Rapid CDN IP Redirection with SDN

Jeffrey Lai
REANNZ
18-19 Feb 2015

2015 Wellington SDN Workshop

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Background

- CDN Basics
- Assumptions...
- ...And breaking them
- The problem

My research

- SDN concepts
- Implementing an SDN solution
- Sample code

BACKGROUND

CDN BASICS

Content Delivery Networks(CDNs) generally utilise a DNS-based request rerouting scheme

When users want to access content, they perform a DNS request to associate an IP address to a hostname

These users are load balanced/redirected according to geography, server load, etc

ASSUMPTIONS...

IP-hostname mappings are unlikely to change

IP-hostname mappings are correct

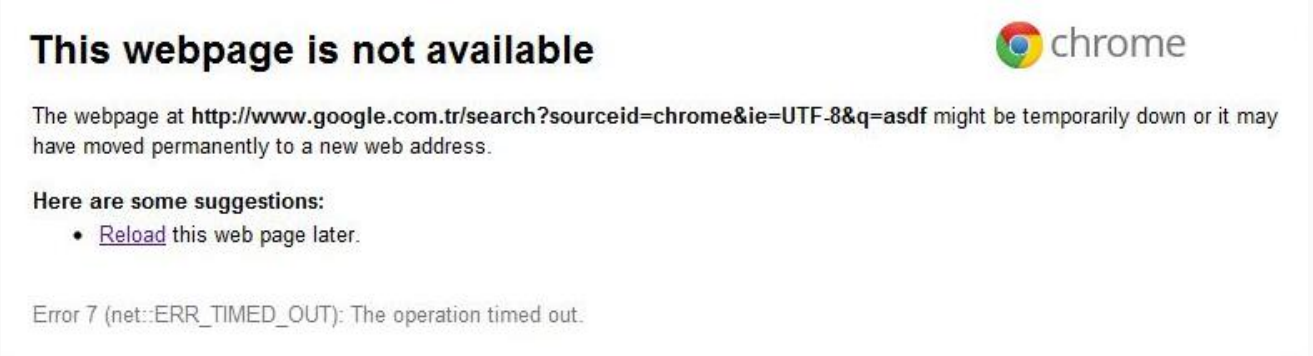
IP-hostname mappings are ideal

...AND BREAKING THEM

What happens when we break these assumptions?

...AND BREAKING THEM

What happens when we break these assumptions?



THE PROBLEM

DNS caching is a thing!

- Browsers
- Operating systems
- Caching resolvers on the LAN
- Routers

MY RESEARCH

GETTING AROUND DNS CACHING

Network operators know the latest server details, users don't

- New server IP addresses
- Network load/status
- Server utilisation

GETTING AROUND DNS CACHING

Network operators know the latest server details, users don't

- New server IP addresses
- Network load/status
- Server utilisation

So why not let the network operator transparently reroute requests?

We have a situation where a network operator can take advantage of

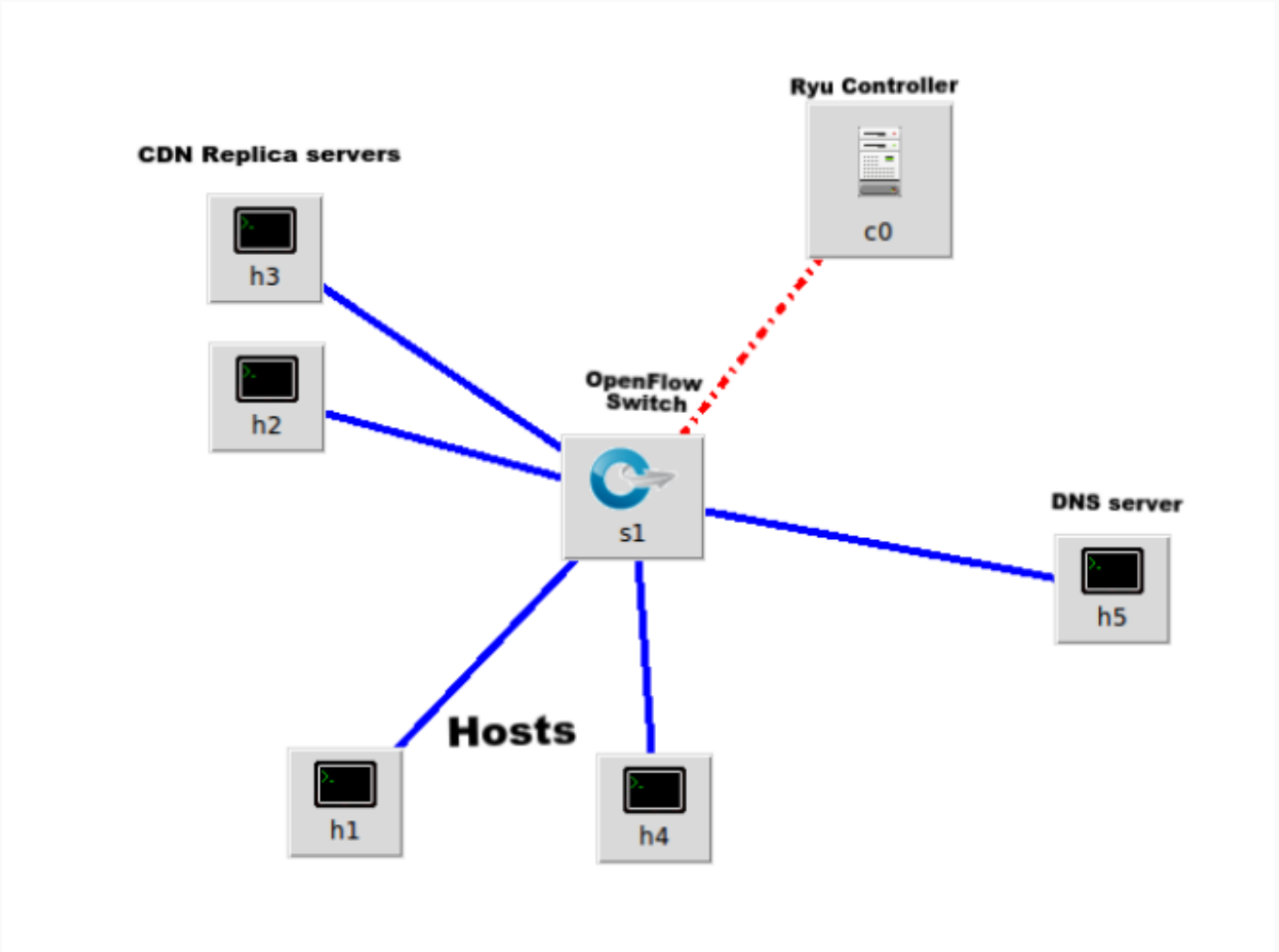
- A global network view
- Centralised control
- Flow-based IP rerouting

Seems like a good candidate for an SDN-based approach...

IMPLEMENTING AN SDN SOLUTION

Why not perform DNAT/SNAT operations on packets passing through a switch at the network edge?

My test environment utilises an openflow switch containing the relevant NAT rules, and an Ryu controller that manages state


Ryu Controller
c0
CDN Replica servers
h3
h2
OpenFlow
Switch
s1
DNS server
h5
Hosts
h1
h4

SAMPLE TOPOLOGY CODE

```
# create a mininet object, with a remote controller
net = Mininet(controller=RemoteController, autoStaticArp=True, autoSetMacs=True)
c0 = net.addController('c0')                    # Add our controller
s1 = net.addSwitch('s1')                        # Add our switch
h1 = net.addHost('h1')                          # Add host 1
...
h5 = net.addhost('h5')                          # Add host 5
TCLink(h1, s1, port2=1, delay='10ms', bw=10)    # Link h1 to s1
...
TCLink(h5, s1, port2=5, delay='10ms')           # Link h5 to s1
net.build()                                     # Build our mininet object
c0.start()                                      # Start our controller
s1.start([c0])                                  # Start our switch, linking to c0
c0.cmdPrint('ryu-manager CDN_NAT_CONTROLLER &') # Tell c0 to execute a command
CLI(net)                                        # Drop us into the mininet CLI
```

SAMPLE FLOW RULES

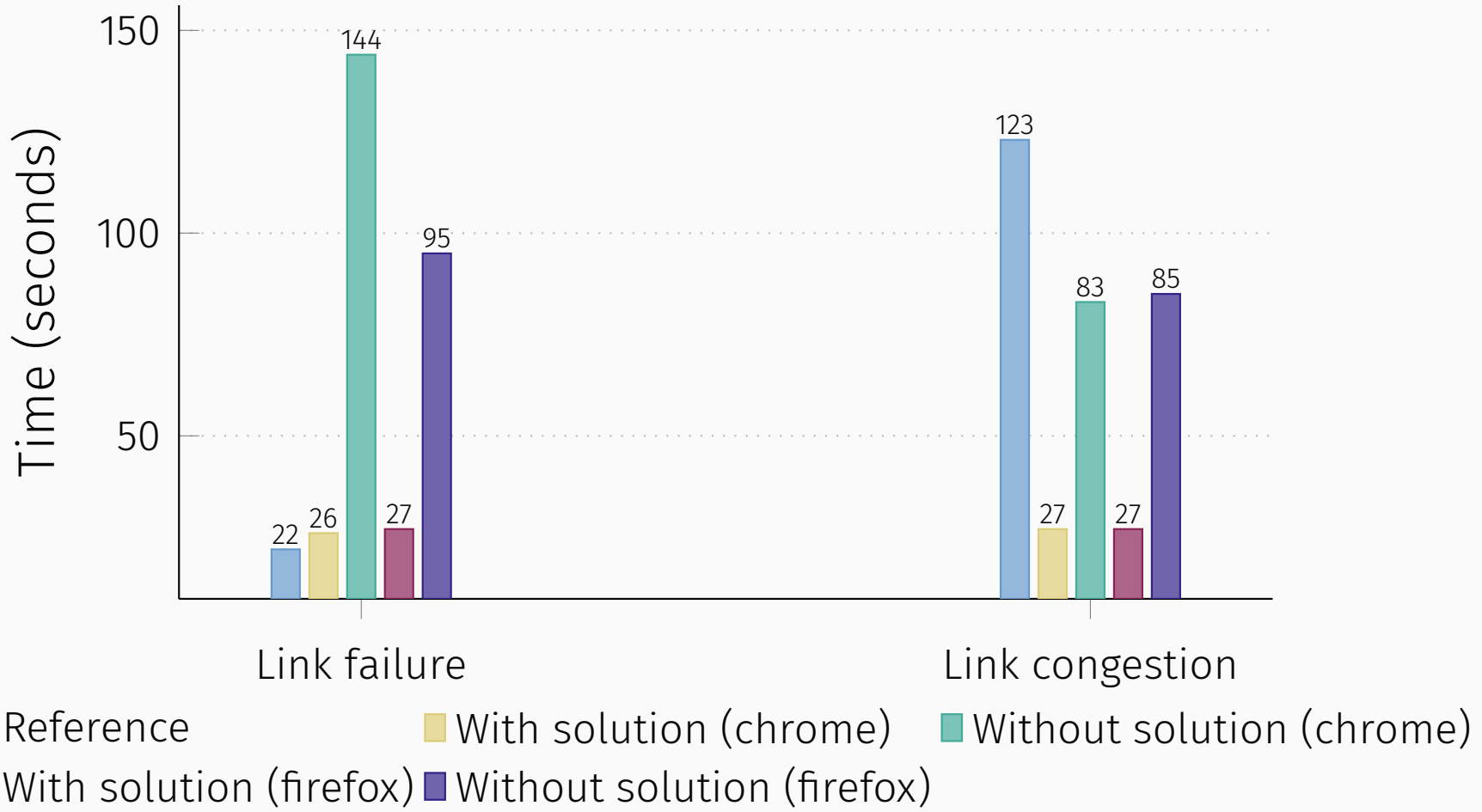
Table miss flow that catches traffic from h1 (10.0.0.1) to the CDN replicas

Priority	In-port	Protocol	IP SRC	IP DST	SRC port	DST port	Actions
10	*	TCP	10.0.0.1	10.0.0.0/24	*	80	Send packet to controller

DNAT and SNAT flows to transparently redirect a user to a different CDN server

Priority	In-port	Protocol	IP SRC	IP DST	SRC port	DST port	Actions
20	*	TCP	10.0.0.1	10.0.0.2	4444	80	DNAT DST IP from 10.0.0.2 to 10.0.0.3, send out switch port 3
20	*	TCP	10.0.0.3	10.0.0.1	80	4444	SNAT SRC IP from 10.0.0.3 to 10.0.0.2, send out switch port 1

RESULTS


Time (seconds)
150
100
50
22
26
144
27
95
123
27
83
27
85
Link failure
Link congestion
Reference
With solution (chrome)
Without solution (chrome)
With solution (firefox)
Without solution (firefox)

QUESTIONS?

Feel free to contact me at jeff@jeffl.ai

QUESTIONS?

Feel free to contact me at jeff@jeffl.ai

I can demo this on my laptop!