Two Separate Problems: Questions and Answers
DNS Answers

• Complex data:
  Time, source, destination, question, answer, additional records.

• What did a name resolve to and when?

• What else resolved to this IP?

• What else is served by this name server?

• Solution: Passive DNS Replication

• https://dnsdb.isc.org/

• http://www.bfk.de/bfk_dnslogger.html

• http://www.enyo.de/fw/software/dnslogger/
DNS Questions

• Simple data:
  Source IP.
  Destination IP.
  Question – qname and qtype.

• Great for forensics.

• Who looked up this name?

• Who is a member of this botnet?

• What did this host look up?
What About Scale?
How Much Data?

- 130k active Windows hosts at any given time.
- 90k active Linux hosts at any given time.
- 10 billion Netflows captured at zone boundaries.
- IDS sensors at all zone boundaries.
- 1 Tb of security event log data.
- 13 data centres, DMZs covered with PDNS.
- 4 billion DNS and NetBIOS packets captured per day.
- 300gb of traffic captured per day.
DNS traffic capture design
Capturing Data

- Pairs of Cisco UCS Appliances at 13 POPs.
- Attached to Cisco CAT6k switches.
  - Configured as DNS load balancers.
  - vlan ACL sends only UDP/53 and UDP/135.
- Data captured with ncaptool.
  - BPF selects DNS questions.
- Compressed and stored to local disk.
- One minute per file, per site.
Making Data Accessible

• Restricted upstream bandwidth.
• Limited physical access to hosts.
• Highly sensitive data.
• Distributed search engine written in Python.
  JSON-based protocol.
  Use SSL certs for authentication.
• Time range + filters.
• Libbind for parsing.
• IPy for address normalization.
Filtering Large Lists

• Heavy use of tries.
• Patricia tries for IPv4 and IPv6 addresses.
• pySubnetTree module rewritten to support IPv6.
• Optimized regular expressions for string matching.
• Rough port of perl’s Regexp::Optimizer.
  abc, aac, abd = (?a(?:ac|b[cd]))
Faster Searching

• Prior work: Netflow, SiLK tools.

• Full indices impractical.
  Large disk storage requirements.
  Computational overhead.
  Aging out of data expensive.

• Bloom filter index.
  pybloomfilter module.
  Quickly determines whether a file contains entries that match a query.
  Pre-computation by cron job.

• Keyspace:
  Domains broken by part: www.cisco.com = www.cisco.com, cisco.com, com
  Addresses broken by supernets based on global allocation stats.
     IPv4 : (8,16,17,18,19,20,21,22,23,24)
     IPv6 : (32,33,34,35,36,40,44,46,48,64)
Even Faster Searching

- Python object creation is very expensive.
  pynap creates several hashes and arrays that must also be destroyed.
- Modified the Pyrex code to create a pre-filter before callback.
- 100x+ speedup.
- Enables other crazy stuff too.
Command Line Interface

usage: pdns-search [-h] [--src-ip [SRC_IP [SRC_IP ...]]]
    [--dst-ip [DST_IP [DST_IP ...]]]
    [--qname [QNAME [QNAME ...]]] [--qtype [QTYPE [QTYPE ...]]]
    [--nbname [NBNAME [NBNAME ...]]]
    [--nbtype [NBTYPE [NBTYPE ...]]]
    [--nbsuffix [NBSUFFIX [NBSUFFIX ...]]]
    [--max-results MAX_RESULTS] [--start START] [--end END]
    [--no-extract] [--no-expand] [--no-progress]
    [--print-server] [--print-protocol]
## Demo – Mariposa Infections

```
$ pdns-search --qname bfisback.no-ip.org --max-results 4
```

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Source</th>
<th>Destination</th>
<th>QName</th>
<th>QType</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012-08-21 12:26:28</td>
<td>ELIDED</td>
<td>64.102.255.44</td>
<td>bfisback.no-ip.org</td>
<td>A</td>
</tr>
<tr>
<td>2012-08-21 12:26:28</td>
<td>64.102.255.43</td>
<td>69.65.40.108</td>
<td>bfisback.no-ip.org</td>
<td>A</td>
</tr>
<tr>
<td>2012-08-21 13:03:19</td>
<td>ELIDED</td>
<td>64.102.255.44</td>
<td>bfisback.no-ip.org</td>
<td>A</td>
</tr>
<tr>
<td>2012-08-21 13:03:19</td>
<td>64.102.255.43</td>
<td>69.72.255.8</td>
<td>bfisback.no-ip.org</td>
<td>A</td>
</tr>
</tbody>
</table>

Search: 100% | #FFFFFFFFFFFFF| Time: 0:00:03 Files: 780/780
Freebie: NetBIOS

- DNS packet format.
- Overrides QTypes NIMLOC and SRV with NB and NBSTAT.
- QName encoded as per RFC1002.
- HESTERN-MAC<0>
- E1E0F8DFEEFFCE0CNEEBEDC4CAAAA IN NB
In-Situ NetBIOS Filtering

• Encode hostnames in RFC1002 format and match like strings.

• Case insensitivity desired.
  
  Match [EG] and [FH] for first quartet of A-Z.

• `^(?:(?::hostname with types)|
  (?::(?!HO)[A-P]{2})*HO)?(?::hostname without types))
  (?::CA)*(?::service suffixes)\$`
Using Our Tools

- Fill in gaps in Netflow coverage.
- Peek into SSL sessions.
- Alert on queries for dangerous domain names.
- Look for patterns in queries to discover C2 servers.
- Monitor queries about high-value targets.