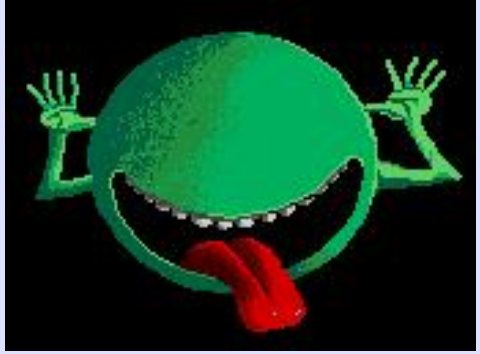
Measuring RPKI Repositories

NANOG / Dallas
2012.10.22

Randy Bush <randy@psg.com>



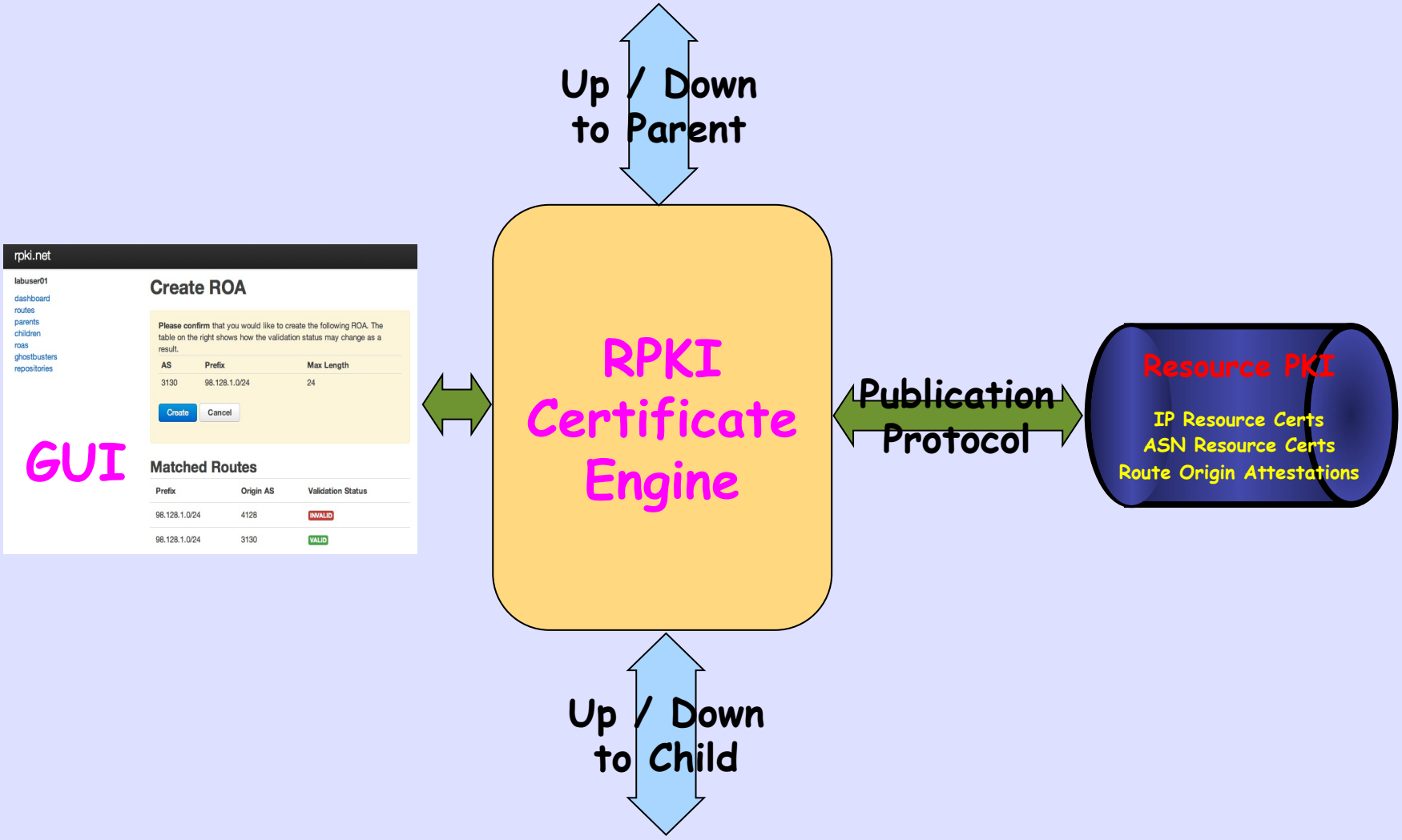
Don't Panic

- I am an Engineer, we always think about the problems
- I am a Researcher, we are only interested in the problems
- The RPKI is going really well
- But I want to talk about the problems

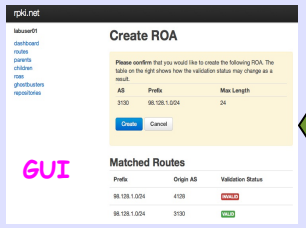
We're All Friends

Review of RPKI Structure

Publishing / Issuing Party



Issuing Parties Relying Parties



IANA

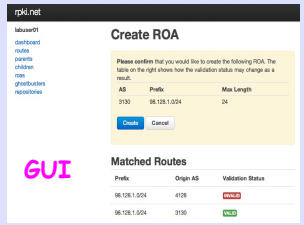
Publication Protocol

Resource PKI

Trust Anchor

Up Down

SIA Pointers



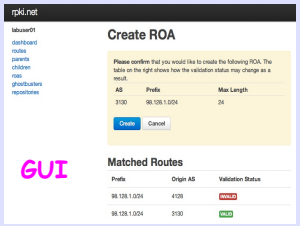
APNIC

Publication Protocol

Resource PKI

Up Down

SIA Pointers



IIJ

Publication Protocol

Resource PKI

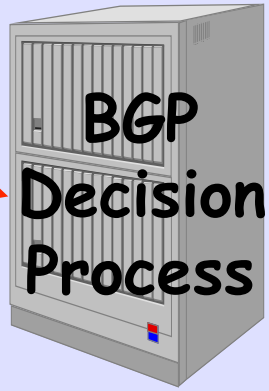
RCynic Gatherer

Validated Cache

Pseudo IRR

```
route:    147.28.0.0/16
descr:    147.28.0.0/16-16
origin:   AS3130
notify:   irr-hack@rpki.net
mnt-by:   MAINT-RPKI
changed:  irr-hack@rpki.net 20110606
source:   RPKI
```

NOC Tools



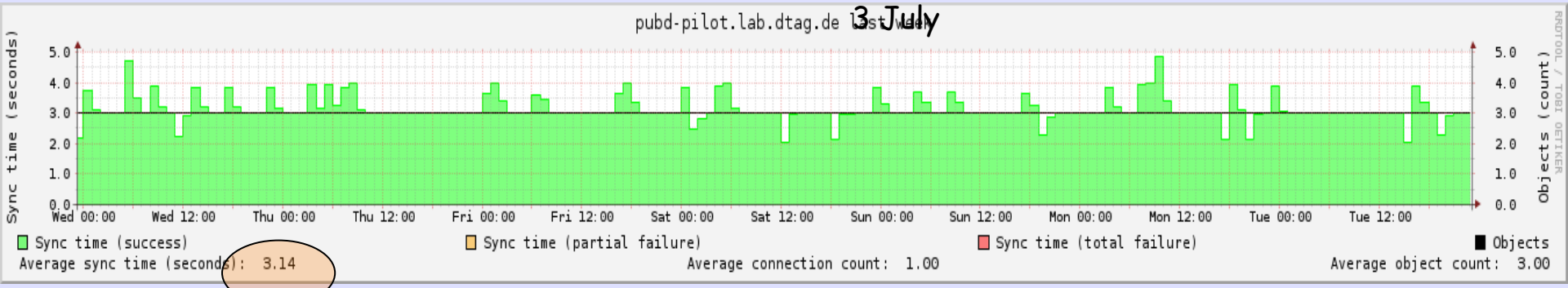
My Routing Relies on It!

- If my routing relies on the RPKI, then I care a lot about publication reliability
- Of course, good relying party software will expect failures, so this is not a killer
- But when we look at current publication, much is not operational quality
- This has to be fixed

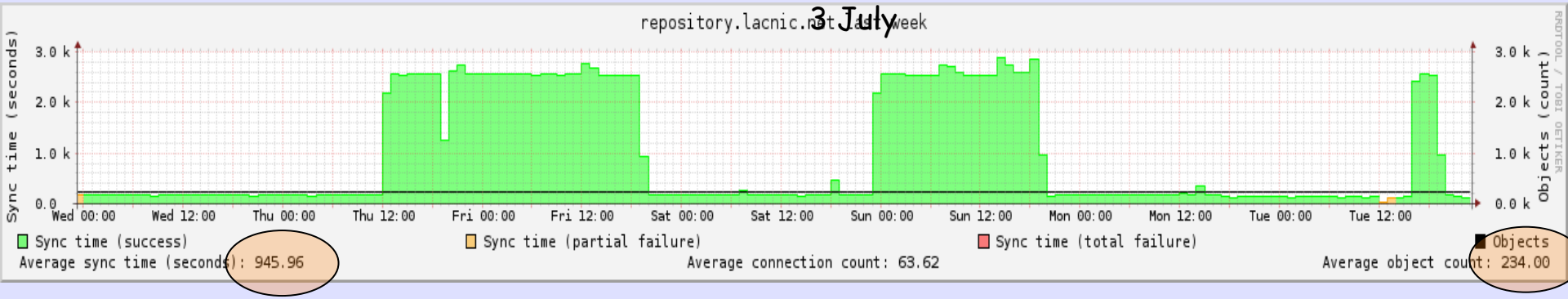
These Graphs are from rpki.net's Relying Party Software Web Page it Makes for You

Not Bad

An ISP



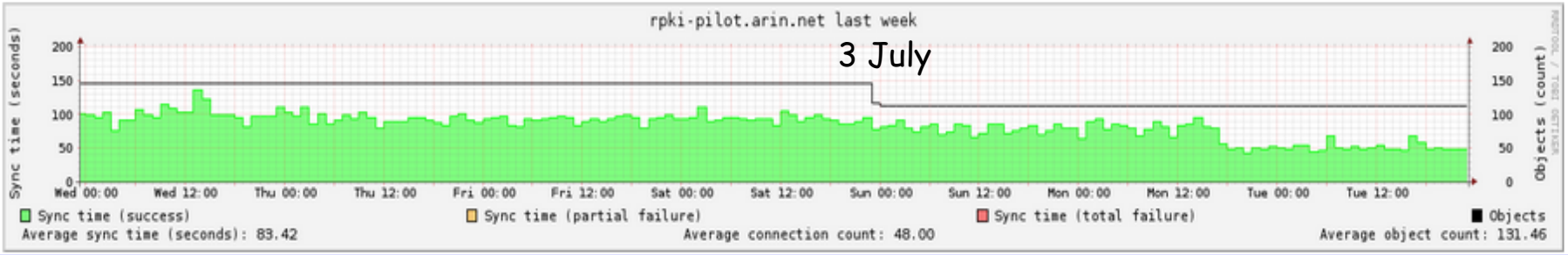
An RIR



Not So Good

Overview for repository rpki-pilot.arin.net

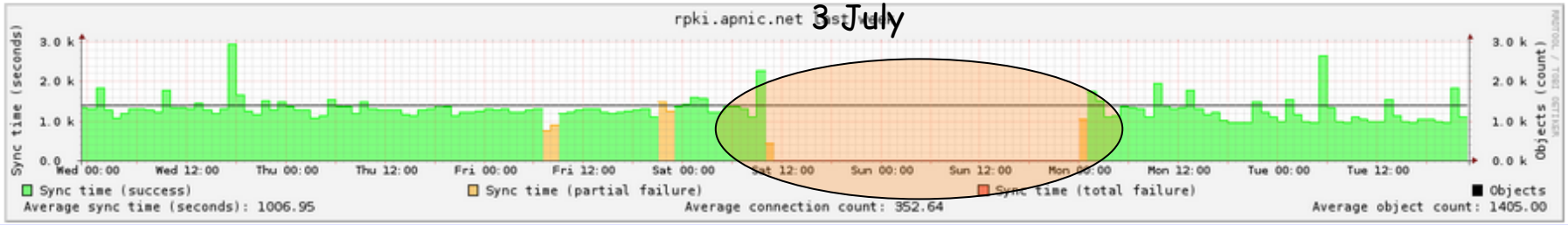
	certificate has expired	Bad keyUsage	Certificate failed validation	CRL not yet valid	CRLDP doesn't match issuer's SIA	Manifest not yet valid	Object rejected	EE certificate with 1024 bit key	Nonconformant X.509 issuer name	Nonconformant X.509 subject name	rsync partial transfer	Stale CRL or manifest	Tainted by stale CRL	Tainted by stale manifest	Tainted by not being in manifest	Non-rsync URI in extension	Object accepted	rsync transfer succeeded
																		48
current .cer										18					48		48	
current .crl																	2	
current .mnf		48					48		18	11								
current .roa		14					14		5	5					14			
Total		62					62		23	34					62		50	48



Very Bad

Overview for repository rpki.apnic.net

	certificate has expired	Bad keyUsage	Certificate failed validation	CRL not yet valid	CRLDP doesn't match issuer's SIA	Manifest not yet valid	Object rejected	EE certificate with 1024 bit key	Nonconformant X.509 issuer name	Nonconformant X.509 subject name	rsync partial transfer	Stale CRL or manifest	Tainted by stale CRL	Tainted by stale manifest	Tainted by not being in manifest	Non-rsync URI in extension	Object accepted	rsync transfer succeeded
																		459
current .cer									457	1							459	
current .crl									1								459	
current .mft									1								459	
current .roa								15									28	
Total								**15**	**459**	**1**							**1405**	**459**



- They do not monitor and have no real NOC
- They do not work weekends
- I had to write a friend in APNIC Engineering

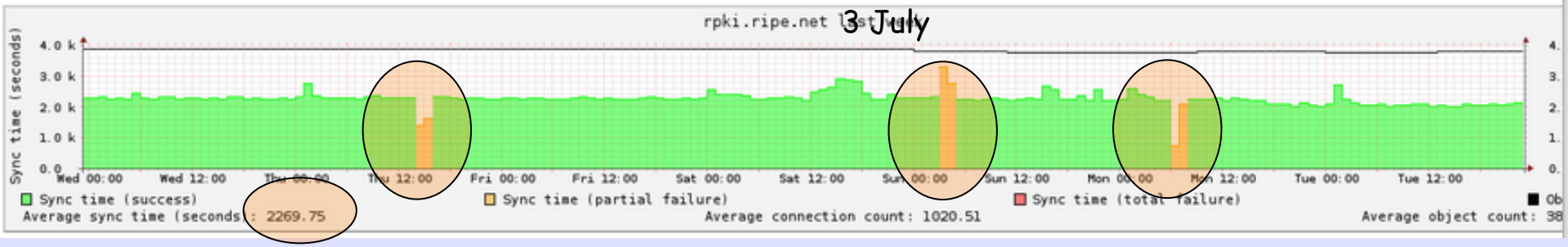
RIPE Stayed Up

Repository details for rpki.ripe.net 2012-07-03T23:10:13Z

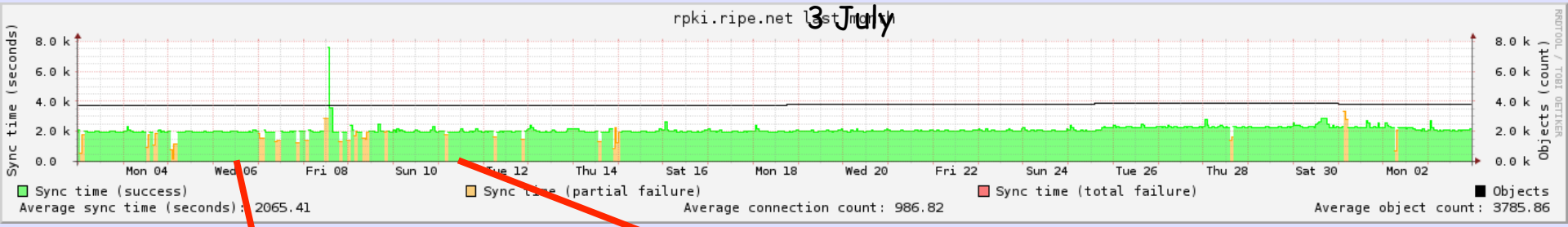
Overview Repositories Problems All Details

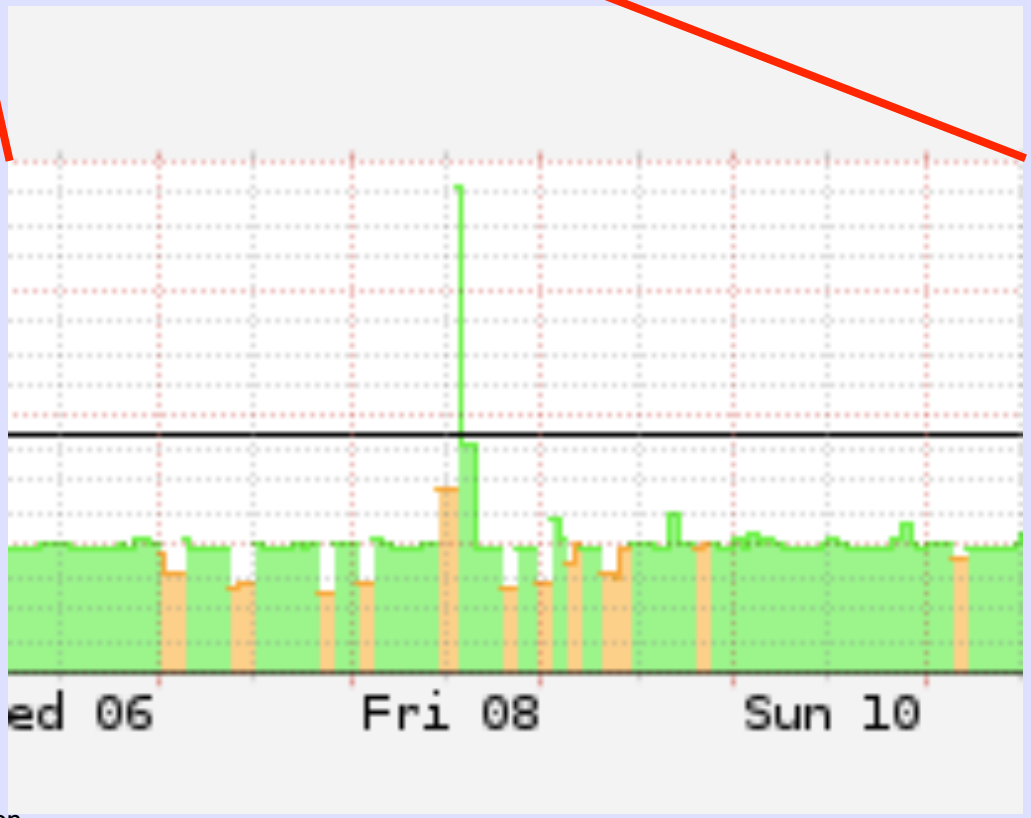
	certificate has expired	Bad keyUsage	Certificate failed validation	CRL not yet valid	CRLDP doesn't match issuer's SIA	Manifest not yet valid	Object rejected	EE certificate with 1024 bit key	Nonconformant X.509 issuer name	Nonconformant X.509 subject name	rsync partial transfer	Stale CRL or manifest	Tainted by stale CRL	Tainted by stale manifest	Tainted by not being in manifest	Non-rsync URI in extension	Object accepted	rsync transfer succeeded
																		1036
current .cer									1033	101							1035	
current .crl									101								1035	
current .mft									101	1							1035	
backup .roa								17	6						35		35	
current .roa								500	78								693	
Total								**517**	**1319**	**102**					**35**		**3833**	**1036**

rpki.ripe.net over last week



RIPE had Bad History

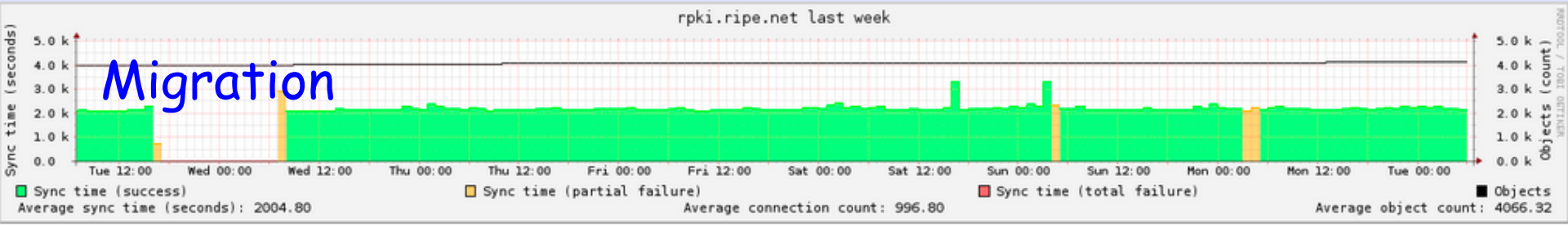


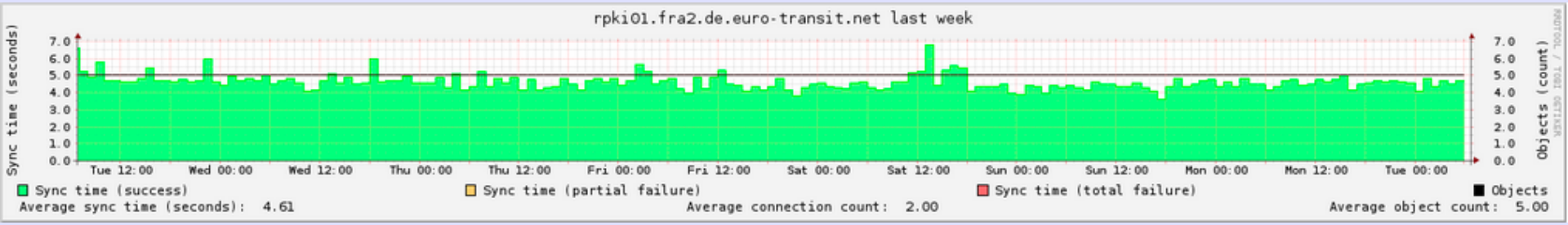


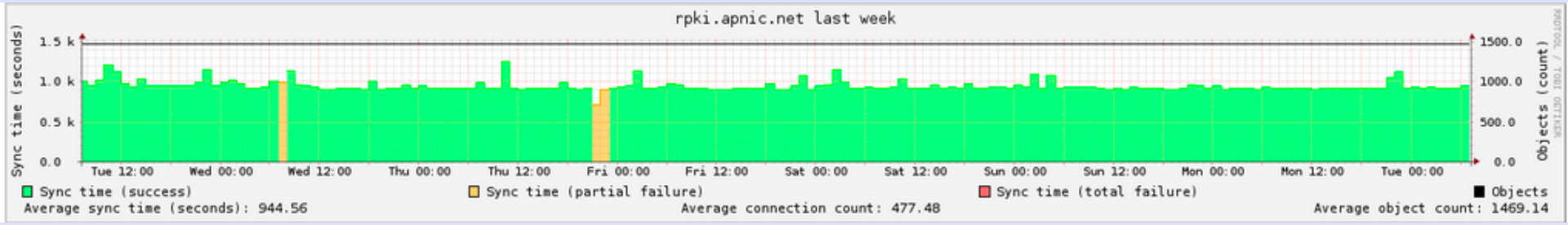
Cause

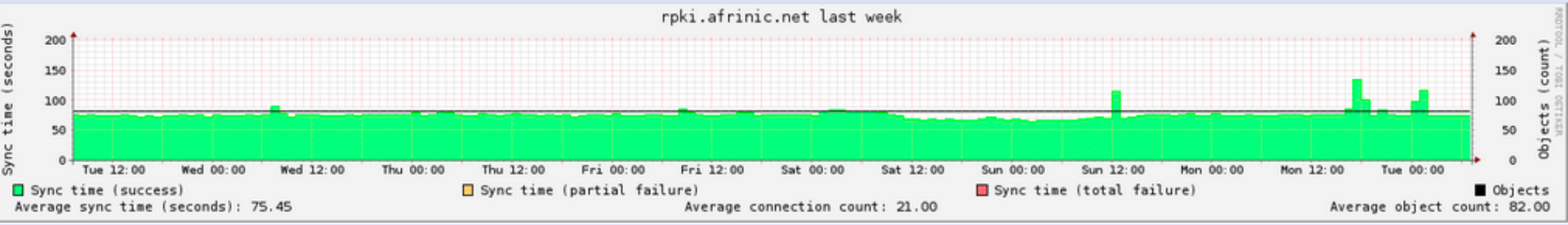
- This was an NFS problem (NFS is Evil!)
- It went on for months
- RPKI.NET logs had full detail showing "NFS"
- But "Nothing Can Be Wrong at the RIR"
- Finally it was fixed, but small problems remain

Three Weeks Ago









AfriNIC is Just Weird

•871553 -rw-r--r-x 4 rcynic rcynic 1969 Feb 17 13:26:29 2012 /usr/home/rpki/rcynic/data/authenticated.2012-07-11T00:00:00Z/rpki.afrinic.net/member_repository/F3634D22/92EF8890119911E0A59EB577833A7E19/79FBE550468F11E19086CABE31FFE8A0.roa

•871602 -rw-r--r-x 4 rcynic rcynic 2009 Feb 17 13:26:26 2012 /usr/home/rpki/rcynic/data/authenticated.2012-07-11T00:00:00Z/rpki.afrinic.net/member_repository/F3634D22/92EF8890119911E0A59EB577833A7E19/82331D8C6C2011E0890EBAC0A0C76497.roa

And we wrote to them multiple times and received only snarky responses

The RIRs are PTTs, "There can be no problem"

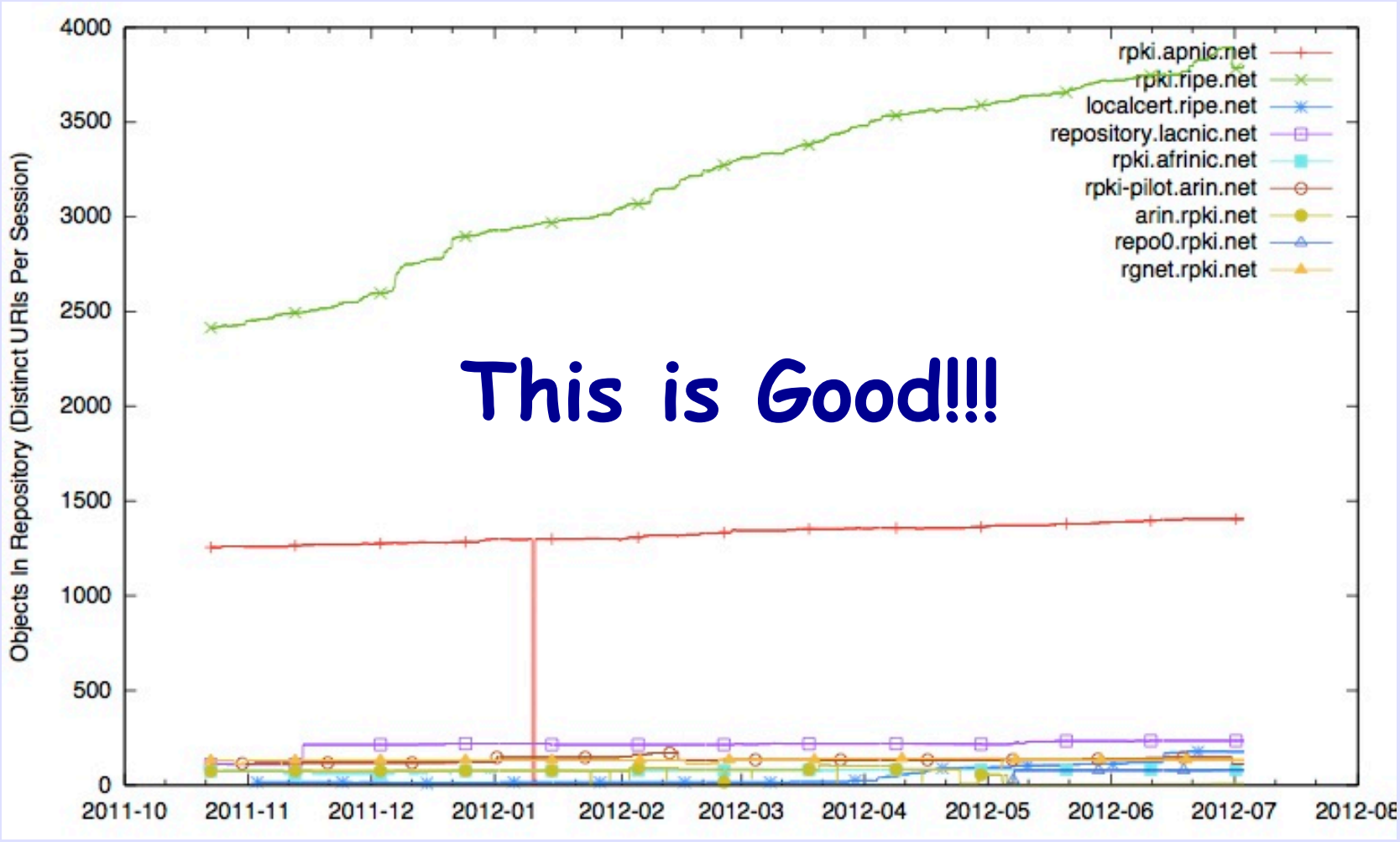
Relying Party Software Saves Us

- Of course, good relying party software will expect failures, so this is not a killer
- rpki.net relying party software uses old data if it can not fetch new
- As RPKI data are fairly stable, this is OK
- But RIPE's in-addr disaster lasted more than five days!

Some Statistics

Again, from rpki.net Relying Party Software

Number of Objects



Conclusions

- RPKI Deployment is serious, especially in the RIPE region. Thanks Alex and Tim!
- RIRs are not Operator Quality/Reliability
- JPNIC hopes to set an example, we'll see
- APNIC & RIPE Publication Structure needs to be fixed, and RIPE's is being fixed
- Relying Party software works around these
- More Measurement and Monitoring

Where to Go?

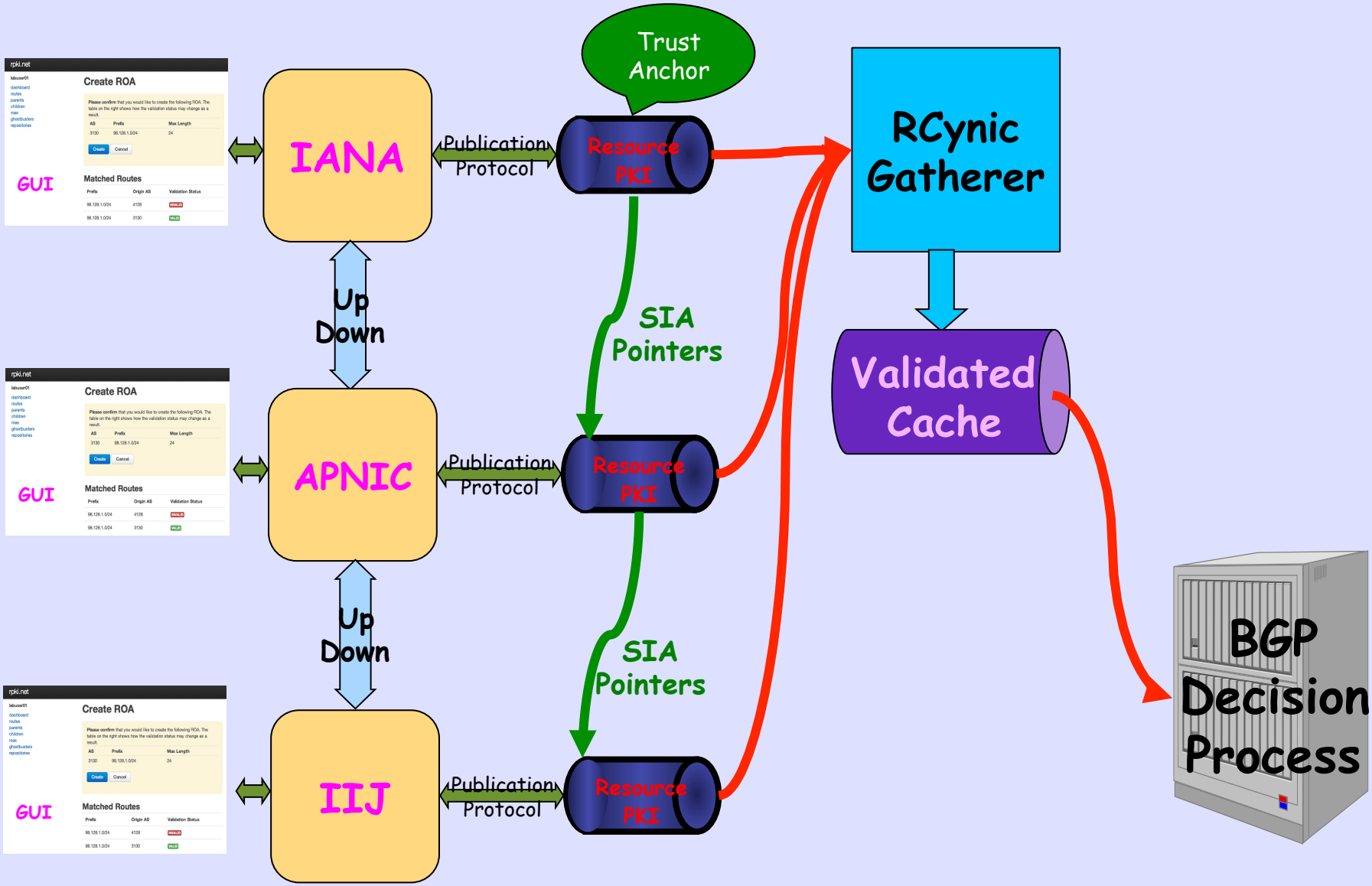
- Asking All Publishers to provide 99.999% uptime is silly, expensive, and doomed
- The Internet is about building a reliable network from unreliable components
- We need to develop deployment based on distributed systems which give reliable service through diversity and redundancy

RPKI Propagation Emulation Measurement: an Early Report

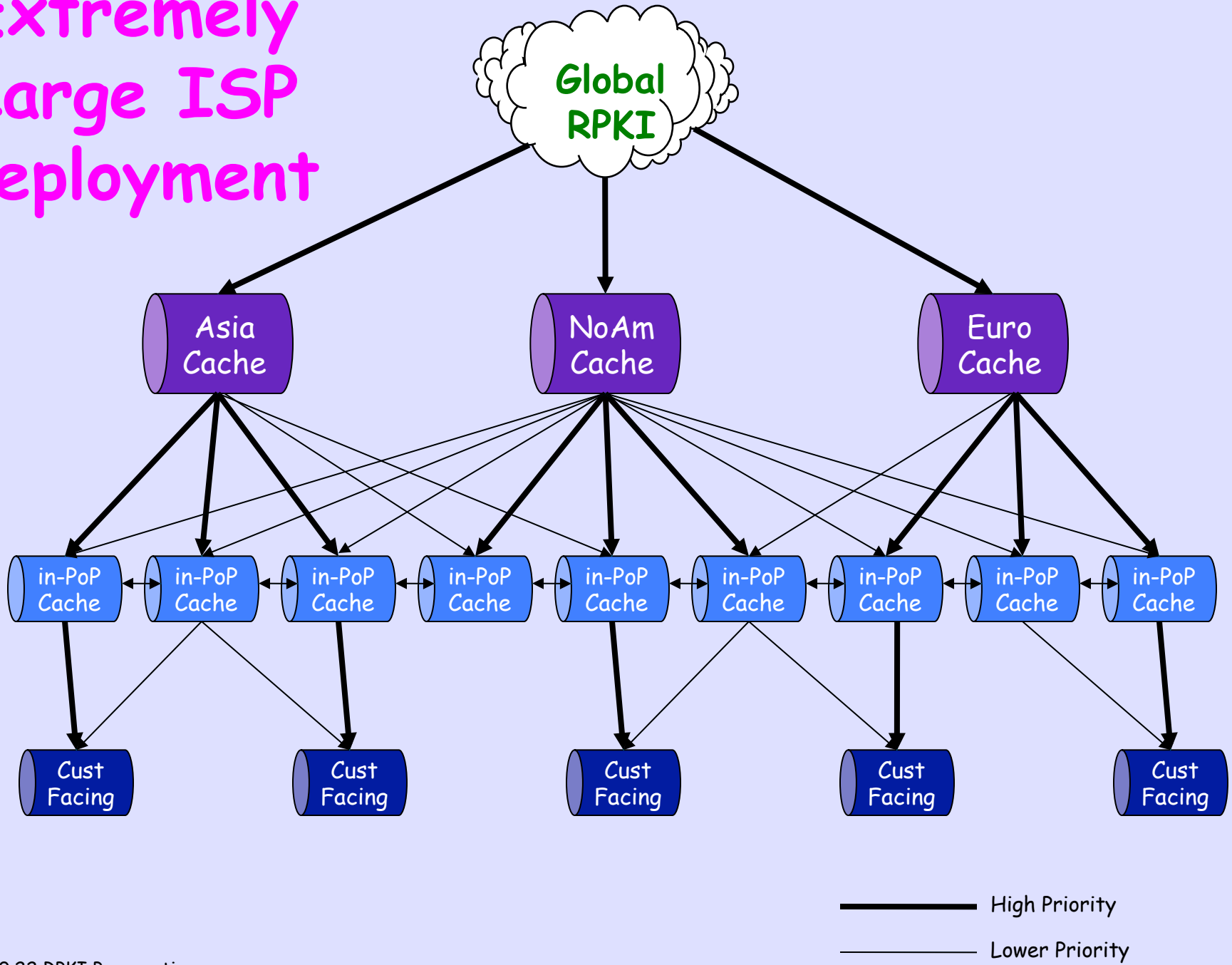
NANOG / Dallas
2012.10.22

Iain Phillips <i.w.phillips@lboro.ac.uk>
Olaf Maennel <o.m.maennel@lboro.ac.uk>
Debbie Perouli <depe@cs.purdue.edu>
Rob Austein <sra@hactrn.net>
Cristel Pelsser <cristel@iij.ad.jp>
Keiichi Shima <keiichi@iijlab.net>
Randy Bush <randy@psg.com>

Issuing Parties Relying Parties



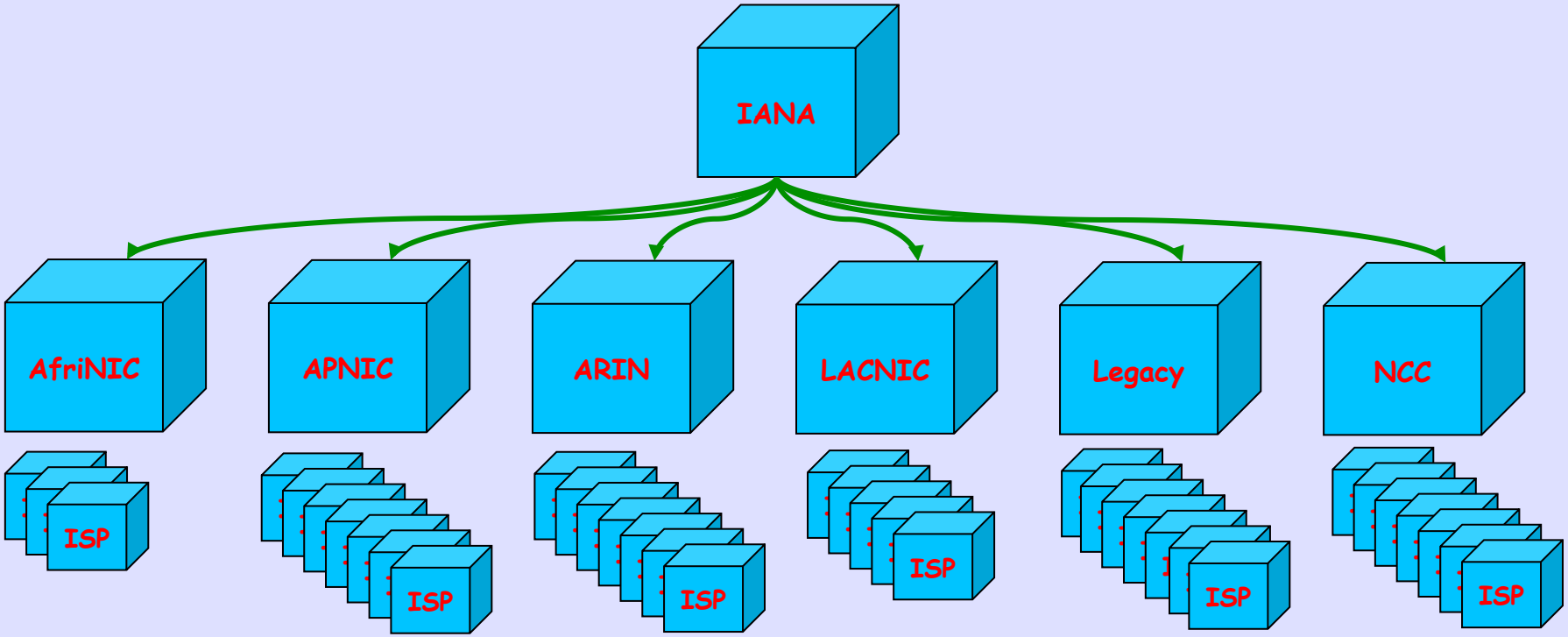
Extremely Large ISP Deployment



Questions

- What are the propagation characteristics of Relying Part (RP) infrastructure?
- How sensitive is propagation to RP and cache fetch timers?
- How much is propagation and how much is validation?
- Is it sensitive to inter-cache RTT?

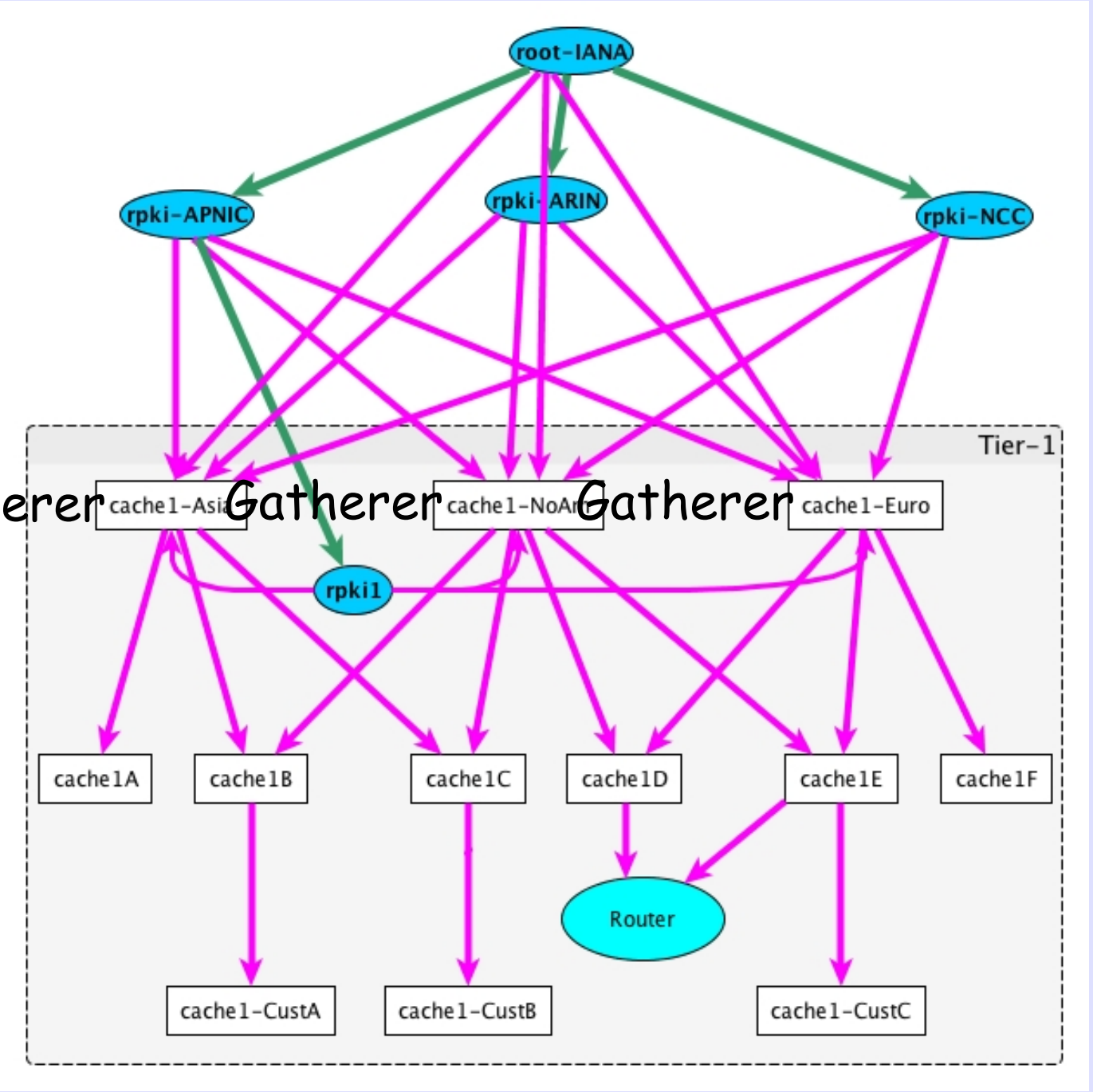
Publication Hierarchy



Not Critical as our Interest is Inter-Cache

Publication

Inter-Cache



Caches

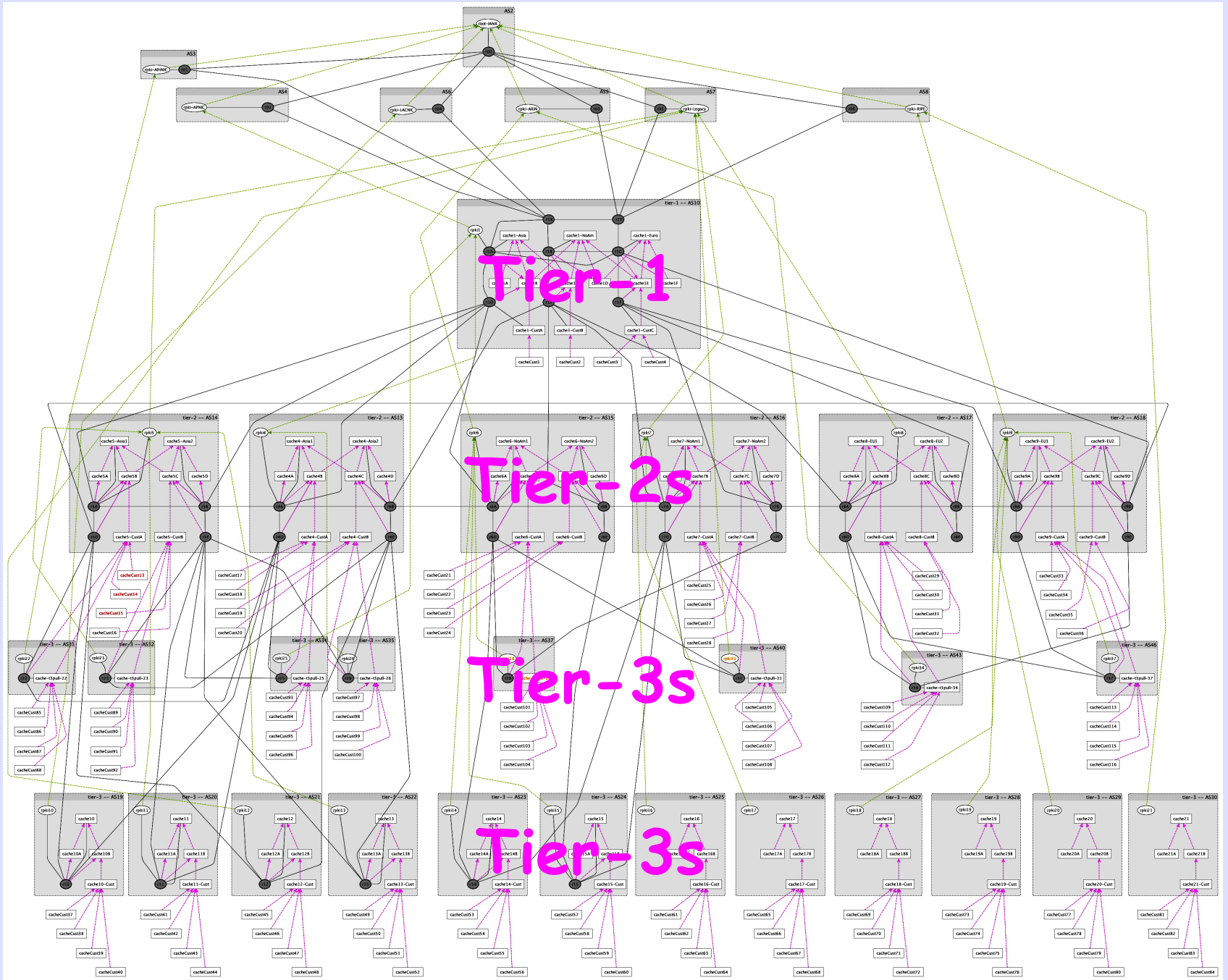
- Each cache rsyncs entire data from *parent* cache(s) or gatherers
- Each cache has a root TAL
- Every cache validates the data it has fetched

What is Propagation?

- The time from when a CA publishes an object (Cert or ROA) to when a Relying Party receives it.
- A Relying Party is a validated cache or a router via the rpki-rtr protocol.
- Measured by caches and routers logging every received object.

Architecture

- Do not care about routers, BGP, … as they do not contribute to measurement
- Use *pseudo-router*, an rpki-rtr client which logs each incoming VRP (ROA PDU)
- Caches also log receipt of objects


Tier-1
Tier-2s
Tier-3s
Tier-3s

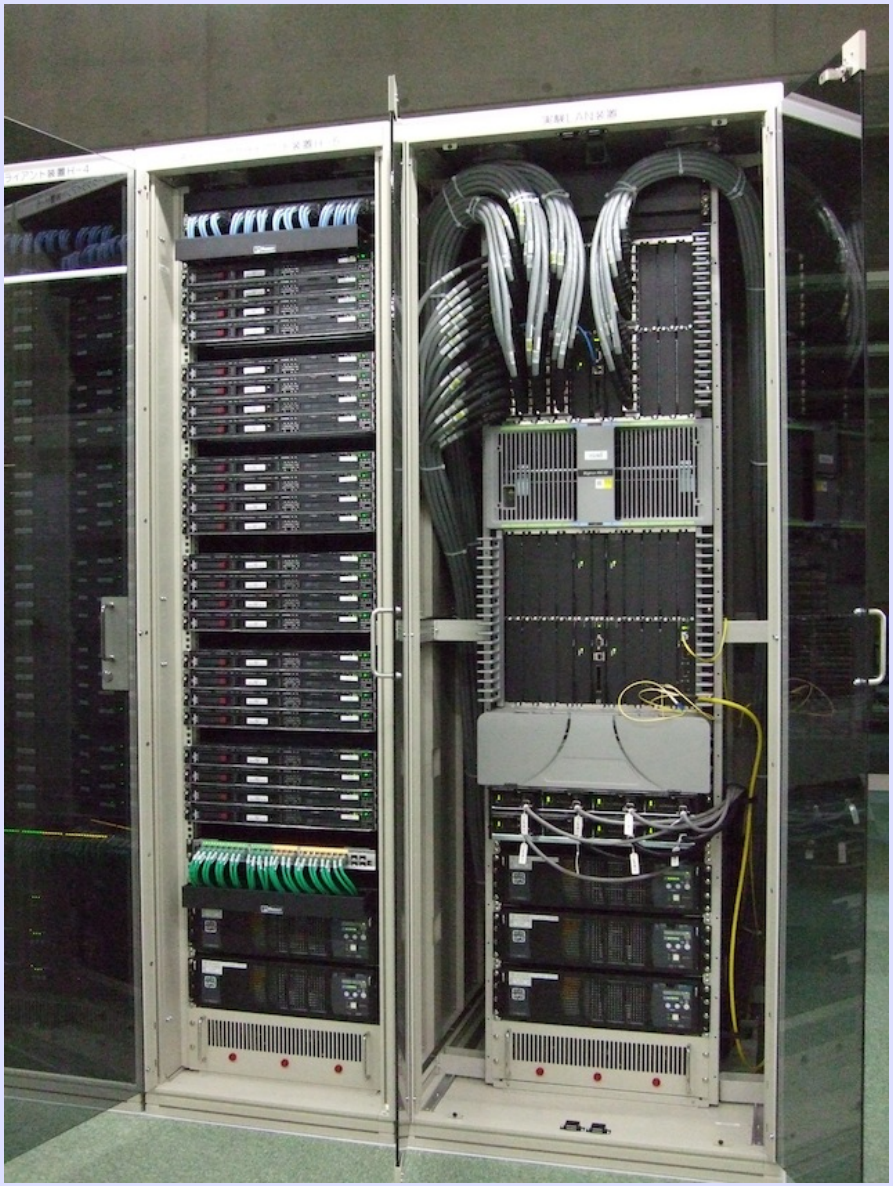
Small Testbed

- ~~3~~ 1 Tier-1, each with 3 Gatherers
- 6 Tier-2 per Tier-1, each with 2 Gatherers
- 20 Tier-3s per Tier-1 - 12 have gatherer, 8 use upstreams' caches

	Count	Gatherers	Caches	CAs
Tier-1	1	1x3=3	1x16=16	1=1
Tier-2	1x6=6	6x2=12	6x12=72	6=6
Tier-3	1x20=20	1x12=12	8x5=40 12x8=96	12=12
Totals	27	27	224	19+7=26

How Do You Deploy a Testbed of About ~~1,000~~ 300 Machines?

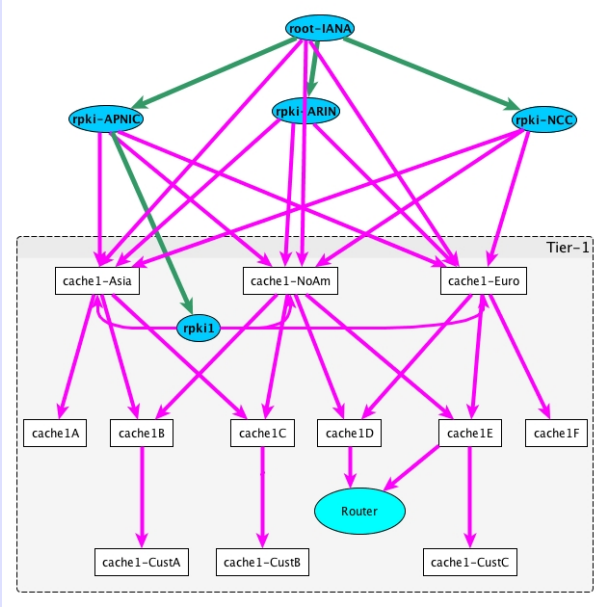
StarBED ~ 1000 KVMs



But You Don't Configure ~~1,000~~ 300 Servers by Hand

L'Borough AutoNetKit

You draw this on your Mac using yEd



Yes, I am Serious

AutoNetKit reads the graphml, Builds Server Configurations and Deploys them on StarBED, Junosphere, etc.

AutoNetKit

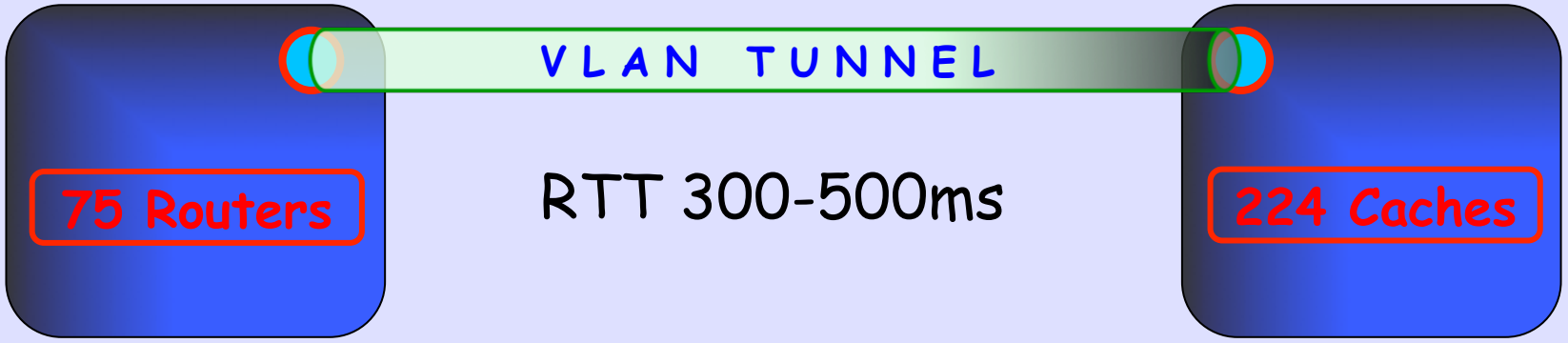
- NetKit originally Roma Tre University by Andrea Cecchetti, Lorenzo Colitti, Federico Mariani, Stefano Pettini, Flavia Picard, and Fabio Ricci
- AutoNetKit by Matt Roughan and Simon Knight at University of Adelaide
- Further Developed at University of Loughborough by Iain, Debbie, and Olaf

Enhancing AutoNetKit

- Was only routers and routing
- Address assignment was poor
- Needed to add concept of servers and services
- Needed to understand RPKI components: rpkid, pubd, caches, rtr-client, …
- Needed to handle RPKI object creation

Inducing Delay

RTT to Remote Routers Induces Delay



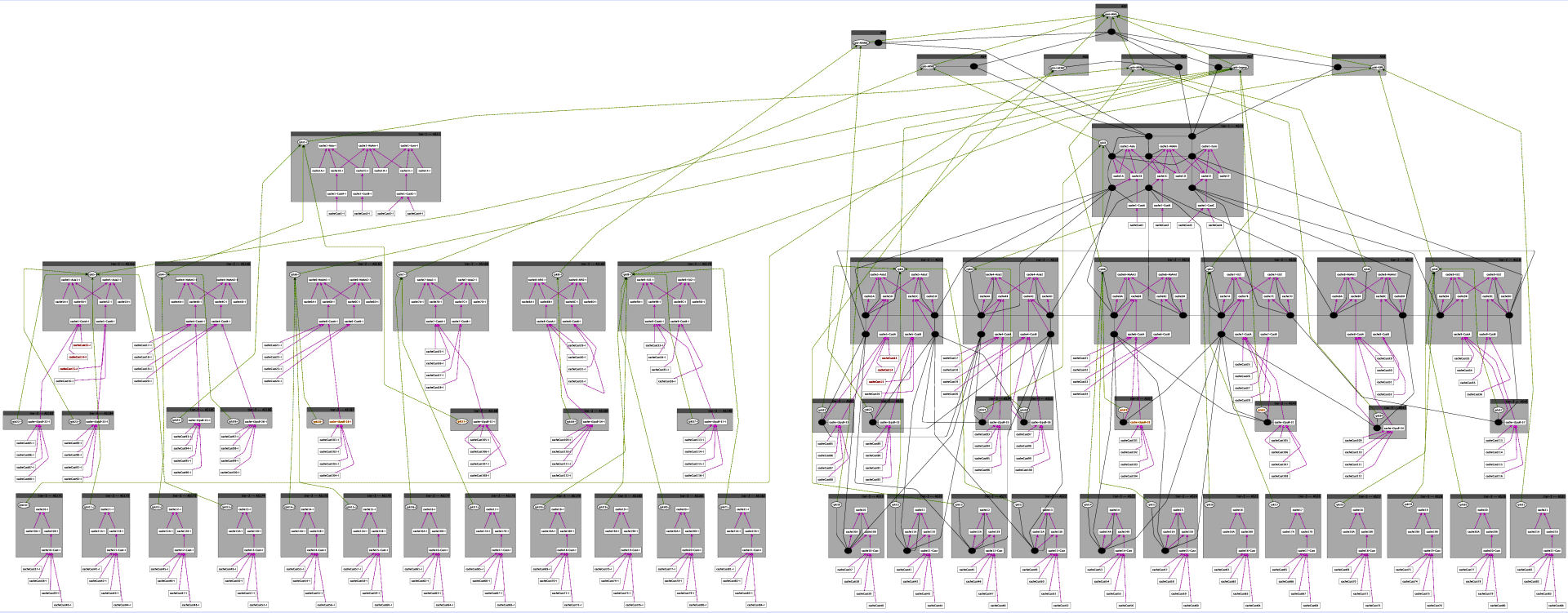
vmm.dfw
Four Server VMM
Junosphere
Dallas Texas US

StarBED
1,000 Servers
We're Using 50
Nomi-city, Ishikawa Pref

Notation for Delay

- If two pubds/caches/... are both connected to routing by a solid line, then the traffic between them is routed, i.e. goes StarBED to Junosphere back to StarBED, inducing a very large delay.
- Sequential router hops stay within Junosphere/Dallas, so do not add significantly more delay

Two Tier-1 Model



One Without Delay One With

Delay Had No Effect

- We ran fairly large scale with delay and without
- Between Publication and Gatherers and Between Caches
- The numbers were essentially the same
- Network Latency is not Important
- So Cache Deployment Architecture can be based on other things

Creating Objects

We will buy a two or three star dinner for the code to take a real BGP table dump and create a realistic hierarchy of well aggregated certificate requests and subsequent ROAs.

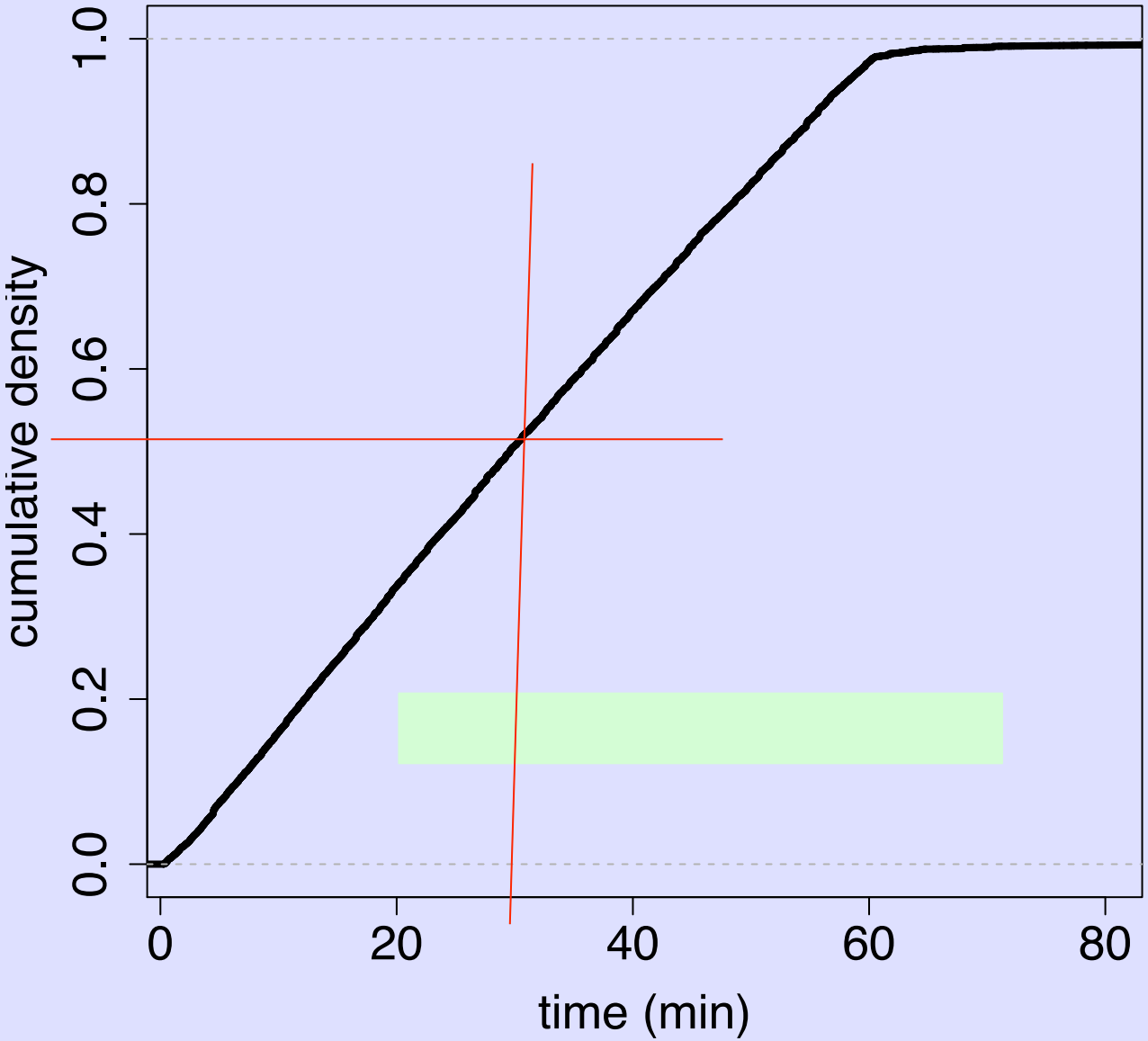
Creating Objects

- One CA/ROA Per Entity on Start
- 7,500 – 14,000 Added During Run
  - 250-270 per RIR for ISPs who use RIR web pages
  - 45 per Tier-1 ISP
  - 10 per Tier-2 ISP
  - 1-2 per Tier3 ISP

Running the Model

- About two hours to run and upload to StarBed in Japan from Loughborough
- 250MB Uploaded, and 2G image to copy
- 1:1 Time Ratio, so it runs for a full day
- Produces 1-3.3G of log files
- Which we then have to transfer the logs to a compute server
- Analysis of logs takes 10-15 minutes

Ideal Pubd to Gatherers



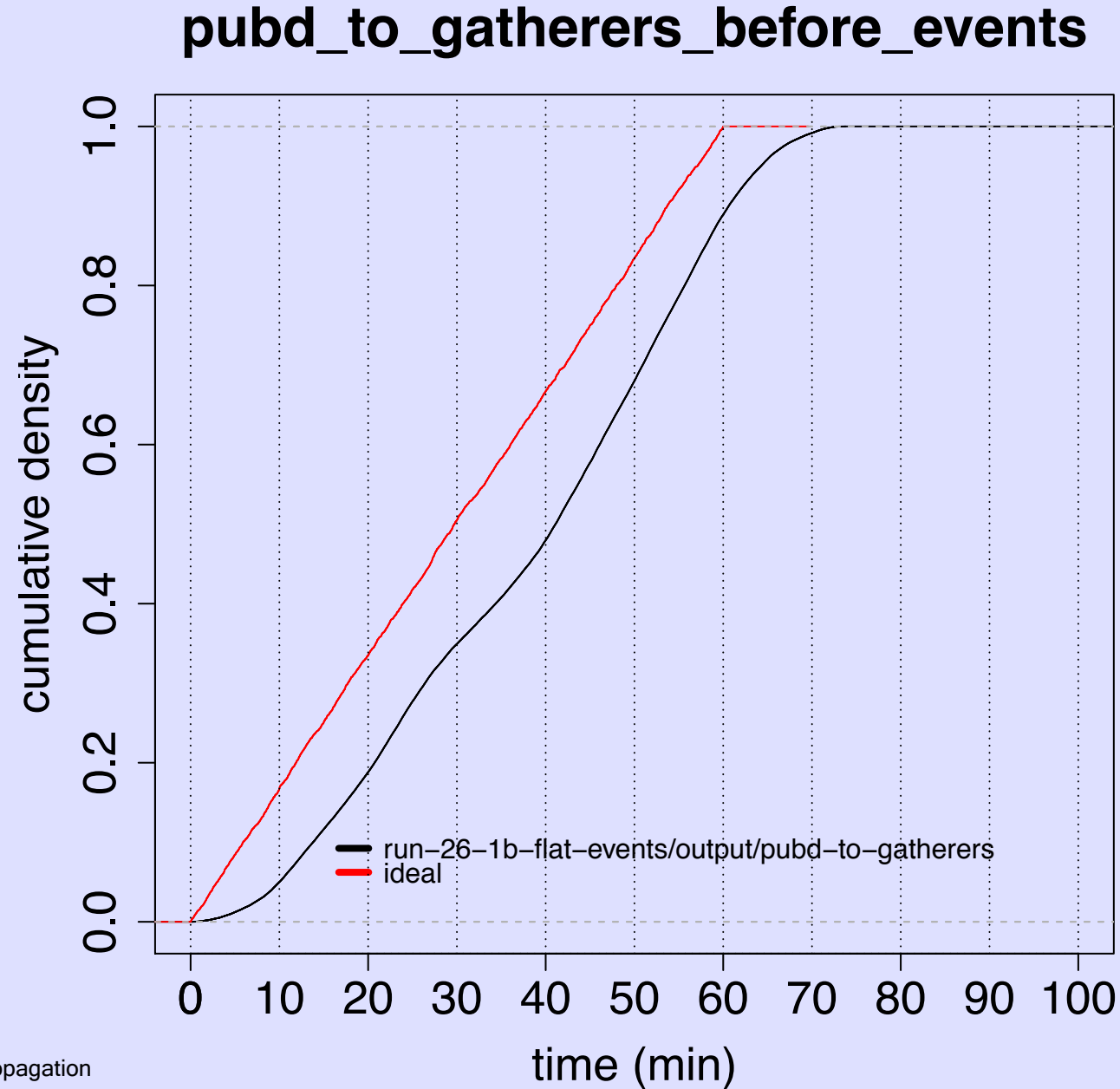
RPKI-Rtr Negligible

The time for the RPKI-Rtr compute and the protocol to transfer over to the routers is negligible.

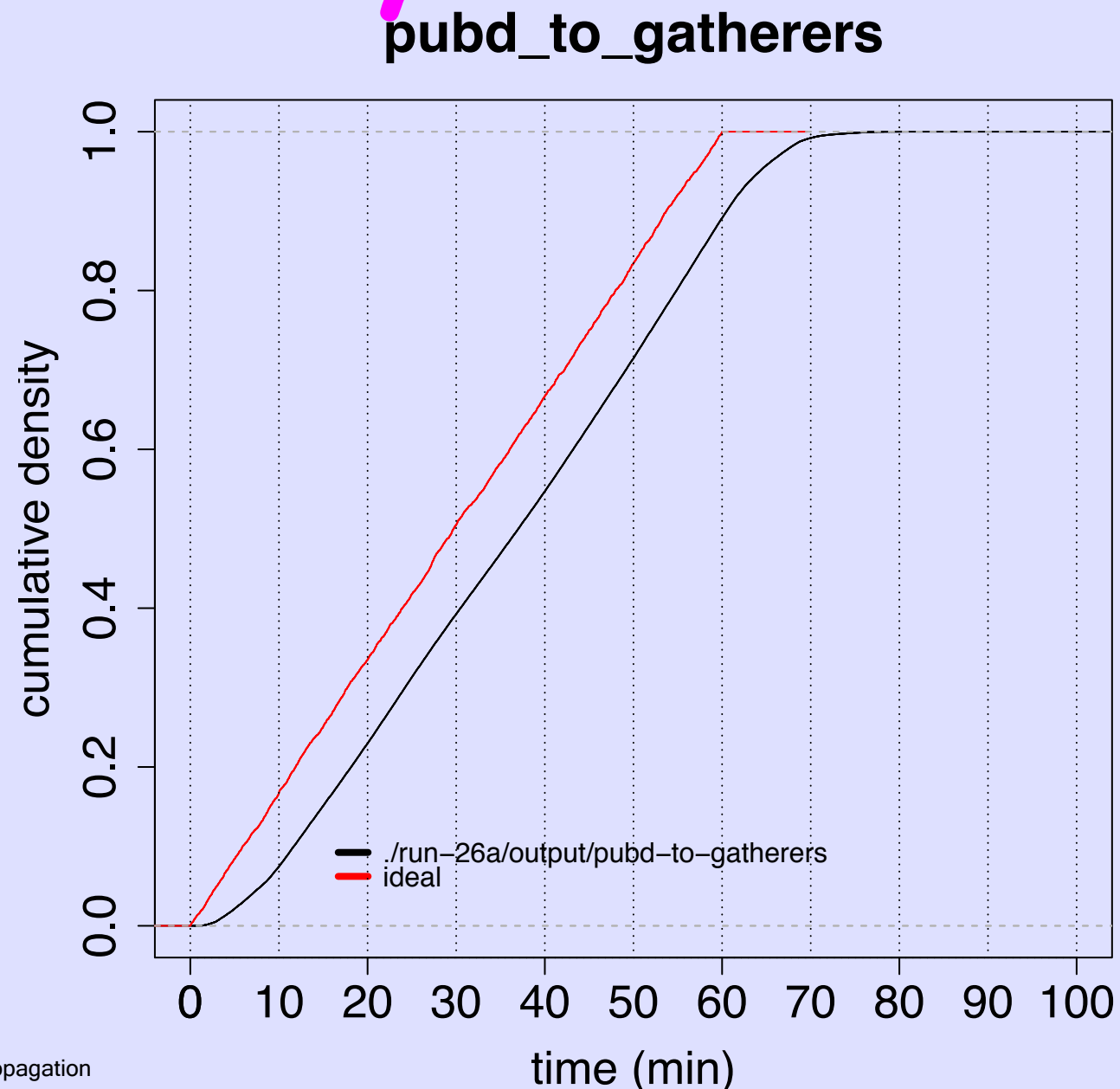
The only somewhat expensive operation is one O(n*log(n)) sort of the new state, everything one would get if one were to perform a cache reset query

Feeding new data from rpki-rtr server to rpki-rtr client is just dropping a pre-computed file into the network connection

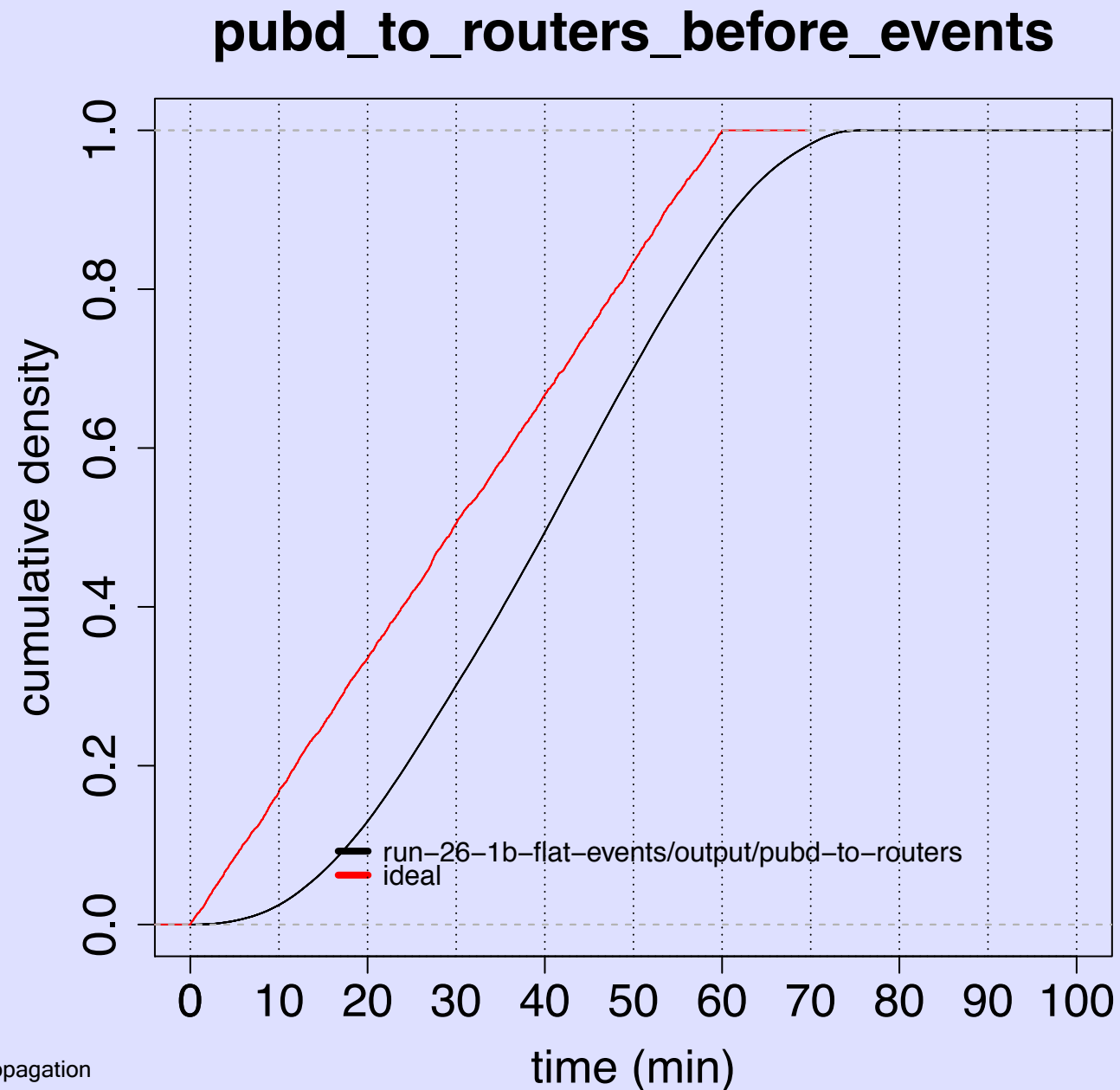
Initial Load at Start



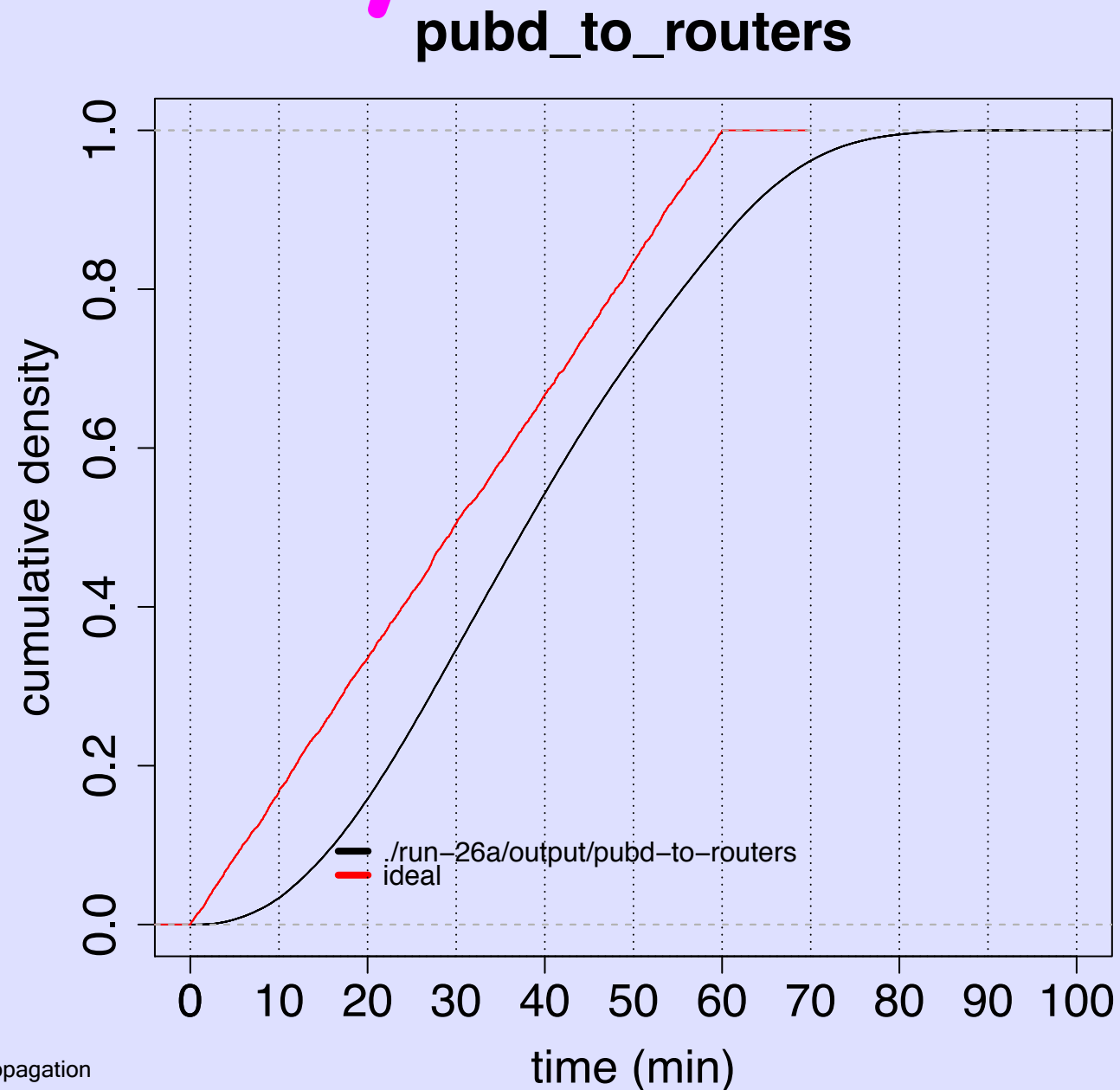
Steady State Flow



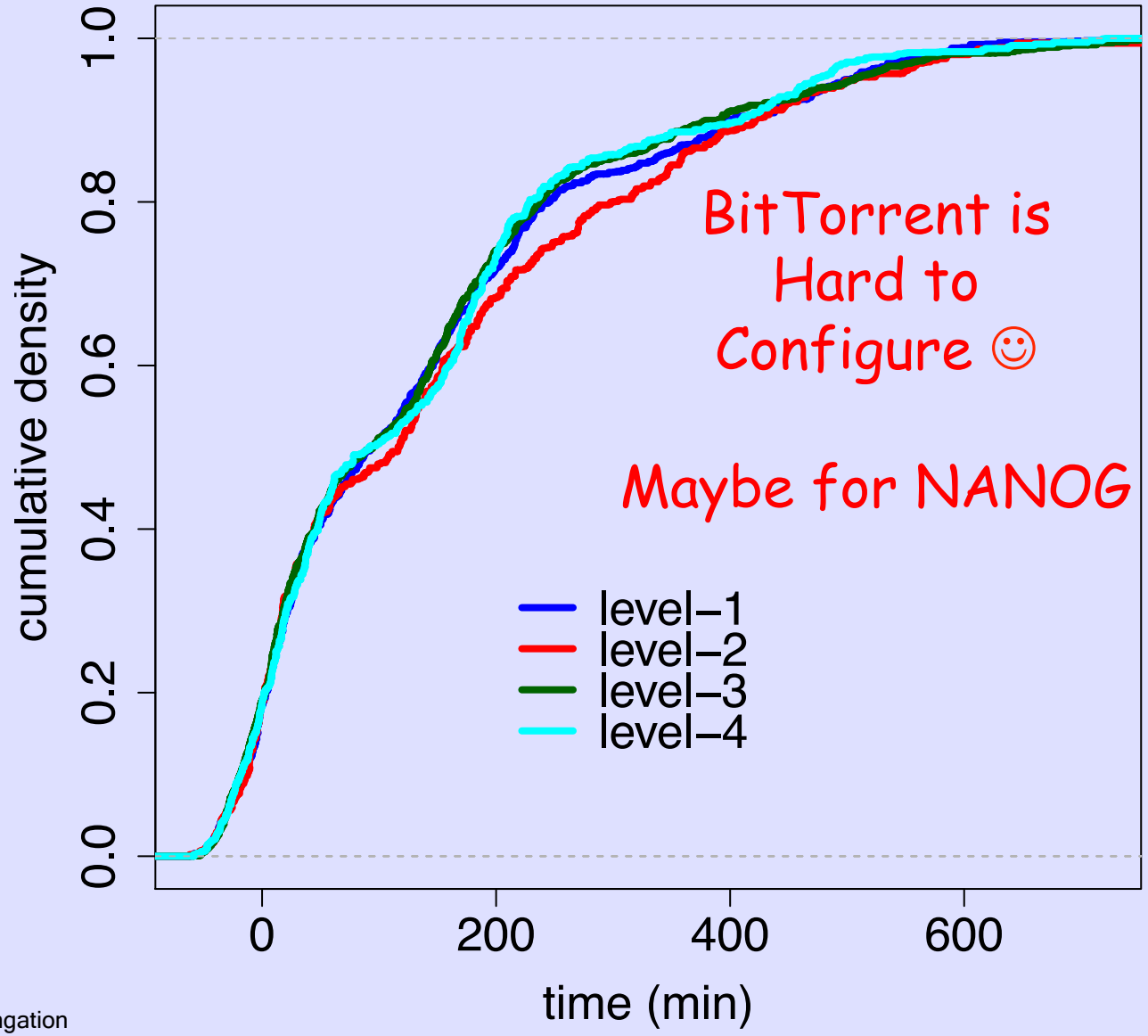
Initial Load at Start



Steady-State Flow



BitTorrent Gather/Cache



Thanks

- StarBED
- Juniper and Cisco
- DHS [0]
- University of Adelaide
- Loughborough University, Purdue, & IIJ

[0] THIS WORK IS SPONSORED IN PART BY THE DEPARTMENT OF HOMELAND SECURITY UNDER AN INTERAGENCY AGREEMENT WITH THE AIR FORCE RESEARCH LABORATORY (AFRL).

TCP Behavior of BGP

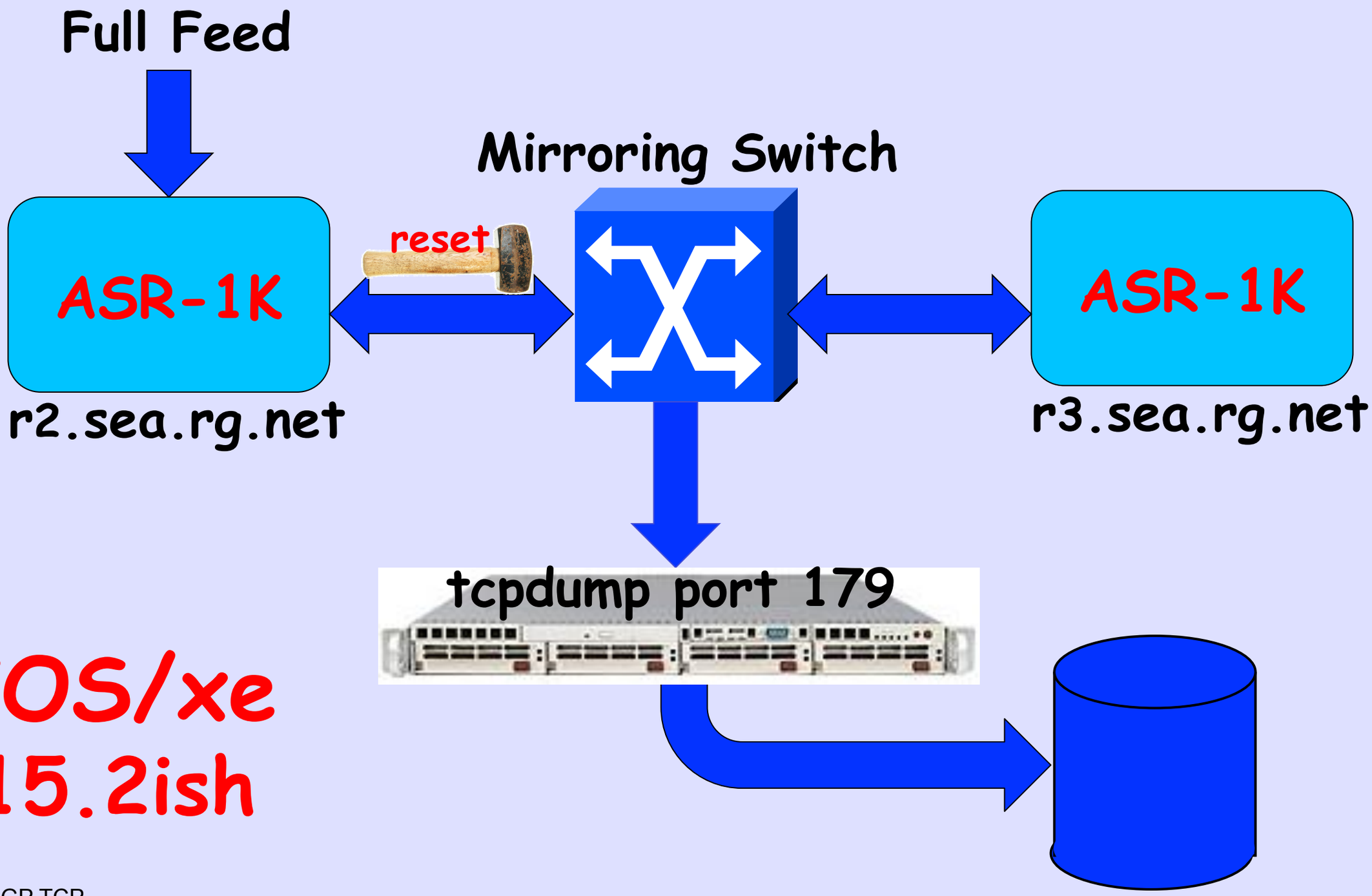
NANOG / Dallas

2012.10.22

Randy Bush <randy@psg.com>
Mark Allman <mallman@icir.org>
Keyur Patel <keyupate@cisco.com>
Balaji Pitta Venkatachalapathy <bvenkata@cisco.com>
Manoj Pandey <mpandey@cisco.com>
Vern Paxson <vern@icir.org>
Cristel Pelsser <cristel@iij.ad.jp>
Ed Kern <ejk@cisco.com>

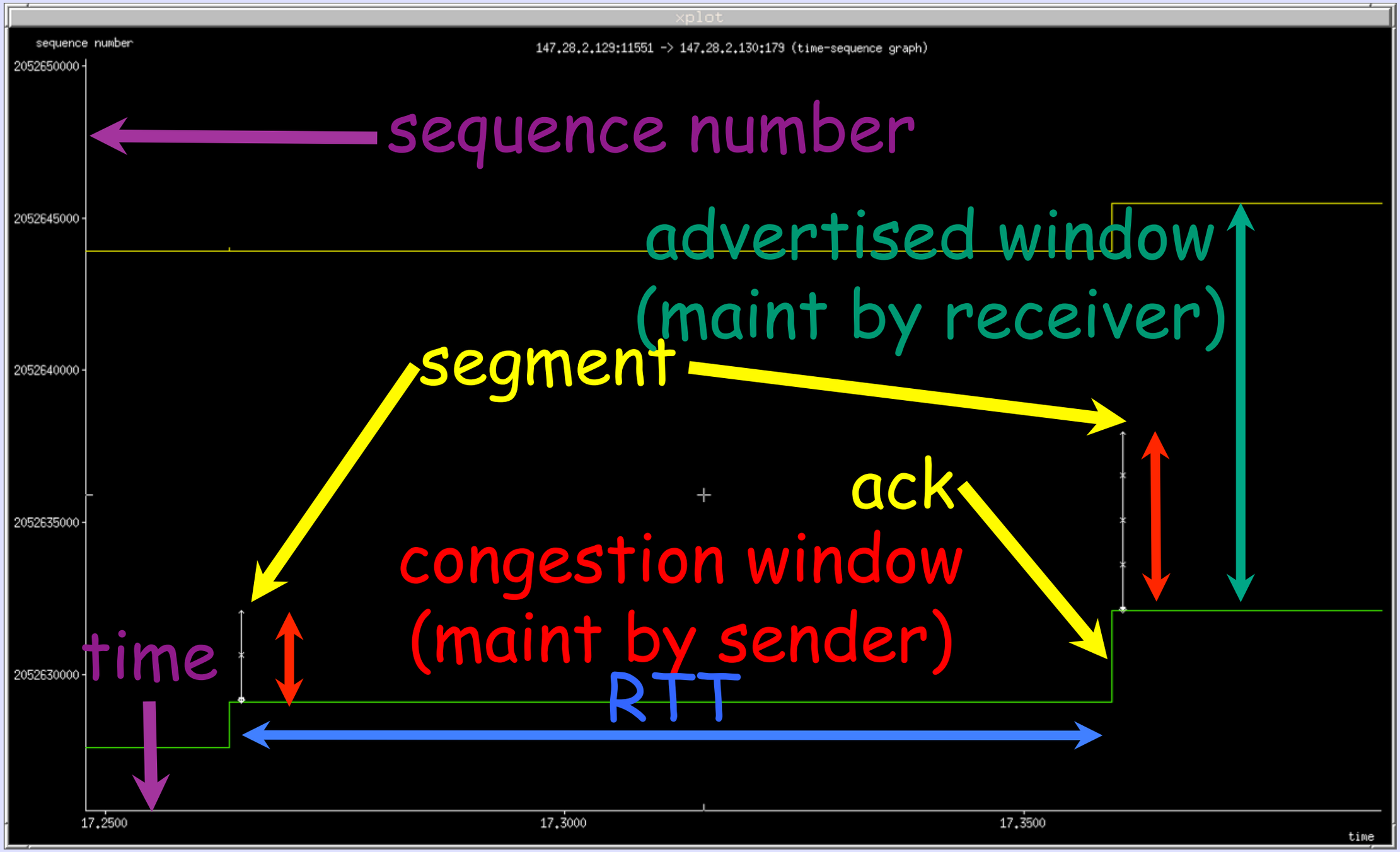
Goal:
Can We Make BGP Convergence Even Faster?

So We Measure



IOS/xe
15.2ish

Time Sequence Plots

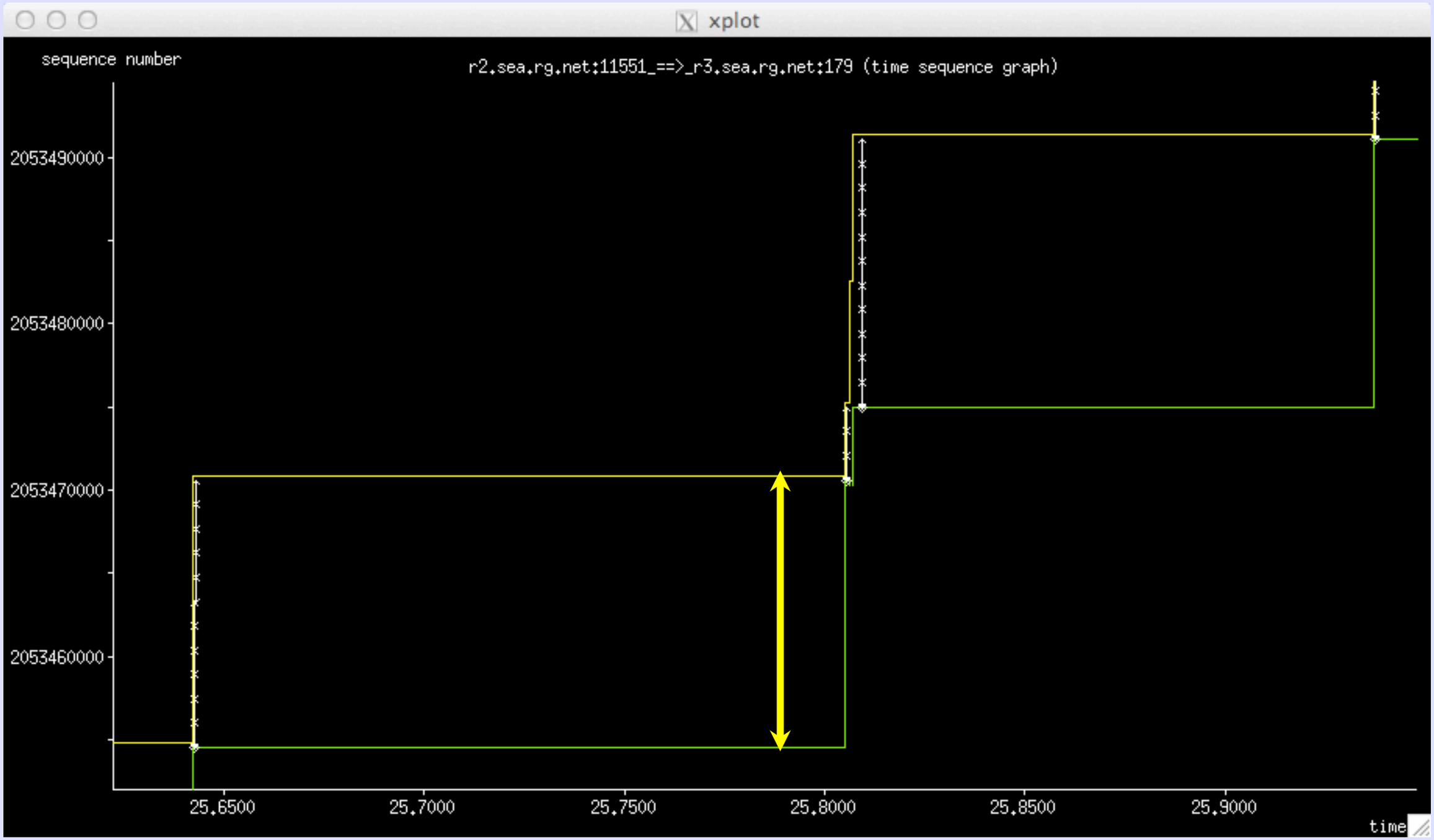


Warning: In xplot, relative scaling of axes is completely arbitrary, i.e., one can zoom one without the other and often does by accident.

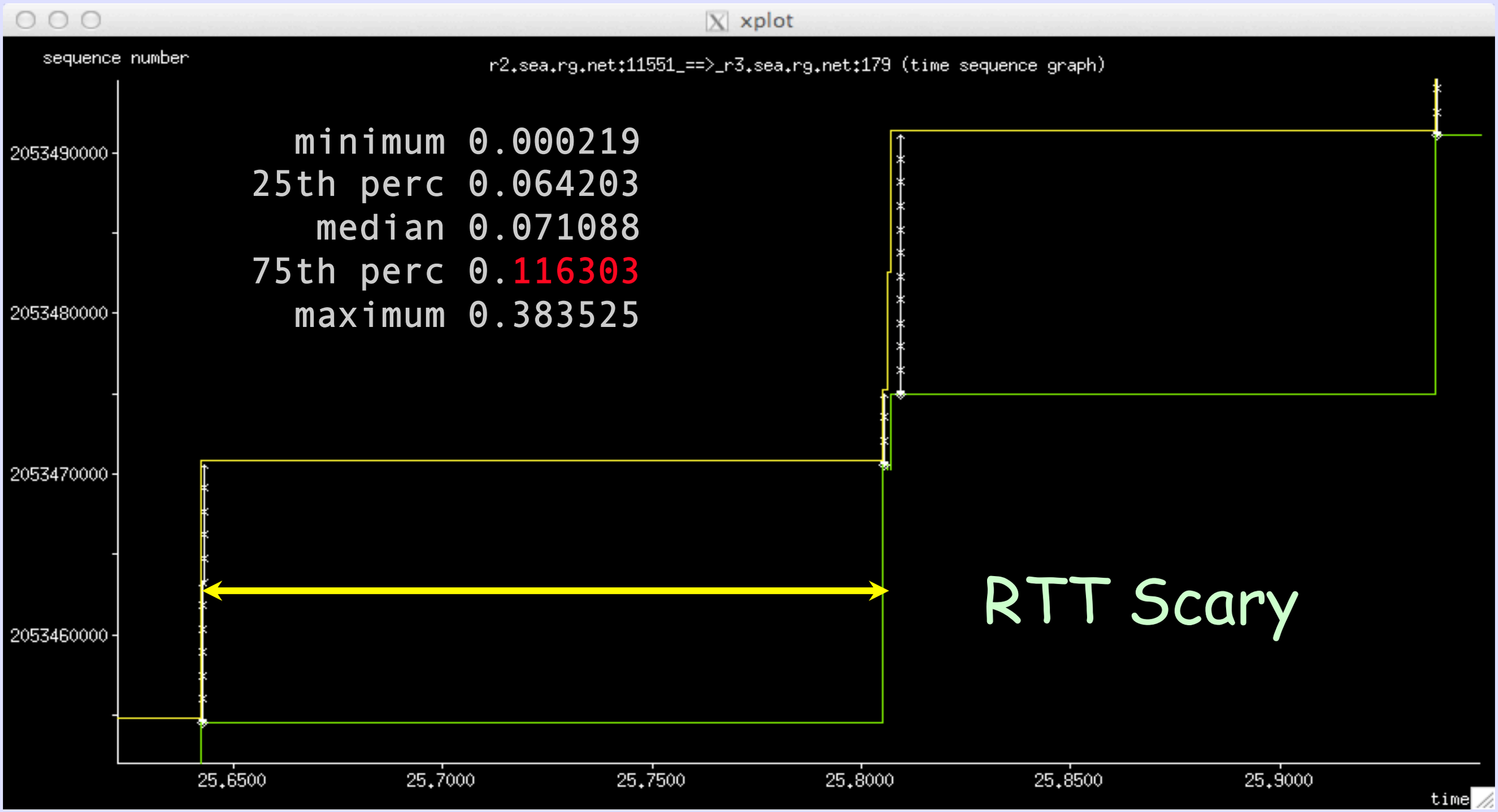
So viewers should not read too much into flat vs. steep curves, only shape patterns.

And there are no calipers, and the axis labels suck. But it's free ☺.

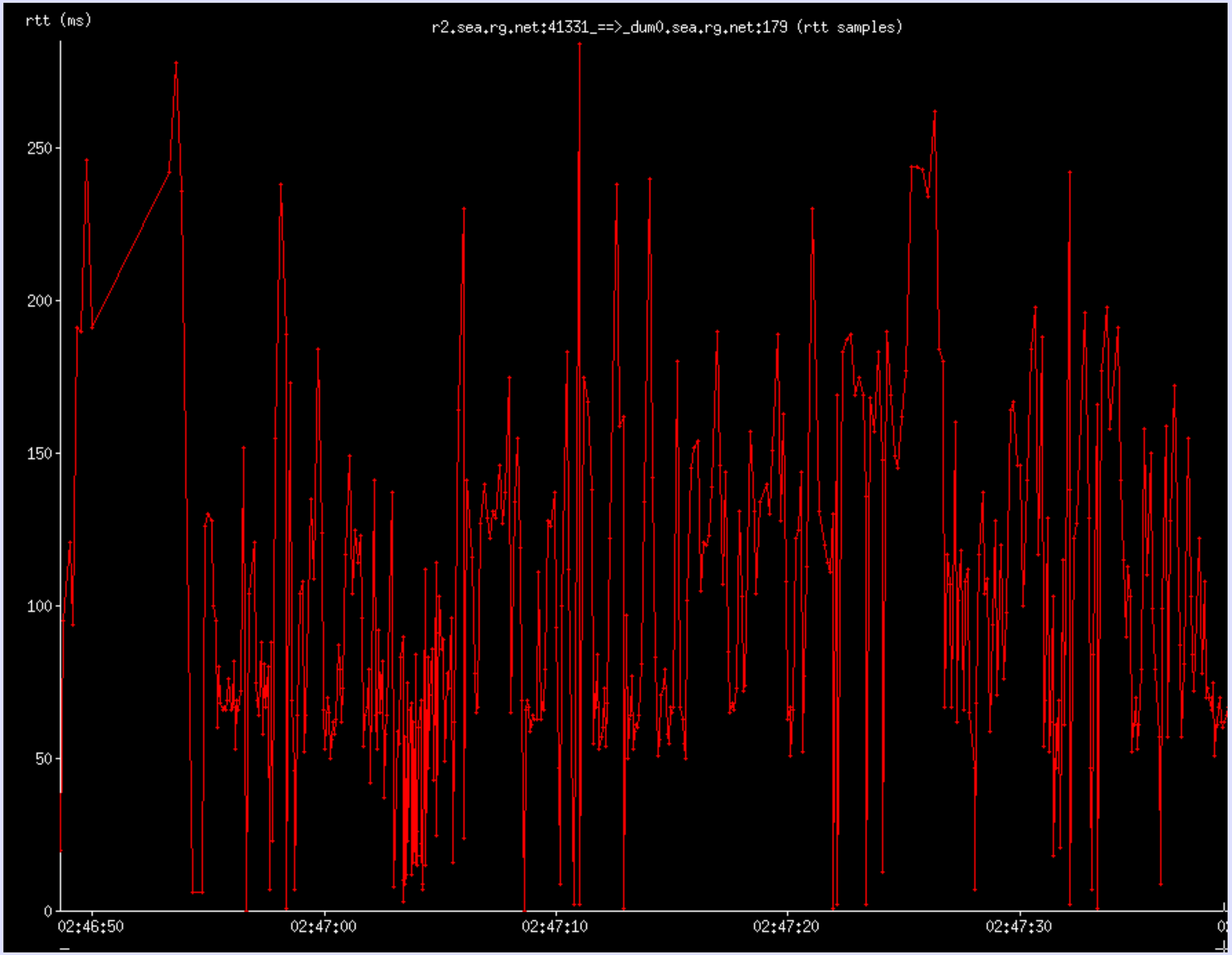
cwnd limited by adv.win



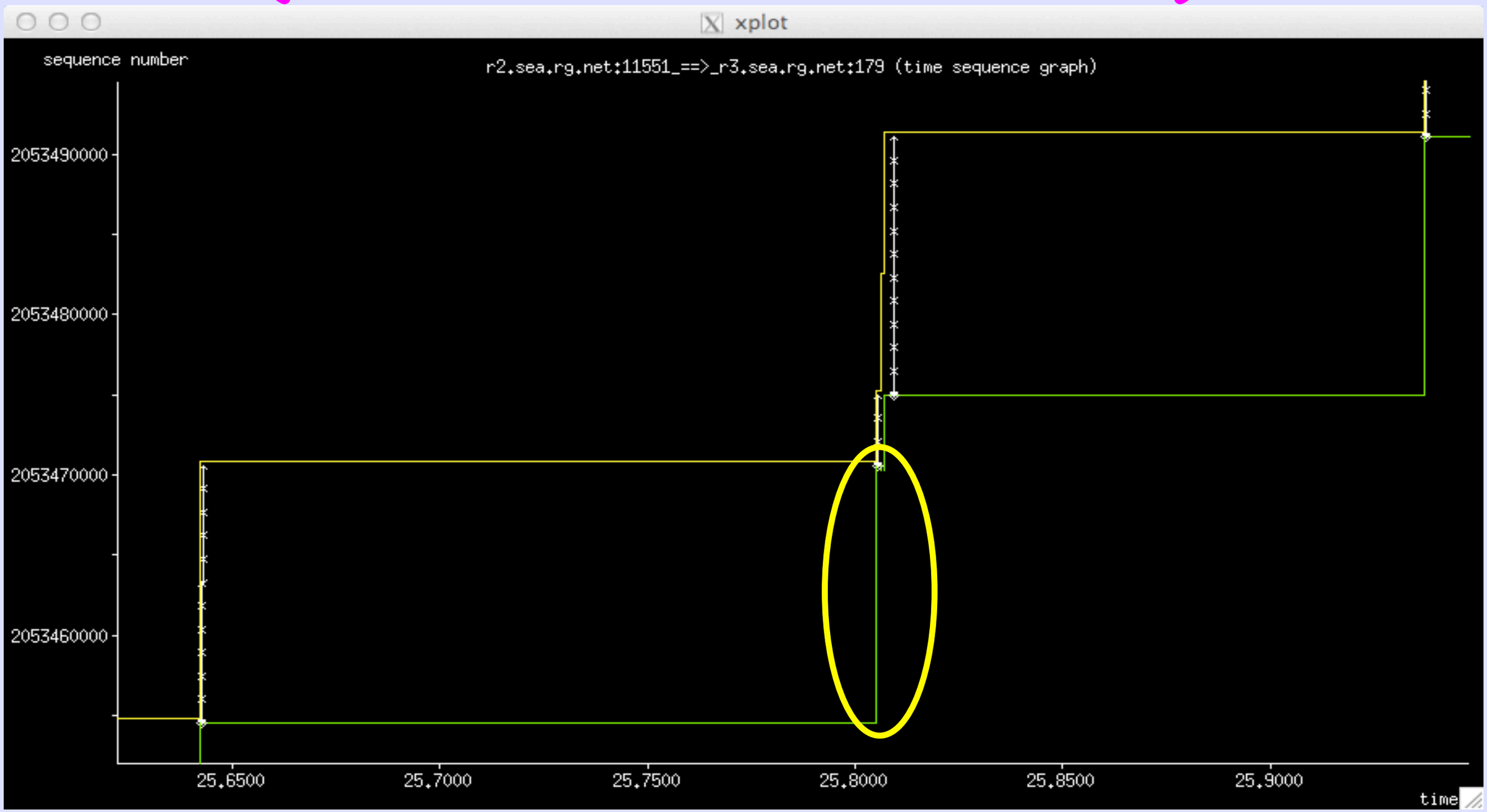
RTT~110ms-On a LAN!



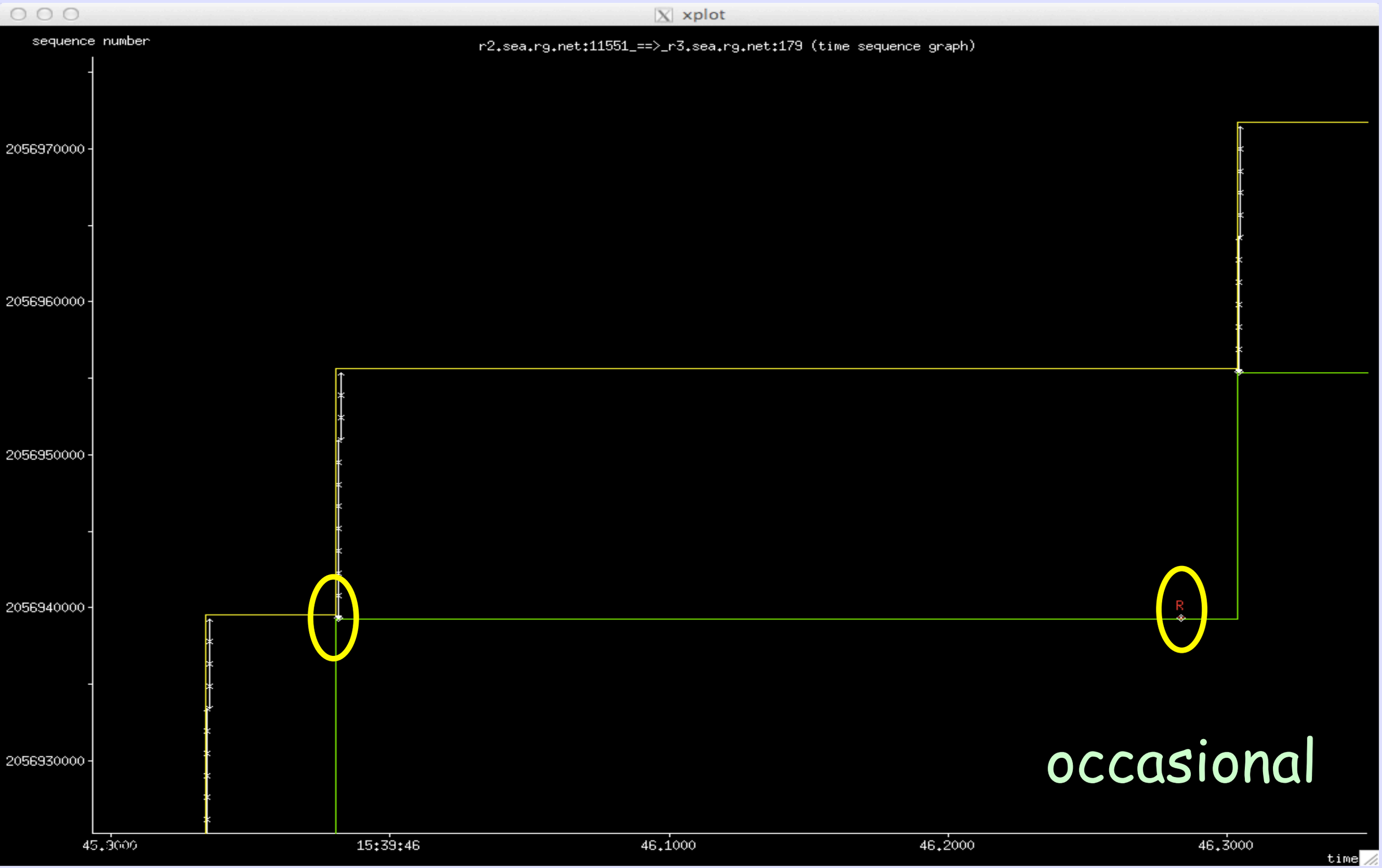
RTT over Full Session



One ACK for Entire Window (AKA 'Stretch' ACK)



Loss and Retransmit



Is Stretch ACK OK?

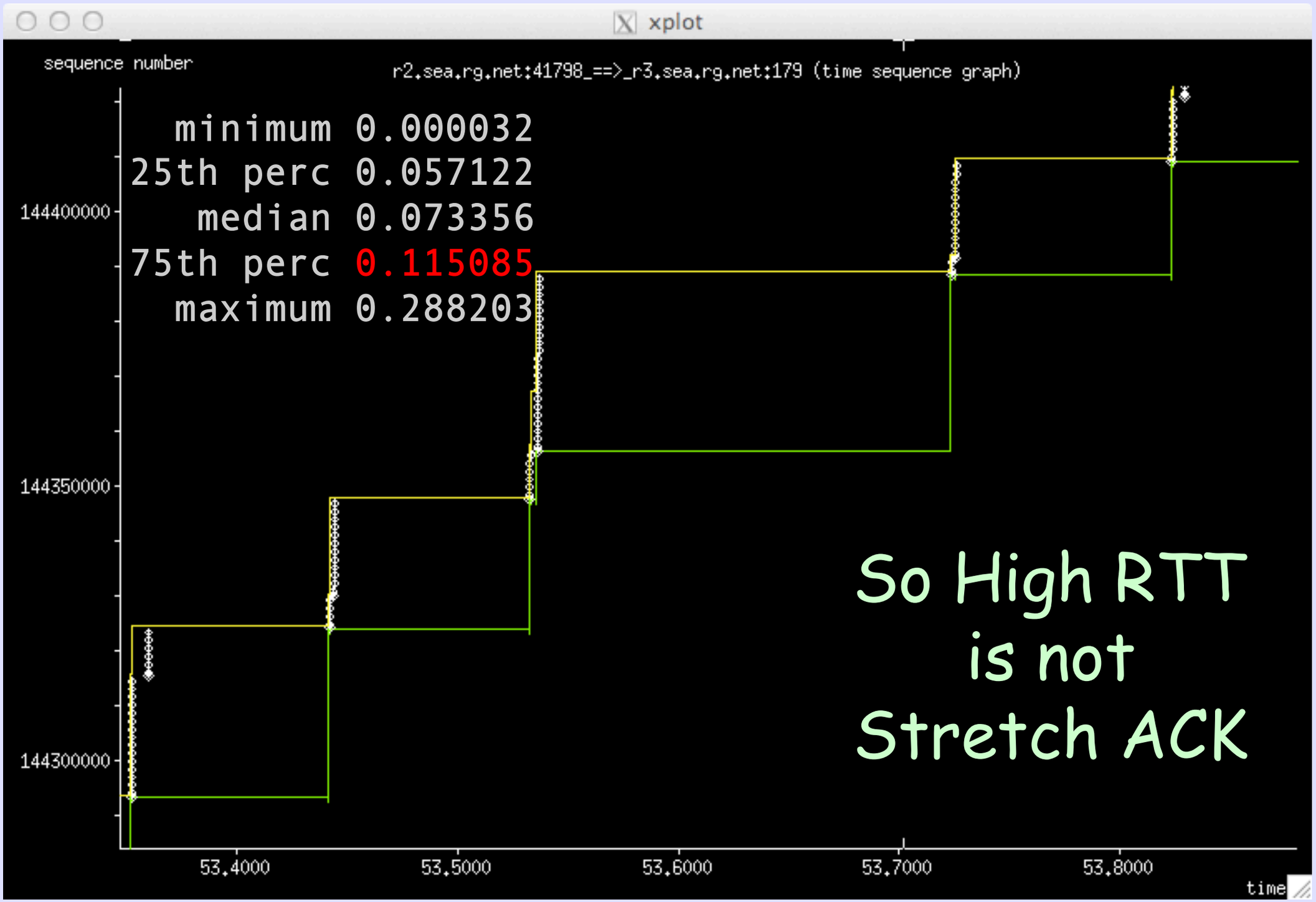
- Stretch ACK for the entire window
  - May contribute to long RTTs, as we wait to coalesce ACKs
  - Very Bursty, as big ACKs cause large window shifts
  - Loss, as bursts overwhelm a buffer (maybe NIC?)

Small Advertised Window

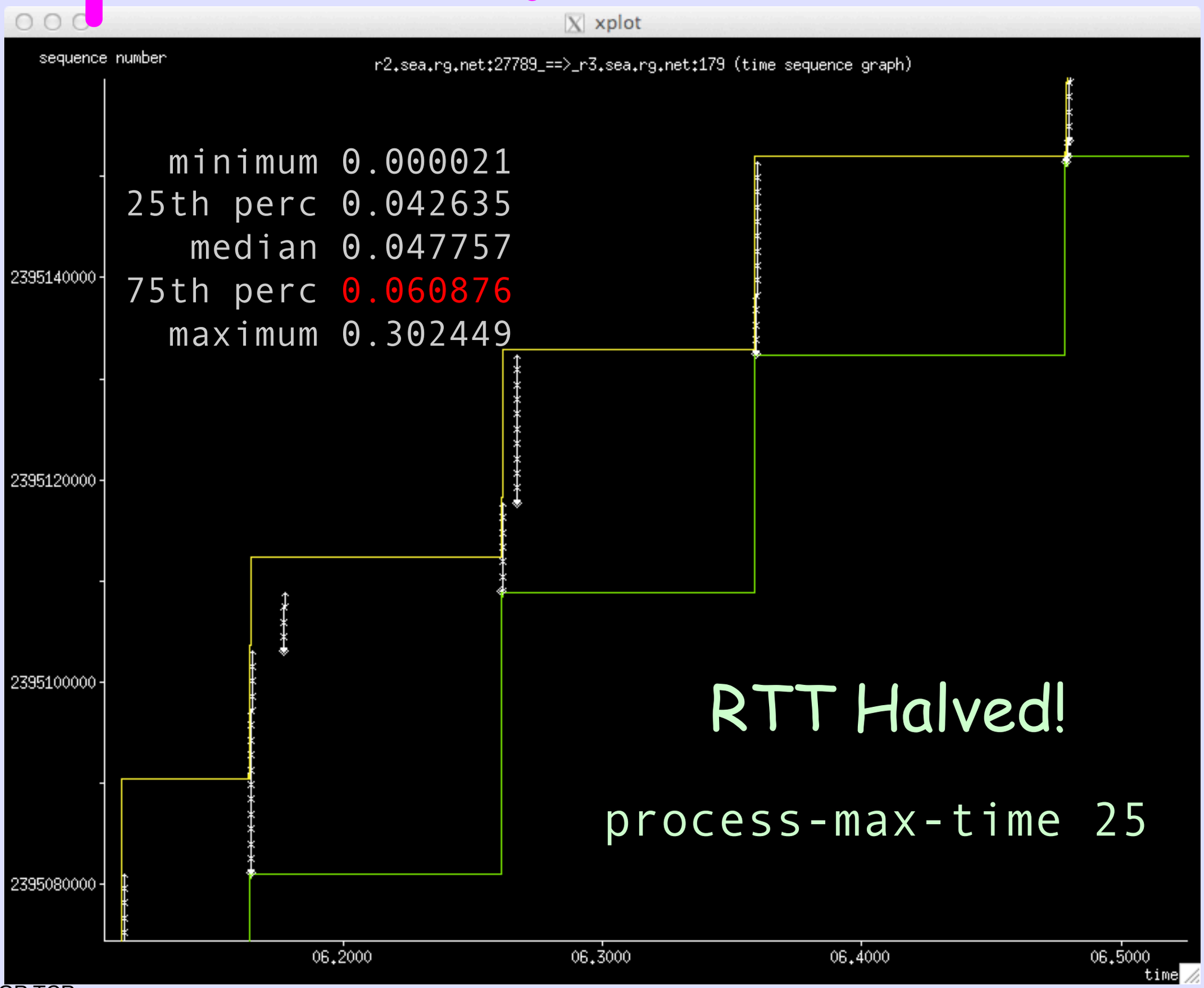
- Would like at least RTT * Bandwidth for TCP to fully utilize the path capacity
- Window size issue is exacerbated by artificially long RTTs
- We increased the advertised window size but it had no impact

Removed Stretch ACK from Code

Stretch ACK Removed



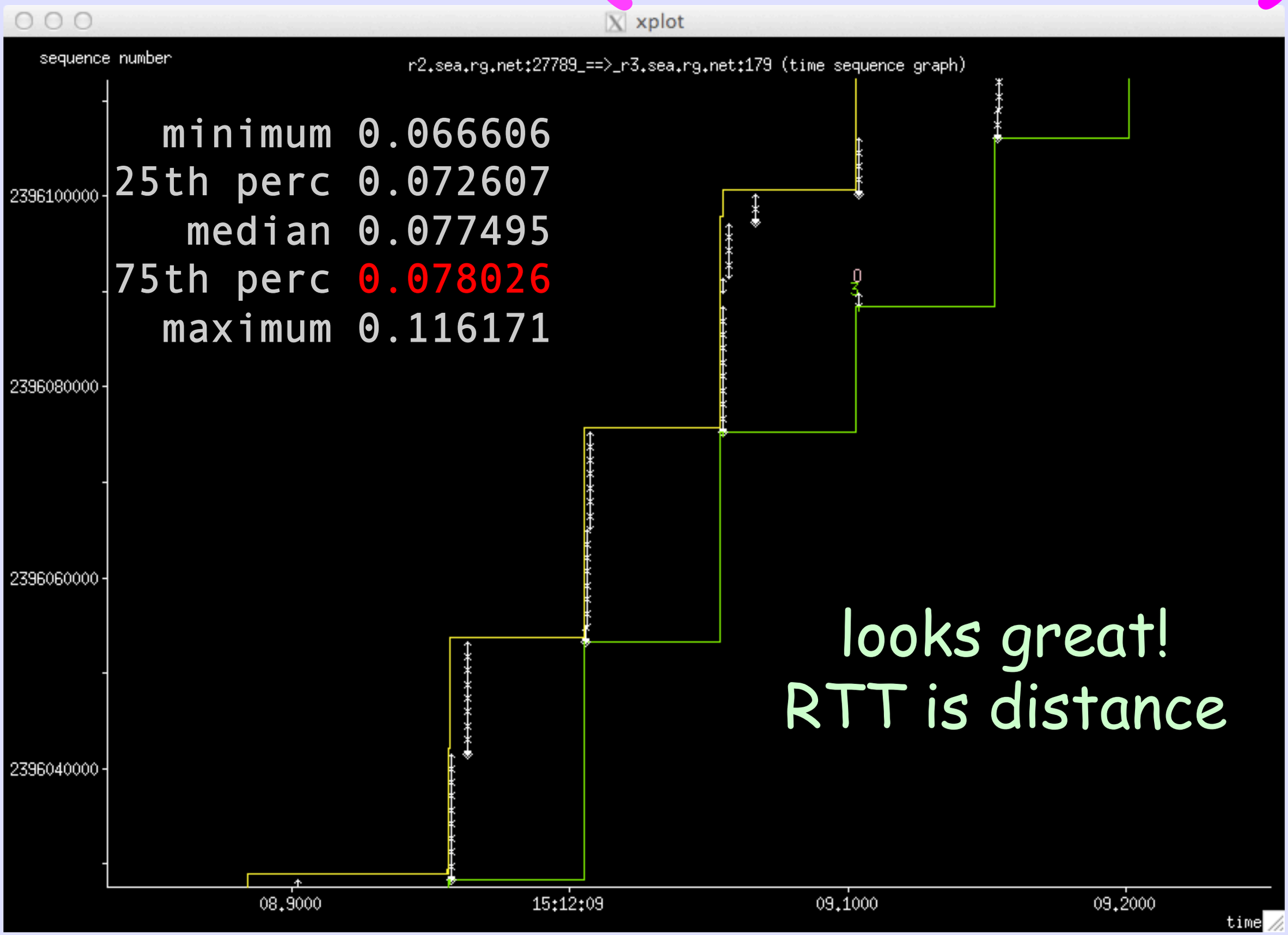
Drop RTC from 50 to 25



What if it is the TCP Stack?

So Let's Measure a Non-BGP Protocol

RPKI-Rtr (SEA-DFW)

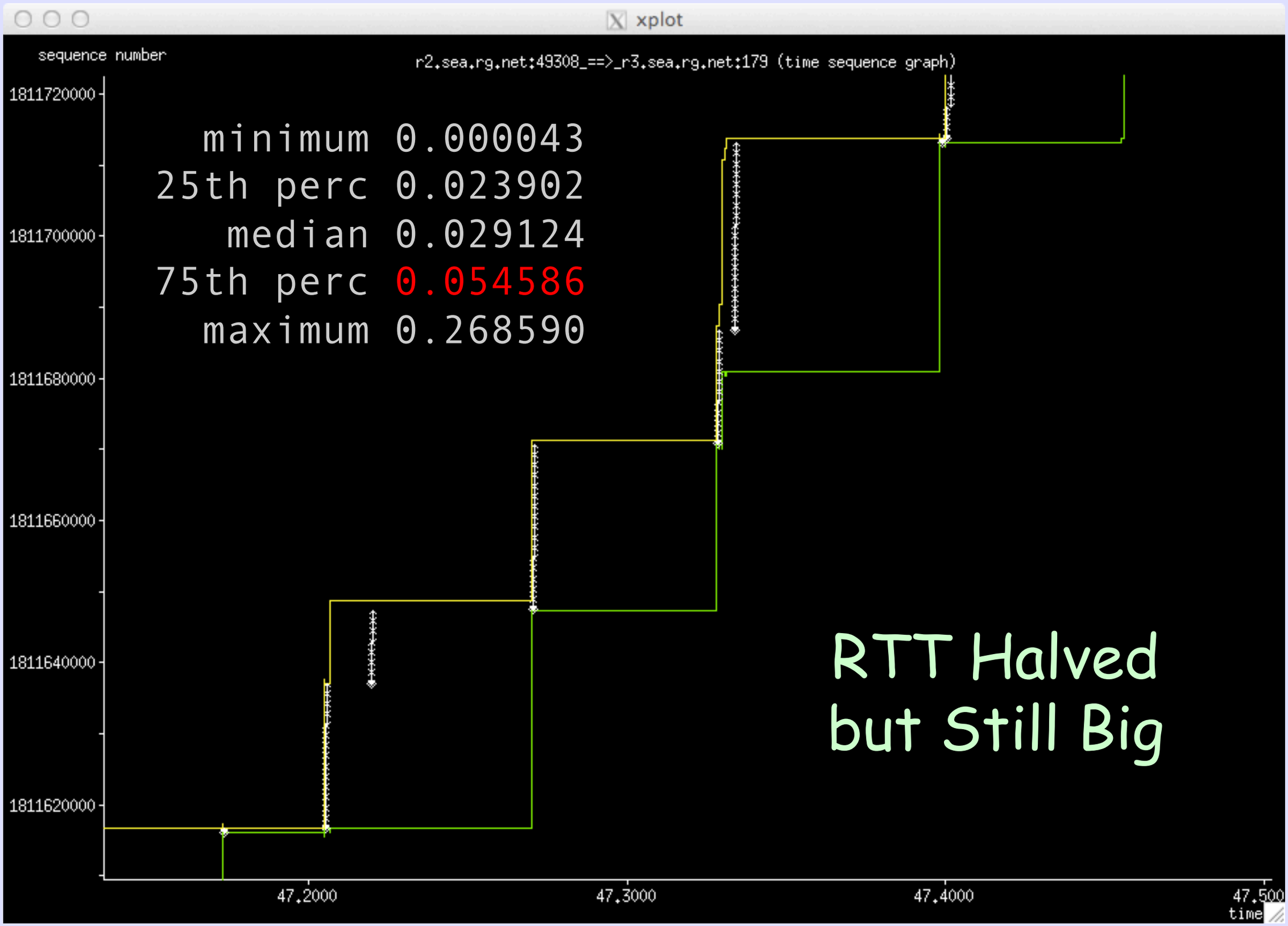


Stack Looks Good!

- Looks like what one would really expect TCP to look like
- ACKs are generated correctly
- Sender fills the window
- RTT looks to be roughly right for the underlying path, Dallas to Seattle
- Window is too small for the path, but buffer small as they're saving RAM

So is it BGP or Could it Be RIB->FIB?

BGP w/o RIB->FIB



Better, Not Yet Beautiful

- Get rid of Stretch ACK
- Open the Window to >= 32K
- RIB->FIB is a known 'opportunity'
- Is Run To Completion keeping the RTT high?
- The stack is not so bad. Yay!
- We really want to measure XR!!!

We got the RTTs Down

They are Still Too Long

We are Still

Chasing This

And

We Saw Something

Very Strange

in Dallas

RPKI (DFW-DFW) but XR