Measuring the End User

Geoff Huston

APNIC Labs, February 2014

What's the question?

How many users do <x>?

- How many users can retrieve a URL using IPv6?
- How many users perform DNSSEC validation when they resolve a domain name?
- How many users follow DNAME chains in the DNS

etc

“Measurable” Questions

- How much traffic uses IPv6?
- How many connections use IPv6?
- How many routes are IPv6 routes?
- How many service providers offer IPv6?
- How many domain names have AAAA RRs?
- How many domains are DNSSEC signed?
- How many DNS queries are made over IPv6?

…

Users vs Infrastructure

- None of these specific measurement questions really embrace the larger questions about the end user behaviour
- They are all aimed at measuring an aspect of of behaviour within particular parameters of the network infrastructure, but they don’t encompass how the end user assembles a coherent view of the network

For example… IPv6

- To make an IPv6 connection everything else (routing, forwarding, DNS, transport) has to work with IPv6
- So can we measure how many connected devices on today's Internet are capable of making IPv6 connections?

How to measure a million end users for their IPv6 capability

How to measure a million end users for their IPv6 capability

- Be Google (or any other massively popular web service provider)

How to measure a million end users for their IPv6 capability

- Be Google (or any other massively popular web service provider)
  - And insert measurement code on the web page that is executed as part of the page load

```
</div>

<script src="http://labs.apnic.net/measure-ipv6.php" type="text/javascript"></script>

<script src="http://www.google-analytics.com/urchin.js" type="text/javascript"> </script>
<script type="text/javascript">_uacct = "UA-597837-1"; urchinTracker(); </script>

</body>
</html>
```

How to measure a million end users for their IPv6 capability

- Be Google (or any other massively popular web service provider)

or

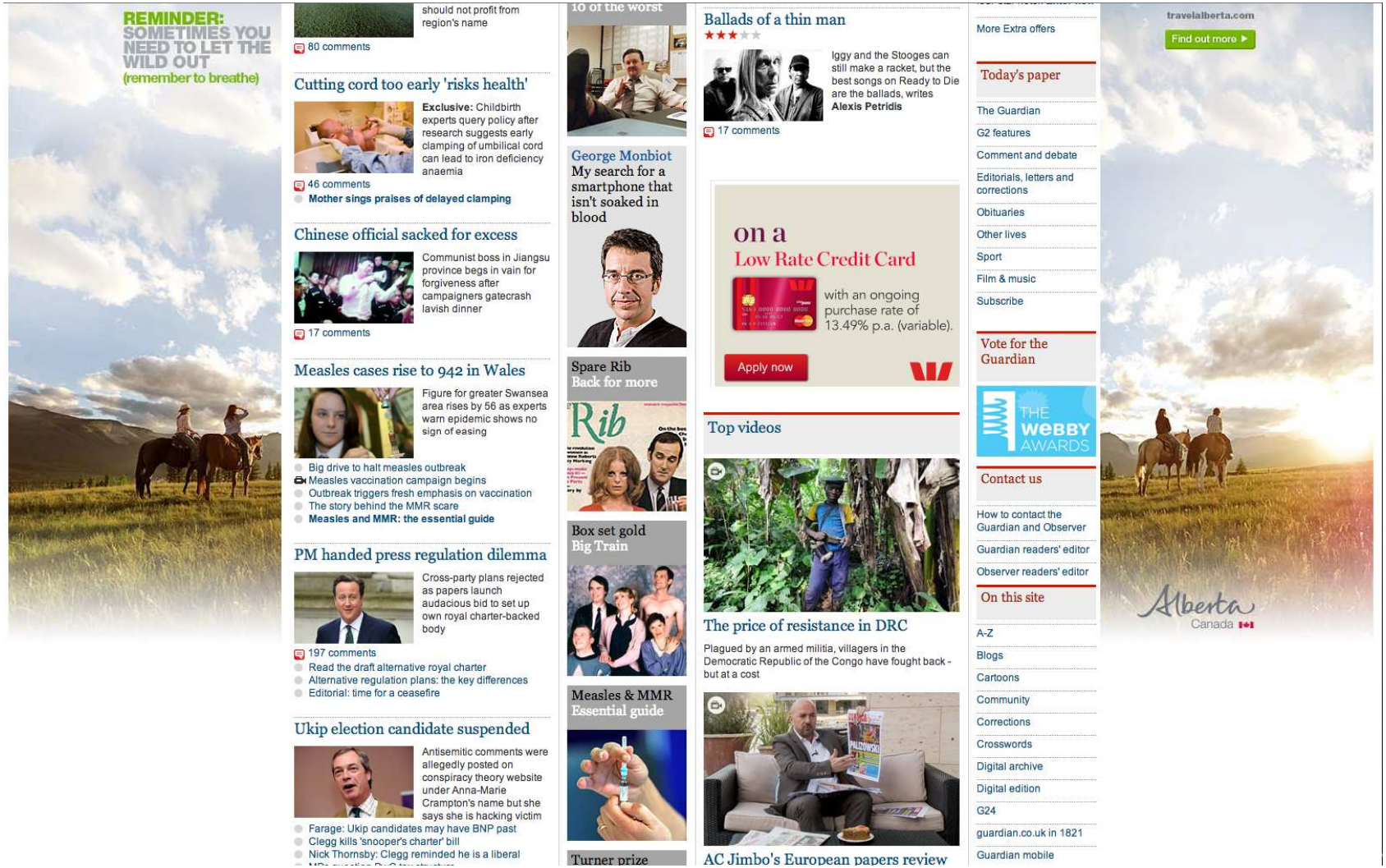
How to measure a million end users for their IPv6 capability

- Be Google (or any other massively popular web service provider)

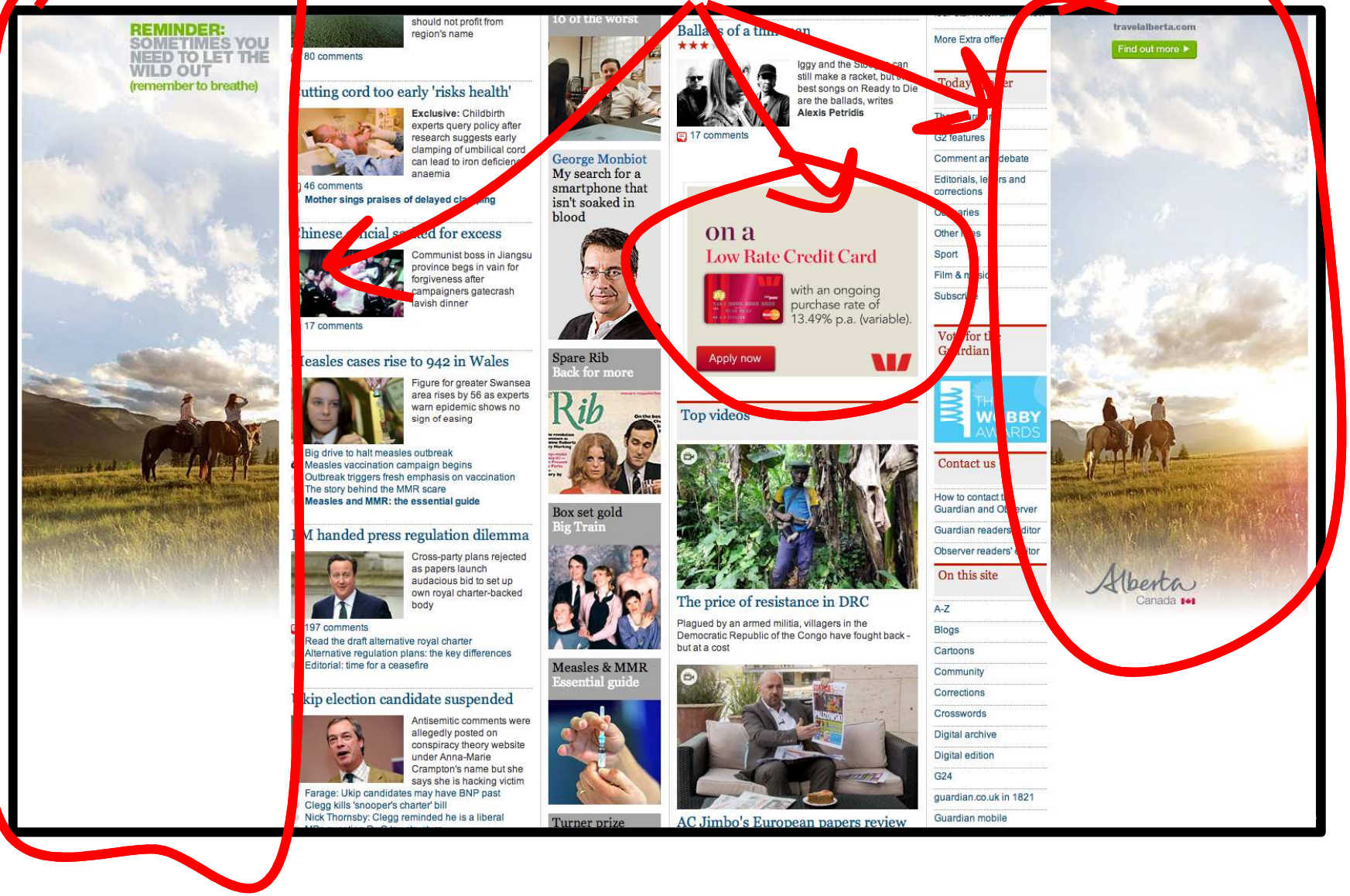
or

- Get your code to run on a million users' machines through another delivery channel

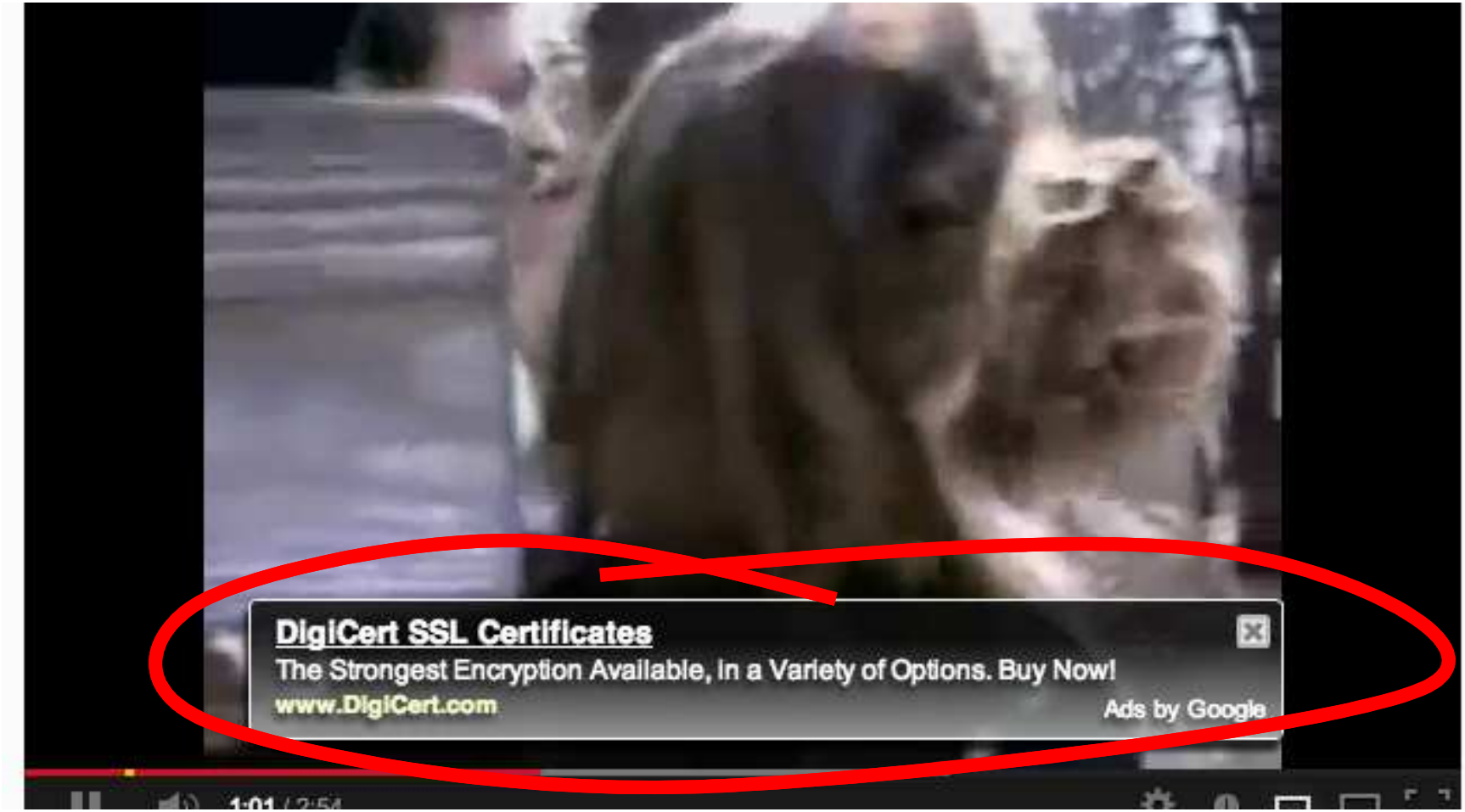
Ads are ubiquitous



Ads are ubiquitous



Ads are ubiquitous



Ads are implemented in Adobe Flash

- Advertising channels use Flash to make ads interactive
  - This is not just an ‘animated gif’

Flash makes ads interactive

- [Apply Now] hover-over is interactive, and responds when selected.

Flash and the network

- Flash includes primitives in ‘actionscript’ to fetch ‘network assets’
  - Typically used to load alternate images, sequences
  - Not a generalized network stack, subject to constraints:
    - Port 80
    - crossdomain.xml on hosting site must match source name (wildcard syntax)
- Flash has asynchronous ‘threads’ model for event driven, sprite animation

APNIC's measurement technique

- Craft flash/actionscript which fetches network assets to measure.
- Assets are reduced to a notional '1x1' image which is not added to the DOM and is not displayed
- Assets can be named (DNS resolution via local gethostbyname() styled API within the browser's Flash engine) or use literals (bypass DNS resolution)
- Encode data transfer in the name of fetched assets
  - Use the DNS as the information conduit:
    - Result is returned by DNS name with wildcard
  - Use HTTP as the information conduit
    - Result is returned via parameters attached to an HTTP GET command

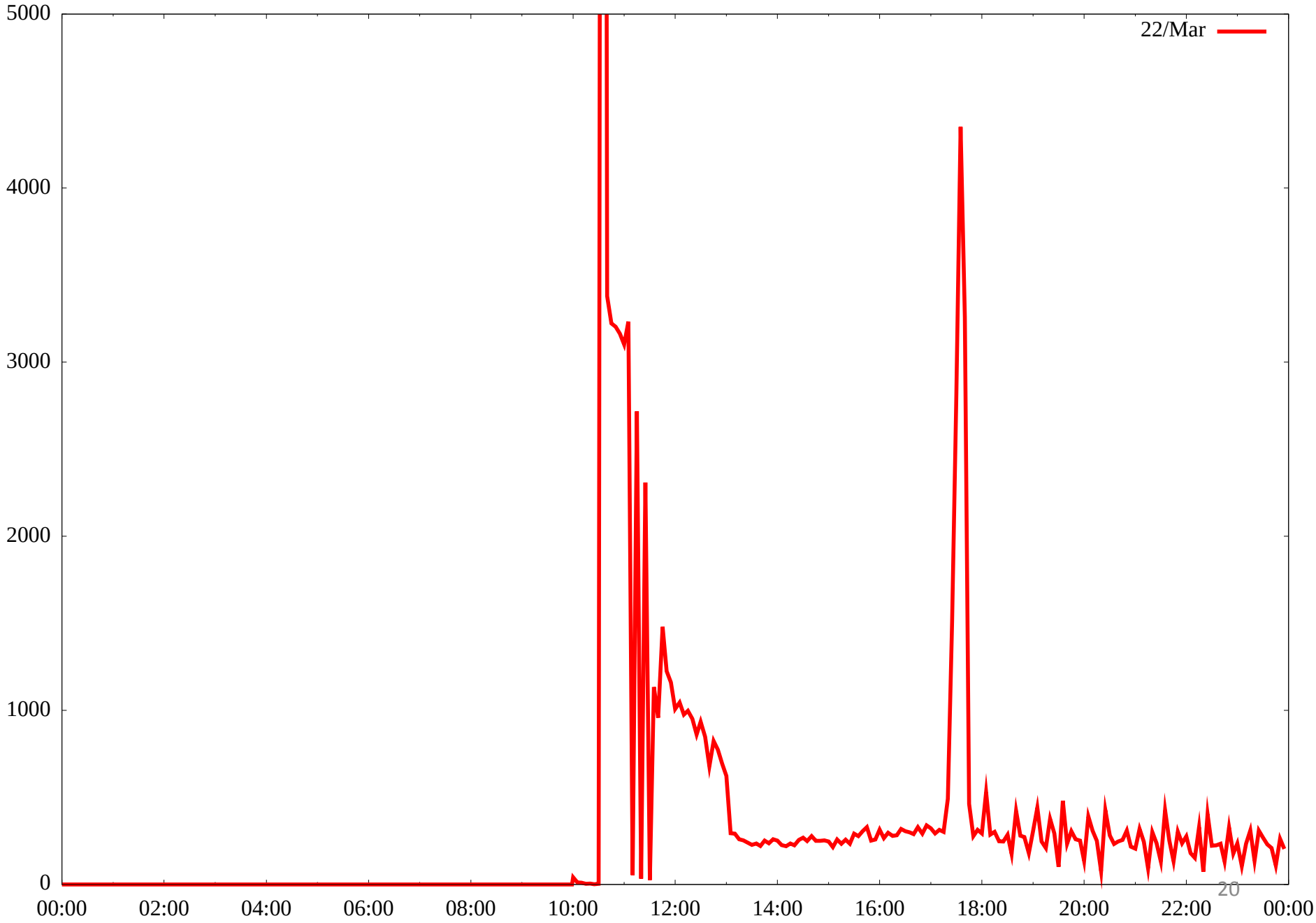
Advertising placement logic

- Fresh Eyeballs == Unique IPs
  - We have good evidence the advertising channel is able to sustain a constant supply of unique IP addresses
- Pay by click, or pay by impression
  - If you select a preference for impressions, then the channel tries hard to present your ad to as many unique IPs as possible
- Time/Location/Context tuned
  - Can select for time of day, physical location or keyword contexts (for search-related ads)
  - But if you don't select, then placement is generalized
- Aim to fill budget
  - If you request $100 of placement a day, then inside 24h algorithm tries hard to even placement but in the end, will 'soak' place your ad to achieve enough views, to bill you $100

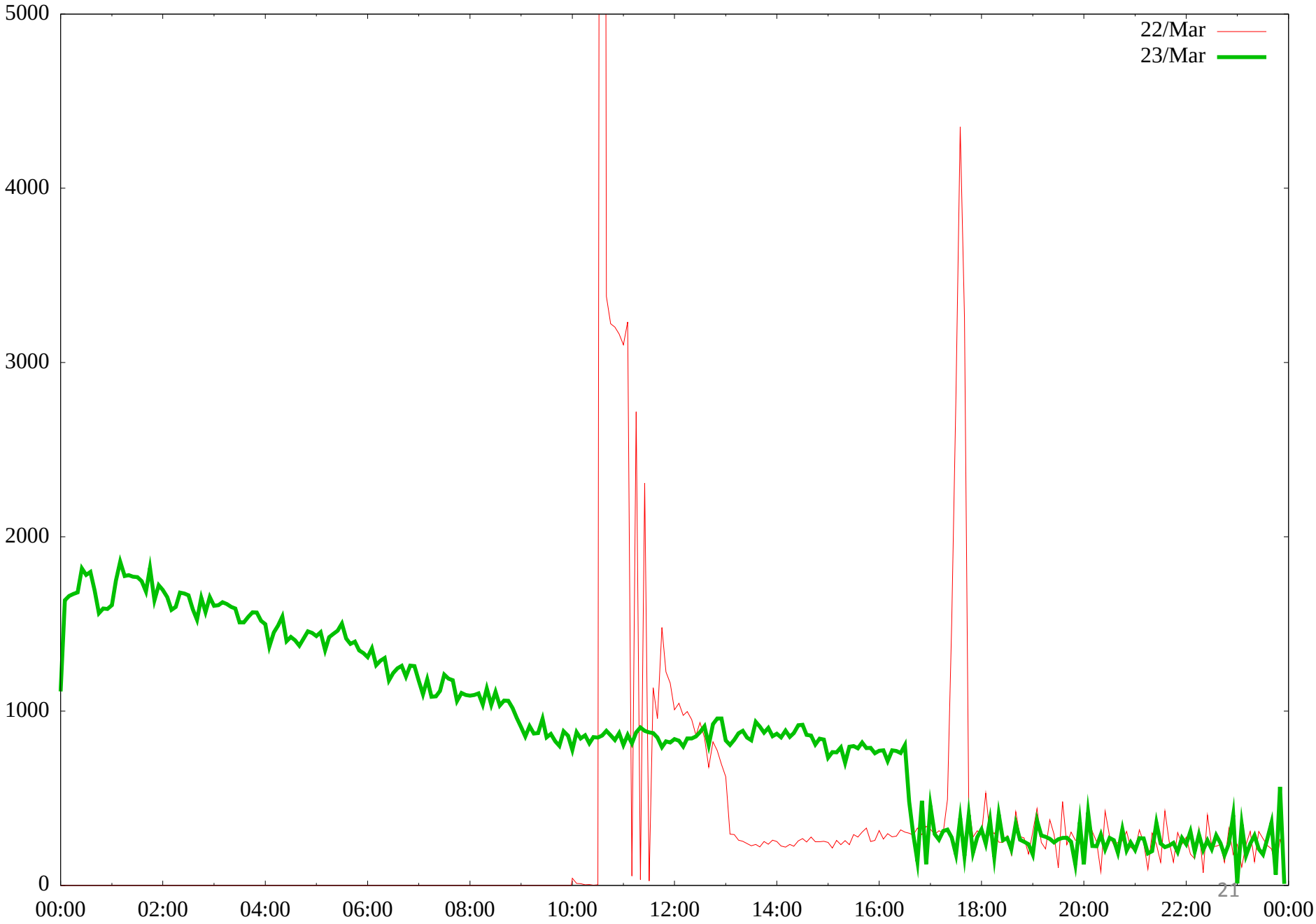
Advertising placement logic

- Budget: $100 per day, at $1.00 ‘CPM’ max
  - Clicks per millepressions: aim to pay no more than $1 per click but pay up to $1 for a thousand impressions
- Even distribution of ads throughout the day
- No constraint on location, time
- Outcome: 350,000 placements per day, on a mostly even placement model with end of day ‘soak’ to achieve budget goal

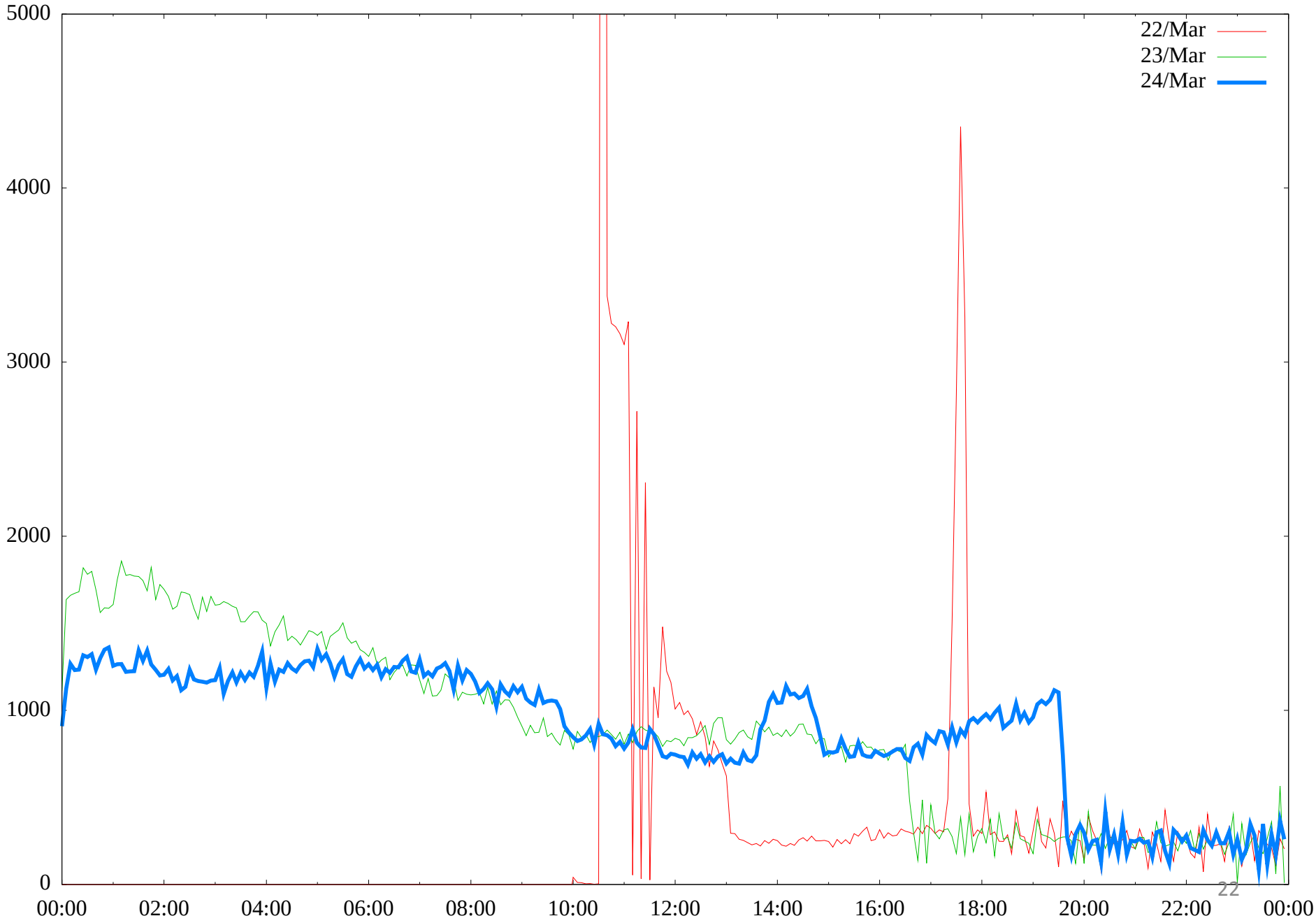
Ad Placement Training – Day 1



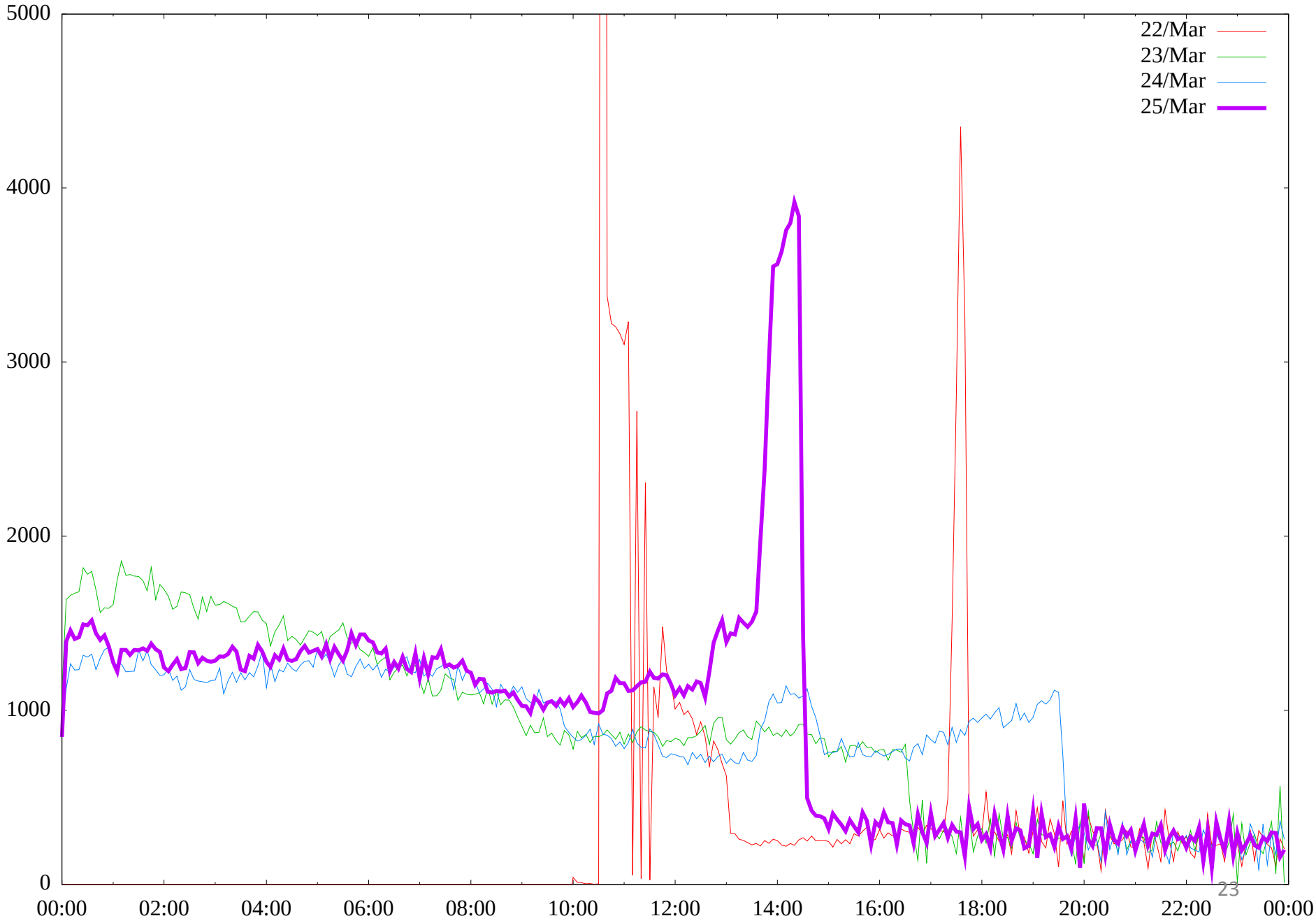
Ad Placement Training – Day 2



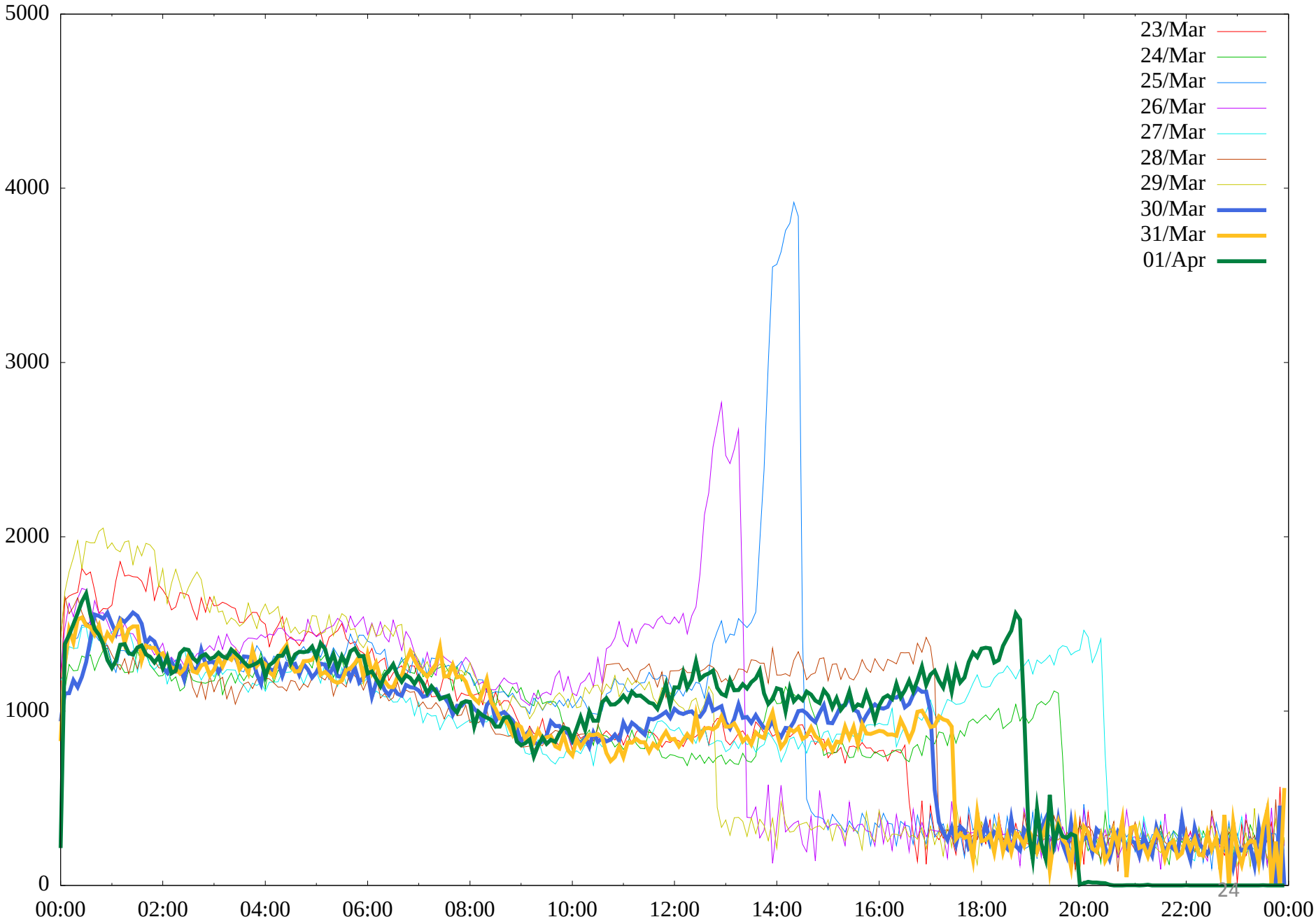
Ad Placement Training – Day 3



Ad Placement Training – Day 4



Ad Placement Training – Days 5, 6 & 7



Measurement Control Channel

- Use Flash code that is executed on ad impression that retrieves the actual measurement script
  - Ad carries code to send the client to retrieve an ad-controller URL

    http://drongo.rand.apnic.net/measureipv6id.cgi?advertID=9999
  - Client retrieves set of “tests” from the ad-controller as a sequence of URLs to fetch and a “result” URL to use to pass the results to the ad-server

- This allows us to vary the measurement experiment without necessarily altering the ad campaign itself – the ad, and its approval to run, remain unchanged so that measurements can be activated and deactivated in real time.

Experiment Server config

- There are currently three servers, identically configured (US, Europe, Australia)
- Server runs Bind, Apache and tcpdump
- Experiment directs the client to the “closest” server (to reduce rtt-related timeouts) based on simple /8 map of client address to region

Measuring IPv6 via Ads

Client is given 5 URLs to load:

- Dual Stack object
- V4-only object
- V6-only object
- V6 literal address (no DNS needed)
- Result reporting URL (10 second timer)

All DNS is dual stack

Measuring DNSSEC via Ads

Client is given 4 URLs to load:

- DNSSEC-validly signed DNS name
- DNSSEC-invalidly signed DNS name
- Unsigned DNS name (control)
- Result reporting URL (10 second timer)

All DNS is IPv4

Discovering Routing Filters via Ads

Client is given 3 URLs to load:

- DNS name that resolves into the test prefix
- DNS name the resolves to a control prefix
- Result reporting URL (10 second timer)

Caching

- Caching (generally) defeats the intent of the measurement
  - Although some measurements are intended to measure the effects of caching
- We use unique DNS labels and unique URL GET parameters
  - Ensures that all DNS resolution requests and HTTP fetch requests end up at the experiment's servers
- We use a common "tag" across all URLs in a single experiment
  - Allows us to join the individual fetches to create the per-user view of capability

Collected Data

- Per Server, Per Day:
  - http-access log

    (successfully completed fetches)
  - dns.log

    (incoming DNS queries)
  - Packet capture

    All packets

Collected Data

Web Logs:

```
h.labs.apnic.net 2002:524d:xxxx::524d:xxxx [29/Apr/2013:05:55:05 +0000] "GET /1x1.png?
    t10000.u7910203317.s1367214905.i888.v1794.v6lit
h.labs.apnic.net 2002:524d:xxxx::524d:xxxx [29/Apr/2013:05:55:05 +0000] "GET /1x1.png?
    t10000.u7910203317.s1367214905.i888.v1794.r6.td
h.labs.apnic.net 82.77.xxx.xxx [29/Apr/2013:05:55:05 +0000] "GET /1x1.png?
     t10000.u7910203317.s1367214905.i888.v1794.rd.td
h.labs.apnic.net 82.77.xxx.xxx [29/Apr/2013:05:55:05 +0000] "GET /1x1.png?
    t10000.u7910203317.s1367214905.i888.v1794.r4.td
h.labs.apnic.net 82.77.xxx.xxx [29/Apr/2013:05:55:05 +0000] "GET /1x1.png?
    t10000.u7910203317.s1367214905.i888.v1794&r=zrdtd-348.zr4td-376.zr6td-316.zv6lit-
228
```

(In this case the client is using 6to4 to access IPv6, and prefers to use IPv4 in a dual stack context)

Collected Data

DNS Logs:

```
27-Feb-2014 00:00:07.849 queries: client 12.121.116.213#54311 query:
f.t10000.u3934702783.s1393459207.i1022.v6022.47c34.z.dotnxdomain.net IN A -EDC
(199.102.79.186)
27-Feb-2014 00:00:07.850 queries: client 12.121.116.213#30544 query:
e.t10000.u3934702783.s1393459207.i1022.v6022.47c33.z.dashnxdomain.net IN A -EDC
(199.102.79.186)
27-Feb-2014 00:00:07.851 queries: client 12.121.116.213#55619 query:
d.t10000.u3934702783.s1393459207.i1022.v6022.47c33.z.dotnxdomain.net IN A -EDC
(199.102.79.186)
```

What does this allow?

- In providing an end user with a set of URLs to retrieve we can examine:
  - Protocol behaviour

    e.g.: V4 vs V6, protocol performance, connection failure rate
  - DNS behaviours

    e.g.: DNSSEC use, DNS resolution performance…

The generic approach

- Seed a user with a set of tasks that cause identifiable traffic at an instrumented server
- The user does not contribute measurements
- The server performs the data collection

Collision detection?

There was a thought that this approach could be used to perform collision detection:

Test:

http://<unique_id>-a.TestName.CandidateTLD/1x1.png?<uniqueid>-a

http://<unique_id>-a.TestName.ExistingTLD/1x1.png?<uniqueid>-b

http://results.TestName.ExistingTLD/1x1.png?<uniqueid>?za=<a_result>&zb=<b_result>

Result Analysis:

If the server sees a query for B and NOT A, then we can infer that there is possibly a collision for the use of CandidateTLD between local and globally scoped contexts

Really?

- But is this collision or the opposite?
- This shows the extent of local zone instances occluding a global zone
- But I thought we were looking for the possibility of global zone delegation altering the behaviour of client applications using / assuming a local zone resolution
- Which looks like the opposite

Furthermore …

- Is it the use of a local name or the content of local name search lists that is critical here?
- And what name forms trigger the local name resolution function to invoke the local search list to apply to the given name?
- Are we measuring the extent of name collision itself or the extent of deployment of various forms of name resolution with search lists?

What about…

Test:

http://<unique_id>-single-label-name/1x1.png?<uniqueid>-a

http://second-label.<unique_id>-single-label-name/1x1.png?<uniqueid>-b

http://<unique_id>-single-label-name.Existing.domain.name/1x1.png?<uniqueid>-c

http://results.TestName.Existing.domain.name/1x1.png?<uniqueid>?za=<a_result>&zb=<b_result>&zc=<c_result>

Question:

If we launched a high volume of ads, then what would we see at a root server?

A few observations

- Measuring what happens at the user level by measuring some artifact or behaviour in the infrastructure and inferring some form of user behaviour is going to be a guess of some form
- If you really want to measure user behaviour then its useful to trigger the user to behave in the way you want to study or measure
- The technique of embedding code behind ads is one way of achieving this objective, for certain kinds of behaviours relating to the DNS and to URL fetching

Questions?

APNIC Labs:

Geoff Huston **research@apnic.net**