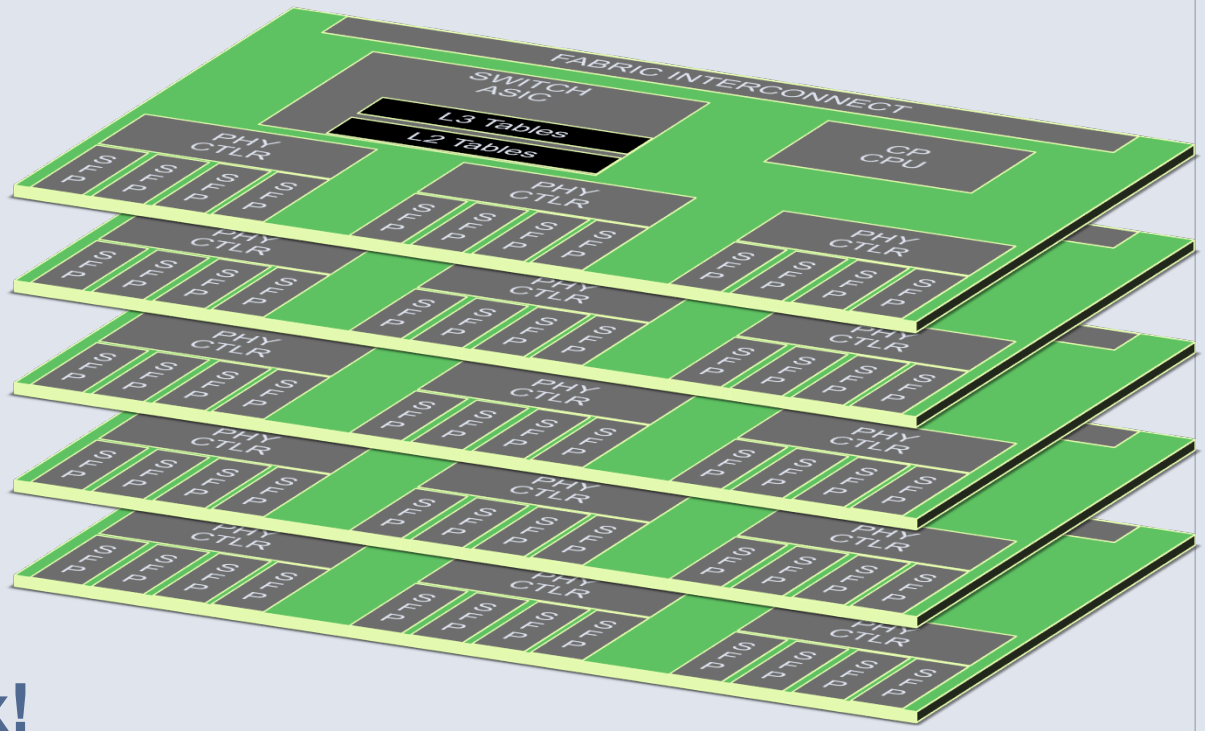


troubleshooting the black box

when a line-card goes crazy

how do you even know about it?

never trust the box!



monitoring everything

all links / BGP sessions

FIBs

TCP retransmit stats

a culture of automation

filter noise in software

automate the repetitive

engineers focus on real problems

our rack switches

the problem

install and forget

configuration drift

inconsistent IPv6 support

IPv6 everywhere! for real!

every rack in 2013

all services in early 2014

why?

- IPv4 won't last forever
- Band-Aids are not fun to troubleshoot

- and it's really cool ☺!

`2a03:2880:2110:df07:face:b00c:0:1`

rolling out IPv6

dual-stacked backbone
and cluster switches

rack switch upgrades

service migration

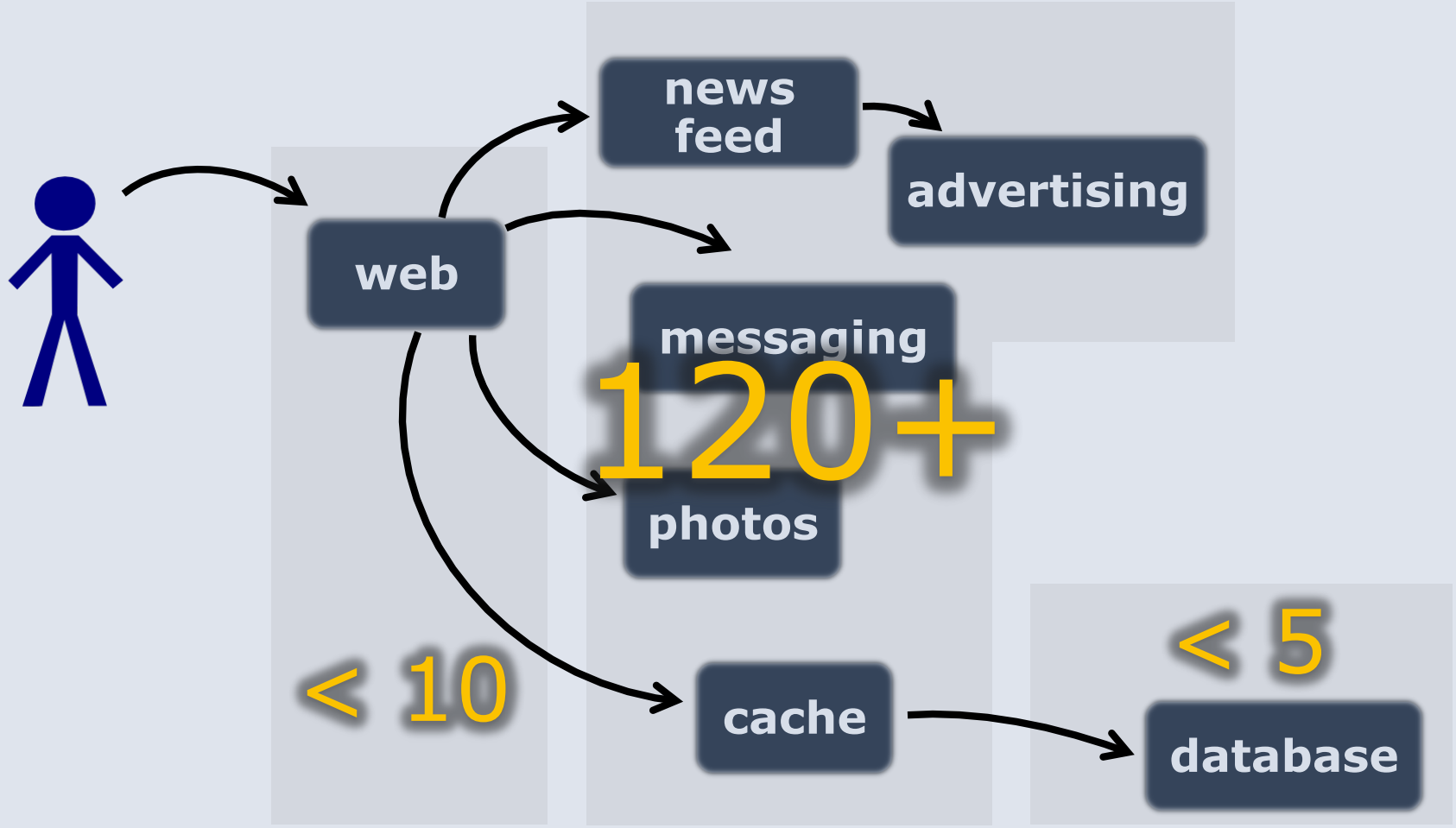
the old way to upgrade a cluster

coordinate an outage window with affected service owners

drain traffic

upgrade

the difficulty – lots of service owners



we needed a change....

why drain?

why a single window?

why is NetEng so heavily involved?

looking at an early attempt

blacklist racks by hostnames,

upgrade the rest

how we solved it

dedicated racks?

- shift the responsibility to the service owner

shared racks?

- schedule every rack under full automation

dedicated racks

shift the responsibility?

- real world – less time on all sides!
- empowered service owners – moved from user to customer
  - **we're friends now!**

focused on a smooth user experience

single button upgrades

detailed reporting from the start

easy job management

shared racks

schedule every rack?

- accurate timing and notification
- some services need to be drained
- why not? software will do the work anyway

where are we now?

service owners handling

rack switch upgrades!

a network that constantly upgrades itself!

how? Aggiornamento!

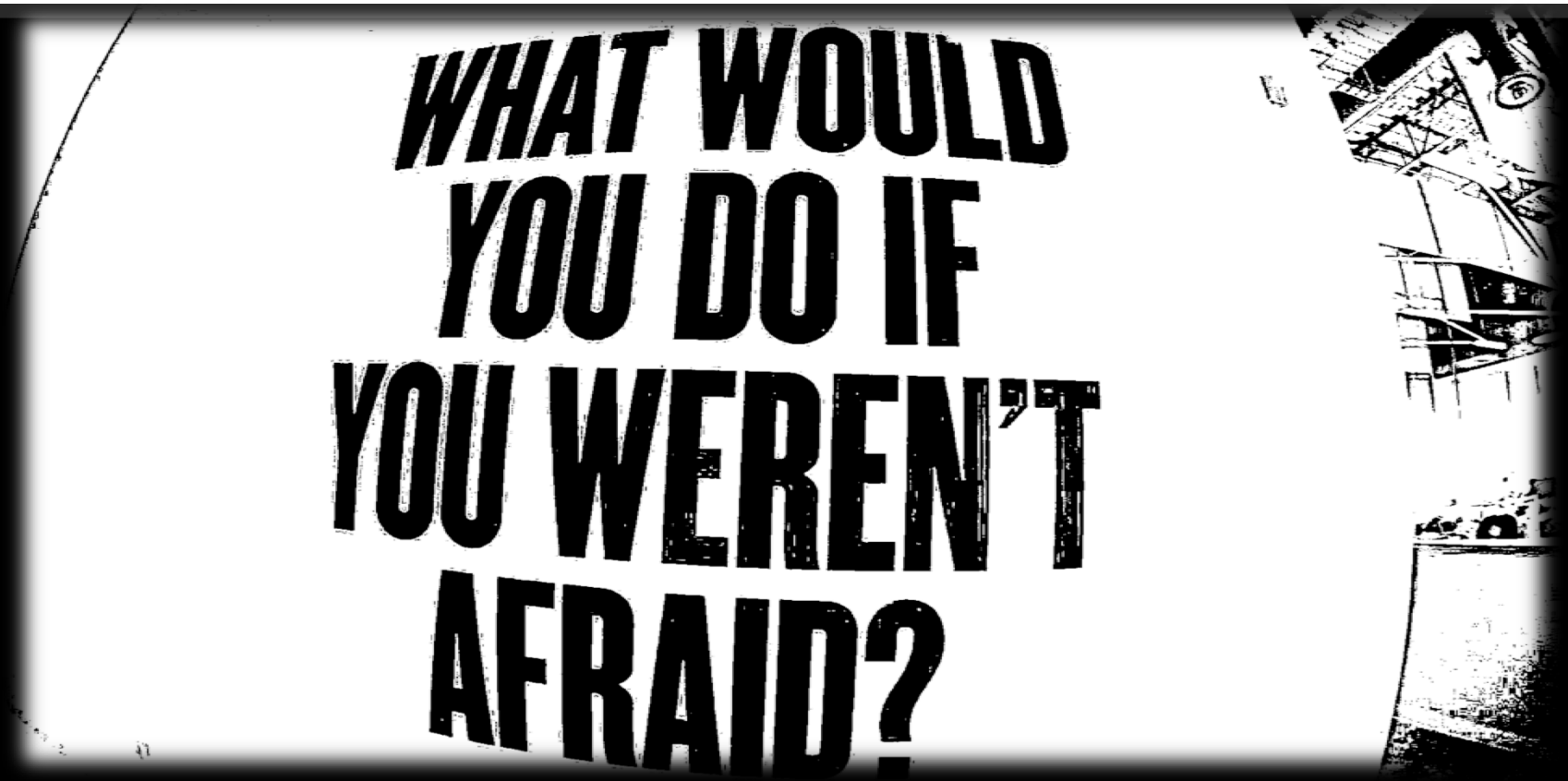
client / server model based on Thrift

runs on Linux, written in Python,

backed by MySQL

integrates across all internal systems:

- impact analysis and notification
- scheduling
- job management



automate your job!

fate is on the way

impossible!