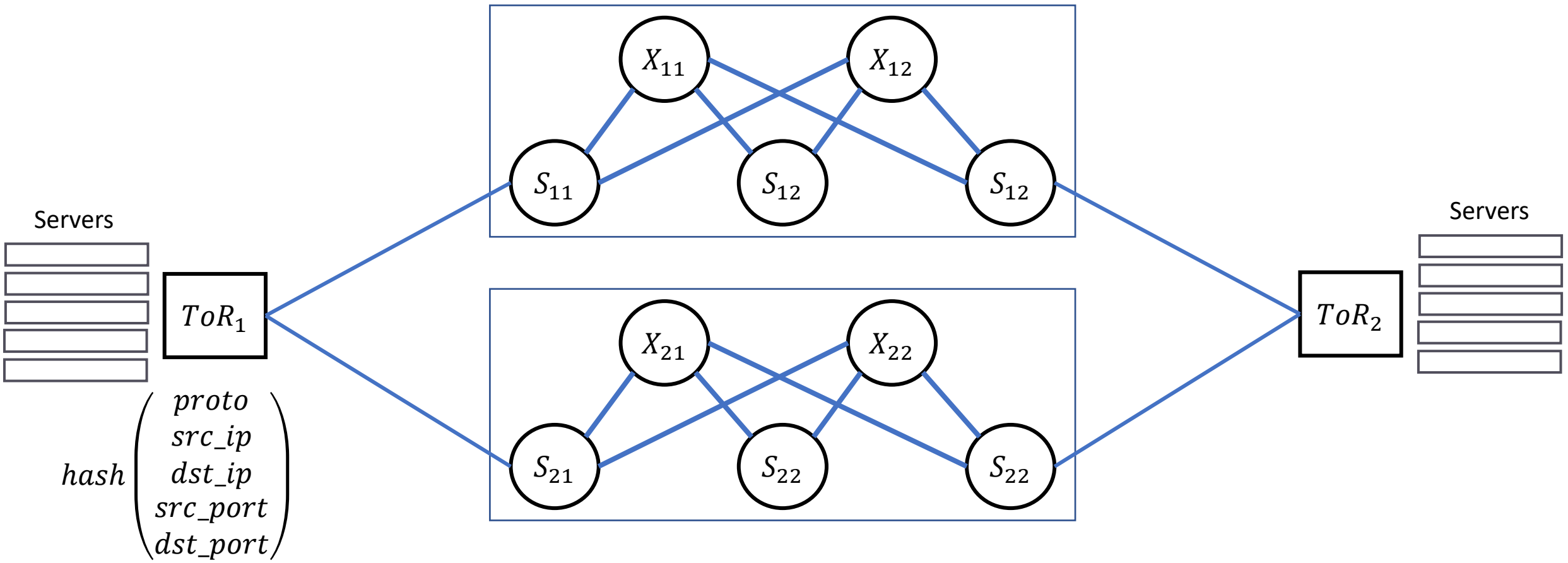
Self-healing Networking with Flow Label

Alexander Azimov mitradir@yandex-team.ru

ToR + 2xPlanes + ToR



Theory DC: Many-Many Paths

N_PLANES: Number of planes in DC;

N_X_SPINES: Number of super spines (X) in each plane;

- Inside ToR: 1
- Inside PoD: N_PLANES
- Between PoDs: N_PLANES x N_X_SPINES

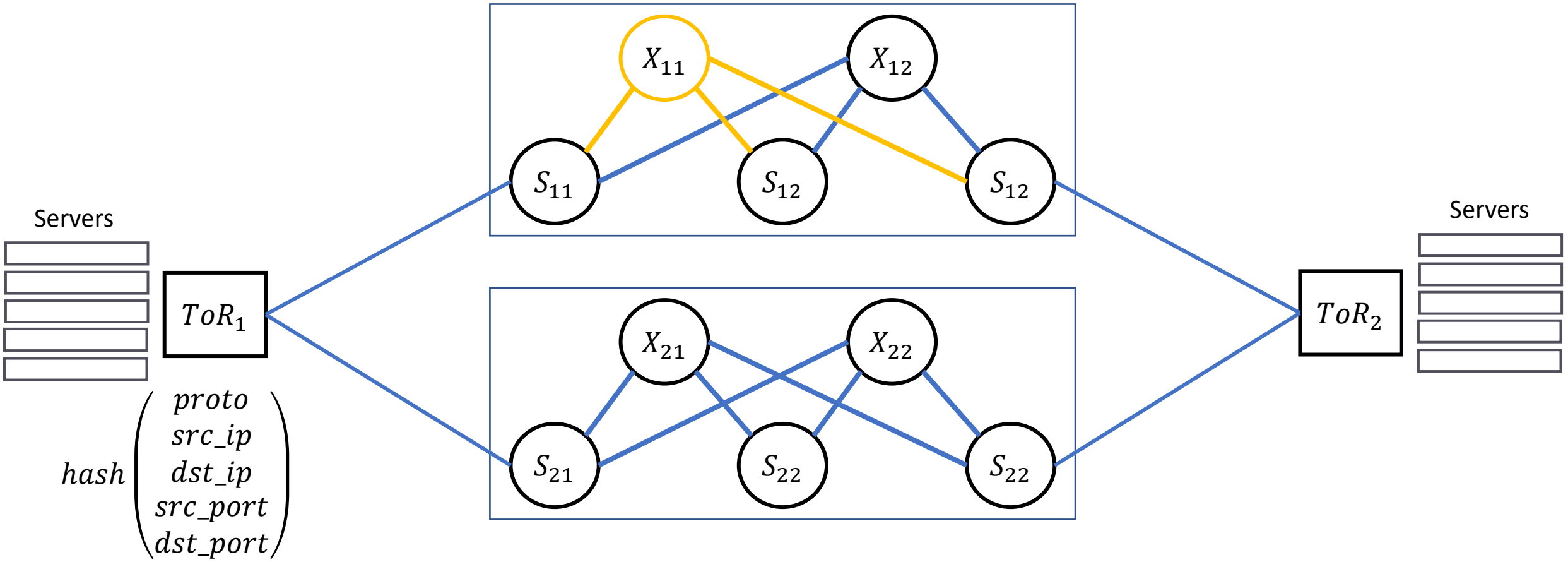
Real DC: Many-Many Paths

N_PLANES: Number of planes in DC; (8)

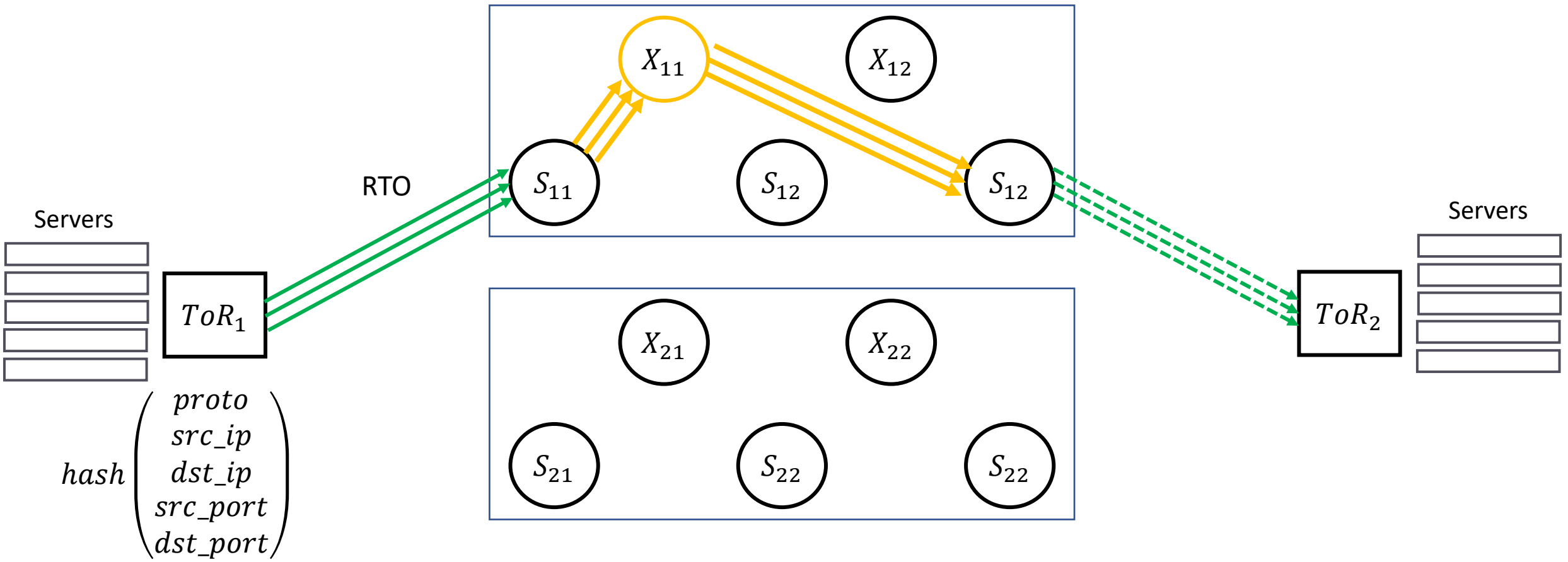
N_X_SPINES: Number of super spines (X) in each plane; (32)

- Inside ToR: 1
- Inside PoD: N_PLANES = 8
- Between PoDs: N_PLANES x N_X_SPINES = 256

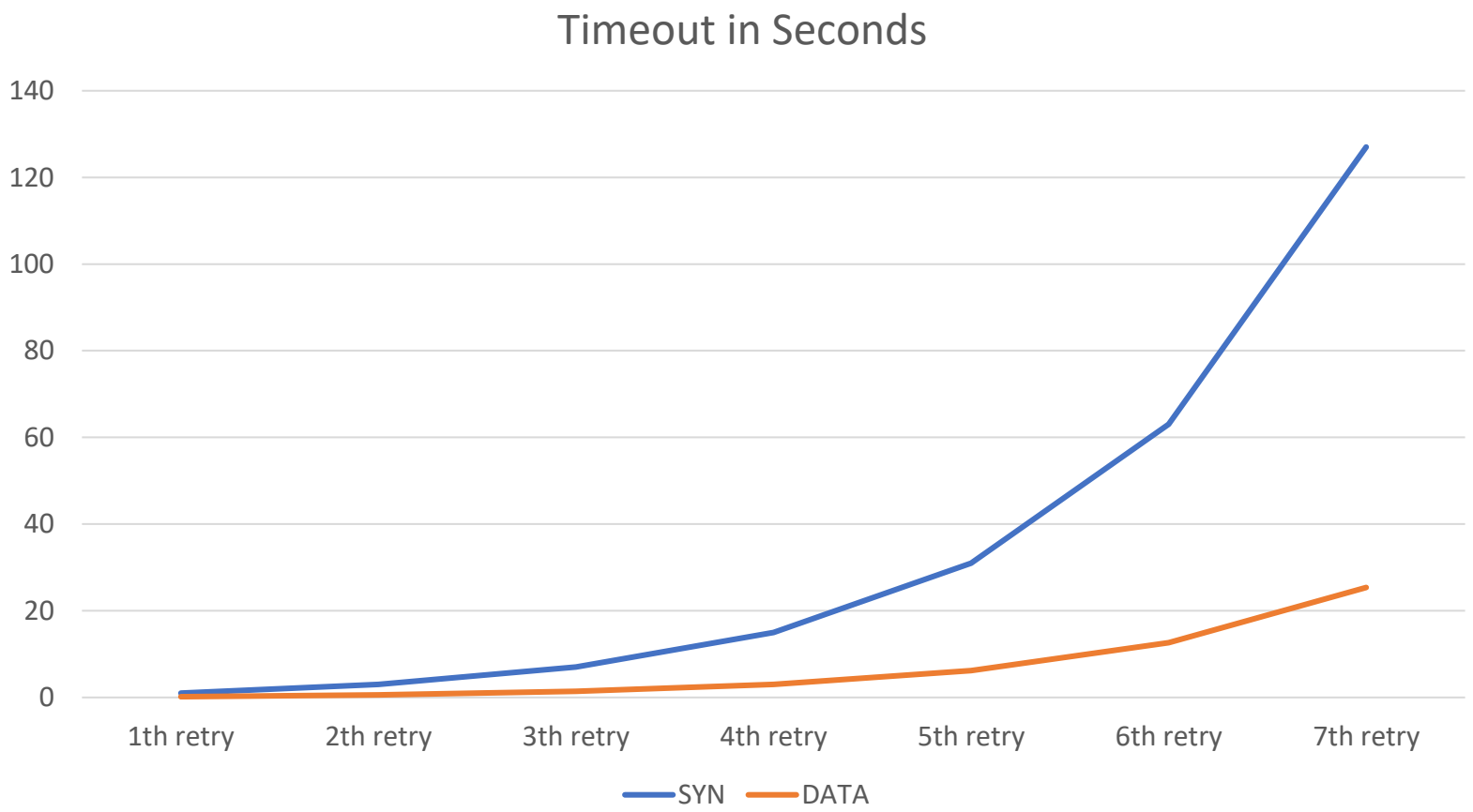
X_{11} is Broken: Constant Loss



Unhappy TCP Flow



RTO & SYN_RTO Timeouts



Timeouts

RTO_MIN	SYN_RTO
200ms	1s

Real RTT

1ms

RTO = MAX(RTO_MIN, RTT)

Linux Kernel

2014

From: Tom Herbert @ 2014-07-02 4:33 UTC (permalink / raw)
To: davem, netdev

Automatically generate flow labels for IPv6 packets on transmit.
The flow label is computed based on skb_get_hash. The flow label will
only automatically be set when it is zero otherwise (i.e. flow label
manager hasn't set one). This supports the transmit side functionality
of RFC 6438.

Added an IPv6 sysctl auto_flowlabels to enable/disable this behavior
system wide, and added IPV6_AUTOFLOWLABEL socket option to enable this
functionality per socket.

By default, auto flowlabels are disabled to avoid possible conflicts
with flow label manager, however if this feature proves useful we
may want to enable it by default.

It should also be noted that FreeBSD has already implemented automatic
flow labels (including the sysctl and socket option). In FreeBSD,
automatic flow labels default to enabled.

Linux Kernel

2015

From: Tom Herbert <tom@herbertland.com>
To: <davem@davemloft.net>, <netdev@vger.kernel.org>
Cc: <kernel-team@fb.com>
Subject: [PATCH net-next 0/2] net: Initialize sk_hash to random value and res
Date: Tue, 28 Jul 2015 16:02:04 -0700
Message-ID: <1438124526-2129341-1-git-send-email-tom@herbertland.com> (raw)

This patch set implements a common function to simply set sk_txhash to
a random number instead of going through the trouble to call flow
dissector. From dst_negative_advice we now reset the sk_txhash in hopes
of finding a better ECMP path through the network. Changing sk_txhash
affects:
  - IPv6 flow label and UDP source port which affect ECMP in the network
  - Local EMCP route selection (pending changes to use sk_txhash)

Tom Herbert (2):
  net: Set sk_txhash from a random number
  net: Recompute sk_txhash on negative routing advice

Linux Kernel

2016

From: Lawrence Brakmo <brakmo@fb.com>
To: netdev <netdev@vger.kernel.org>
Cc: Kernel Team <kernel-team@fb.com>,
        Eric Dumazet <eric.dumazet@gmail.com>,
        Yuchung Cheng <ycheng@google.com>,
        Neal Cardwell <ncardwell@google.com>
Subject: [PATCH v4 net-next] tcp: Change txhash on every SYN and RTO retransmit
Date: Tue, 27 Sep 2016 19:03:37 -0700
Message-ID: <20160928020337.3057238-1-brakmo@fb.com> (raw)

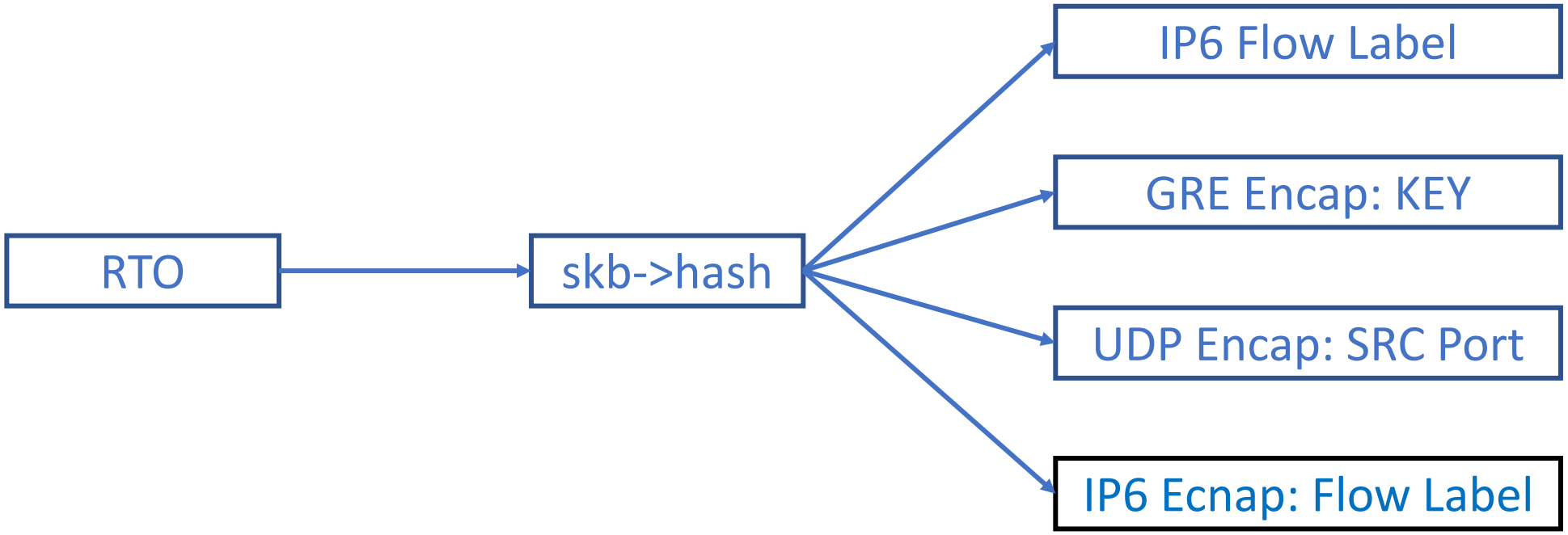
The current code changes txhash (flowlables) on every retransmitted
SYN/ACK, but only after the 2nd retransmitted SYN and only after
tcp_retries1 RTO retransmits.

With this patch:
1) txhash is changed with every SYN retransmits
2) txhash is changed with every RTO.

The result is that we can start re-routing around failed (or very
congested paths) as soon as possible. Otherwise application health
checks may fail and the connection may be terminated before we start
to change txhash.

v4: Removed sysctl, txhash is changed for all RTOs
v3: Removed text saying default value of sysctl is 0 (it is 100)

TCP RTO & skb->hash



net.ipv6.auto_flowlabels

0: automatic flow labels are completely disabled

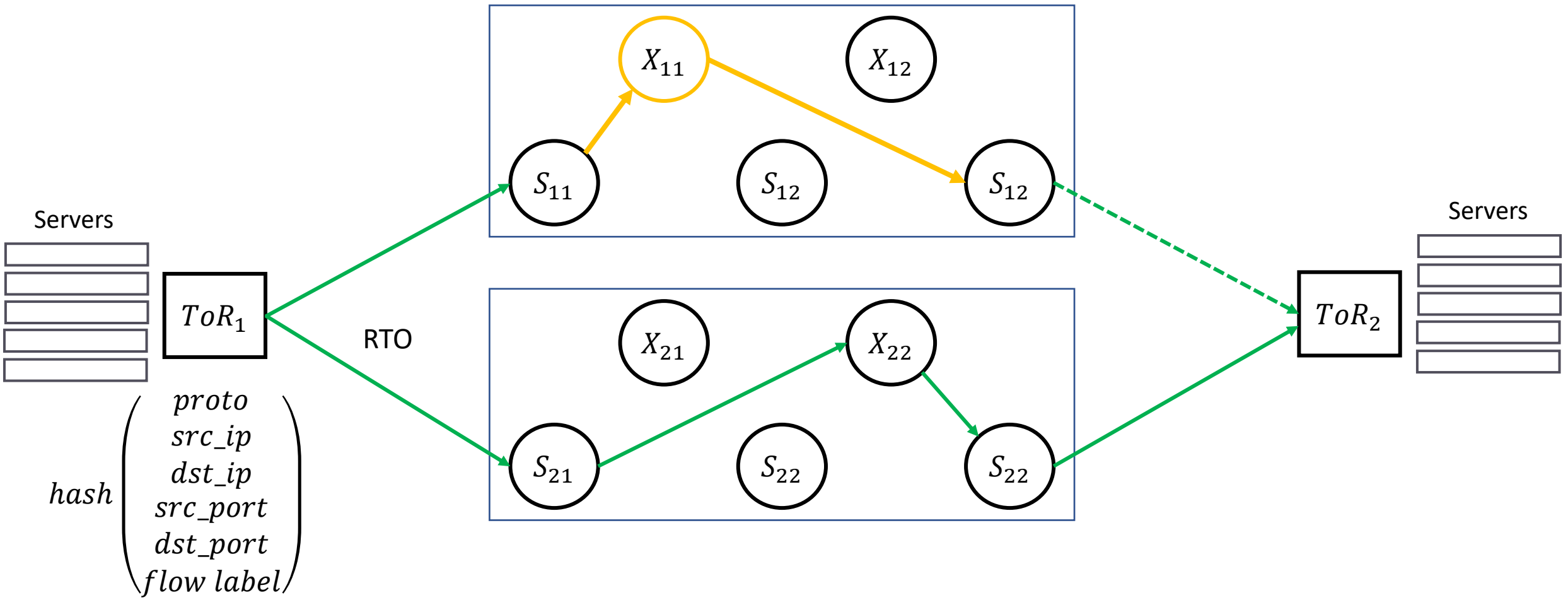
1: automatic flow labels are enabled by default, they can be disabled on a per socket basis using the IPV6_AUTOFLOWLABEL socket option

2: automatic flow labels are allowed, they may be enabled on a per socket basis using the IPV6_AUTOFLOWLABEL socket option

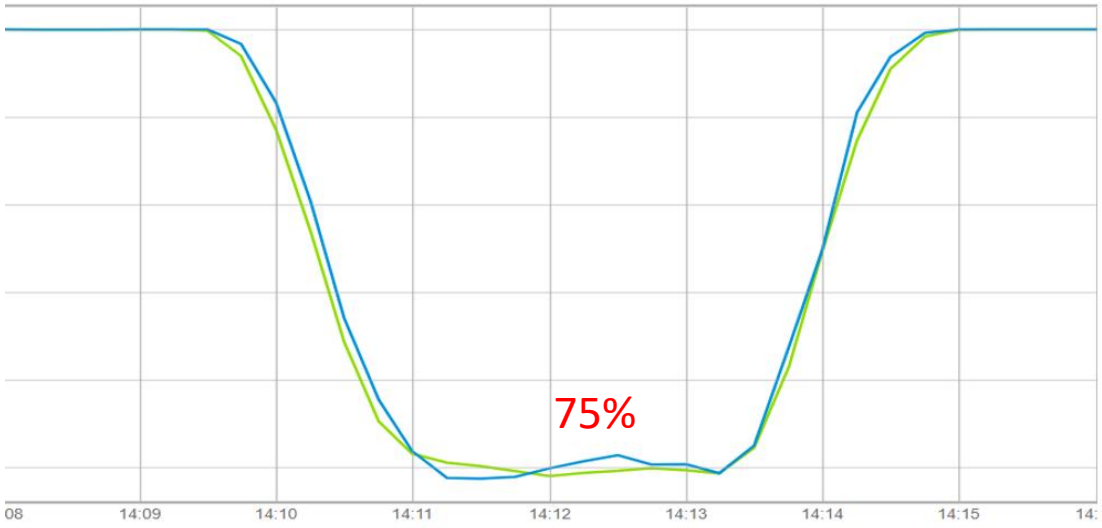
3: automatic flow labels are enabled and enforced, they cannot be disabled by the socket option

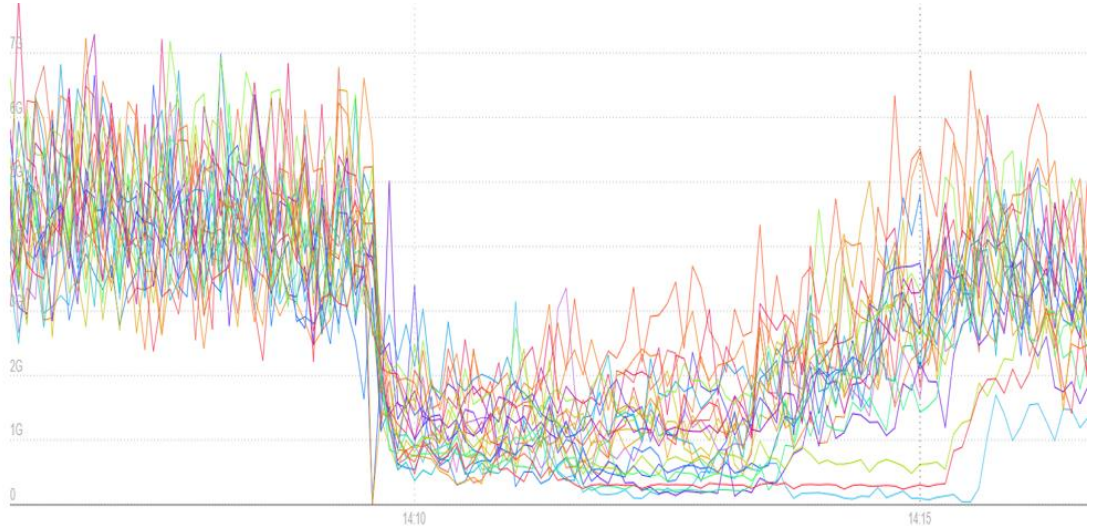
Default: 1

Unhappy TCP Flow Becomes Happier



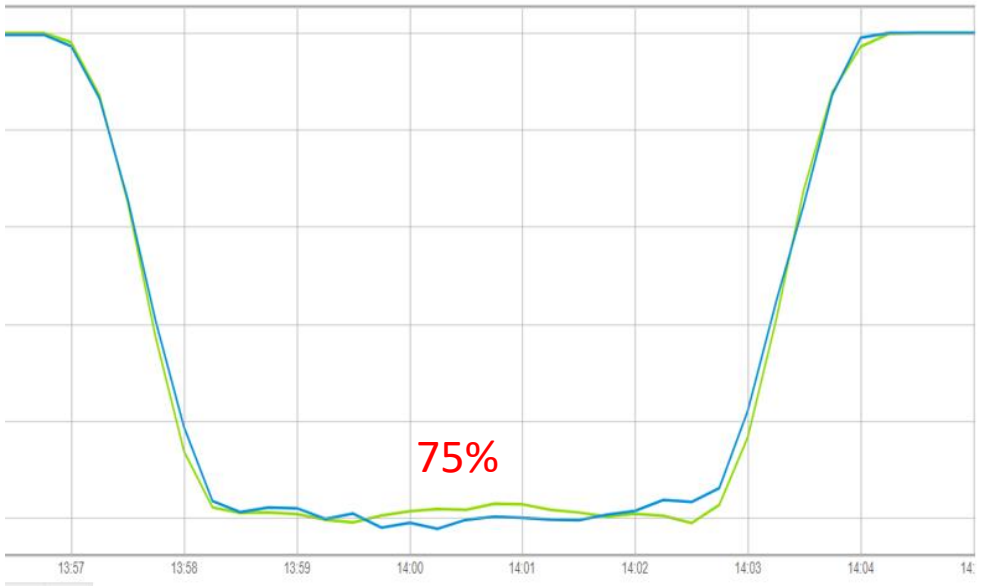
Evaluation: Without Flow Label

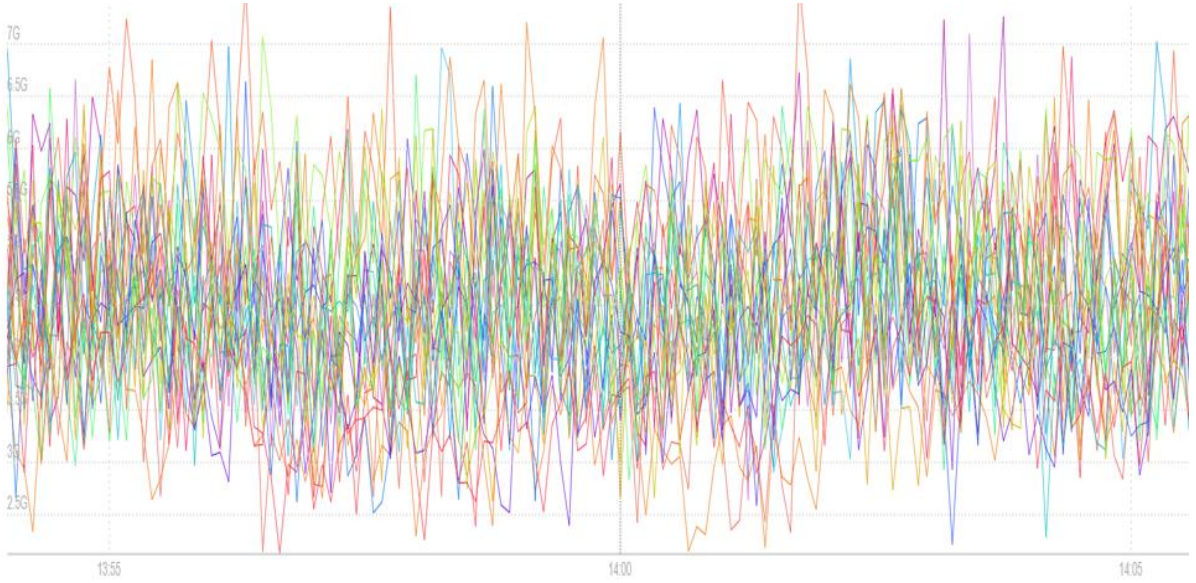




One of four ToR uplinks drops packets, significant service degradation

Evaluation: Flow Label + eBPF

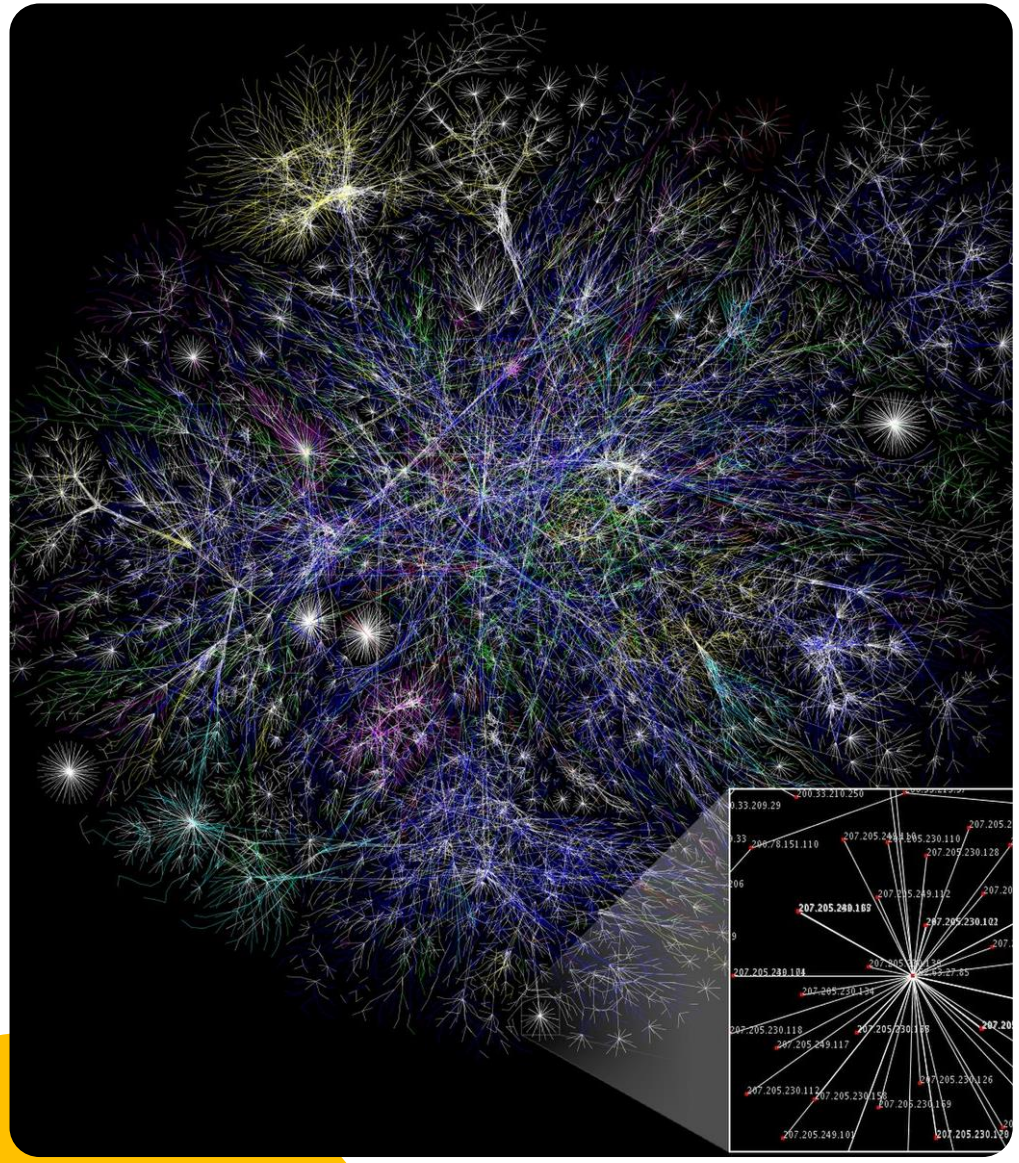




One of four ToR uplink drops packets, no effect on the service!

Self-healing Datacenter: Cookbook

- Does it scale? **Yes!**
- Does it have many paths? **Yes!**
- Does it have fault tolerance? **Use IPv6! Use flow label!**
- How do I change RTO? **eBPF is the answer!**
- Without documentation!



Theory Internet: Many-Many Paths

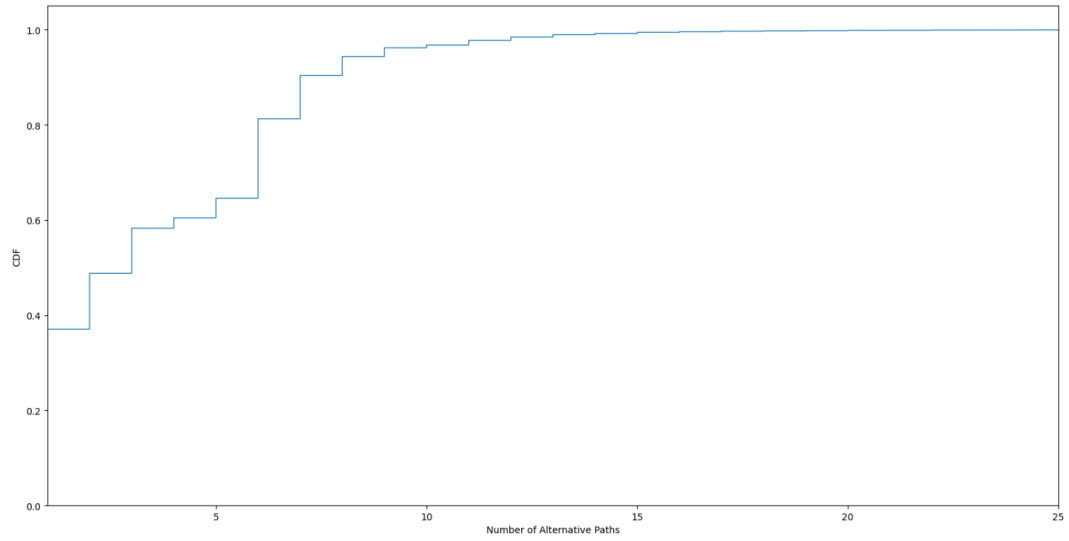
- Multihomed at the edge;
- Multiple connections between peers;
- Multiple connection with upstreams;

Real Internet: Many-Many Paths

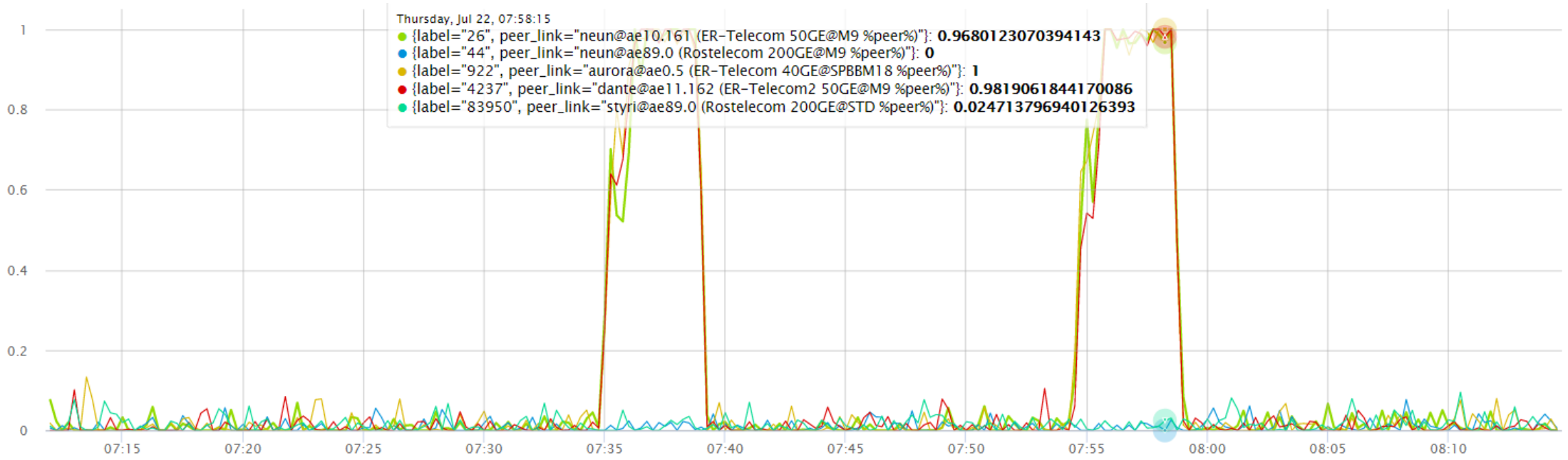
Average number of best paths: 3.8

Maximum number of best paths: 44

>60% of prefixes have more then 1 path

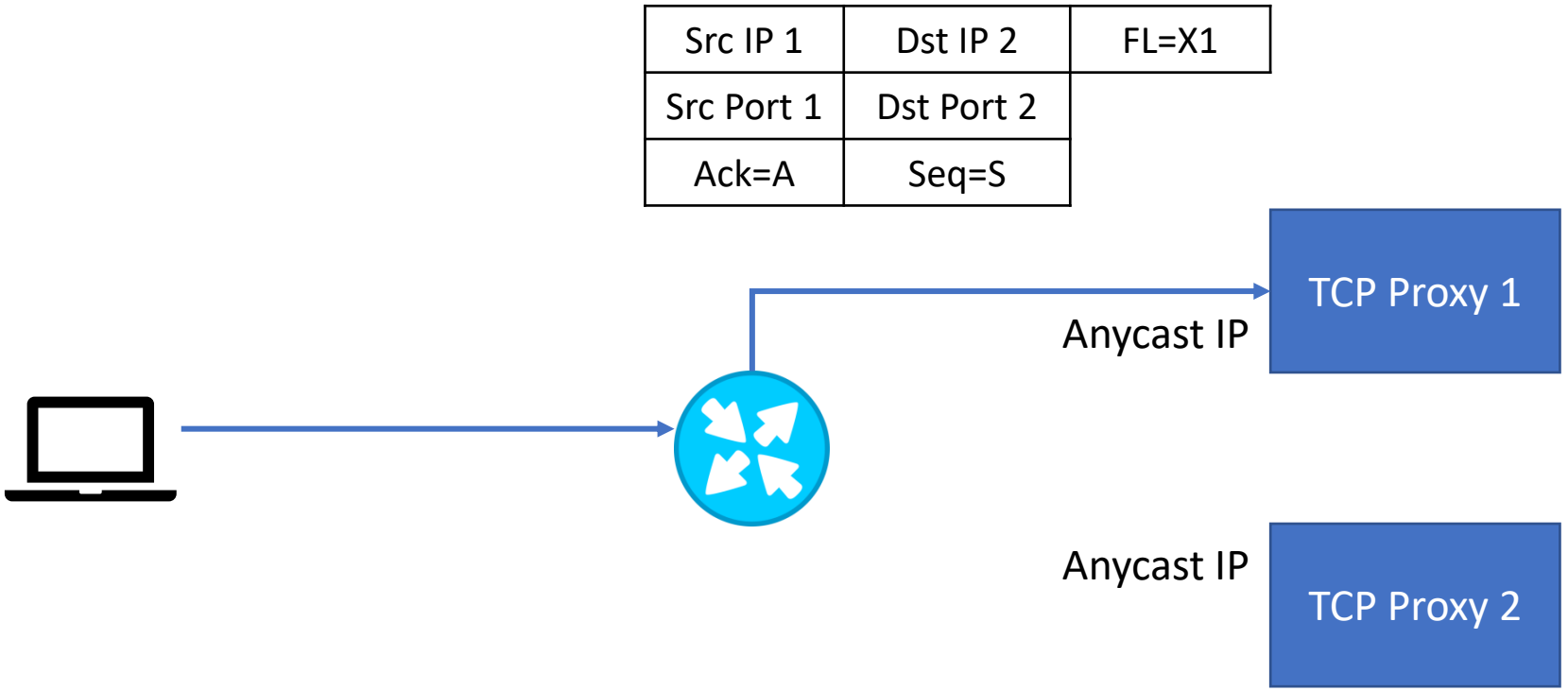


A Real Outage

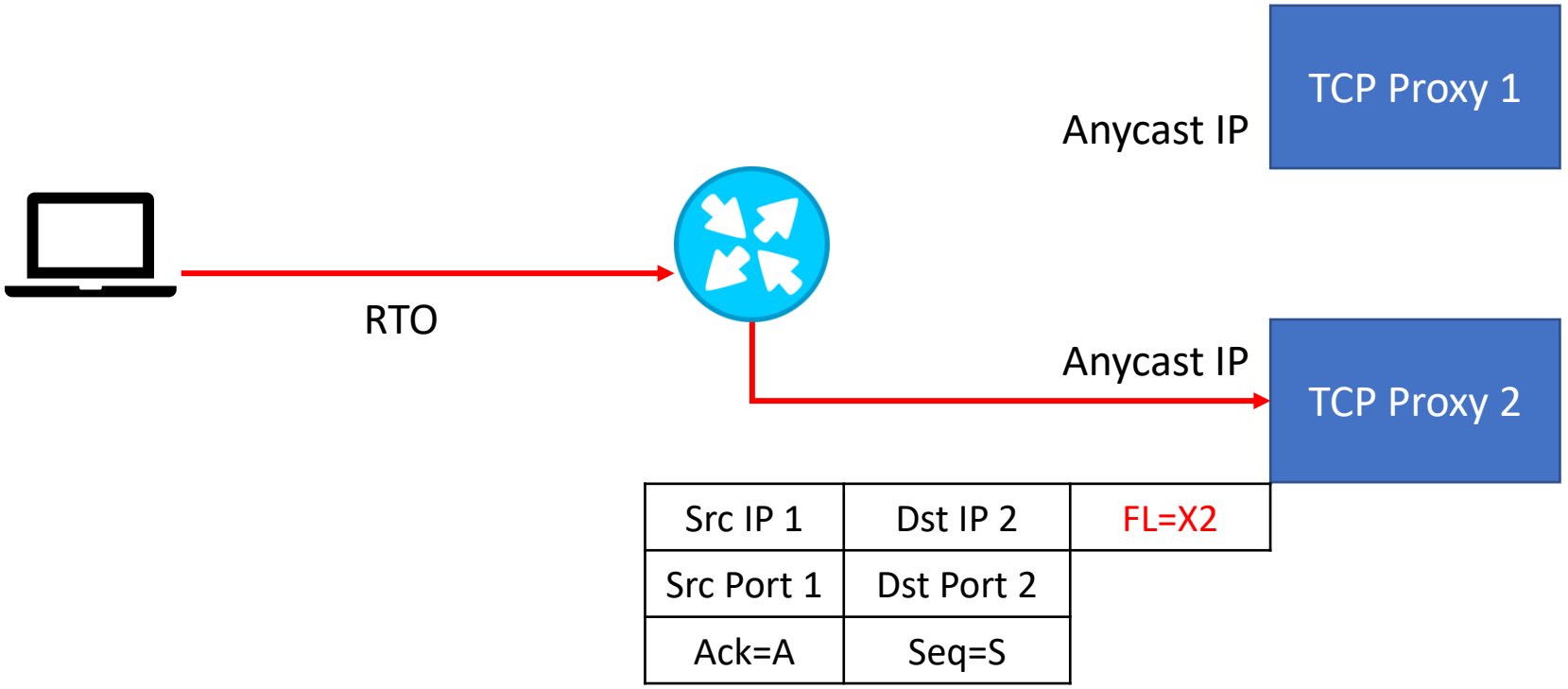



IT'S AN ANYCAST
imgflip.com

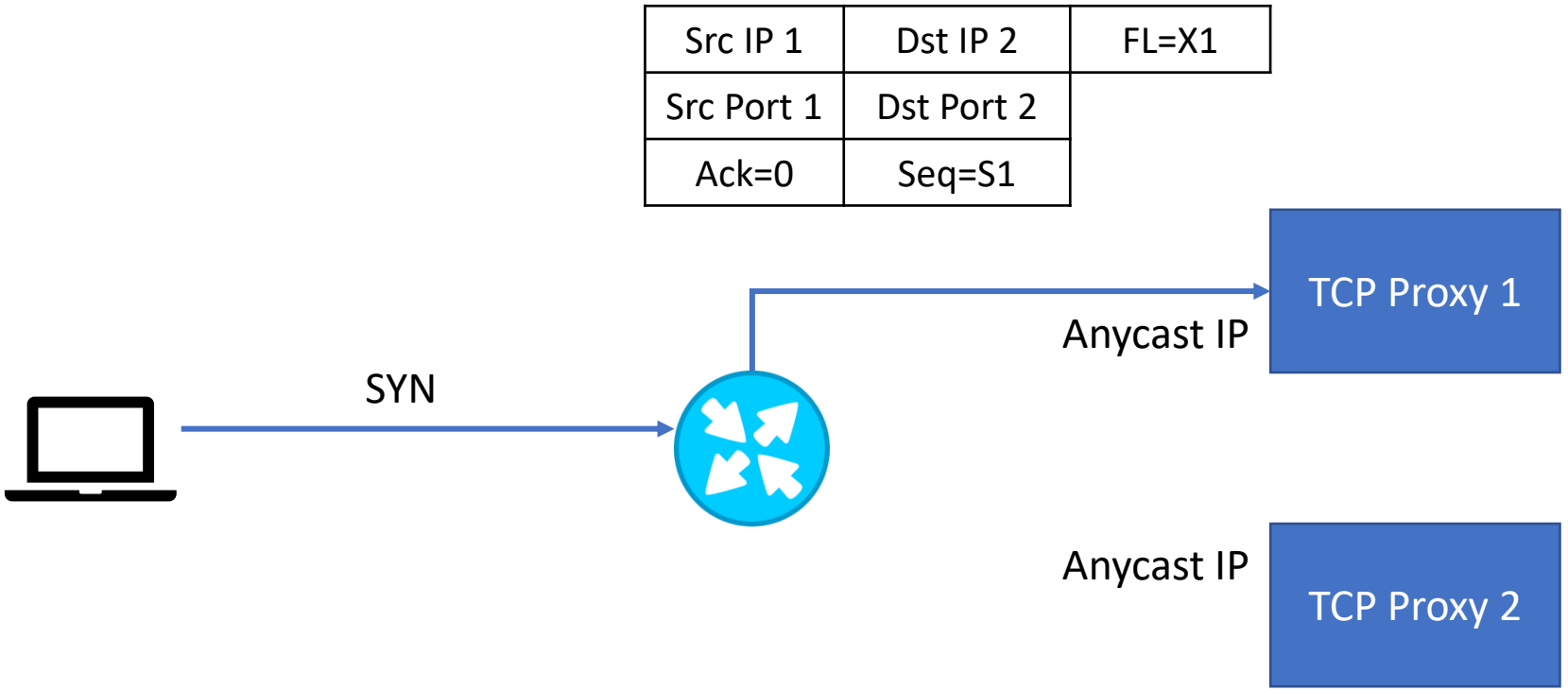
RTO & Anycast


Src IP 1
Dst IP 2
FL=X1
Src Port 1
Dst Port 2
Ack=A
Seq=S
TCP Proxy 1
Anycast IP
Anycast IP
TCP Proxy 2

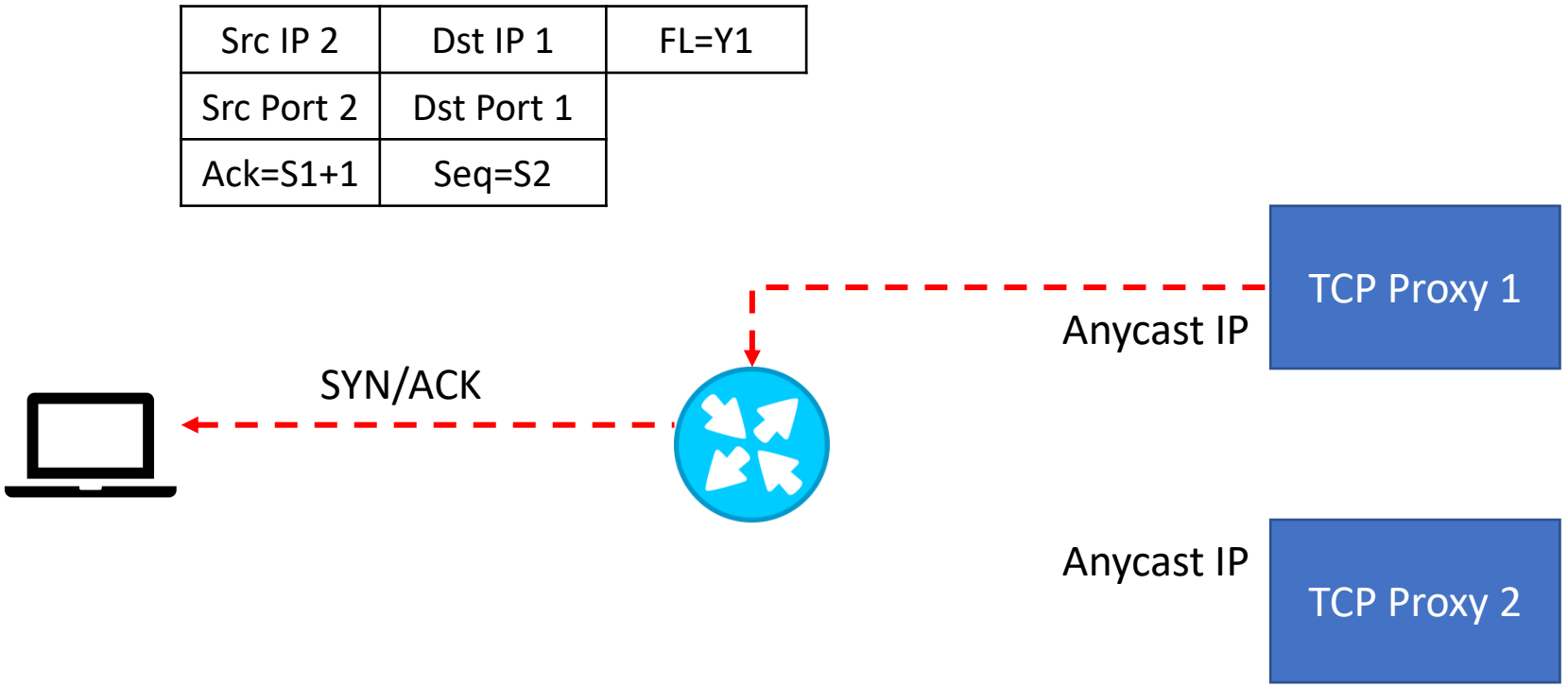
RTO & Anycast


TCP Proxy 1
Anycast IP
RTO
Anycast IP
TCP Proxy 2
Src IP 1
Dst IP 2
FL=X2
Src Port 1
Dst Port 2
Ack=A
Seq=S

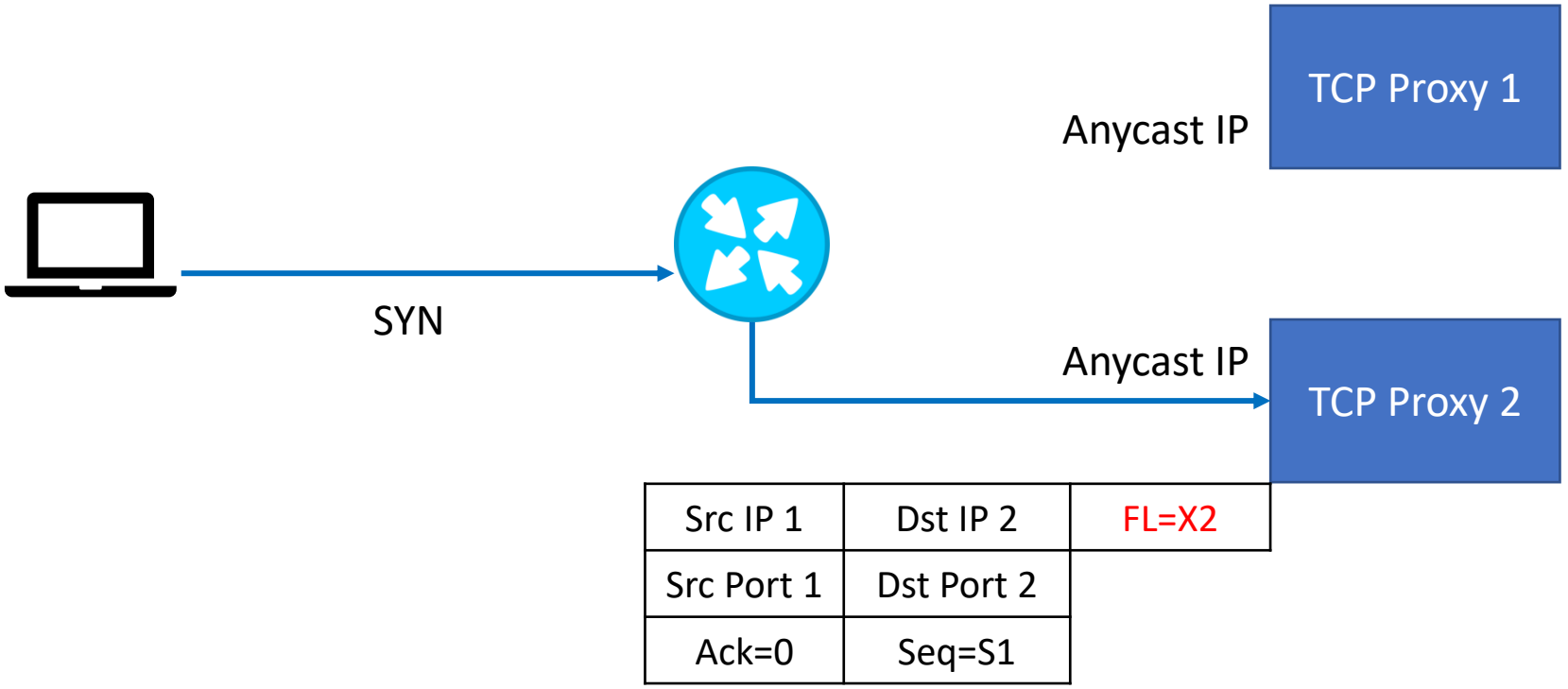
SYN RTO & Anycast



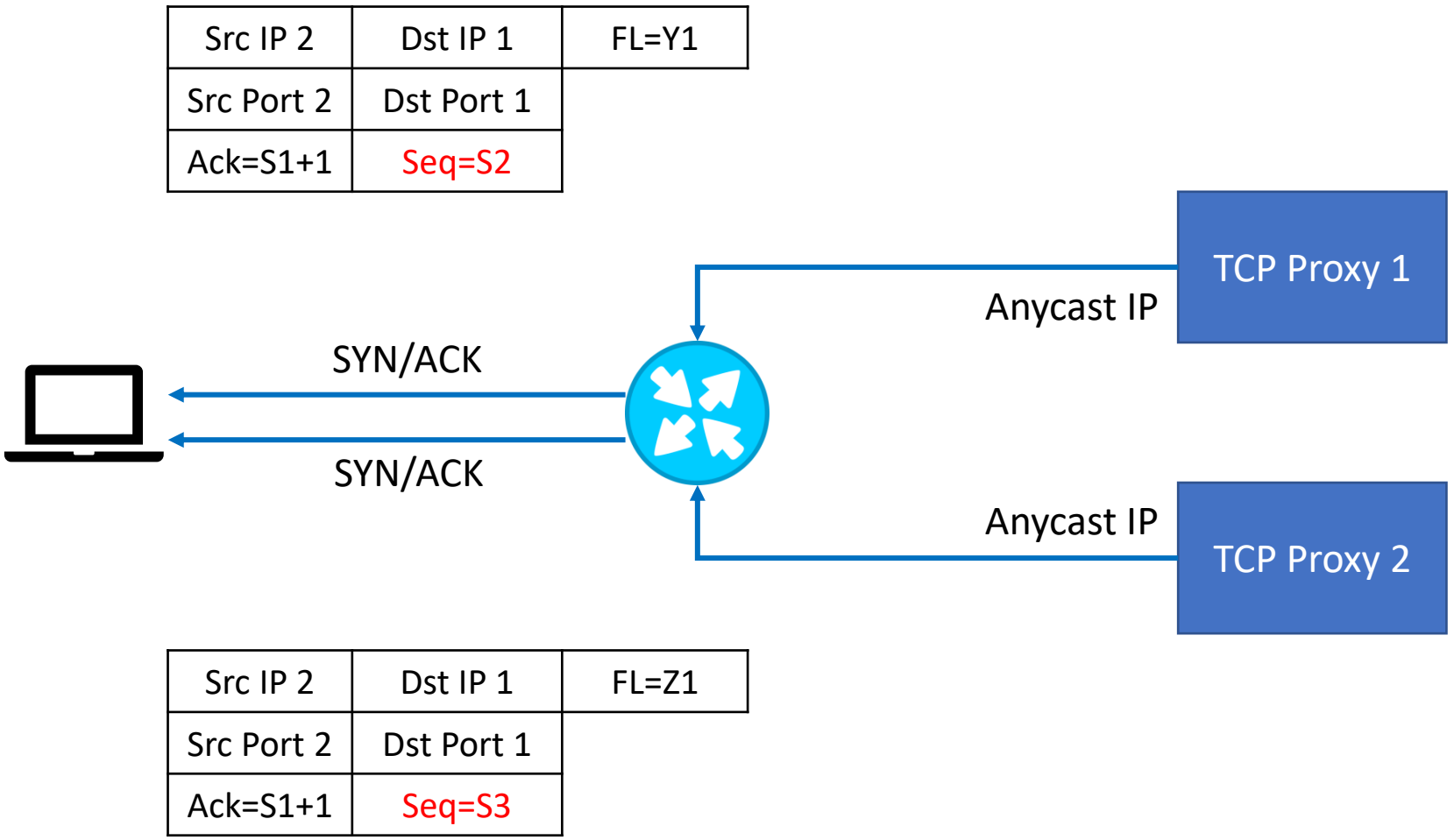
SYN RTO & Anycast



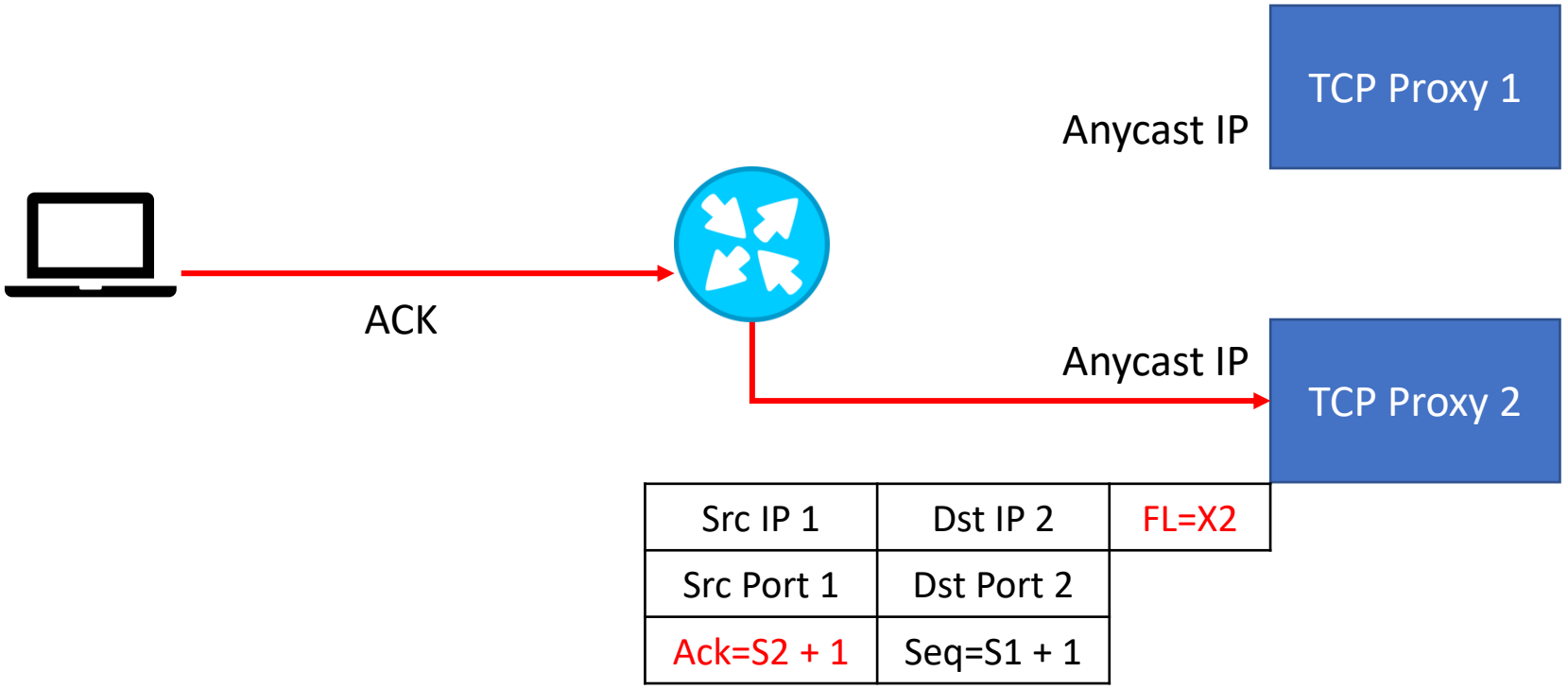
SYN RTO & Anycast



SYN RTO & Anycast



SYN RTO & Anycast



Flow Label: Safe Mode

Client – sends SYN, Server – responds with SYN&ACK

- In case of SYN_RTO or RTO events Server SHOULD recalculate its TCP socket hash, thus change Flow Label. This behavior MAY be switched on by default;
- In case of SYN_RTO or RTO events Client MAY recalculate its TCP socket hash, thus change Flow Label. This behavior MUST be switched off by default;

Self-healing ~~Datacenter~~ TCP: Cookbook

- Flow label provides is a way to ‘jump’ from a failing path;
- Already works in controlled environment;
- Can disrupt TCP connection with stateful anycast services;
- We need to change Linux defaults!
- This time we need to document it!