C-DNS:

A DNS Packet Capture Format

ICANN DNS Engineering Team

Presenter: Sara Dickinson, Sinodun
C-DNS Agenda

• Motivation for a new format
• Design decisions
• Describe Compacted-DNS (C-DNS) format
• Standardisation effort: draft-ietf-dnsop-dns-capture-format
• Implementation status
DNS Packet Capture

- Capturing DNS traffic on the wire: Why do it?
  - General traffic analysis
  - Security - detect attacks in real time
  - Post event analysis - research
    - Community efforts e.g. DITL
What is done today?

• **PCAP** files - full packet capture
• DNS message capture tools
  • **DNSTAP**, **DNSCAP**, **PacketQ**
• **DSC** - counts of DNS of traffic metrics
What is done today?

• BUT no **standard** interchange format for DNS traffic

• PCAP is very common

  • However contains lots of information not directly relevant to DNS analysis

Larger than necessary capture files
## Data Capture Environments

Wide range of possibilities.....

<table>
<thead>
<tr>
<th>Hosting</th>
<th>Conditions</th>
<th>Hardware</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-hosted</td>
<td>Well provisioned, steady state</td>
<td>Port mirroring</td>
<td>Out-of-band upload</td>
</tr>
<tr>
<td>Third-party hosted</td>
<td>Heavily loaded</td>
<td>Network tap</td>
<td>Limited out-of-band upload</td>
</tr>
<tr>
<td>Third-party hardware</td>
<td>Under attack</td>
<td>Same h/w as nameserver</td>
<td>Everything in-band</td>
</tr>
</tbody>
</table>

Increasingly constrained
C-DNS Project Background

- ICANN DNS Engineering team is responsible for the Root Server operated by ICANN

- DNS-STATS.org created in 2014: A covering entity for the implementation of open source DNS statistics collection and presentation software.

- Sinodun contracts for DNS Eng Team on DNS-STATS work e.g. Hedgehog - a DNS traffic statistics presentation tool (DSC XML)
C-DNS Project Background

- ICANN operates 155+ anycast instances, 7 billion q/day
- Majority are hosted, in many different types of networks, by many different organisations. Some constrained, AND all on a 1RU server
- Historically used a combination DSC XML + PCAP
- Now need a more general purpose and scalable solution
C-DNS Project Goal: Target most limited use case

• Data collection on same hardware as nameserver

• Constrained instance resources: CPU, bandwidth (Focus on serving DNS, not capturing DNS!)

• Collected data stored on same hardware (At least temporarily)

• Upload will use the same interface as DNS traffic (Can be artificially throttled)
Technical requirements

1. **Minimise the file size** for storage and transmission

2. **Minimise the overhead** of producing the packet capture files
   - And further general purpose compression

3. Desirable: **Re-construction** e.g. PCAP
Design Considerations (1)

1. **Basic Unit is Q/R item:** Combine DNS Query and associated Response
   - A single DNS ‘transaction’, commonality in data

2. **Collect ‘default’ Q/R data:** Optionally capture other data
   - Storage constraints vs ability to reconstruct fully

3. **Block storage:** Collected data into blocks of Q/R items
   - Abstract common data and reference by indexing e.g. IP addresses, QNAMES

Use DNS specific domain knowledge to achieve compression
4. **Also collect metadata**: ICMP, TCP resets, malformed DNS counts

5. **Optionally collect malformed DNS packets**: Any structured format is limited in what can be recorded

   - Malformed queries generate well formed responses
   - Attack traffic may be malformed
What storage format to use?

- Considered several binary representations: CBOR, Apache Avro, Protobuffers

- Assume further compression with a general purpose tool (xz, gzip,...)

  Testing showed only minor size differences between the formats

  => So consider other factors
CBOR

- What is it?
  A serialisation format comparable to JSON but with binary representation

- Why use it?
  - IETF standard ([RFC7049](https://tools.ietf.org/html/rfc7049))
  - Simple format, simple to implement (16 languages)
  - CDDL draft - CBOR data definition language
  - Converts to JSON nicely
C-DNS Format
C-DNS Conceptual Overview

- File type identifier
- Preamble
  - Format version
  - Capture configuration
  - Capture metadata
- Data block
  - Block start time
  - Block statistics
  - Block tables
  - Query/Response data items
  - Address/Event count data items
- Further data blocks ...

ICANN DNS Symposium, Madrid, May 2017
C-DNS Conceptual Overview

File ID and Preamble

Further data blocks ...

Data block
Block start time
Block statistics
Block tables
Query/Response data items
Address/Event count data items

Data block
Block start time
Block statistics
Block tables
Query/Response data items
Address/Event count data items

File type identifier
Preamble
Format version
Capture configuration
Capture metadata
C-DNS Conceptual Overview

File ID and Preamble

Data Block 1
C-DNS Conceptual Overview

File ID and Preamble

Data Block 1

Data Block 2
C-DNS Conceptual Overview

File ID and Preamble

Data Block 1

Data Block 2
C-DNS Conceptual Overview

- File ID and Preamble
- Data Block 1
- Data Block 2

- File type identifier
- Preamble
- Format version
- Capture configuration
- Capture metadata

- Block start time
- Block statistics
- Block tables
- Query/Response data items
- Address/Event count data items

- Further data blocks...
C-DNS Conceptual Overview

- File ID and Preamble
- Data Block 1
- Data Block 2
C-DNS Conceptual Overview

File ID and Preamble

Data Block 1

Data Block 2
Q/R Items + Block data
Q/R Items + Block data

Key
Not present if data not available
Optional data, not present unless collection configured
Q/R Items + Block data

Key
Not present if data not available
Optional data, not present unless collection configured
Q/R Items + Block data

Key
- Not present if data not available
- Optional data, not present unless collection configured

Query/Response
- Time offset
- Response delay
- Client address
- Client port
- Client hoplimit
- Transaction ID
- Query signature
- Query name
- Response size
- Extra query info
- Question
- Answer
- Authority
- Additional
- Extra response info
- Answer
- Authority
- Additional

Query signature table
- Server address
- Server port
- Transport flags
- Query QNAME
- Query QNAME flags
- Query QPCODE
- Query QCODE
- Query class/type
- Query QD count
- Query QD count
- Query QD count
- Query QD count
- Query NS count
- Query NS count
- Query EDNS version
- EDNS UDP size
- Query Opt RDATA
- Response RCODE

IP address table
- IPv4 or IPv6 address

Name/RDATA table
- Name/RDATA

Class and type table
- Class
- Type

Question list table
- Question

Question table
- Name
- Class/type

RR table
- Name
- Class/type
- TTL
- RDATA
Q/R Items + Block data

Key
Not present if data not available
Optional data, not present unless collection configured
Q/R Items + Block data
CDDL representation

- Multiple tables - complicated but achieves goals

```c
QueryResponse = {
    time-useconds => uint, ; Time offset from start of block
    ? time-pseconds => uint, ; in microseconds and picoseconds
    client-address-index => uint,
    client-port => uint,
    transaction-id => uint,
    query-signature-index => uint,
    ? client-hoplimit => uint,
    ? delay-useconds => int,
    ? delay-pseconds => int, ; Has same sign as delay-useconds
    ? query-name-index => uint,
    ? query-size => uint, ; DNS size of query
    ? response-size => uint, ; DNS size of response
    ? query-extended => QueryResponseExtended,
    ? response-extended => QueryResponseExtended,
}
```
Draft contains a non-normative proposal for how to match queries and responses:

**Primary key**: 6 point tuple of
- IP Addrs, ports, transport, Msg ID

**Secondary key**: (if present)
- First Question
Draft contains a non-normative proposal for how to match queries and responses:

**Primary key**: 6 point tuple of
- IP Addr, ports, transport, Msg ID

**Secondary key**: (if present)
- First Question

WARNING: Packet capture libraries don’t guarantee to return packets in time order
So... Results: Block size

Tests done using sample data from a Root Server
So... Results: Block size

Tests done using sample data from a Root Server

Optimal block size is around 5-10,000 items
So... results: File size

<table>
<thead>
<tr>
<th>Format</th>
<th>PCAP</th>
<th>C-DNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>File size (Mb)</td>
<td>660</td>
<td>75</td>
</tr>
<tr>
<td>Compressed with ‘xz -9’ (Mb)</td>
<td>49</td>
<td>18</td>
</tr>
<tr>
<td>User time for compression (s)</td>
<td>161</td>
<td>39</td>
</tr>
</tbody>
</table>
So... results: File size

<table>
<thead>
<tr>
<th>Format</th>
<th>PCAP</th>
<th>C-DNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>File size (Mb)</td>
<td>660</td>
<td>75</td>
</tr>
<tr>
<td>Compressed with ‘xz -9’ (Mb)</td>
<td>49</td>
<td>18</td>
</tr>
<tr>
<td>User time for compression (s)</td>
<td>161</td>
<td>39</td>
</tr>
</tbody>
</table>

**COMPRESSED SIZE:** C-DNS is 30-40% size of PCAP
So... results: File size

<table>
<thead>
<tr>
<th>Format</th>
<th>PCAP</th>
<th>C-DNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>File size (Mb)</td>
<td>660</td>
<td>75</td>
</tr>
<tr>
<td>Compressed with <code>xz -9</code> (Mb)</td>
<td>49</td>
<td>18</td>
</tr>
<tr>
<td>User time for compression (s)</td>
<td>161</td>
<td>39</td>
</tr>
</tbody>
</table>

**COMPRESSED SIZE:** C-DNS is 30-40% size of PCAP

**COMPRESSION CPU:** C-DNS uses ~25% of PCAP
Regenerating PCAPs

1. **May not have captured:** Entire message (optional)

2. **Cannot properly reconstruct:**
   - IP Fragmentation, TCP streams

3. **Do not capture:**
   - ICMP, TCP resets (only counts)

4. **Name compression:** Different algorithms used
   - NSD vs Knot detailed in draft

This is a ‘lossy’ process
DNSOP - Draft Status

- Oct 2016: Submitted first draft
- Nov 2016: IPR disclosure relating to pending patent application (No updates to this since then)
- Dec 2016: Draft adopted by WG
- May 2016: Now on -02 revision, no major issues outstanding (more work on malformed packets)
Implementation Status

• Running code to capture C-DNS and perform lossy reconstruction of PCAP

• Deployed in ICANN operated Root Server in Sept 2016

• Architecture allows for separate capture of ‘ignored’ packets in PCAP files i.e. all packets that are not stored in CBOR
Summary

• New DNS traffic capture format progressing as a IETF Proposed Standard

• Significant saving is file size over e.g. PCAP

• Possible to regenerate PCAP in lossy fashion

• Next steps? Consume C-DNS directly for analysis and visualisation
Thank you!

Any questions?