RESTORING TCP SESSIONS WITH A DISTRIBUTED HASH TABLE

Advanced Networking RP2

Peter Boers

June 29, 2016

System and Network Engineering - FNWI - UVA

SCALING INFRASTRUCTURE

- Imagine you are one of the largest providers of web services in the world...
- How do you make sure that you can service your infrastructure **and** make sure your clients never know that this is happening?

LOAD BALANCERS

Why do you balance load?

- To maintain the integrity of the end to end session between the Client who is trying to access a Service.
- To distribute load across multiple end points

TRADITIONAL SOLUTIONS

Traditional hardware and software **Load Balancers** can do some or all of the following:

- Maintain a high available setup
- Layers 3,4 and/or 7 in the OSI stack
- TLS offloading
- Compression
- Marshaling of TCP sessions
- Proxying

TRADITIONAL SOLUTIONS

Traditional hardware and software **Load Balancers** can do some or all of the following:

- Maintain a high available setup
- Layers 3,4 and/or 7 in the OSI stack
- TLS offloading
- Compression
- Marshaling of TCP sessions
- Proxying

However sometimes these solutions require high licensing fees and they are unable to scale enough.

TRADITIONAL SOLUTIONS

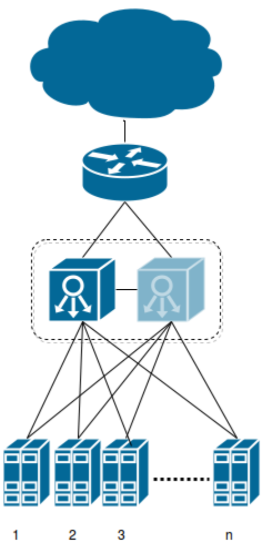


Figure: Simple high available setup

NEW NETWORK DESIGN

In a recent draft RFC by Facebook and Arista Networks, a new network design for very large data centers is discussed[1]:

NEW NETWORK DESIGN

In a recent draft RFC by Facebook and Arista Networks, a new network design for very large data centers is discussed[1]:

- ”Environments of this scale have a unique set of network requirements with an emphasis on operational simplicity and network stability.”

NEW NETWORK DESIGN

In a recent draft RFC by Facebook and Arista Networks, a new network design for very large data centers is discussed[1]:

- ”Environments of this scale have a unique set of network requirements with an emphasis on operational simplicity and network stability.”
- This document proposes the use of EBGP as the only routing protocol.

NEW NETWORK DESIGN

In a recent draft RFC by Facebook and Arista Networks, a new network design for very large data centers is discussed[1]:

- ”Environments of this scale have a unique set of network requirements with an emphasis on operational simplicity and network stability.”
- This document proposes the use of EBGP as the only routing protocol.
- To distribute load and traffic, Anycast in combination with Equal Cost MultiPath routing (ECMP) will be used instead of traditional load balancers.

NEW NETWORK DESIGN

In a recent draft RFC by Facebook and Arista Networks a new network design for very large data centers is discussed[1]:

- ”Environments of this scale have a unique set of network requirements with an emphasis on operational simplicity and network stability.”
- This document proposes the use of an EBGP only as routing protocol.
- To distribute load and traffic, Anycast in combination with Equal Cost MultiPath routing (ECMP) will be used instead of traditional load balancers.

The goal is to achieve greater horizontal scalability and use proven Network protocols for simplicity

NEW NETWORK DESIGN

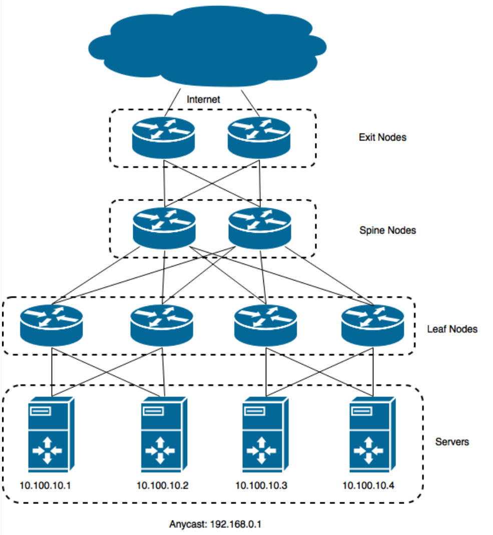


Figure: New Design

NEW NETWORK DESIGN

Features of the new network:

- Balancing no longer done at the edge but at the endpoints
- All hosts take part in the routing protocol
- Layer 3/4 balancing is no longer scalable through traditional means
- How do you maintain the integrity of a TCP session?

RESEARCH QUESTIONS

How can a DHT be leveraged to maintain TCP session state in the case of a failure in a Large BGP networks with thousands of hosts [1]?

- What technical requirements are needed to maintain the TCP session in the case of a failure?
- Does using a DHT to lookup invalid sessions provide enough performance so that the session can continue?

METHOD - WHY A DHT?

What is good about a Distributed Hash Table in this situation?

- Nodes do not have all the information, but know where to look it up.
- Distributes the information evenly over all nodes.
- Scales well: $O(n) = \log(n)$
- Stores key-value pairs.

METHOD - WHY A DHT?

What is good about a Distributed Hash Table in this situation?

- Nodes do not have all the information, but know where to look it up.
- Distributes the information evenly over all nodes.
- Scales well: $O(n) = \log(n)$
- Stores key-value pairs.

Kademlia implementation chosen to build the Distributed Hash Table.

METHOD – HOW TO HANDLE TCP

How do we detect on the node if the TCP session is wrong?

- Nodes must track connections
- If the connection is not NEW or ESTABLISHED do a look up on the DHT.
- The 4-tuple is ideal for storing in the DHT: Client socket = key. Server socket = value

METHOD - HOW TO HANDLE TCP

How do we detect on the node if the TCP session is wrong?

- Nodes must track connections
- If the connection is not NEW or ESTABLISHED do a look up on the DHT.
- The 4-tuple is ideal for storing in the DHT: Client socket = key. Server socket = value

```
{ ”145.100.102.131:12445” : ”10.100.10.1:80”  }
```

METHOD - HOW TO HANDLE TCP

How do we detect on the node if the TCP session is wrong?

- Nodes must track connections
- If the connection is not NEW or ESTABLISHED do a look up on the DHT.
- The 4-tuple is ideal for storing in the DHT: Client socket = key. Server socket = value

```
{ ”145.100.102.131:12445” : ”10.100.10.1:80”  }
```

When a wrong session arrives do a look up and redirect the traffic.

SCENARIO

In the scenario we assume the following:

- N amount of servers hosting a website and taking part in a DHT overlay
- The website is balanced using ECMP and Anycast on the network
- All new TCP sessions are stored in the DHT

SCENARIO

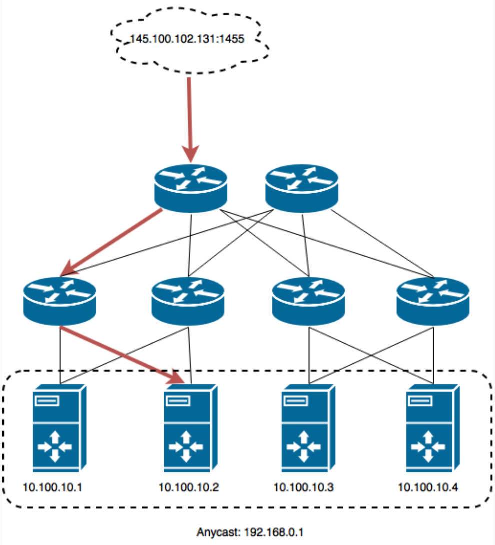
In the scenario we assume the following:

- N amount of servers hosting a website and taking part in a DHT overlay
- The website is balanced using ECMP and Anycast on the network
- All new TCP sessions are stored in the DHT

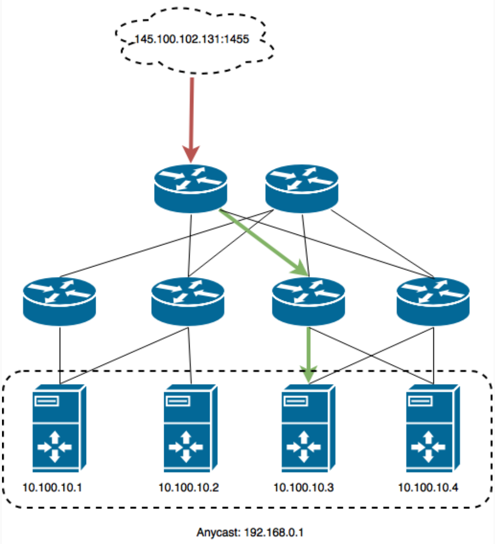
Then we simulate a link failure:

- Let ECMP recalculate the path of the traffic
- Lookup the ”Key” (Client socket)
- Forward traffic to the ”Value” (Server Identifier)

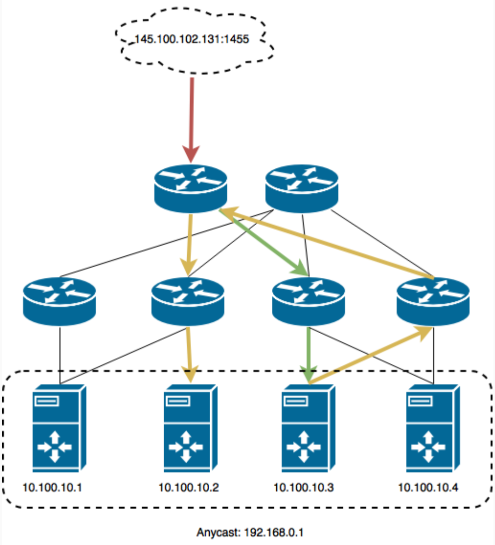
SCENARIO - STEP 1



SCENARIO - STEP 2



SCENARIO - STEP 3



IN WHAT CASE IS THE TEST SUCCESSFUL?

How do you measure when a fail over is within an industry standard acceptable window?

- Amazon Web Services load balancing health check has a default of 30 seconds and a minimum of 5 seconds, with a timeout of 30 seconds[2]

IN WHAT CASE IS THE TEST SUCCESSFUL?

How do you measure when a fail over is within an industry standard acceptable window?

- Amazon Web Services load balancing health check has a default of 30 seconds and a minimum of 5 seconds, with a timeout of 30 seconds[2]
- Kemp technologies has a default health check of 9 seconds and a minimum of 3 seconds, with a timeout of 15 seconds[3]

IN WHAT CASE IS THE TEST SUCCESSFUL?

How do you measure when a fail over is within an industry standard acceptable window?

- Amazon Web Services load balancing health check has a default of 30 seconds and a minimum of 5 seconds, with a timeout of 30 seconds[2]
- Kemp technologies has a default health check of 9 seconds and a minimum of 3 seconds, with a timeout of 15 seconds[3]
- f5 technologies has a default health check every 5 seconds, with a timeout of 15 seconds[4]

IN WHAT CASE IS THE TEST SUCCESSFUL?

How do you measure when a fail over is within an industry standard acceptable window?

- Amazon Web Services load balancing health check has a default of 30 seconds and a minimum of 5 seconds, with a timeout of 30 seconds[2]
- Kemp technologies has a default health check of 9 seconds and a minimum of 3 seconds, with a timeout of 15 seconds[3]
- f5 technologies has a default health check every 5 seconds, with a timeout of 15 seconds[4]

This means: in the worst case scenario there is a timeout of 20 seconds to around one minute before TCP session restoration

RESULTS

Results for the test setup of this research:

- Setting Time - The time that it takes to set a key in the DHT
- Detection Time - The time that it takes to detect a Link Failure
- Lookup Time - The Time it takes to Lookup a key on the DHT.

SETTING TIME

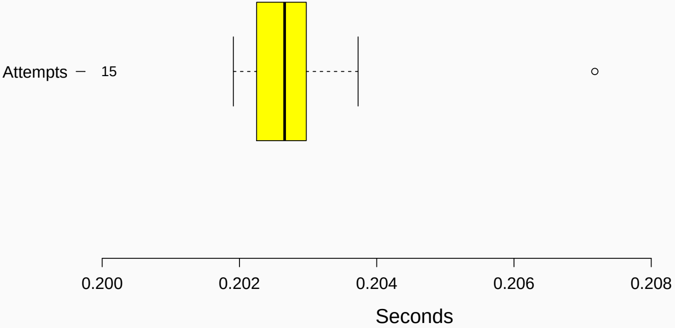


Figure: This plot shows the time in seconds that it takes to set the Key - Value pair on the DHT

DETECTION TIME

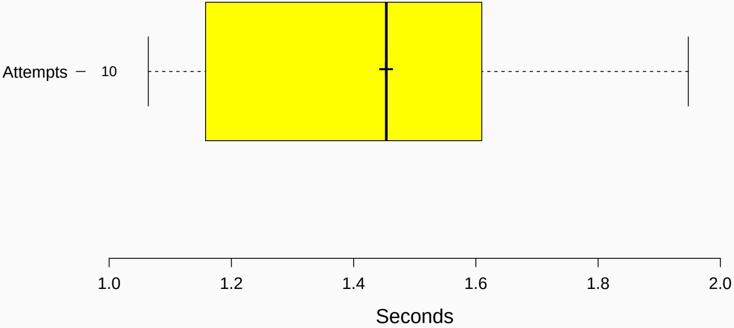


Figure: This plot shows the time in seconds that it takes between the failure of a link and the rerouting of packets

LOOKUP TIME

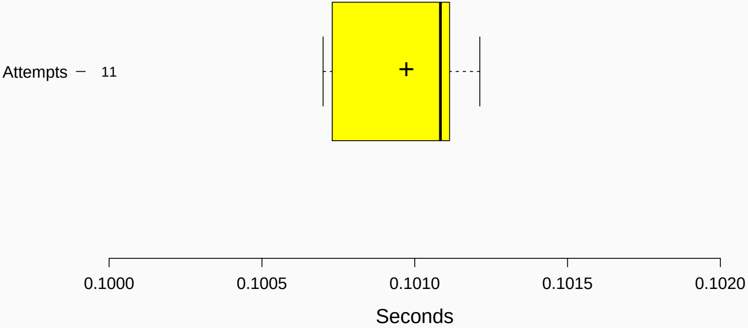


Figure: This plot shows the time in seconds that it takes for a node to lookup a key on the DHT

CONCLUSION - DISCUSSION

Key findings:

- On this small scale it is fast enough to detect a failure and act on it.
- No protocol changes needed.
- Horizontal scalability is very simple in this model

CONCLUSION - DISCUSSION

Key findings:

- On this small scale it is fast enough to detect a failure and act on it.
- No protocol changes needed.
- Horizontal scalability is very simple in this model

Future Efforts:

- What is the performance cost when it scales?
- Convert script to binary and integrate with other software
- How do you make sure it is reliable?

REFERENCES

- P. Lapukhov, A. Premji, and J. Mitchell. Use of BGP for routing in large-scale data centers. Tech. rep. Technical report, IETF, 2016. URL: https://datatracker.ietf.org/doc/draft-ietf-rtgwg-bgp-routing-large-dc/.
- Amazon. Elastic Load Balancing - Configure Health Checks. 2016. URL: http://docs.aws.amazon.com/ElasticLoadBalancing/latest/DeveloperGuide/elb-healthchecks.html (visited on 06/28/2016).
- Kemp Technologies. Frequently Asked Questions. 2016. URL: https://kemptechnologies.com/faq/ (visited on 06/28/2016).
- f5 solutions. Manual Chapter: Configuring Monitors. 2016. URL: https://support.f5.com/kb/en-us/products/big-ip_ltm/manuals/product/ltm_configuration_guide_10_0_0/ltm_monitors.html#1201151 (visited on 06/28/2016).