Domain Name System (DNS)

- Decentralized naming system for resources.
- Hierarchical.

Some resource records (RR):
- A: IPv4
- AAAA: IPv6
- CNAME: Alias
- MX: Mail exchange record
- NS: Name server

Some return codes:
- NOERROR: all ok
- SERVFAIL: server failed to complete the DNS request
- NXDOMAIN: domain does not exist
- NSD: Name server does not exist

Diagram:
- Root
- .com
- .org
- .edu
- .fr
- .cl
- .niclabs
- .ñandú
- .edu
- .com
- .org
- .edu
- .fr
- .cl
- .niclabs
- .ñandú
- Root
- Name server
- Name server
- Name server
- Name server
- Client A
- Client B
- Client C
NIC Chile operations

- Administrator of the "cl" ccTLD.
- More than 550,000 registered domains.
- 26+ nodes directly managed on 10+ countries.
- Two external DNS clouds
  - Netnod
  - Packet Clearing House (PCH)
Why is DNS monitoring interesting?
Context: Why is DNS monitoring interesting?
why is DNS monitoring interesting?

Image from Github

Context: why is DNS monitoring interesting?
Why is DNS monitoring interesting?

- 2016: Dyn DNS attack.
  - More than 1,200 affected domains.
  - Peak of 1.2 Tbps.
  - 2 hours between detection and resolution for every event.
How is DNS Monitored?

- DNS Statistics Collector (DSC)
- Pre-Aggregated Data
- Post-Aggregation
- Stats by server
- ENTRADA
- DNS-STATS
- Transfer pcap files to Hadoop Cluster for processing

QTYPE
OPCODE
RCODE
First Try: Develop our own solution

We developed RaTA DNS (Real Time Analysis of DNS packets)

- Capture and reduce information.
- Transfer results over REDIS Queue.
- Show the information on our own presenter.

Were we reinventing the wheel?
Second Try: Use Open Source Software

- Instead of developing everything, integrate different open source software.
- Many parts of a monitoring system have already been developed.
- Many of them are used in production.
What we wanted to measure?

<table>
<thead>
<tr>
<th>Packet Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Question</td>
</tr>
<tr>
<td>○ ResponseCode</td>
</tr>
<tr>
<td>● DoBit</td>
</tr>
<tr>
<td>○ Edns0</td>
</tr>
<tr>
<td>○ Type</td>
</tr>
<tr>
<td>○ Class</td>
</tr>
<tr>
<td>○ Opcode</td>
</tr>
<tr>
<td>○ QR</td>
</tr>
<tr>
<td>● DNS Query/Response</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DNS/ARP QR/Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Size</td>
</tr>
<tr>
<td>○ Network Protocol</td>
</tr>
<tr>
<td>○ IP Prefix</td>
</tr>
<tr>
<td>○ IP Version</td>
</tr>
<tr>
<td>○ Server Name</td>
</tr>
<tr>
<td>○ DateTime</td>
</tr>
<tr>
<td>○ Packet Metadata</td>
</tr>
</tbody>
</table>
Requirements

- DNS Packet Capture
- Storage
- Visualization
Requirements

- Secure
- Fast
- Low Cost

DNS Packet Capture

- Scalable
- Information
- Big Volume of
- Fast to process
- Compressed
- Unitary
- Fast Access
- Relevant Information
- Alert Abnormalities

Visualization

Storage
Software to analyze

Capture

Storage

Visualization

PacketBeat

DSC

gopassive

Fleev

ClickHouse

Prometheus

ElasticSearch

InfluxDB

Kibana

Graphite

Grafana

Graphite

ElasticSearch

InfluxDB

ClickHouse

Prometheus

OpenTSDB

Graphite

ElasticSearch

InfluxDB

ClickHouse

Prometheus

OpenTSDB
<table>
<thead>
<tr>
<th></th>
<th>gopassivedns</th>
<th>dsc</th>
<th>collectd</th>
<th>Packetbeat</th>
<th>Fievel</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPv4</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>IPv4</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>IPv6</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>IPv6</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>UDP</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>TCP</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Disaggregated Information</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
Packet Capture

DnsZeppelin: DNS Packet capturer.
- Based on PacketBeat and gopassive.
- Fragmented IP Assembly.
- TCP Assembly.
- Direct connection to database system.

Source code: https://github.com/niclabs/dnszeppelin

Packet Capture
Software to analyze

- PacketBeat
- Collectd
- Fievel
- DSC
- gopassivedns
- DnsZeppelin

Capture

- Prometheus
- Druid
- ClickHouse
- InfluxDB
- ElasticSearch
- OpenTSDB

Storage

- Kibana
- Grafana
- OpenTSDB
- Elasticsearch
- InfluxDB
- Druid
- Prometheus

Visualization

- Packetbeat
- Fluentd
- Collector
- DSC
- gopassivedns
- DnsZeppelin

Software to analyze
Benchmark

- CPU Usage
- Primary Memory
- Secondary Memory
- Query Time

CPU: Intel(R) Core(TM) i5-4200U.
Cores: 2.
Threads: 2.
Primary Memory: 8GiB DDR3 1600.
Operating System: Ubuntu 14.04 LTS.
Architecture: x64

Testing rate: 3,000 Packets/Second.
ElasticSearch stopped answering queries after 3 hours of the benchmark.
Software to analyze

Capture
PacketBeat
Collectd
Dnscache
DsC
Fleev

Storage
Prometheus
Druid
ClickHouse
ElasticSearch
InfluxDB
ClickHouse

Visualization
Kibana
Grafana
Graphite
Graphite
<table>
<thead>
<tr>
<th></th>
<th>Graphite</th>
<th>Grafana</th>
<th>Kibana</th>
<th>Prometheus</th>
<th>OpenTSDB</th>
<th>Elasticsearch</th>
<th>InfluxDB</th>
<th>ClickHouse</th>
<th>Druid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphite</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grafana</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kibana</td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prometheus</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenTSDB</td>
<td></td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
</tr>
</tbody>
</table>
Software to analyze

PacketBeat
Collectd
Fievel
DSC
gopassive dns
DnsZeppelin

Capture

Graphite
Grafana
Kibana

Visualization

Prometheus
Druid
ClickHouse
InfluxDB
Elasticsearch
OpenTSDB

Storage

PacketBeat
Collected
Fierval
gopassive dns
DSC

Capture to analyze
Resulted System
Load Simulation

- Normal Simulation:
  - Packets/Second: ~7,000 pps
  - Time running: 36 Hours
  - Packets/Second: ~7,000 pps

- Compressed packet size: ~8.3 Bytes
- Total compressed data: 7.1 GB
- Total uncompressed data: 52 GB
- Total packet count: ~927,000,000
Load Simulation

Normal Simulation:
- Packets/Second: ~7,000 qps
- Time running: 36 Hours
- Total packet count: ~927,000,000
- Total uncompressed data: 52 GB
- Total compressed data: 7.1 GB
- Compressed packet size: ~8.3 Bytes

Flood Simulation:
- Packets/Second: 120,000 qps
- Average CPU usage: 30%
- Time running: 36 Hours
- Total packet count: ~927,000,000

Normal Simulation:
Grafana Panel
Grafana Panel
Grafana Panel
SQL Interface

- Query individual DNS packet
- Show Last SERF/EI
Alerting

- Grafana Alerting
  - Define thresholds.
  - Send messages on start/end of events.
What type of attack is it?

- Typical DNS packet flood.
Random DNS Query Attack.

ISP don't have query cached.

<randomstring>.cl
Attack Example
- Packets are easier to craft.
- ISP have query cached.
- Example: cl
Limitations

- Currently it's not handling all the data in the DNS packet.
- The alert system is too simple.
- Requires small modifications to use the distributed capabilities of ClickHouse.
Conclusion

- Use open source software.
- Make our monitoring more intelligent.
- Grafana
- ClickHouse
- DnsZeppelin
- Working DNS Monitoring Solution
Questions?

Source code: https://github.com/niclabs/dnszeppelin

Javier Bustos - jbustos@niclabs.cl
Felipe Espinoza - fespin@niclabs.cl

Felipe Espinoza - fespin@niclabs.cl