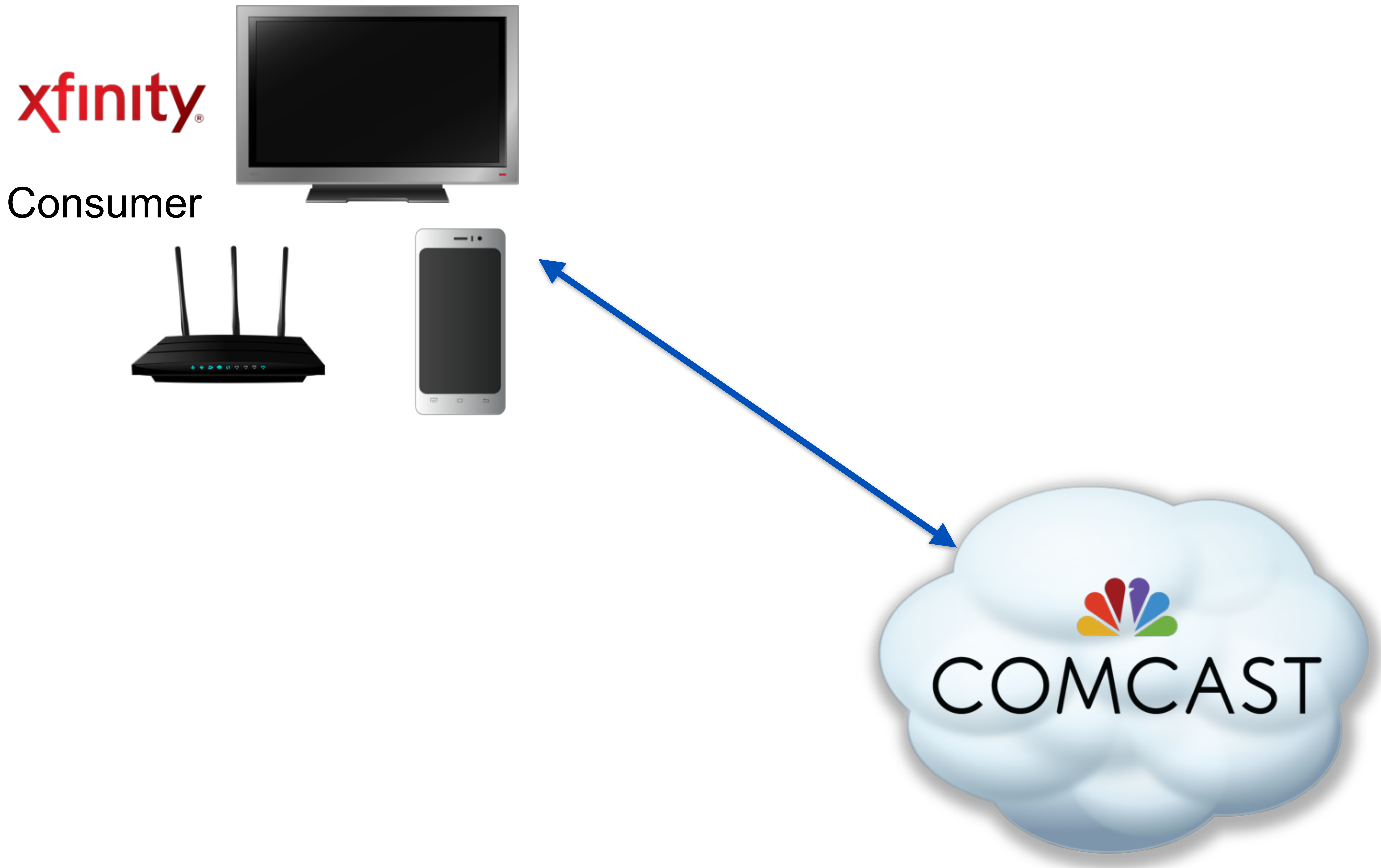
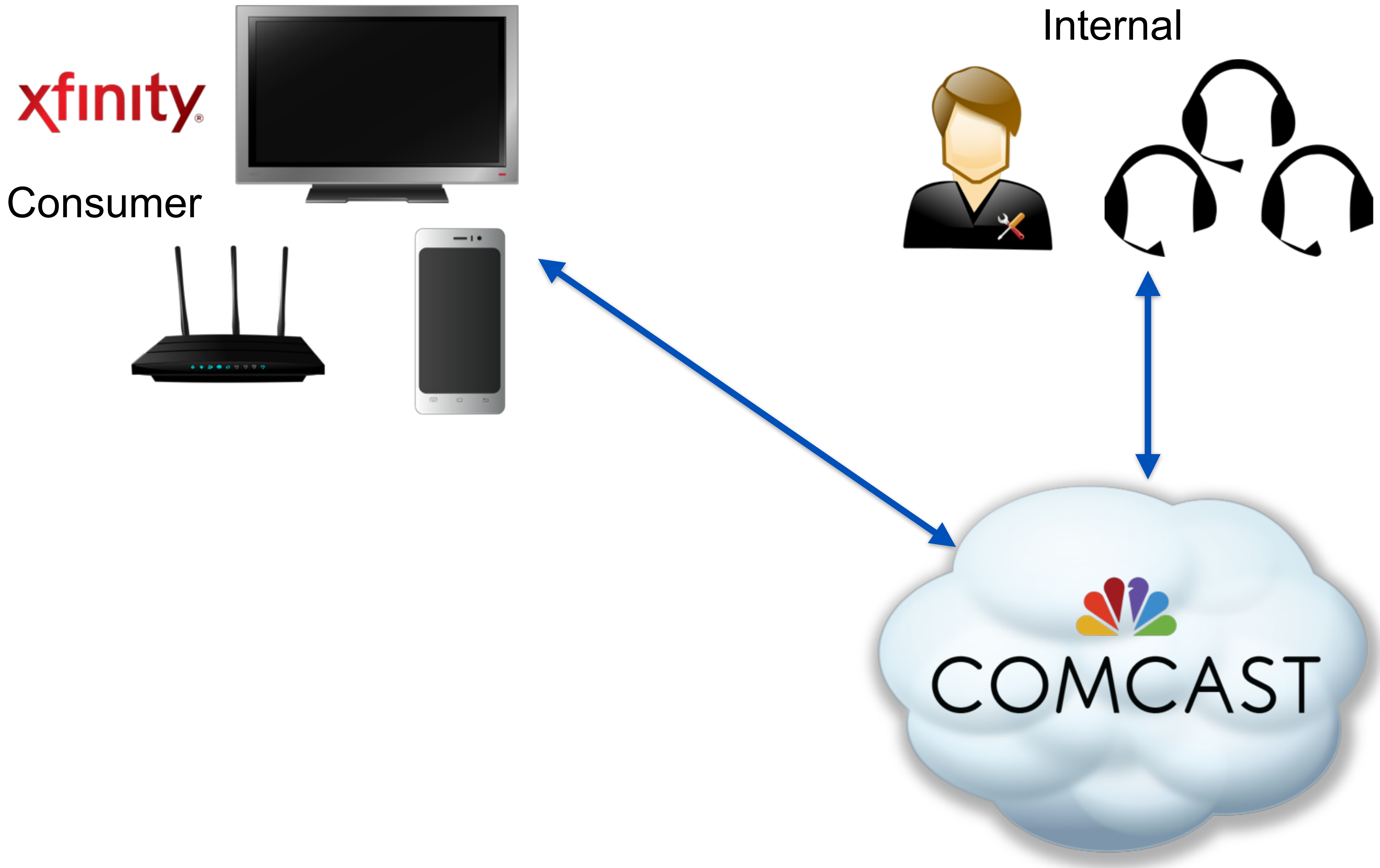
Reliable High-Performance HTTP Infrastructure with nginx and Lua

Sean Cribbs
Senior Principal Engineer, Comcast Cable
@seancribbs

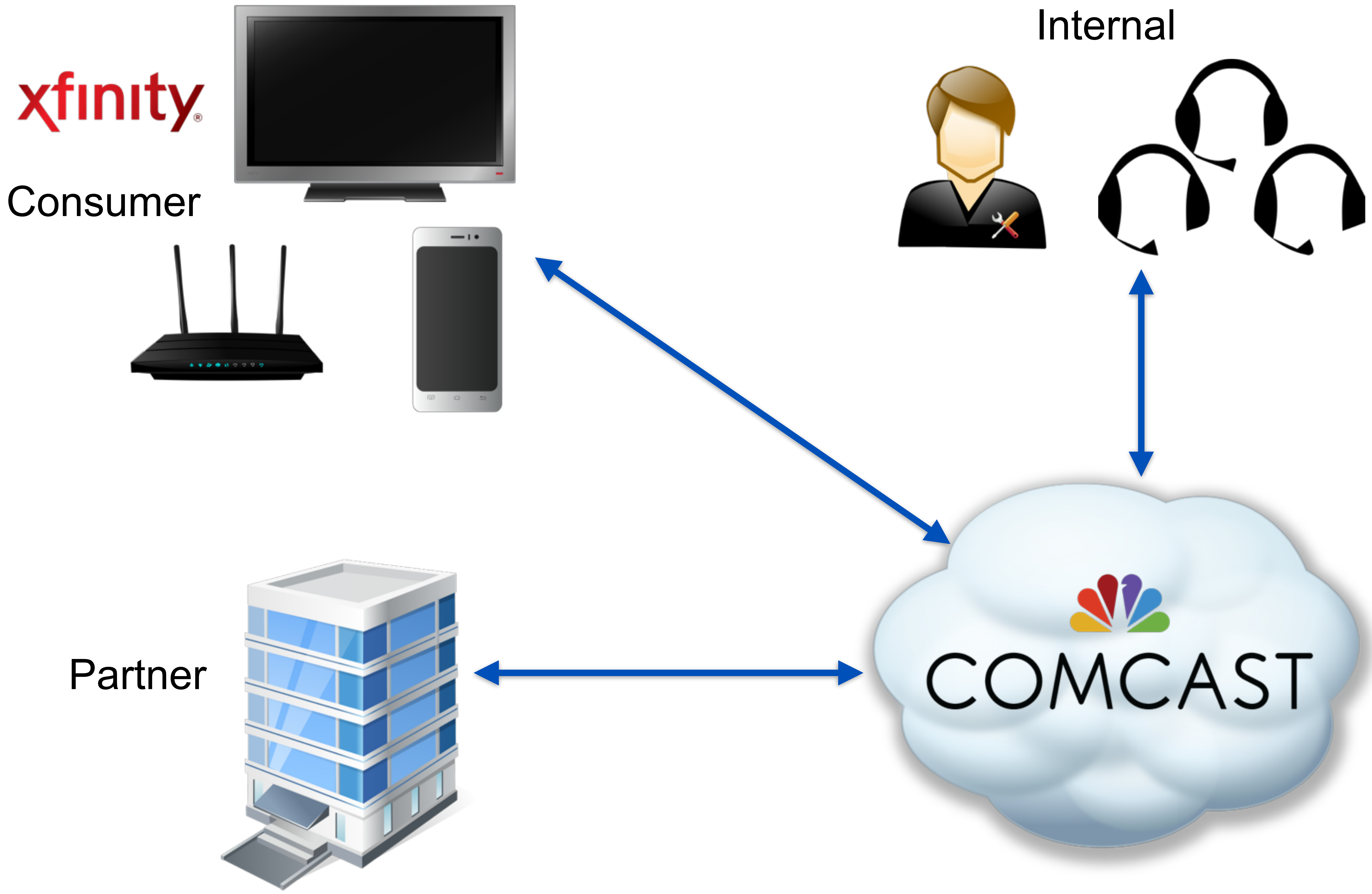
Background

COMCAST


xfinity
Consumer
COMCAST



COMCAST


xfinity
Consumer
Internal
Partner
COMCAST

API Management

API Management

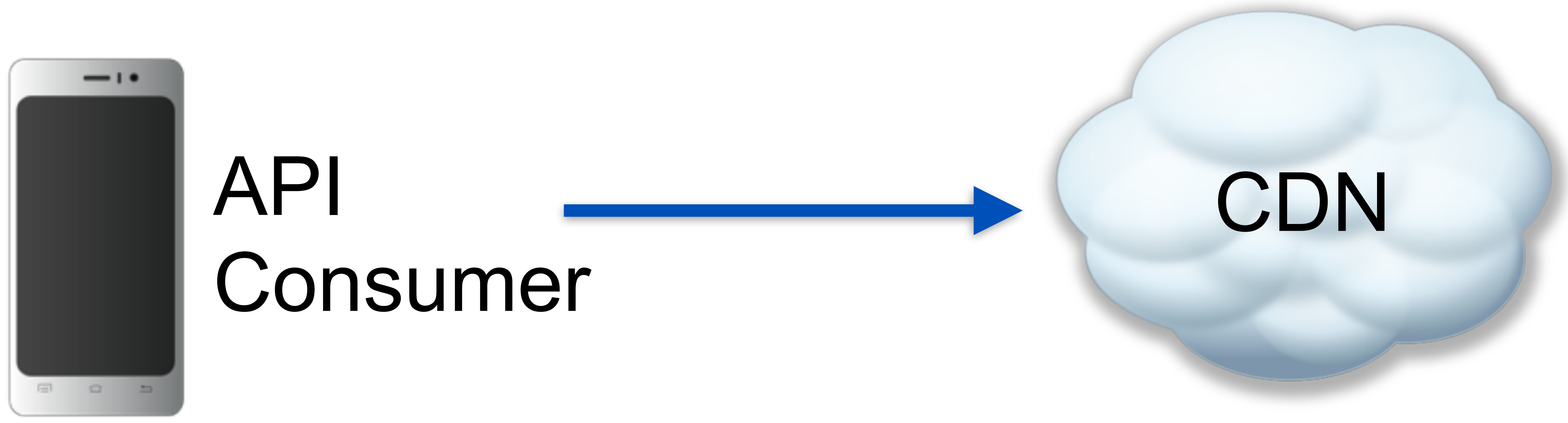
access control

capacity management

CodeBig 1

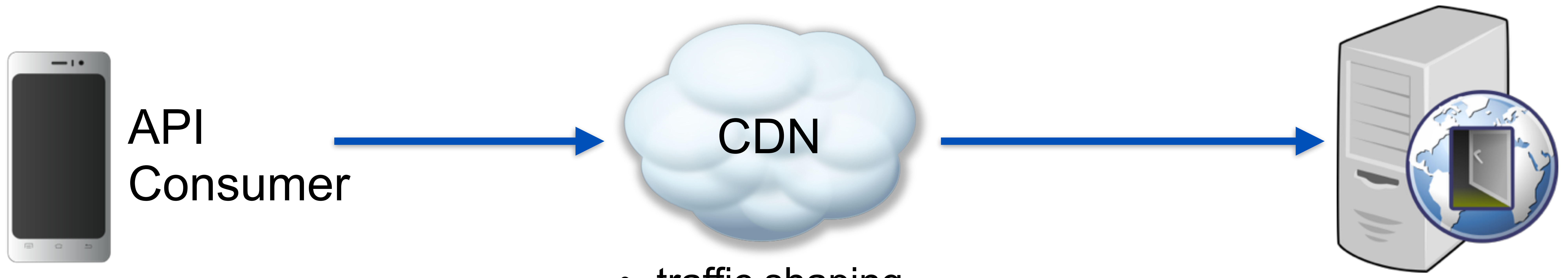
API
Consumer

CodeBig 1



- traffic shaping
- caching

CodeBig 1

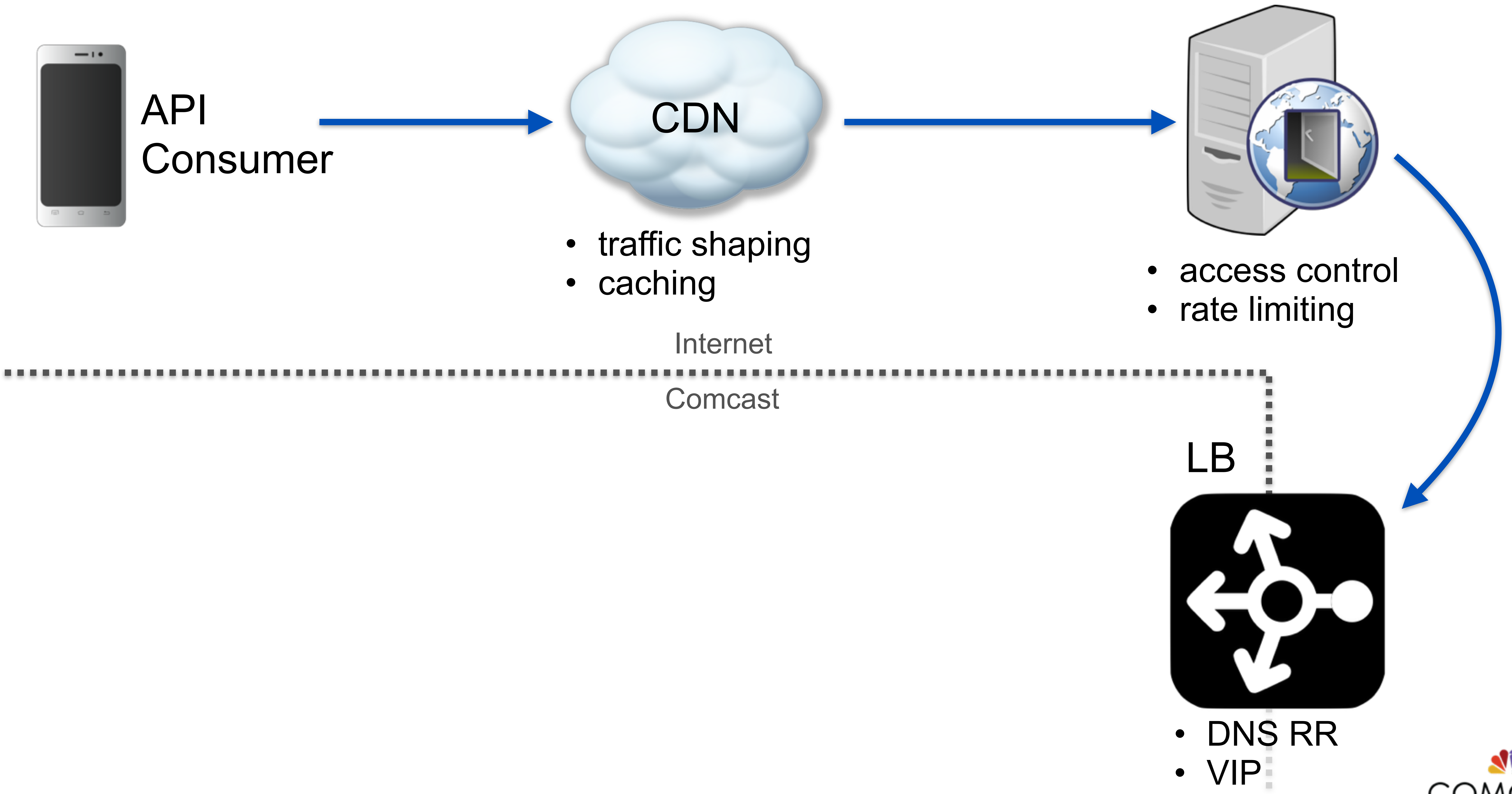


- traffic shaping
- caching

- access control
- rate limiting

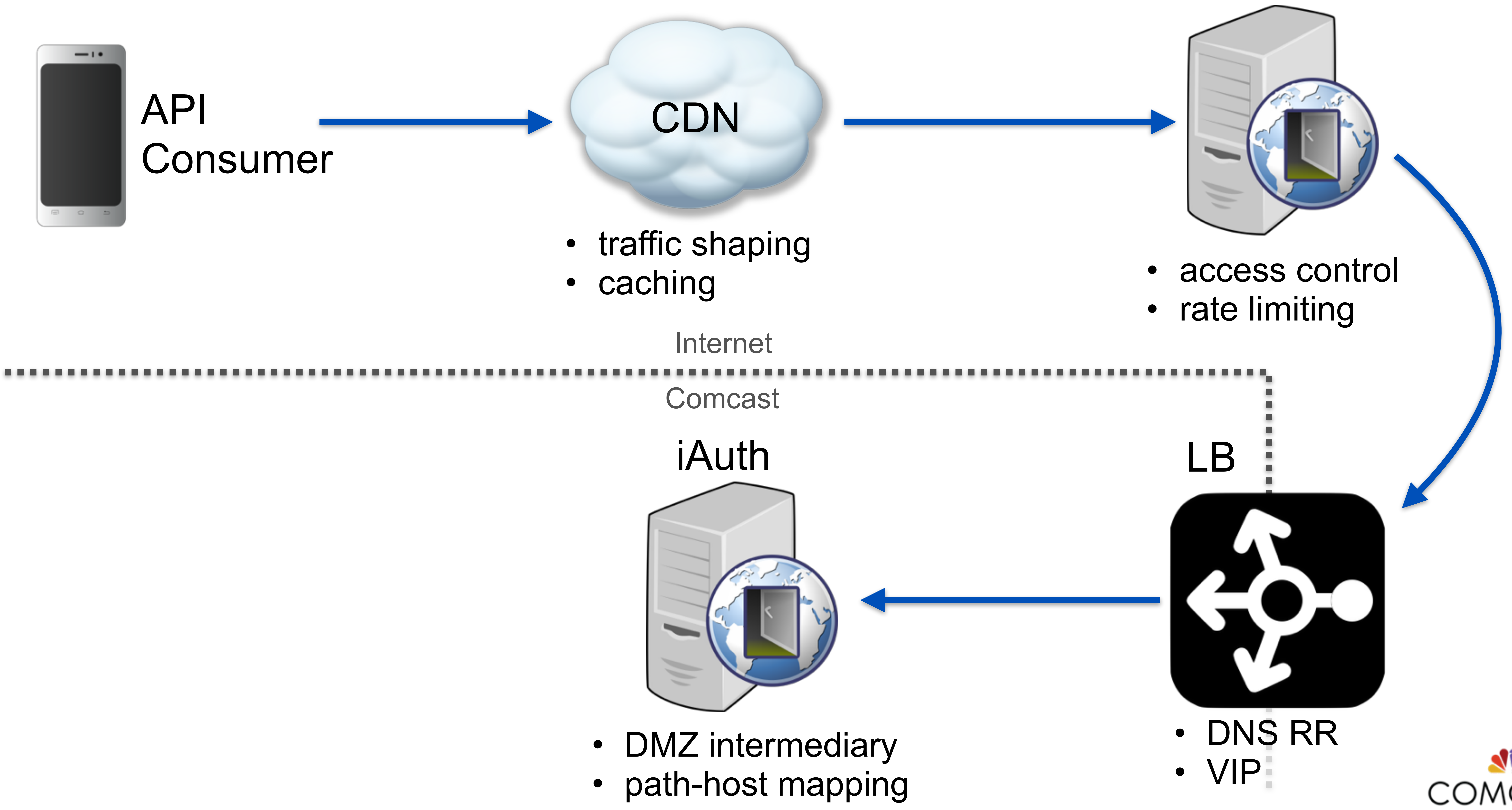
COMCAST

CodeBig 1

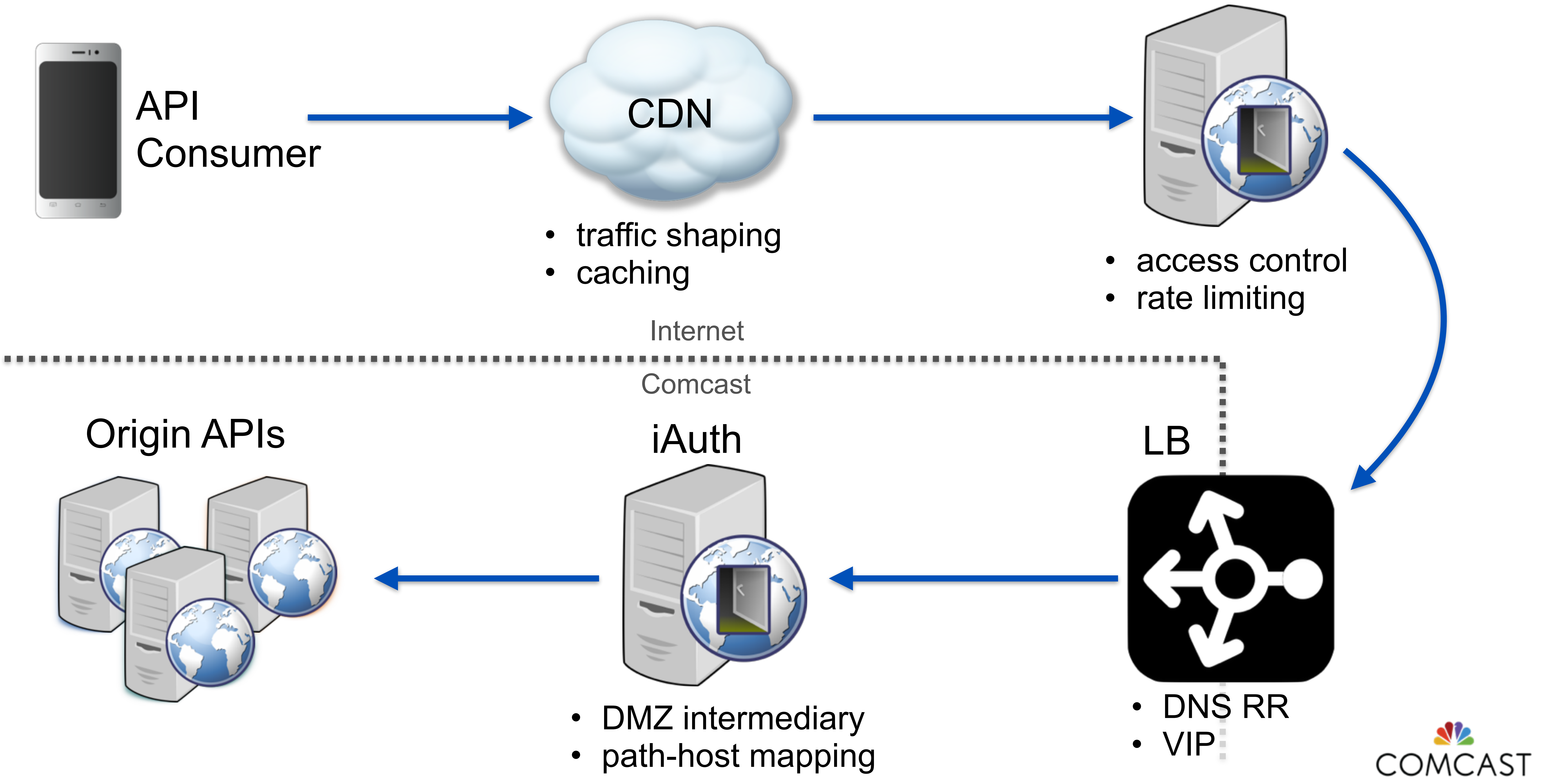


COMCAST

CodeBig 1


API Consumer
CDN
• traffic shaping
• caching
Vendor APIM
• access control
• rate limiting
Internet
Comcast
iAuth
• DMZ intermediary
• path-host mapping
LB
• DNS RR
• VIP

CodeBig 1


API Consumer
CDN
Vendor APIM
traffic shaping
caching
access control
rate limiting
Internet
Comcast
Origin APIs
iAuth
LB
DMZ intermediary
path-host mapping
DNS RR
VIP
COMCAST

Challenges

Challenges

visibility

Challenges

visibility
responsibility

Challenges

visibility
responsibility
scope

Challenges

visibility
responsibility
scope
latency

Challenges

visibility
responsibility
scope
latency
security

CodeBig 2

CodeBig 2

simplify architecture

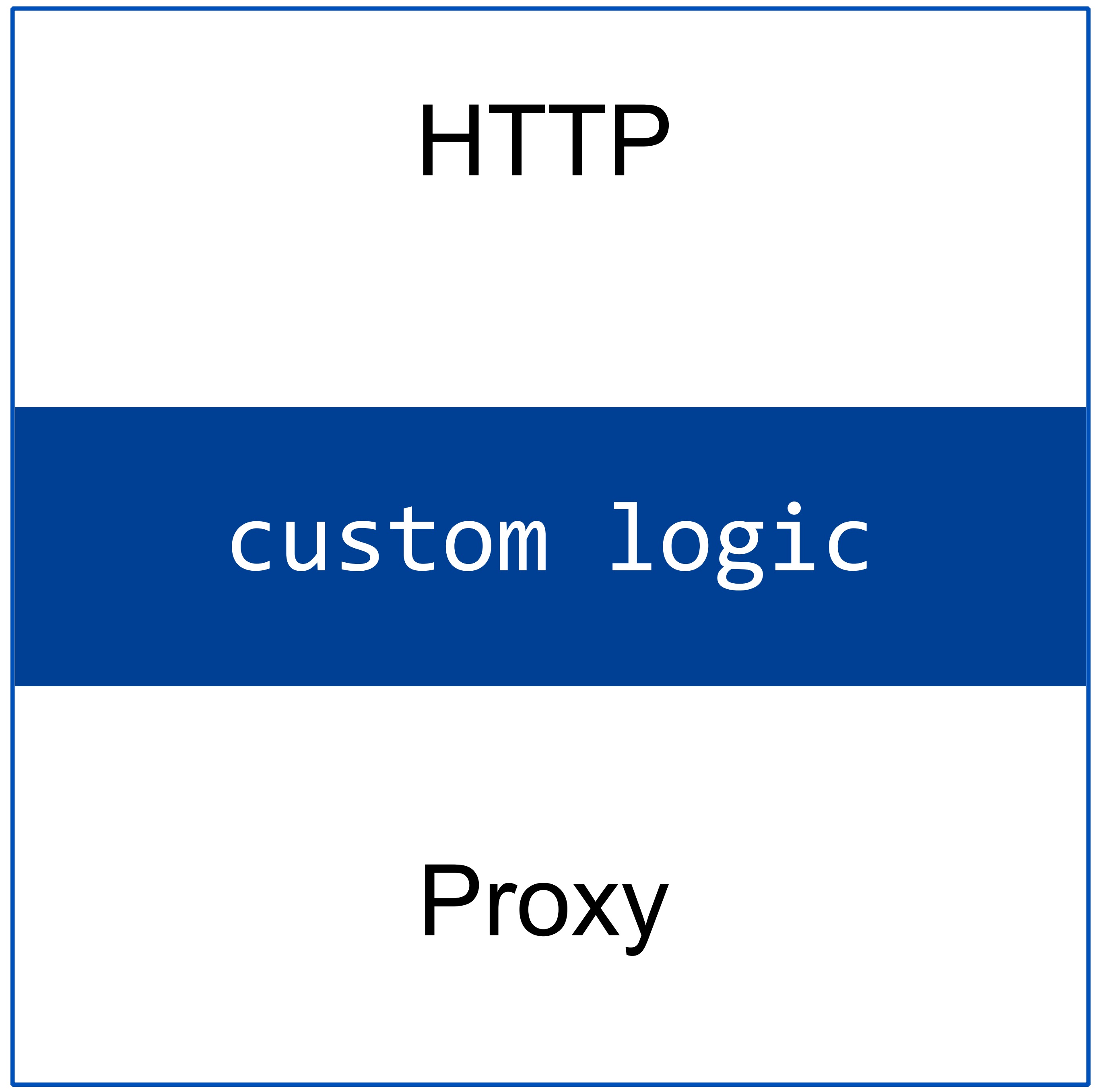
CodeBig 2

simplify architecture
increase visibility

CodeBig 2

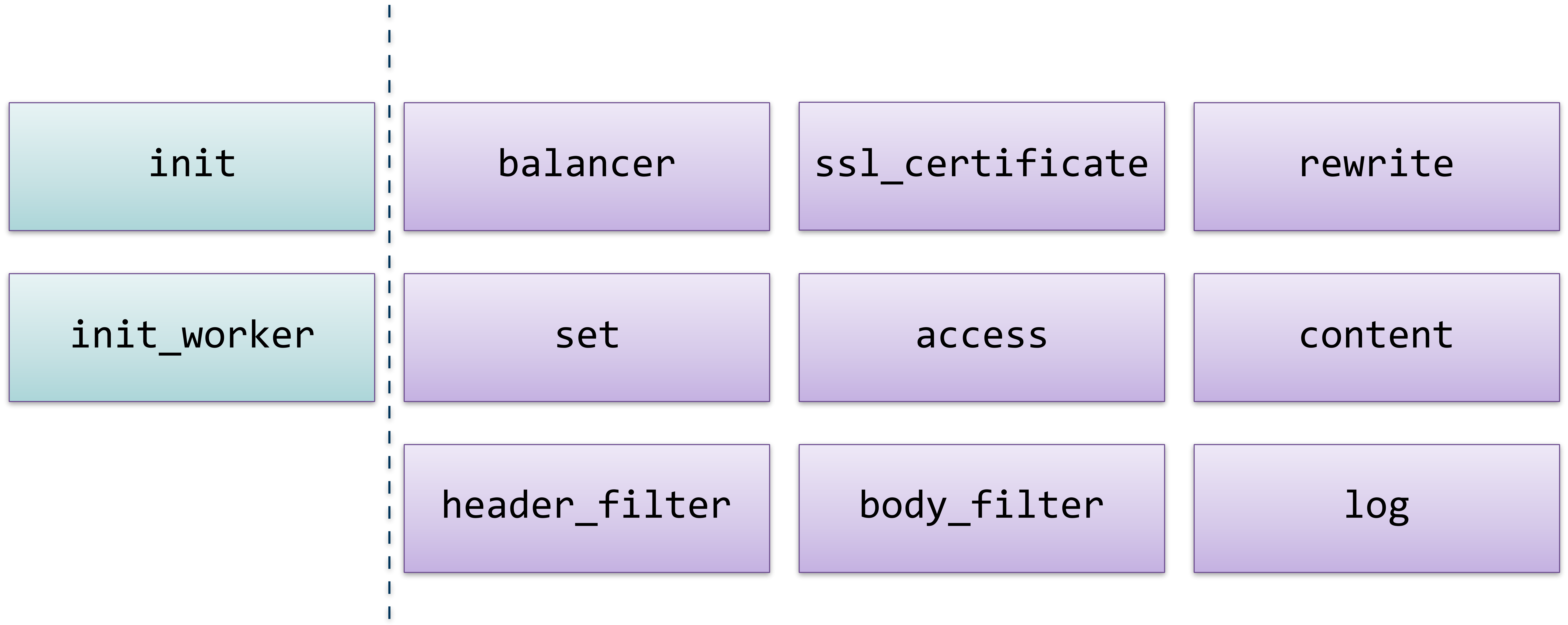
simplify architecture
increase visibility
use open-source tools

Architecture


HTTP
custom logic
Proxy

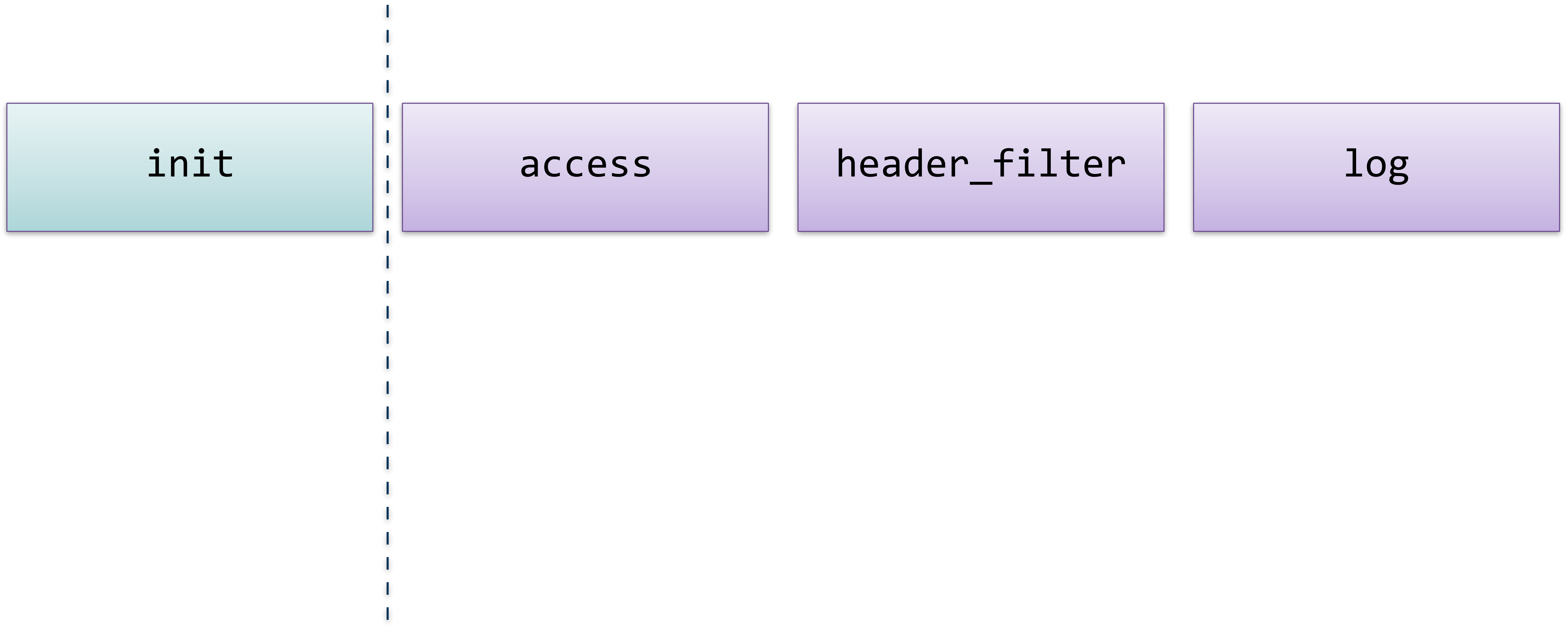
NGINX
Lua

nginx+Lua extension points



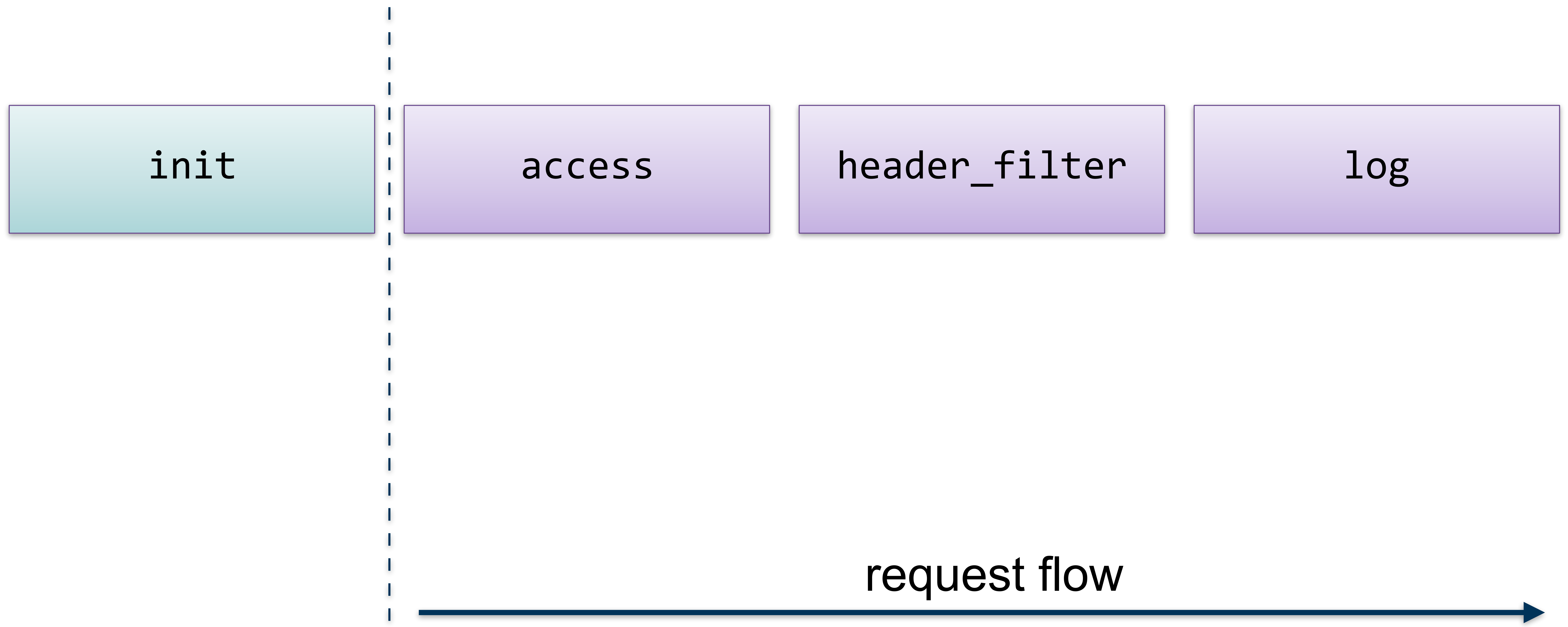
+ _by_lua
_by_lua_file
_by_lua_block

CodeBig Request Phases

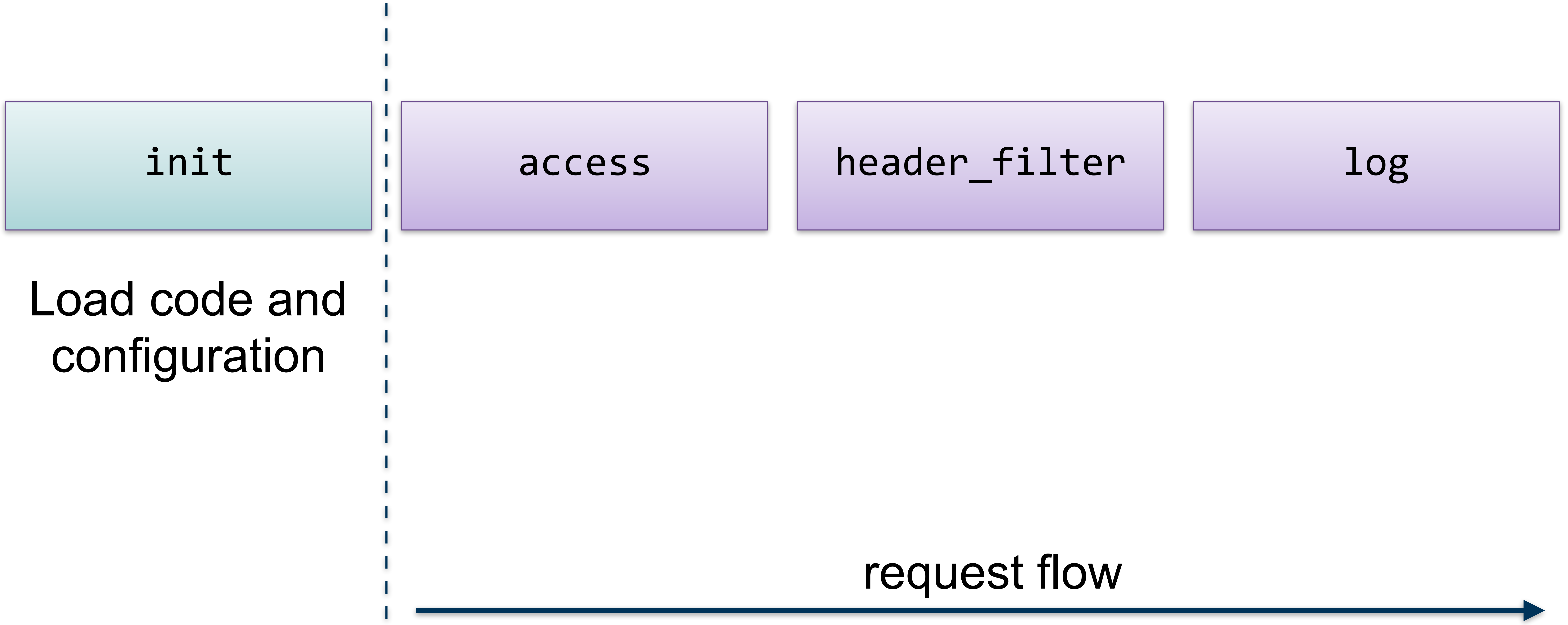


COMCAST

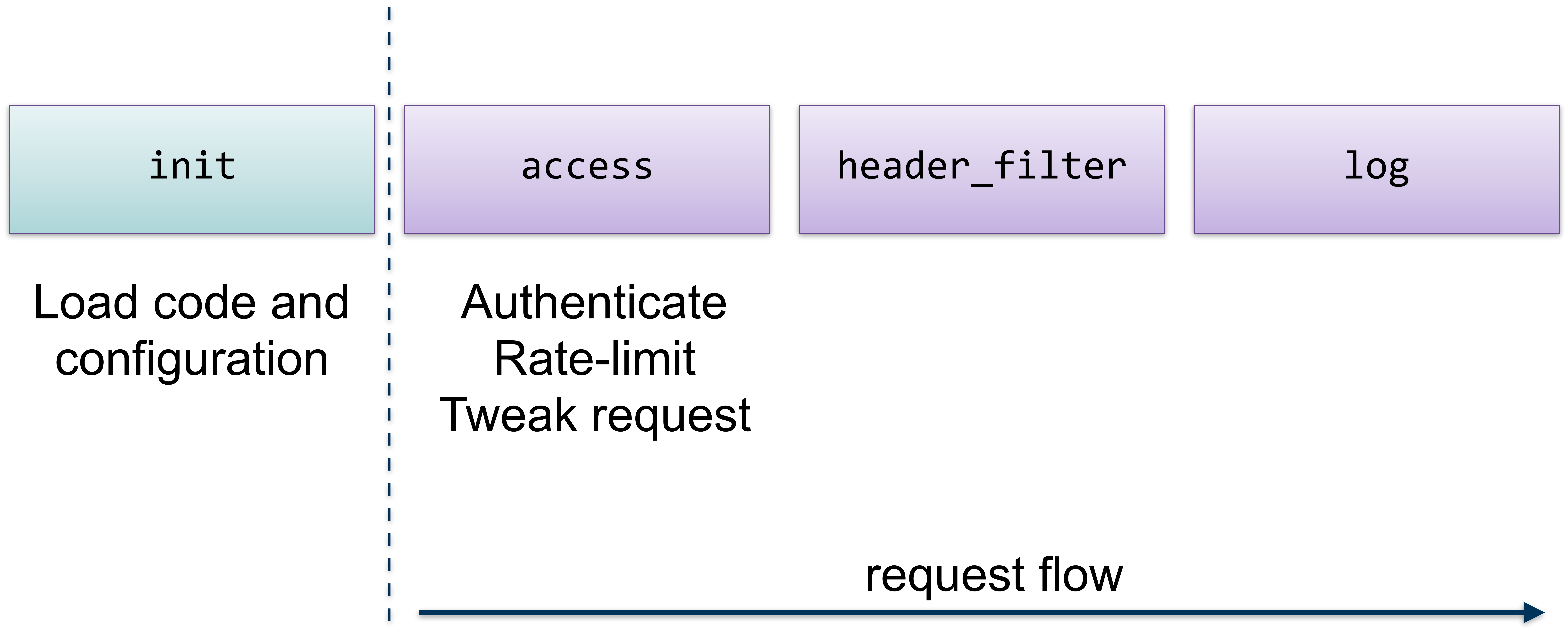
CodeBig Request Phases



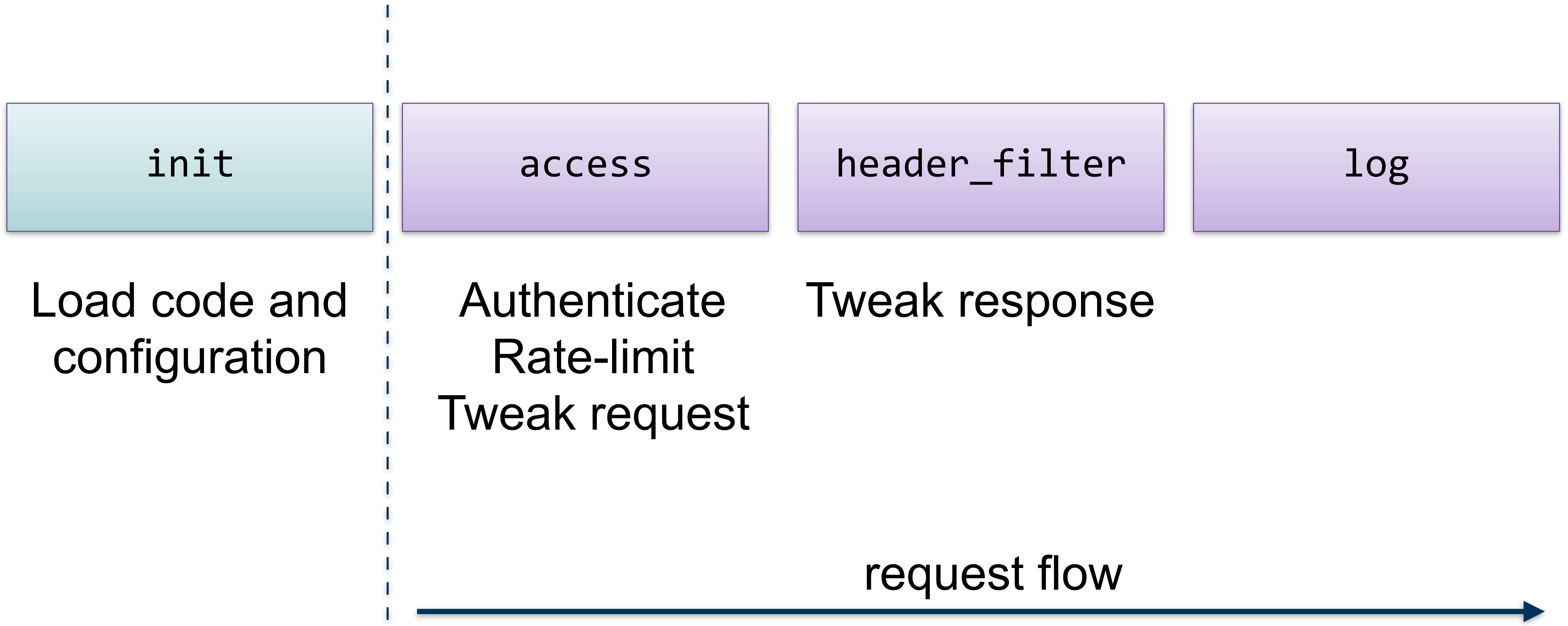
CodeBig Request Phases



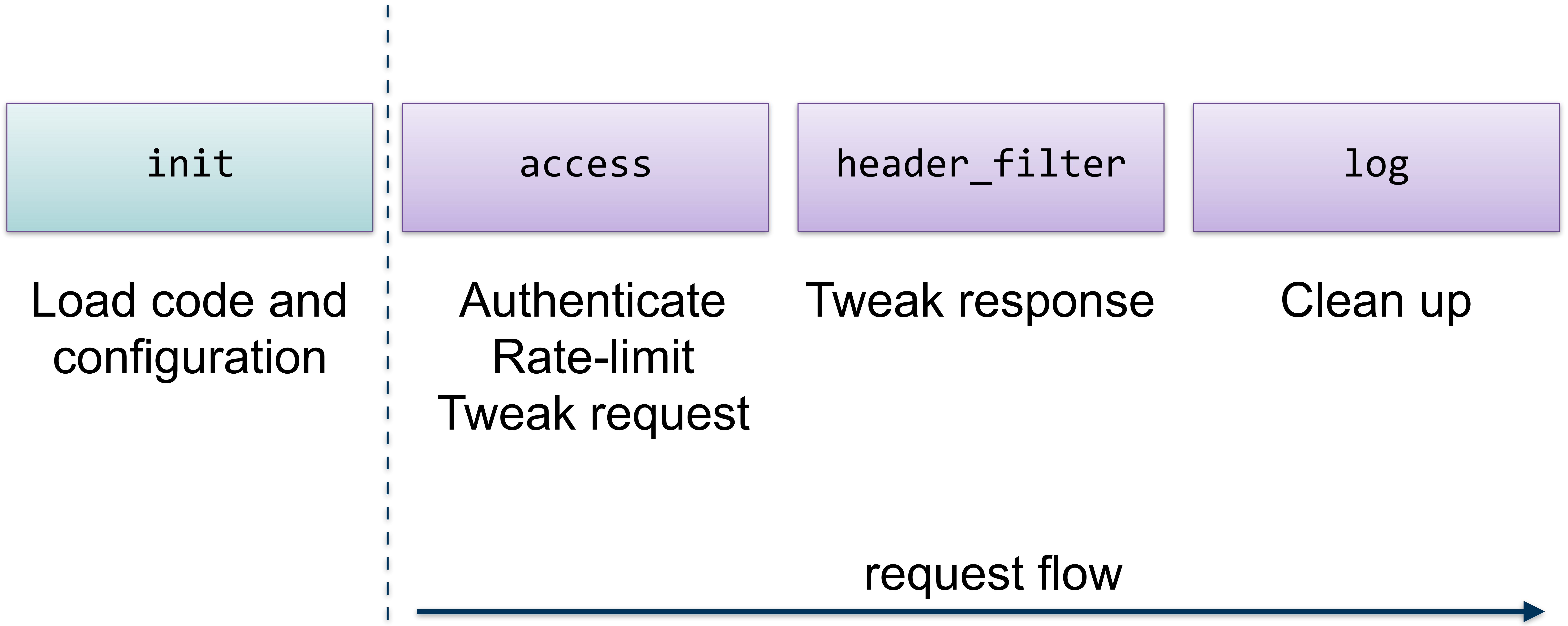
CodeBig Request Phases



CodeBig Request Phases



CodeBig Request Phases



```lua
local setmetatable = setmetatable

local _M = {}

function _M:new(ctx, conf)
    local o = {
        _ctx = ctx,
        _conf = conf
    }

    o.super = self
    setmetatable(o, self)
    self.__index = self
    return o
end

function _M:access()
    return true
end
```

```lua
function _M:post_access()
    -- nop
end

function _M:header_filter()
    -- nop
end

function _M:log()
    -- nop
end

return _M
```

COMCAST

```lua
for _, name in ipairs(conf.plugins) do
    -- load plugin by fully qualified name
    local plugin = require(name):new(ctx, conf)

    -- exit immediately upon first rejection
    local is_ok, err = plugin:access()
    if not is_ok then
        ngx.status = err.code
        ngx_say(err.error)
        ngx.var.access_error = err.error
        return ngx_exit(ngx.HTTP_OK)
    end

    insert(plugins, plugin)
end

for _, plugin in ipairs(plugins) do
    plugin:post_access()
end
```

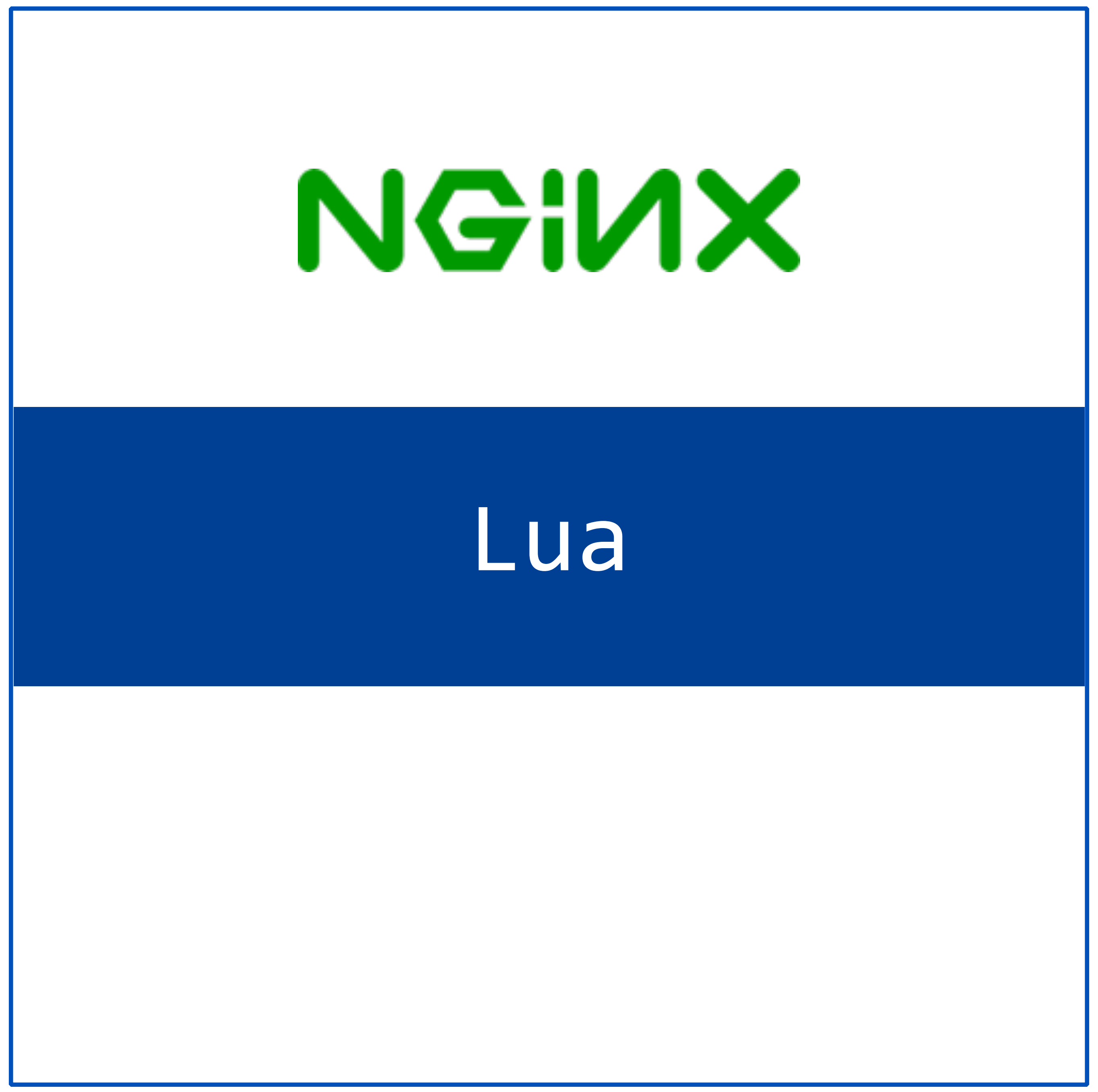
COMCAST

```lua
function _M.header_filter()
    local plugins = ngx.ctx.plugins or {}

    for _, plugin in ipairs(plugins) do
        plugin:header_filter()
    end
end
```

COMCAST

```
# nginx.conf
  lua_package_path      '/usr/share/?/init.lua;/usr/share/?.lua;;';

  lua_shared_dict       memory  50M;

  init_by_lua_block         {
    codebig = require("codebig")
    codebig.init(“/path/to/configs“)
  };

# vhost.conf
    location        / {
        access_by_lua 'return codebig.access("somehost")';

        header_filter_by_lua 'return codebig.header_filter()';

        log_by_lua 'return codebig.log()';
    }
```

COMCAST

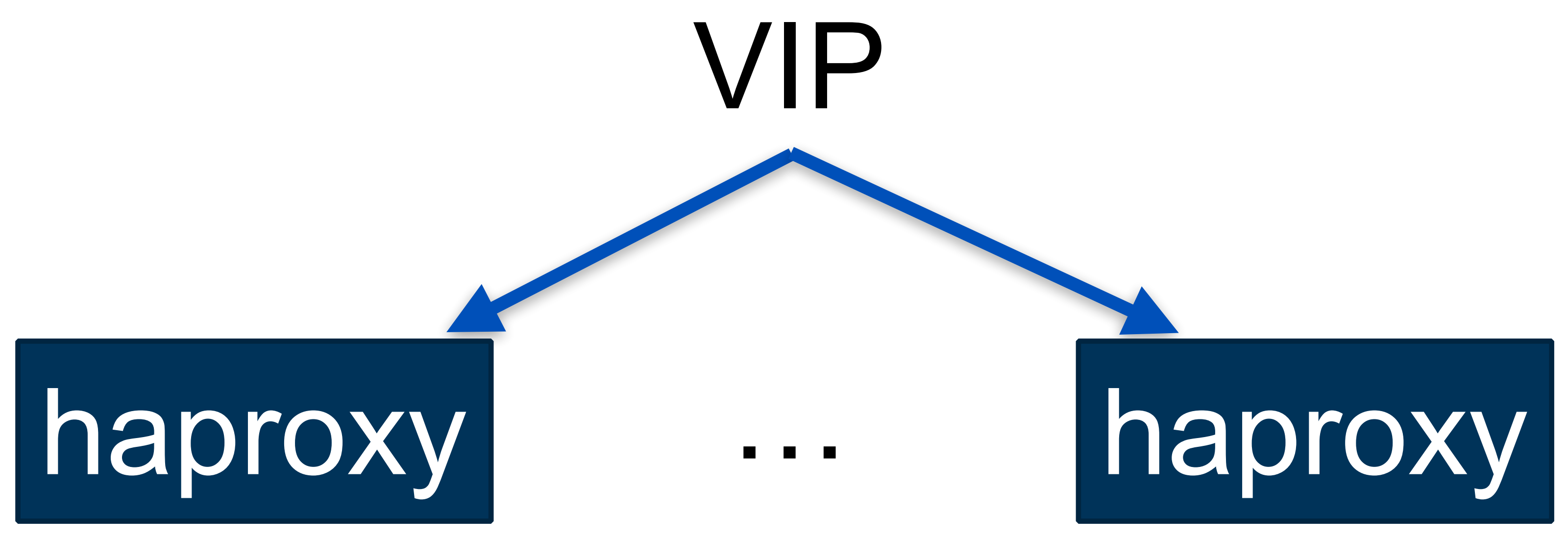


~ 3K LoC!!

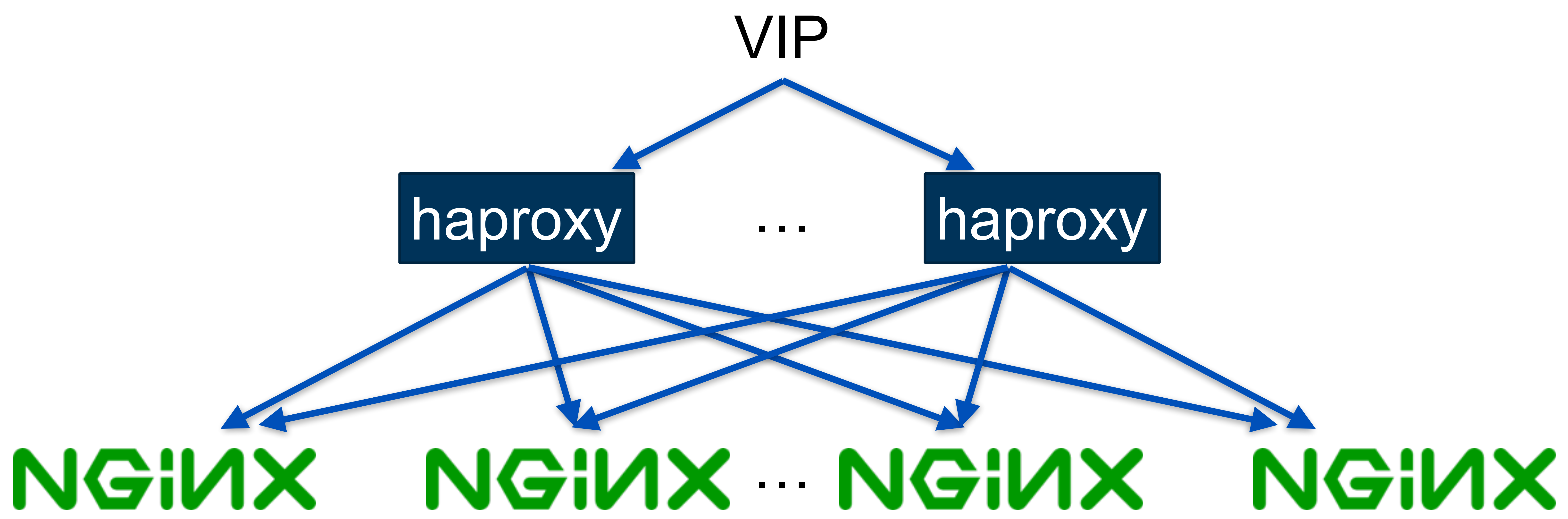
Intra-Datacenter

VIP

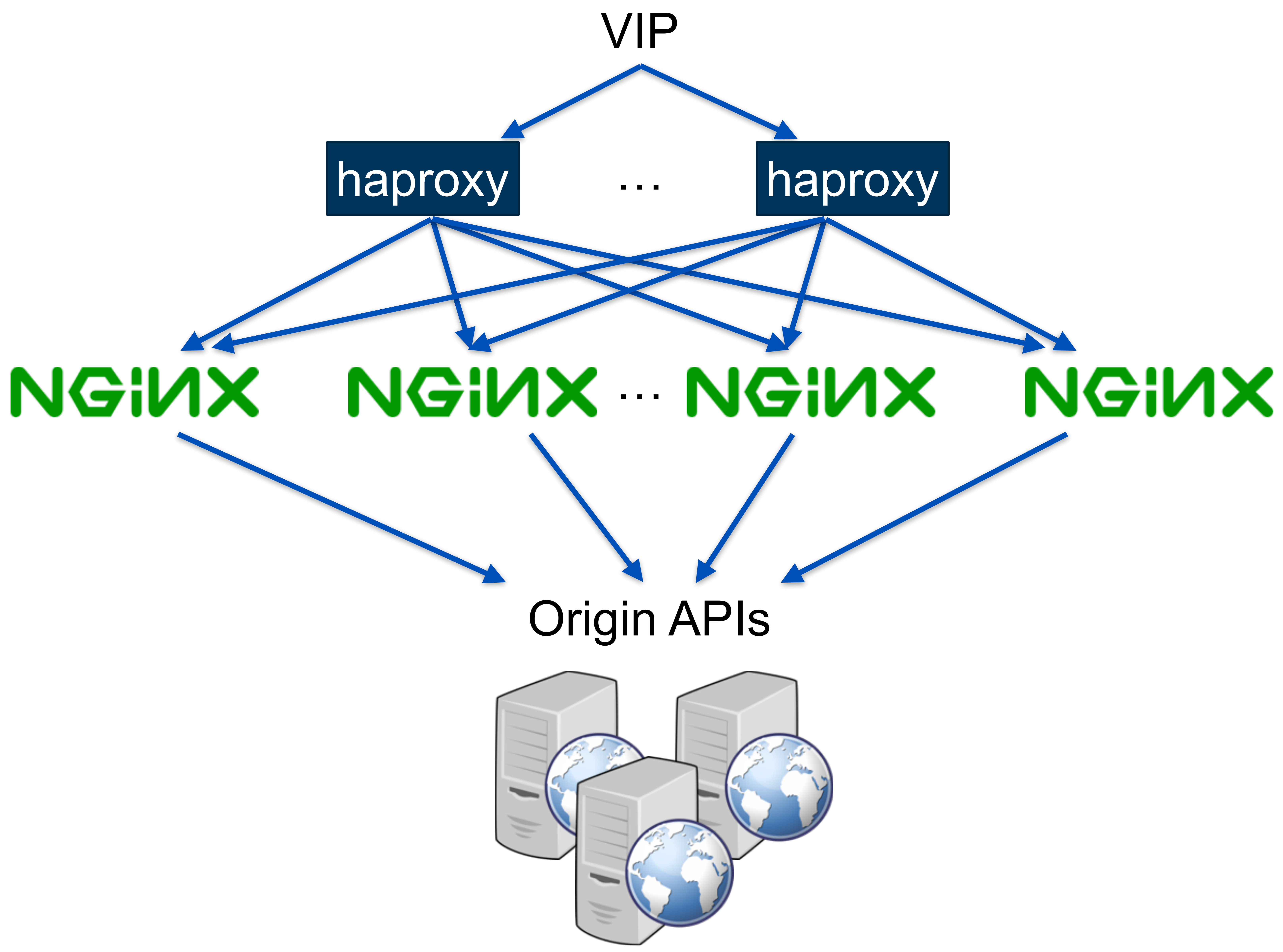
Intra-Datacenter



Intra-Datacenter



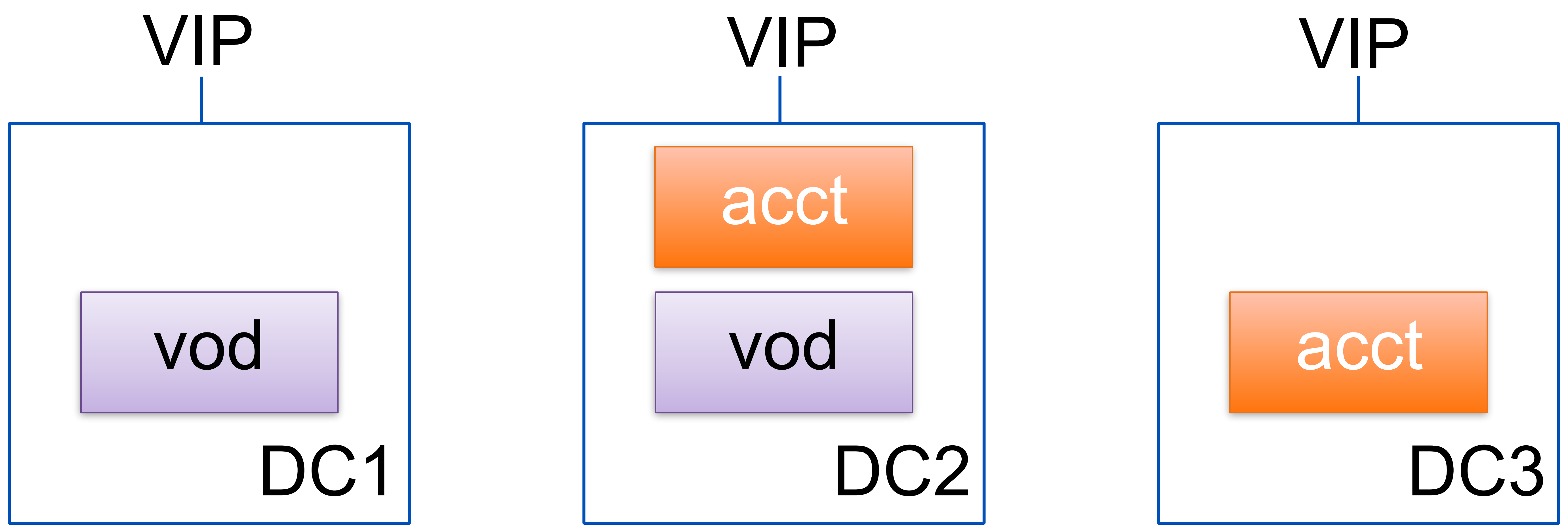
Intra-Datacenter



COMCAST

Cross-Datacenter

```
vod.            CNAME   vod-dc1.
vod-dc1-fo.     CNAME   entry-vip-dc1.
vod-dc1.        CNAME   entry-vip-dc1.
vod-dc2.        CNAME   entry-vip-dc2.
entry-vip-dc1.  A       10.1.0.1
entry-vip-dc2.  A       10.2.0.1
entry-vip-dc3.  A       10.3.0.1
```



Capacity Management

$$N = XR$$

$$\overset{\text{\# concurrent requests}}{\searrow}$$

$$N = XR$$

$$\overset{\text{\# concurrent requests}}{\downarrow}\hspace{-1em}N = \underset{\uparrow \atop \text{transaction rate}}{X}R$$

$$\overset{\text{\# concurrent requests}}{N} = \underset{\text{transaction rate}}{X}\,\underset{\text{response time}}{R}$$

# concurrent
requests

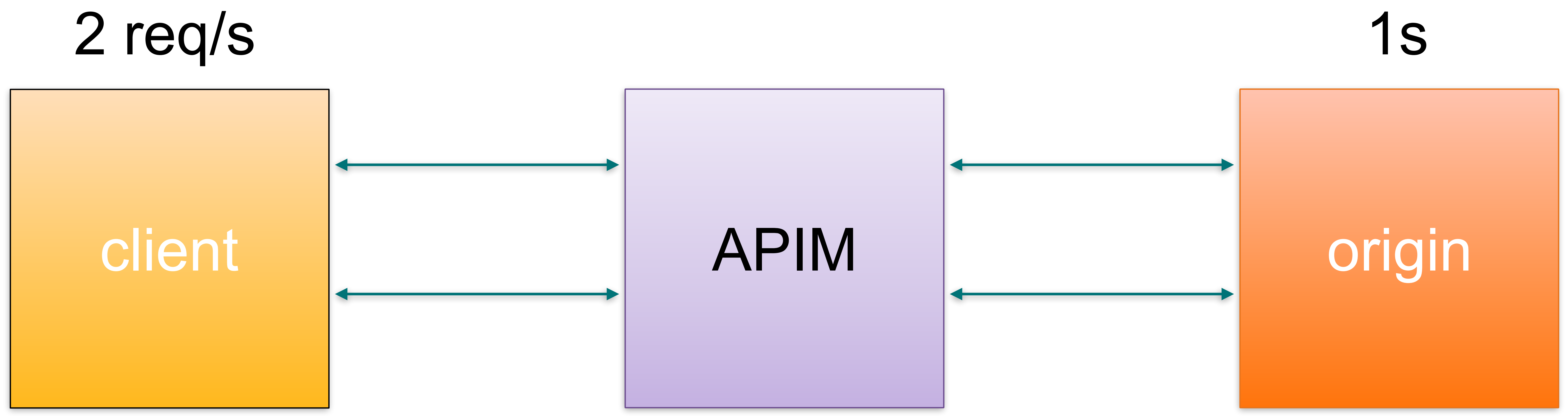
$$N = XR$$

transaction
rate

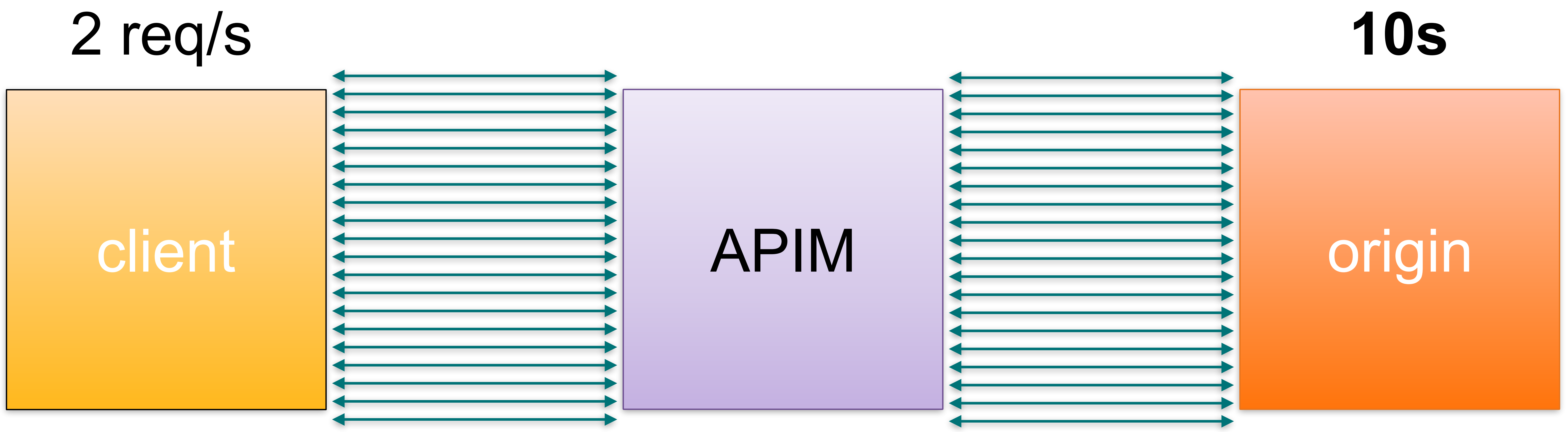
response
time

Little’s Law

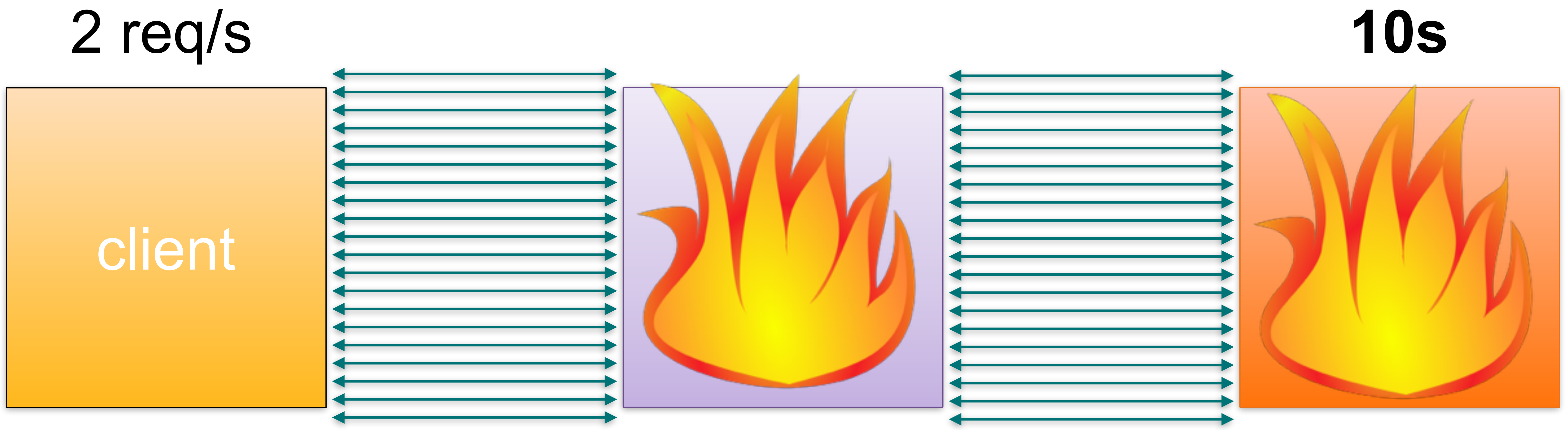
$$N = XR = 2 \text{ req/s x } 1\text{s} = 2 \text{ concurrent}$$



$$N = XR = 2 \text{ req/s} \times \mathbf{10s} = \mathbf{20\ concurrent}$$



$$N = XR = 2 \text{ req/s} \times \mathbf{10s} = \mathbf{20\ concurrent}$$



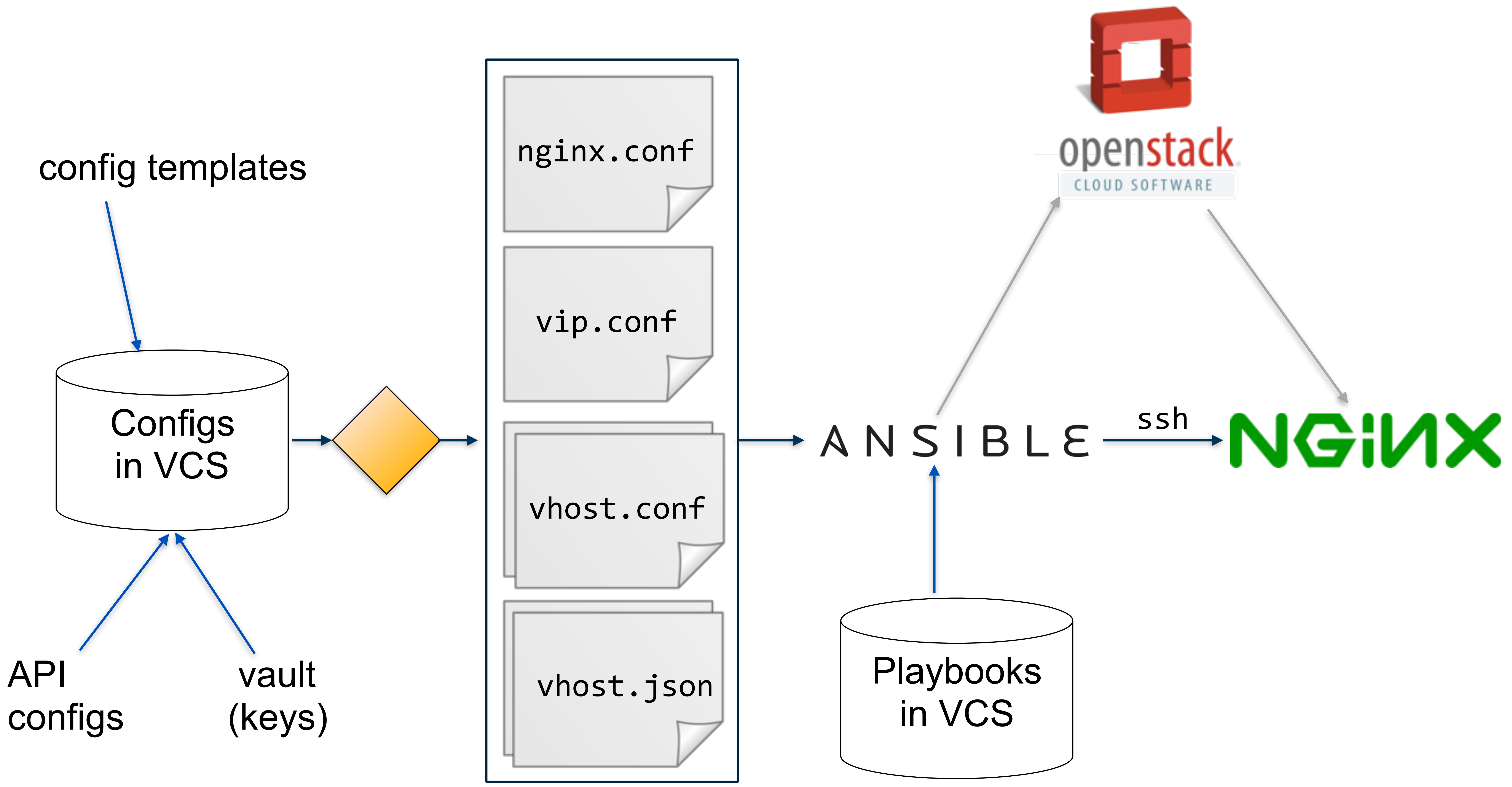
Concurrent Request Limiting

```
lua_shared_dict       memory  50M;

access_by_lua     …   +1

log_by_lua        …   -1
```

Deployment

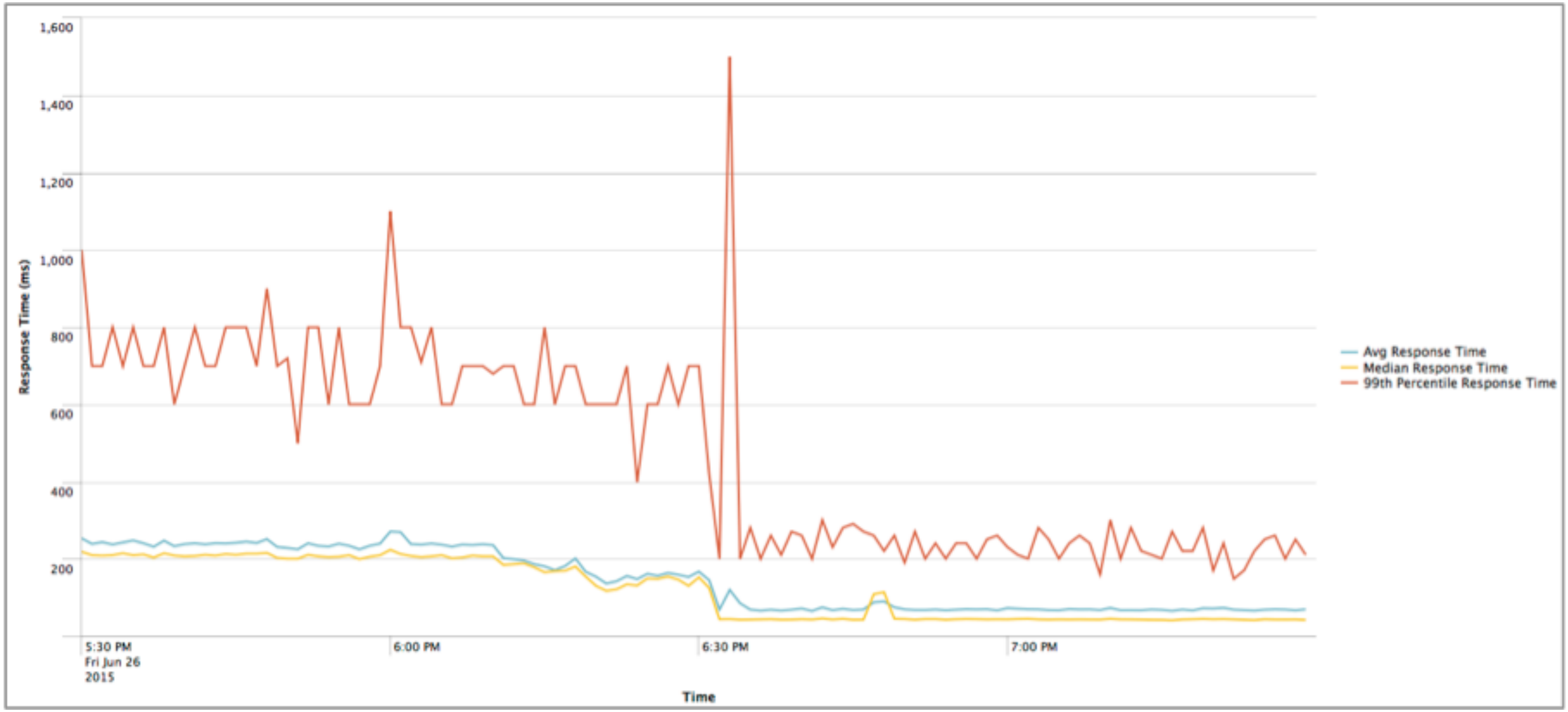
ANSIBLE

COMCAST


config templates
Configs in VCS
API configs
vault (keys)
nginx.conf
vip.conf
vhost.conf
vhost.json
ANSIBLE
Playbooks in VCS
openstack CLOUD SOFTWARE
ssh
NGINX

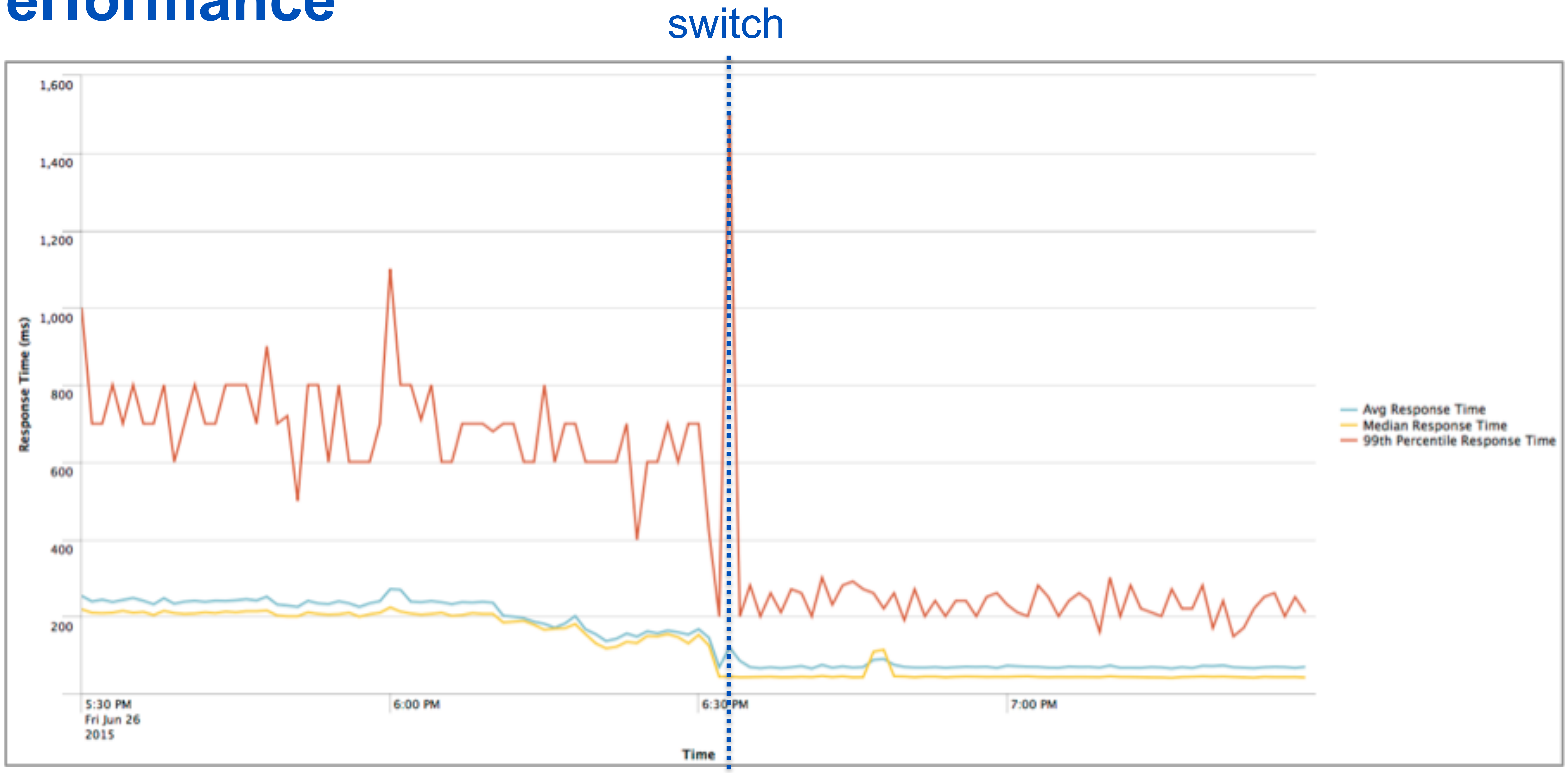
Results

Performance



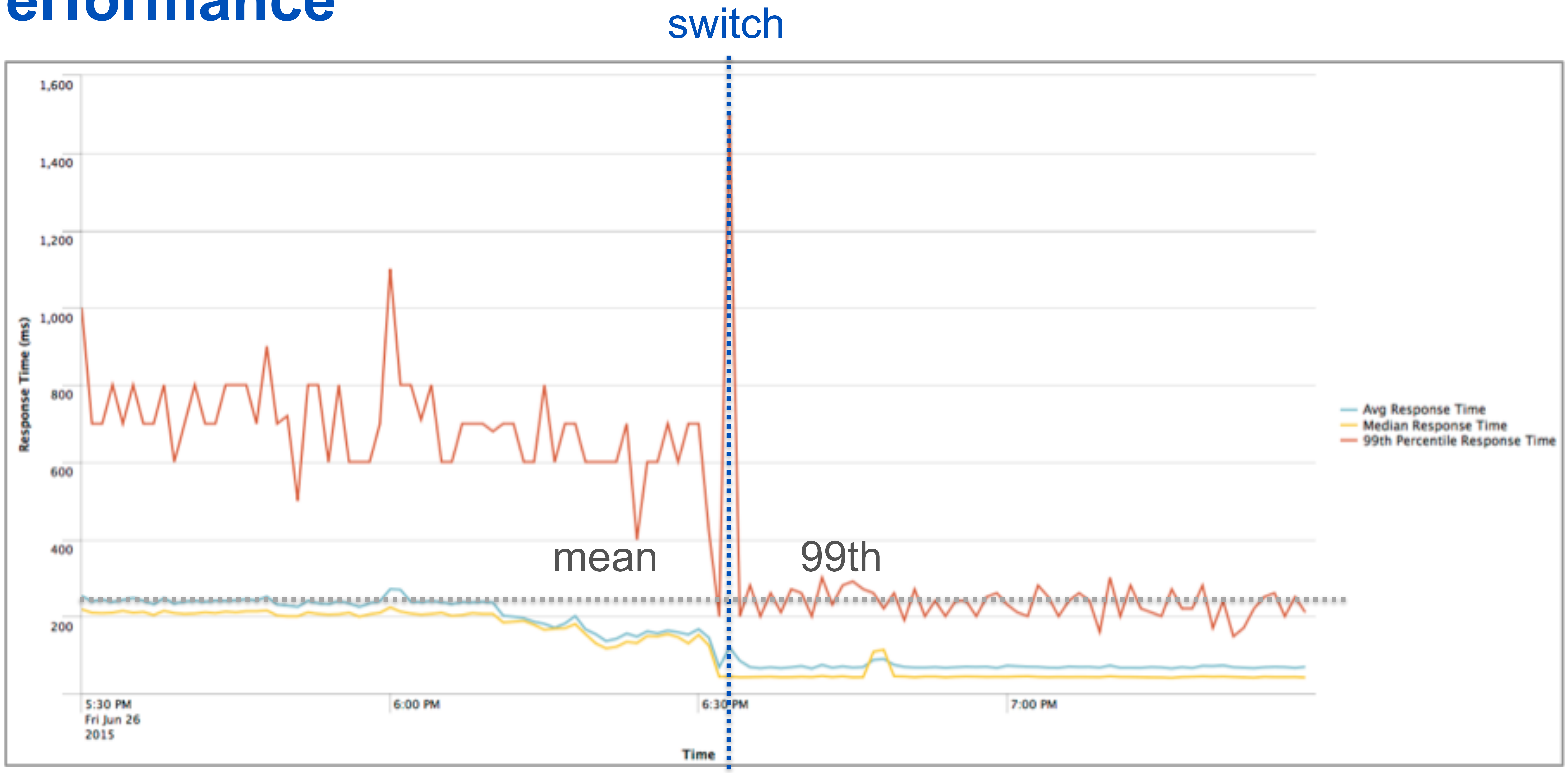
COMCAST

Performance



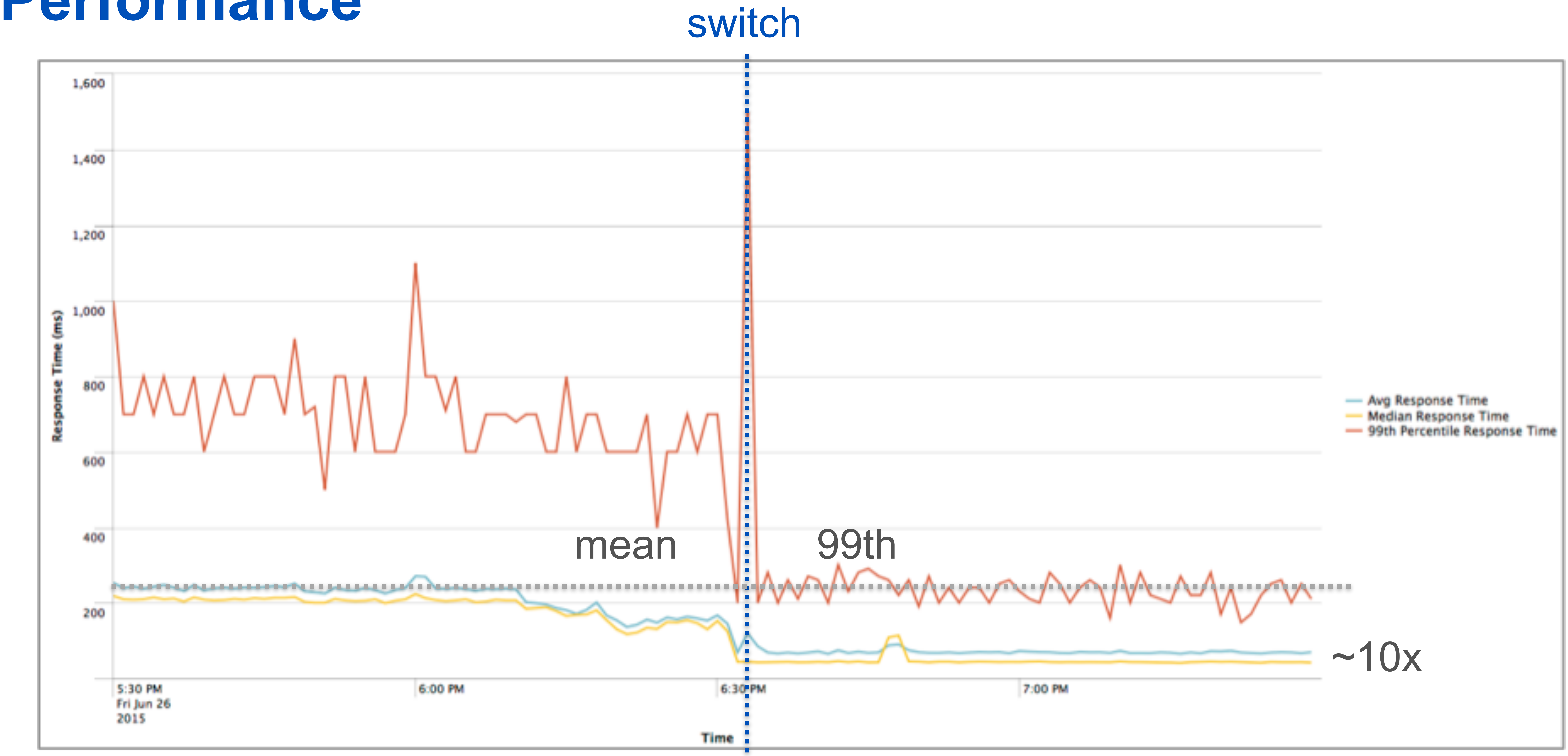
COMCAST

Performance

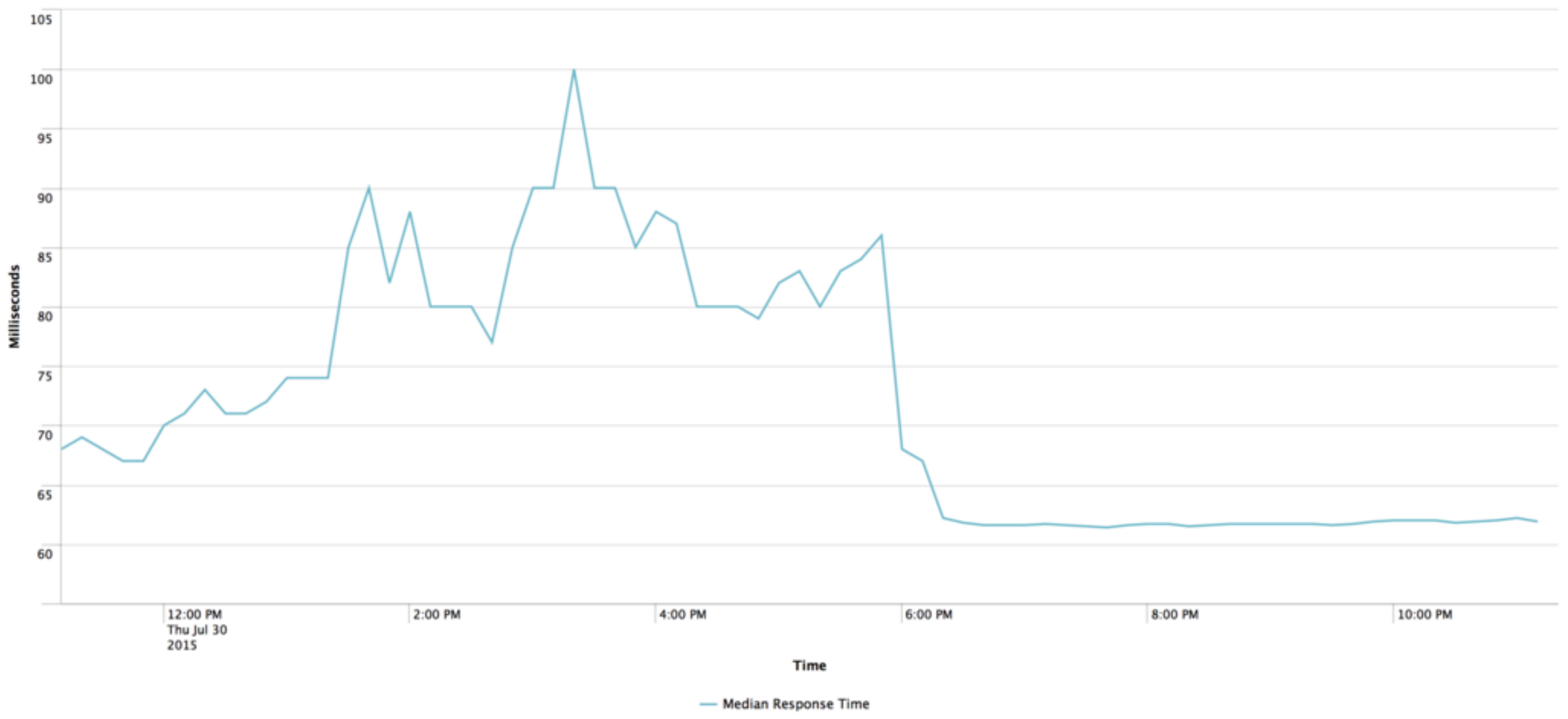


COMCAST

Performance

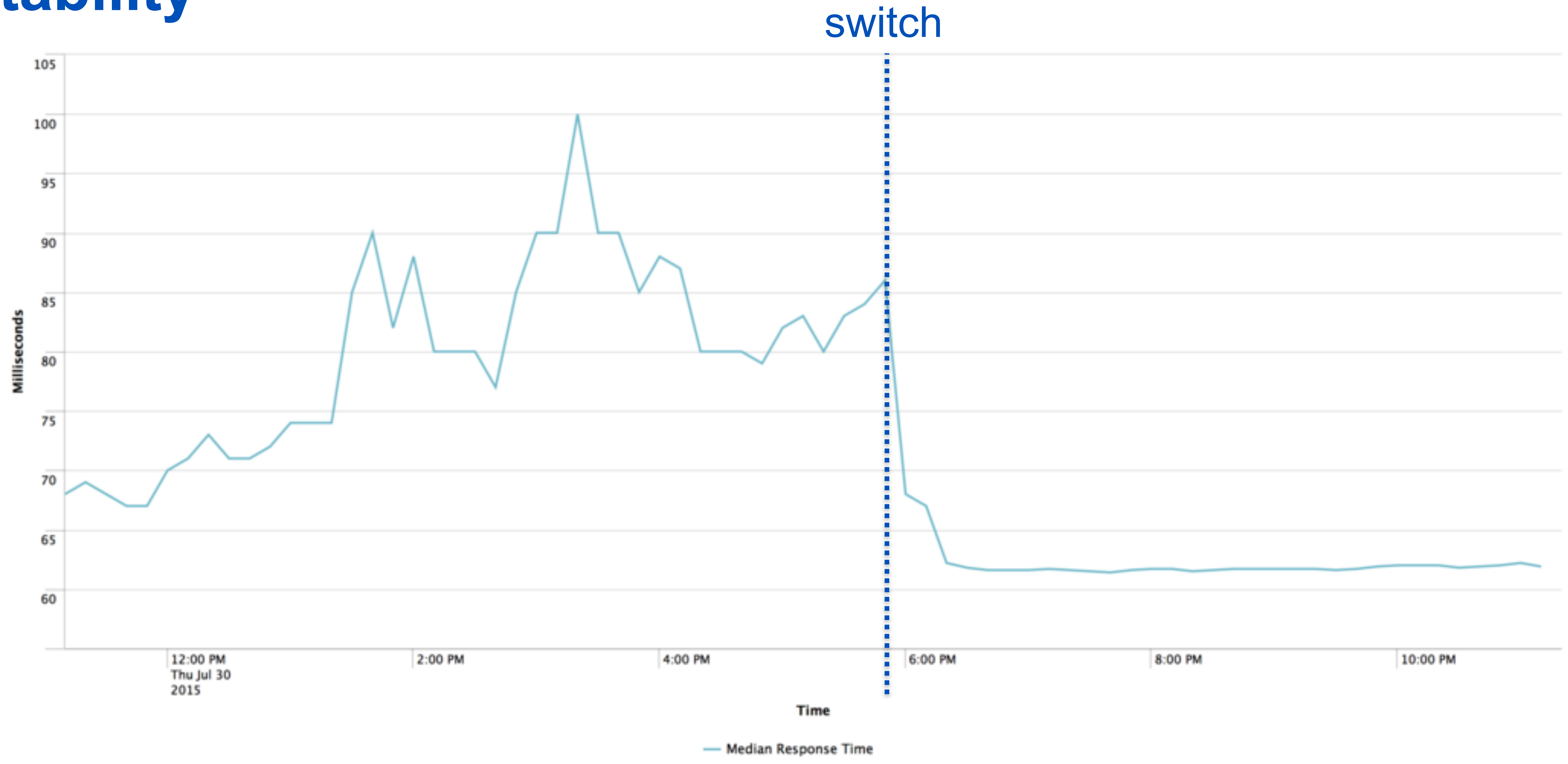


Stability



COMCAST

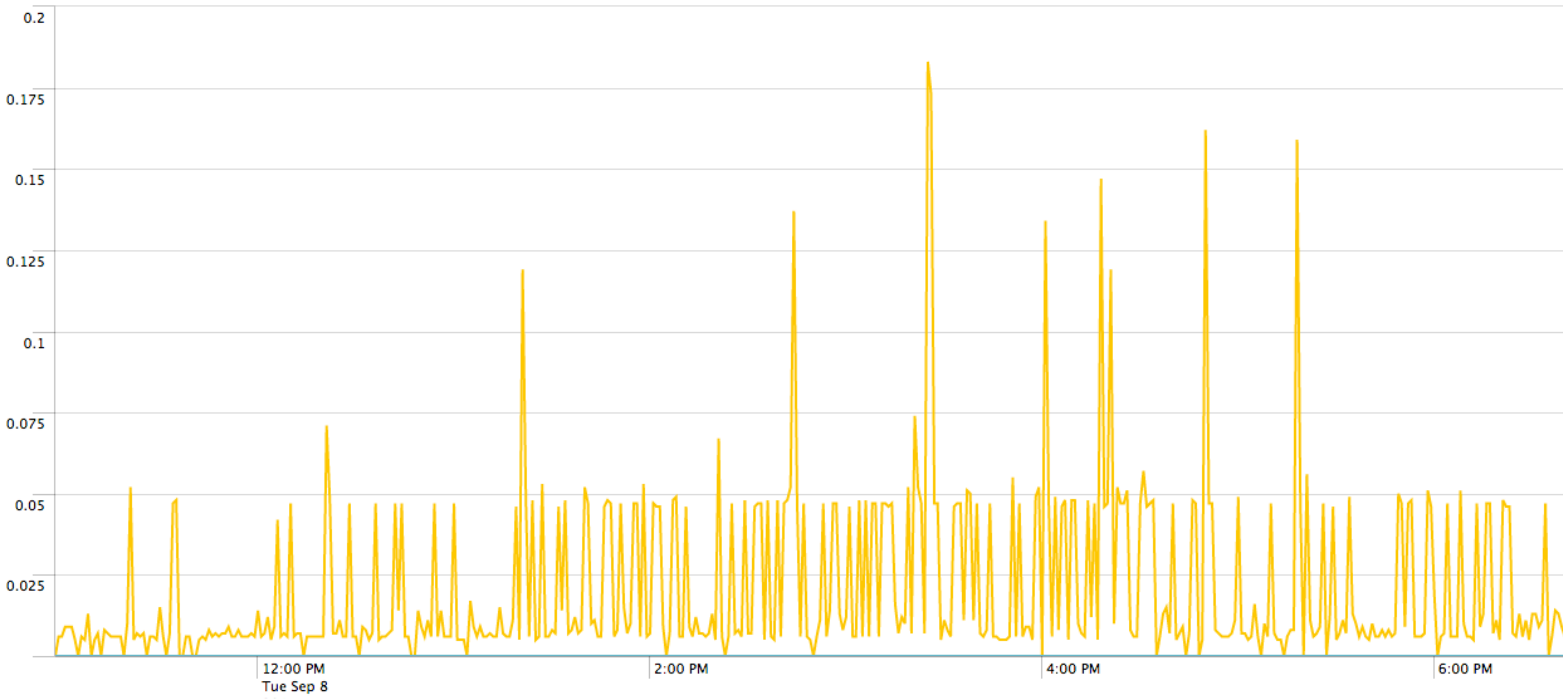
Stability



COMCAST

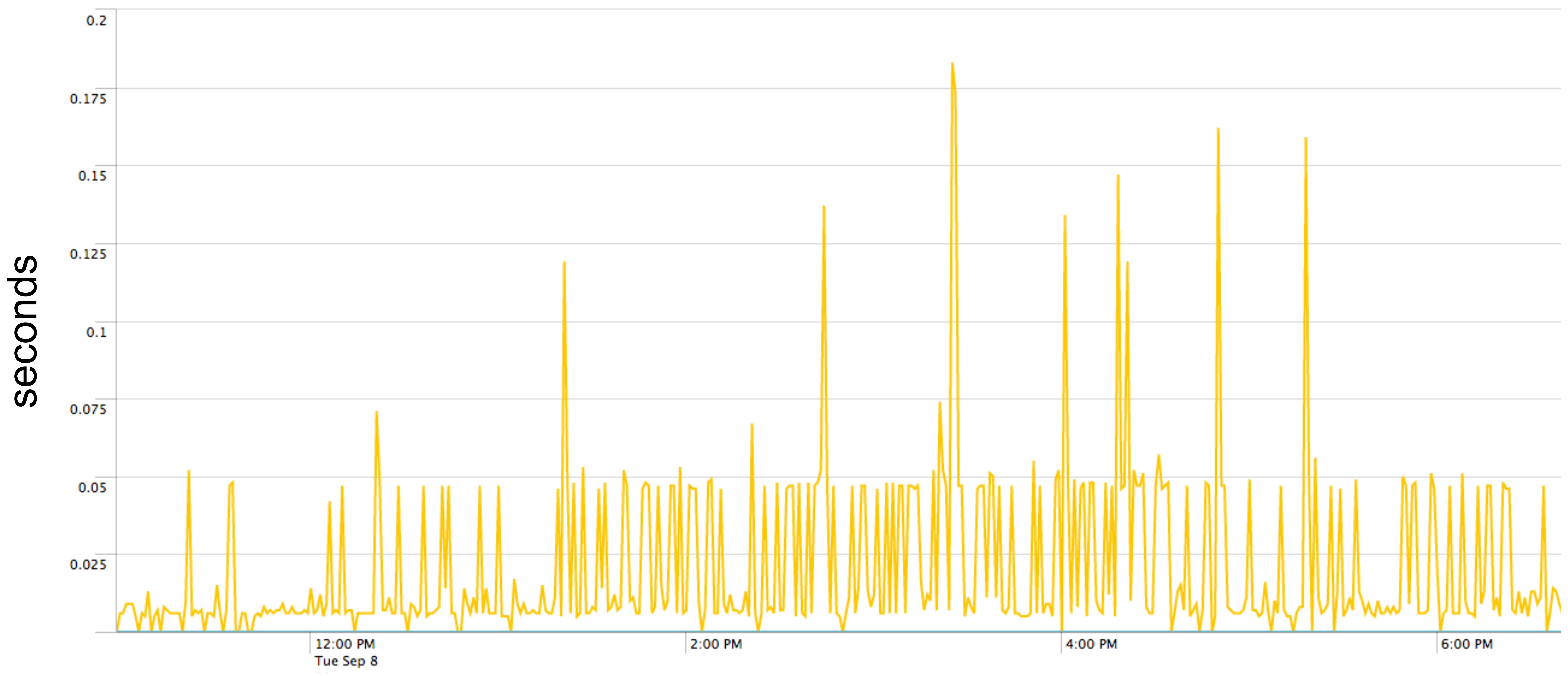
Impact

```
index=codebig host=*.cimops.net source="/var/log/nginx/access.log" |
eval d = request_time - upstream_response_time |
timechart span=1m perc99(d) max(d)
```



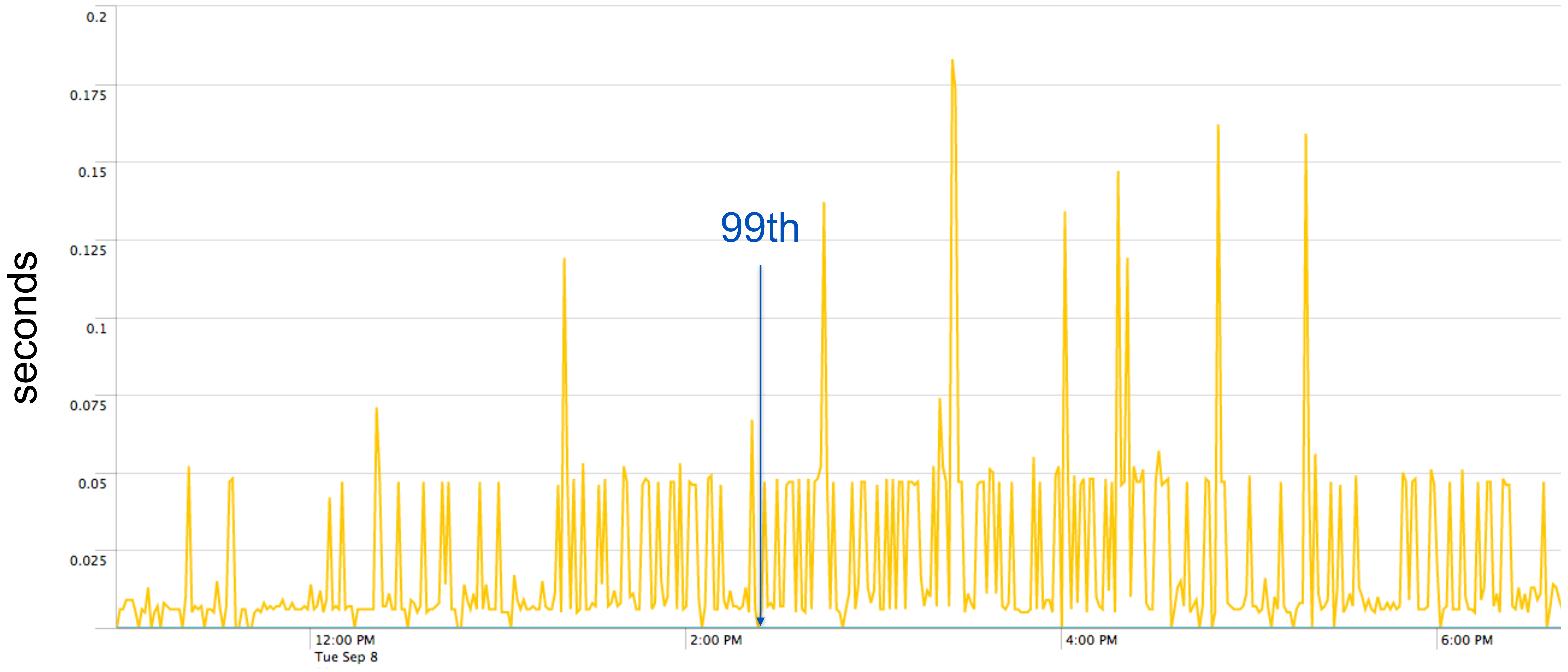
Impact

```
index=codebig host=*.cimops.net source="/var/log/nginx/access.log" |
eval d = request_time - upstream_response_time |
timechart span=1m perc99(d) max(d)
```



Impact

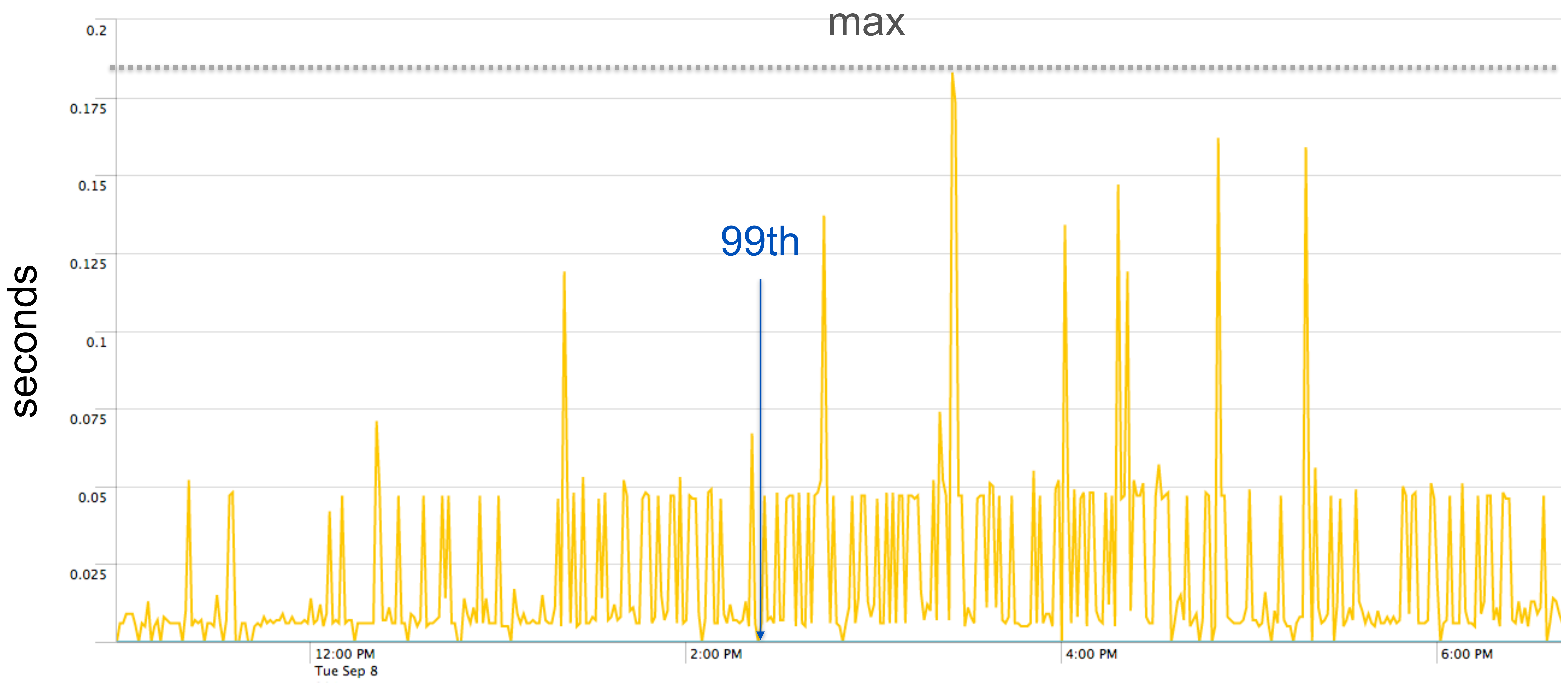
```
index=codebig host=*.cimops.net source="/var/log/nginx/access.log" |
eval d = request_time - upstream_response_time |
timechart span=1m perc99(d) max(d)
```



COMCAST

Impact

```
index=codebig host=*.cimops.net source="/var/log/nginx/access.log" |
eval d = request_time - upstream_response_time |
timechart span=1m perc99(d) max(d)
```



Successes

Successes

great performance improvements

Successes

great performance improvements

hosting ~400 endpoints

Successes

great performance improvements

hosting ~400 endpoints

> 367MM requests a day

Successes

great performance improvements
hosting ~400 endpoints
> 367MM requests a day
prevented upstream downtime

Challenges

Challenges

3rd-party Lua ecosystem

Challenges

3rd-party Lua ecosystem
not self-service yet

Challenges

3rd-party Lua ecosystem
not self-service yet
configuration file size

Challenges

3rd-party Lua ecosystem
not self-service yet
configuration file size
kernel tuning

Challenges

3rd-party Lua ecosystem
not self-service yet
configuration file size
kernel tuning
owning availability

Conclusion

Conclusion

NGINX + Lua for HTTP middleware

Conclusion

NGINX + Lua for HTTP middleware
Automated deployment pipeline

Conclusion

NGINX + Lua for HTTP middleware
Automated deployment pipeline
Concurrent request limiting

Conclusion

NGINX + Lua for HTTP middleware
Automated deployment pipeline
Concurrent request limiting
Operational flexibility

Thanks