Measuring DNS Source Port Randomness

Duane Wessels
DNS-OARC

1st CAIDA/WIDE/CASFI Workshop
August 15, 2008

Kaminsky

- DNS sucks.
  - Okay, I'm paraphrasing...
- Use random source ports to protect from poisoning
- But what about NATs?

How do you know if your DNS ports are random?

- http://www.doxpara.com
  - Web-only
  - Needs javascript
  - /etc/resolv.conf nameservers only
- Why not something strictly DNS-based?
- porttest.dns-oarc.net was born.

Lots of Queries

- We need lots of queries from a resolver in order to detect source port randomness.
  - CNAMEs
  - Delegations
- Resolvers typically limit CNAME chain lengths
  - To solve looping?
  - Probably on the order of 10–15?
  - doxpara uses CNAME chains (5)
  - Neils Provos test also
- Delegation chain
  - length not limited to my knowledge
  - requires unique IP per delegation
- Make resolvers query for long name like z.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.b.a.example.com
- Use a CNAME to start to avoid typing the long name

porttest.dns-oarc.net

- Implemented in Perl (Net::DNS::Nameserver)
- 26 delegations (a–z) and 26 IP addresses
- Return TXT record reporting measure of randomness
- Use short TTLs to allow test to be repeated from the same location.
- Log the results

Measuring Randomness

- There are various statistical tests for randomness, but:
  - I'm not very good with statistics
  - Some tests assume a Normal distribution
  - Some tests require a lot of samples.
- So I cheat and use standard deviation as a measure of randomness.
- It's easy to imagine samples that have high standard deviation but low randomness.
- To account for repeated ports, I multiply the calculated standard deviation by the ratio of unique samples to total samples.
- Its not perfect, but its pretty good and at least some people can understand it.

Standard Deviation

- The standard deviation of a sample from a discrete uniform distribution of size N is:

$$\sigma = \sqrt{\frac{N^2-1}{12}}$$

- Given the standard deviation of a sample, we can estimate the number of $bits$ in the sample size as:

$$bits = \log_2 \sqrt{12\sigma^2 + 1}$$

- Scoring:

Score	σ Range	$bits$ Range
GREAT	3980 – 20,000+	13.75 – 16.0
GOOD	296 – 3980	10.0 – 13.75
POOR	0 – 296	0 – 10.0

How It Looks

- with *dig*:

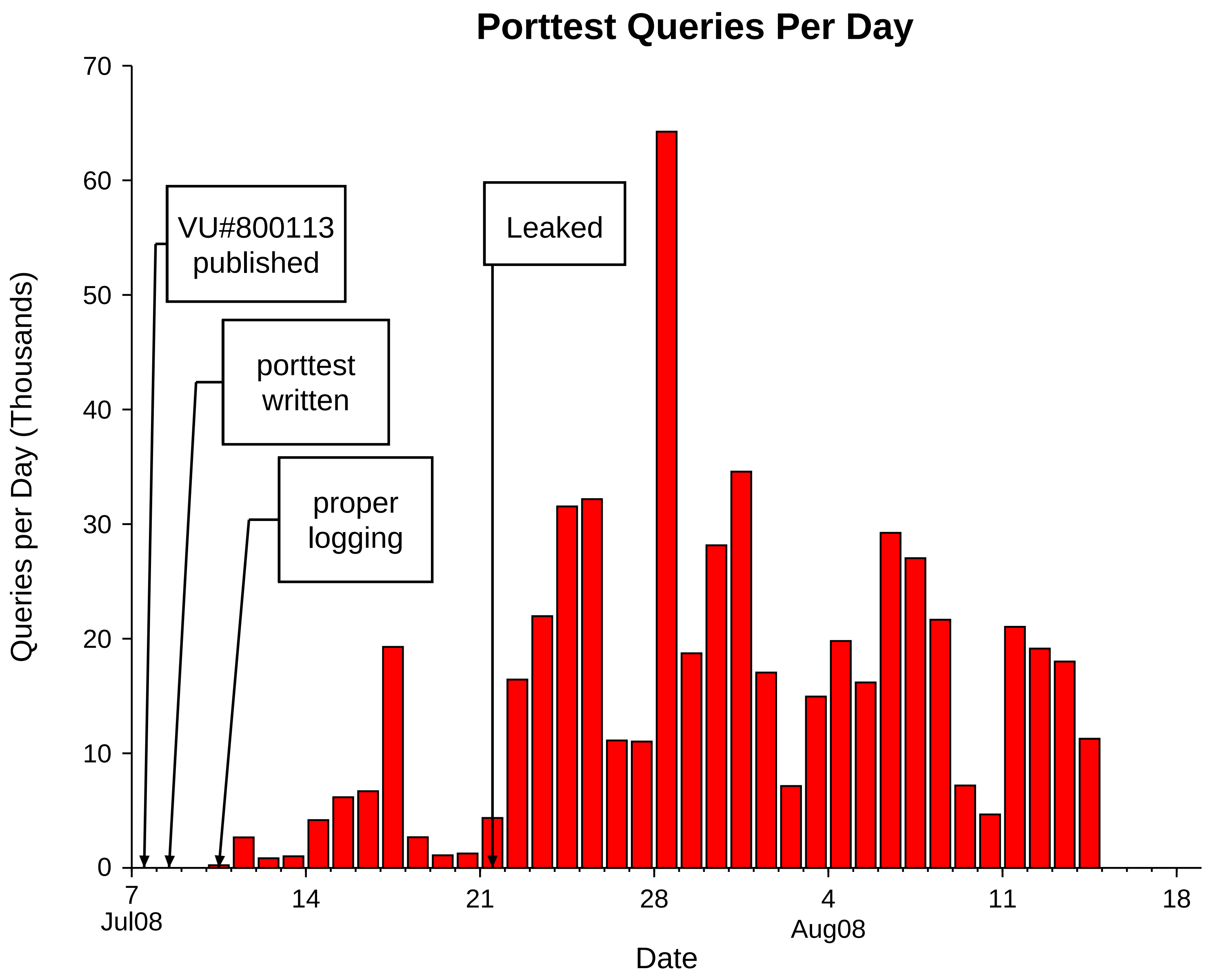
```
$ dig +short porttest.dns-oarc.net txt
porttest.y.x.w.v.u.t.s.r.q.p.o.n.m.l.k.j.i.h.g.f.e.d.c.\
b.a.pt.dns-oarc.net.
"12.160.37.12 is GREAT: 26 queries in 3.1 seconds \
from 26 ports with std dev 19551"
```

- or...

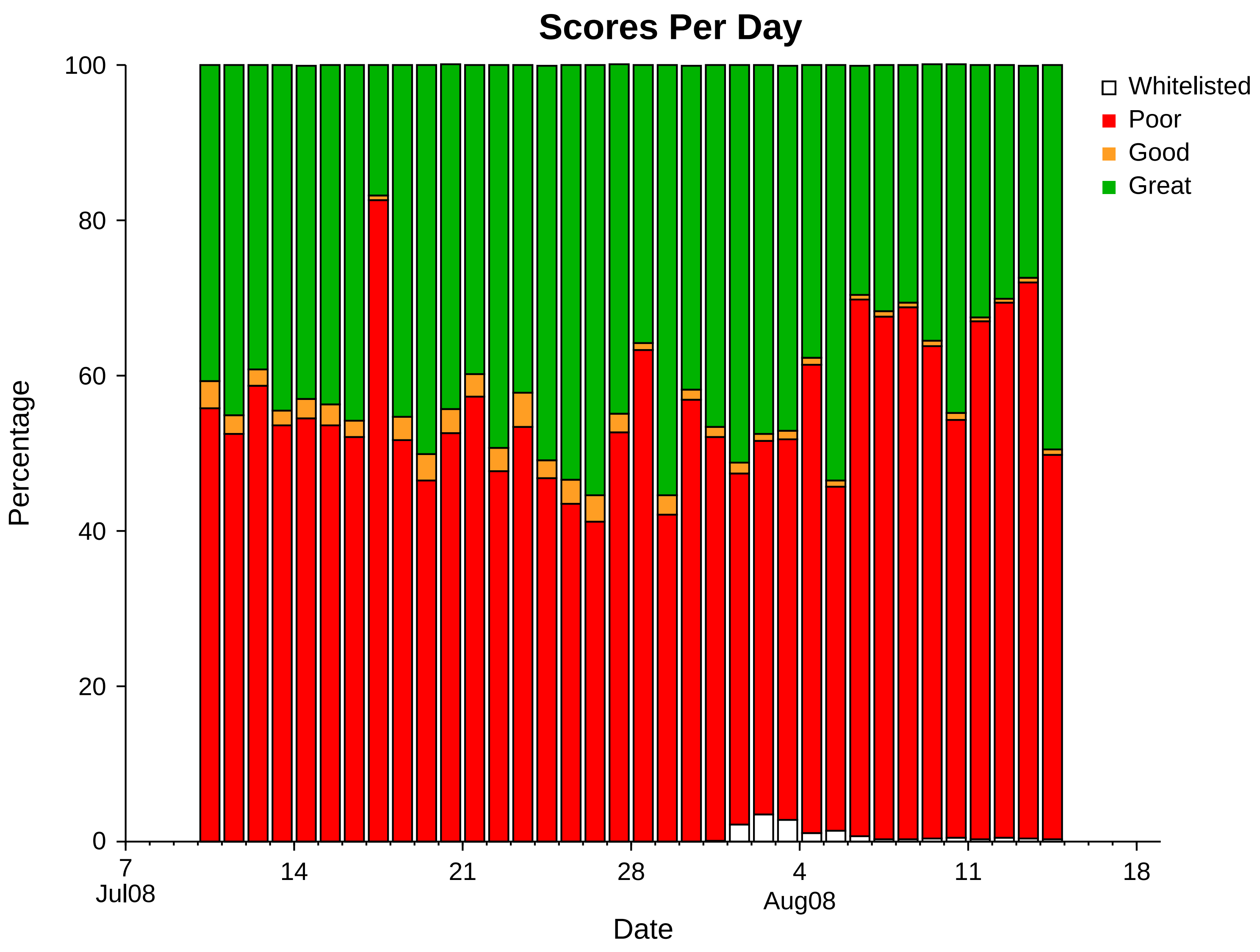
```
85.196.68.238 is POOR: 26 queries in 45.5 seconds \
from 24 ports with std dev 69
```

- or...

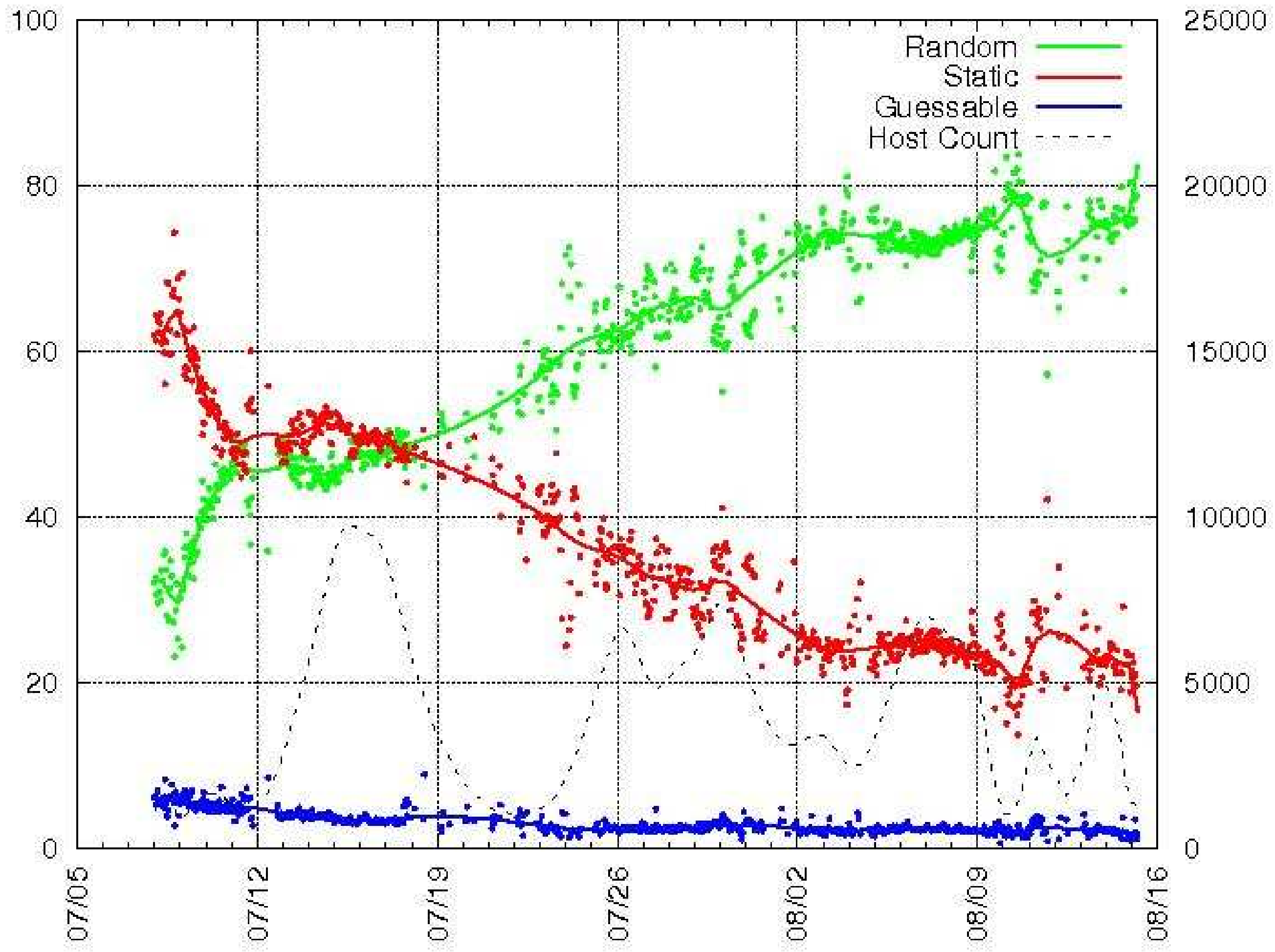
```
216.55.97.81 is POOR: 26 queries in 1.9 seconds \
from 1 ports with std dev 0
```



Porttest Queries Per Day
Queries per Day (Thousands)
0
10
20
30
40
50
60
70
VU#800113 published
porttest written
proper logging
Leaked
7 Jul08
14
21
28
4 Aug08
11
18
Date


Scores Per Day
Percentage
100
80
60
40
20
0
7
Jul08
14
21
28
4
Aug08
11
18
Date
Whitelisted
Poor
Good
Great

Compare to Sid's SIE Data



Nominum

- Nominum didn't want to a lot of bits of source port randomness, for whatever reason.
- Implemented additional anti-spoofing/anti-poisoning features.
  - Such as switching to TCP upon detection of a spoof attempt.
- Upset that their nameservers not rated "GREAT."
- Now whitelisted (as of 2008-07-31) based on list of addresses they provide.

Web-based Tool

- Vixie suggested to Myself, Dagon, and Neils that OARC should host a web-based randomness test. Google ads would direct users to the page.
- The Google ads didn't quite pan out, but I think the tool turned out nicely.
- Advantages:
  - Good for people that can't use *dig*.
  - Provides lots more information that a TXT response.
  - Might end up testing more than one resolver at a time.
- Disadvantages:
  - Can only test system-configured resolvers.

Implementation

- Begins with an HTTP request. The HTTP response is a redirect to a URL with randomly generated name:

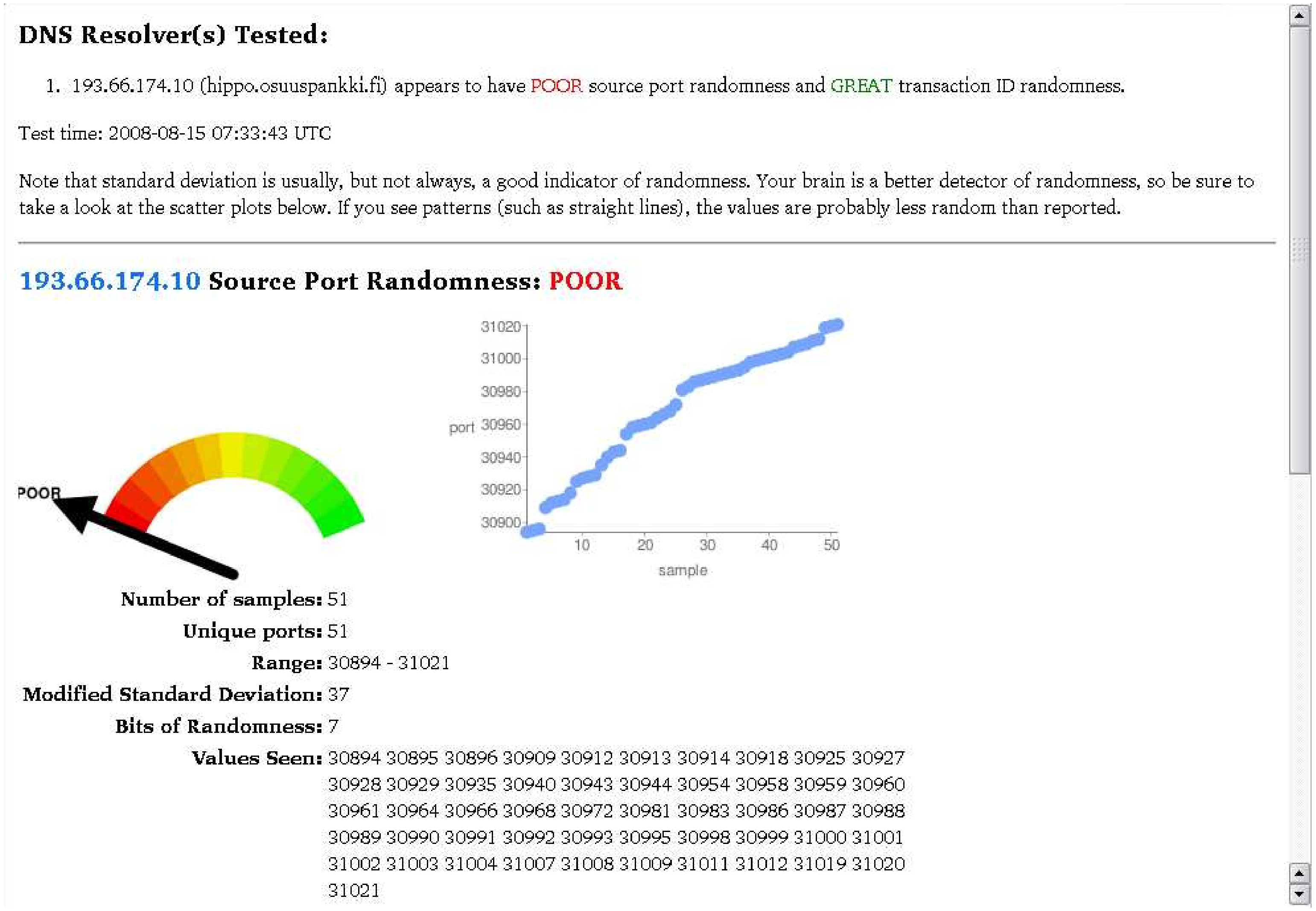
```
Location: http://bd0974adaae13c8268077657.et.dns-oarc.net
```

- The random string becomes a “cookie.” It contains random parts and a timestamp.
- The first DNS request returns a CNAME with the cookie expanded to a sequence of separate zones:

```
bd0974adaae13c8268077657.et.dns-oarc.net. 3600 IN CNAME \
b.d.0.9.7.4.a.d.a.a.e.1.3.c.8.2.6.8.0.7.7.6.5.7.et.dns-oarc.net.
```

- The last nameserver returns the web server address where a CGI script uses the cookie to read the query history from an SQL database and present the results.

http://entropy.dns-oarc.net/test/



DNS Resolver(s) Tested:

1. 193.66.174.10 (hippo.osuuspankki.fi) appears to have POOR source port randomness and GREAT transaction ID randomness.

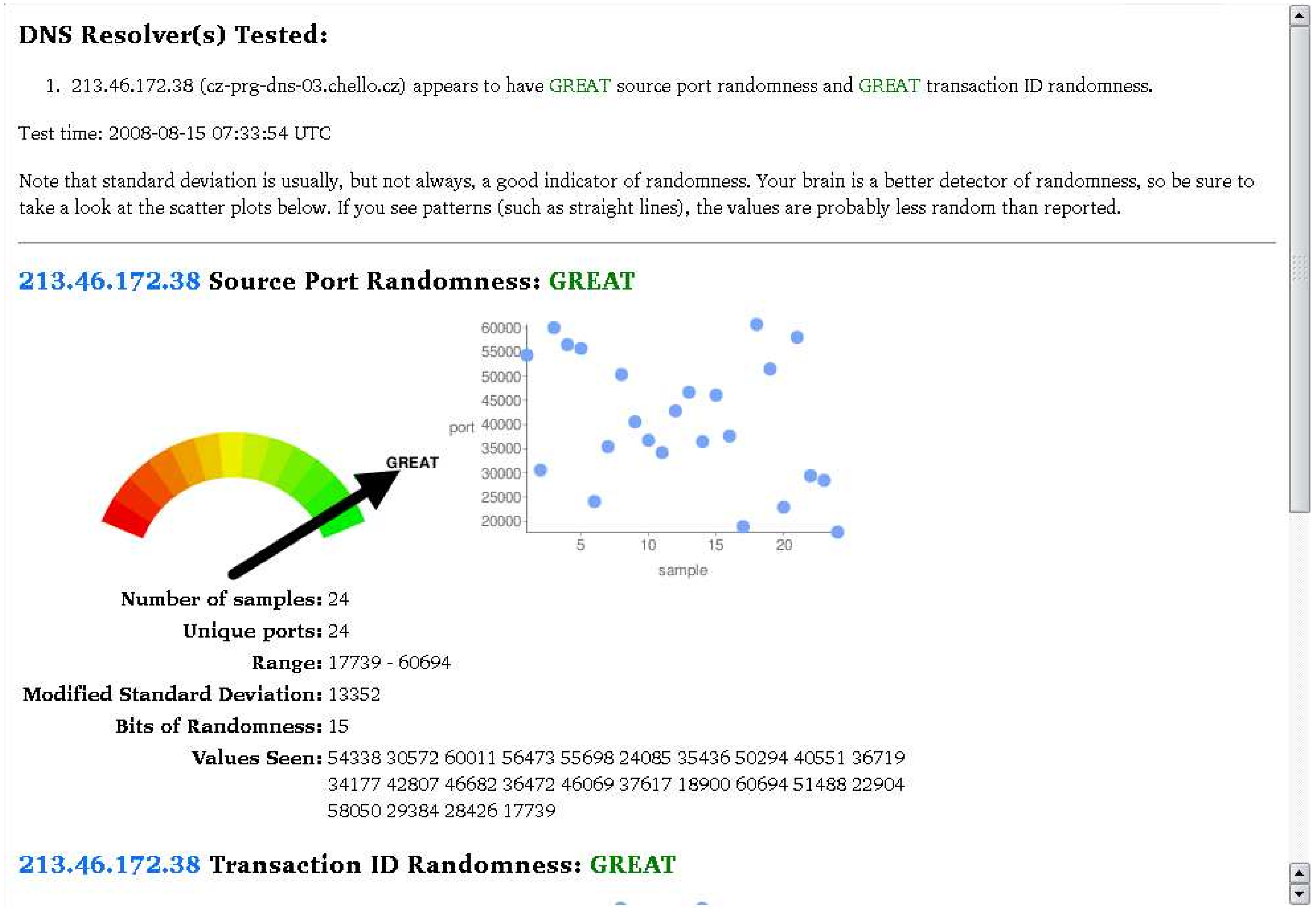
Test time: 2008-08-15 07:33:43 UTC

Note that standard deviation is usually, but not always, a good indicator of randomness. Your brain is a better detector of randomness, so be sure to take a look at the scatter plots below. If you see patterns (such as straight lines), the values are probably less random than reported.

193.66.174.10 Source Port Randomness: POOR

Number of samples:	51
Unique ports:	51
Range:	30894 - 31021
Modified Standard Deviation:	37
Bits of Randomness:	7
Values Seen:	30894 30895 30896 30909 30912 30913 30914 30918 30925 30927 30928 30929 30935 30940 30943 30944 30954 30958 30959 30960 30961 30964 30966 30968 30972 30981 30983 30986 30987 30988 30989 30990 30991 30992 30993 30995 30998 30999 31000 31001 31002 31003 31004 31007 31008 31009 31011 31012 31019 31020 31021

http://entropy.dns-oarc.net/test/



DNS Resolver(s) Tested:

1. 213.46.172.38 (cz-prg-dns-03.chello.cz) appears to have GREAT source port randomness and GREAT transaction ID randomness.

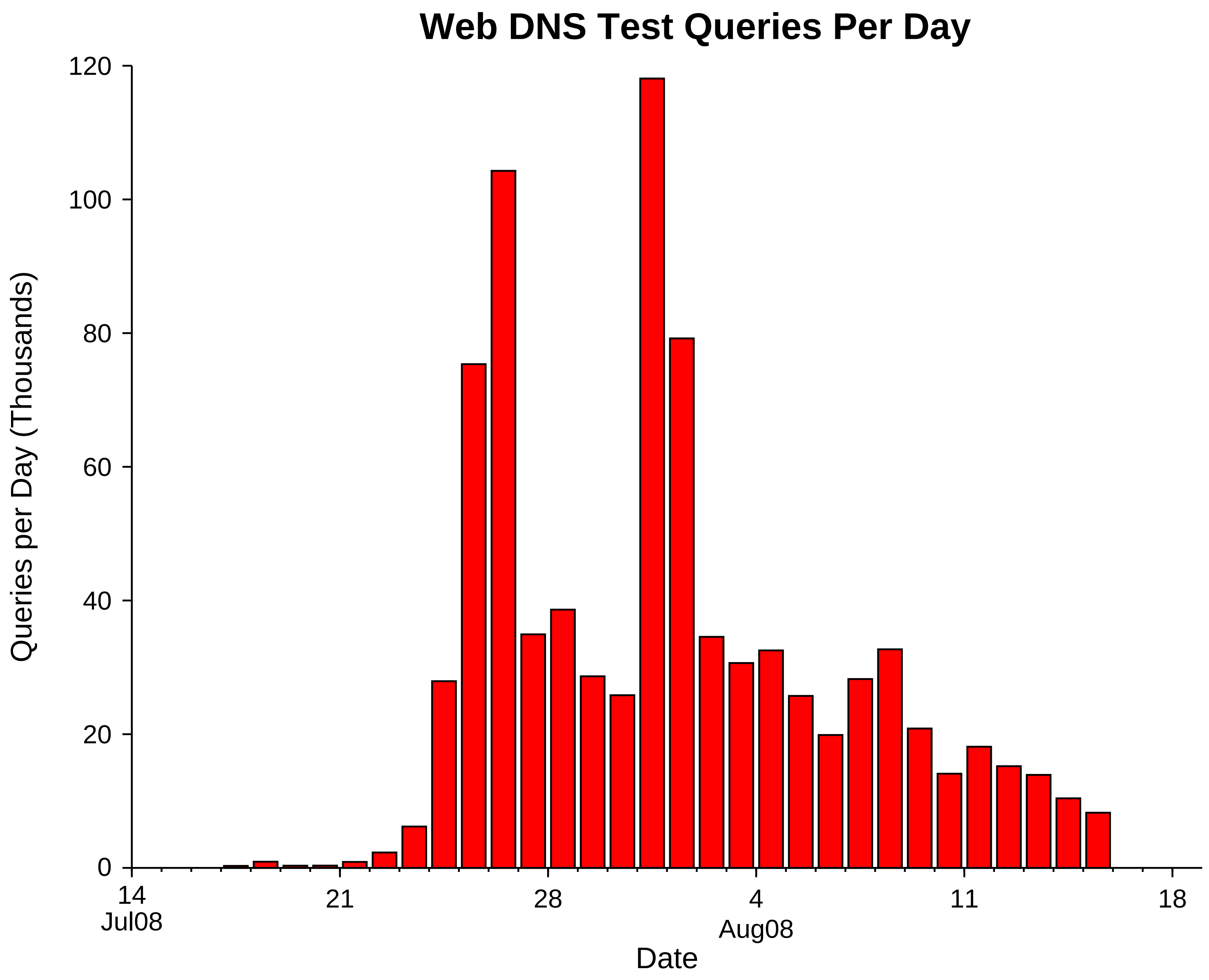
Test time: 2008-08-15 07:33:54 UTC

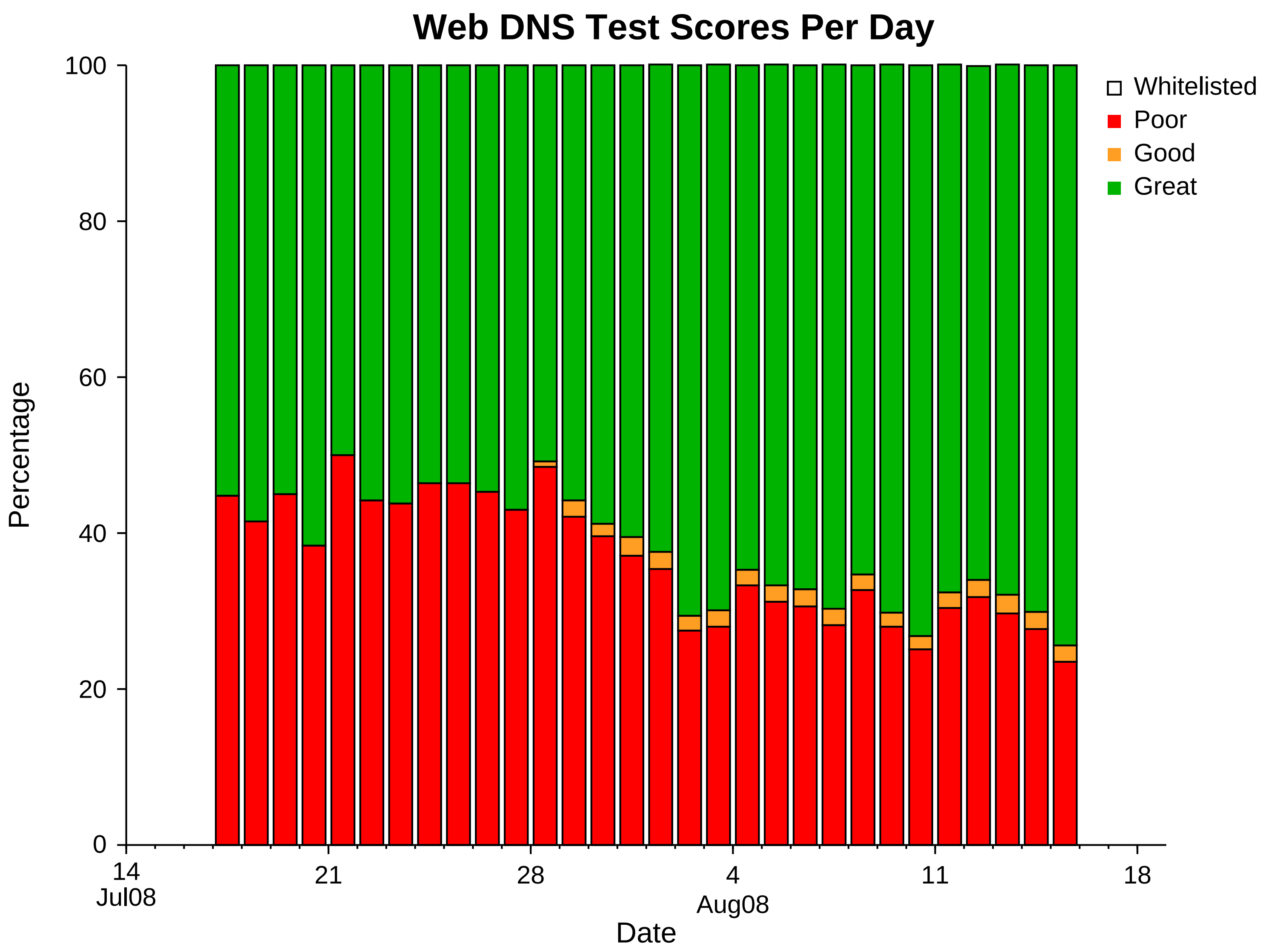
Note that standard deviation is usually, but not always, a good indicator of randomness. Your brain is a better detector of randomness, so be sure to take a look at the scatter plots below. If you see patterns (such as straight lines), the values are probably less random than reported.

213.46.172.38 Source Port Randomness: GREAT

Number of samples: 24
Unique ports: 24
Range: 17739 - 60694
Modified Standard Deviation: 13352
Bits of Randomness: 15
Values Seen: 54338 30572 60011 56473 55698 24085 35436 50294 40551 36719 34177 42807 46682 36472 46069 37617 18900 60694 51488 22904 58050 29384 28426 17739

213.46.172.38 Transaction ID Randomness: GREAT


Web DNS Test Queries Per Day
Queries per Day (Thousands)
0
20
40
60
80
100
120
14
Jul08
21
28
4
Aug08
11
18
Date


Web DNS Test Scores Per Day
Percentage
0
20
40
60
80
100
14
Jul08
21
28
4
Aug08
11
18
Date
Whitelisted
Poor
Good
Great

How To Not Be Poisoned

- Deploy DNSSEC
- Have good transaction ID randomness
- Have good source port randomness
- Implement dns-0x20 (random upper-/lower-casing of query name)
- Use multiple source addresses (unbound, powerdns)
- Detect spoof attempts (nominum, powerdns)
- Require multiple matching authoritative answers
- Add nonce via EDNS0.
- TCP

Final Thoughts

- This testing tool is probably “self selecting” such that it tends to attract sources that are not yet updated. It is not a good indicator of patching rates.
- Should calculate Wald-Wolfowitz Z-scores and see if they correlate to standard deviation.
- Notify network operators of still-vulnerable resolvers.

The End