DNS Monitoring tools

Hands-on Workshop with Prometheus and Grafana

Mauricio Vergara Ereche

Taller Técnico LACTLD 2019 Guatemala
Sep 2019
Intro

Why monitoring? (First the theory)
Monitoring and Analyzing DNS Servers

What to monitor?

○ Internal Service status
  • Is the service available/responding/answering
  • How fast are we responding
  • What’s the server capacity
  • More complex questions
    Client characterization
    Group/classify bulks of data
    Grouping set of servers into different views.
    Analyze traffic and search of patterns

○ External service status
  • Is the service available everywhere?
  • Are we giving the same answer consistently to every client?
  • Perception of service from the client side
Possible solutions (external monitoring)

- Using monitoring distributed services (à la “looking glass”)
  - RIPE: DNSMON
  - ThousandEyes: DNS Monitoring
  - Uptrends Monitoring
  - DNSChecker Propagation and Resolution tool

- DIY approach
  - RIPE Atlas
  - NLNOG Ring
  - Any cloud hosting service (build your own farm of monitors)

- OUT OF SCOPE FOR THIS PRESENTATION
Possible solutions (internal monitoring)

- Let’s do some graphs!
  - RRDtools or similar approaches.
    - Nagios, Icinga, MRTG, Cacti, Observium, Zabbix, Prometheus
  - Let’s do elastic graphs!
    - Kibana, Grafana

- What about the complex questions?
  - Analyze syslog and daemon logs
  - Command line tools
    - dnstop, tcpdump, wireshark
  - Collect traffic and then analyze
    - Capture: pcap, dnscap, dnstap, dsc
    - Analyze: packetq or your usual swiss army knife (Perl, Python, awk) with their own DNS libraries

- Build one solution for most of these requirements
What is Prometheus?

- Open-Source system Monitoring and Alerting toolkit
  - Mostly written in Go lang

- Features:
  - Time series: metric name and key/value pairs
  - **PromQL** query language
  - Time series collection happens with pull model via HTTP
  - **Push** time series supported via intermediary gateway (exporter)

- Components:
  - Prometheus server
  - Client libraries
  - Push gateway
  - Exporters
  - AlertManager
What is Grafana?

- Open-Source metric analytics and visualization suite
  - Most commonly used for visualize time-series data

- Features:
  - Web based
  - Dashboard oriented: graphs, heatmaps, histograms, etc.
  - Alarms, Plugins, Public engaged community

- Several data sources / plugins
  - Graphite
  - InfluxDB
  - Prometheus
  - OpenTSDB
  - MySQL
  - PostgreSQL
  - ClickHouse
  - ElasticSearch
Hands on!
Lab environment

- Ubuntu 18.04LTS
  - VirtualBox machine: [https://bit.ly/2m58gU2](https://bit.ly/2m58gU2)
    - If you didn’t installed it locally, I have a few over here
    - It has already installed and configured BIND, NSD and Knot
    - Pre-configured apt-get repositories to make things faster
      Check out `/etc/apt/sources.list.d`

- Login
  - User: lactld
  - Password: lactld2019
  - User has sudo password 😊

- Check if it is able to connect to internet.
  - TIP: Virtualbox connected Network as Bridged Adapter
Prometheus
Install & Usage
Install Prometheus

- [https://prometheus.io/docs/prometheus/latest/installation/](https://prometheus.io/docs/prometheus/latest/installation/)

- Ubuntu provides packages a bit outdated, so we grab another:

  curl -s https://packagecloud.io/install/repositoriess/prometheus-deb/release/script.deb.sh | sudo os=ubuntu dist=xenial bash
  
apt -y install prometheus
  
  # vim /etc/prometheus/prometheus.yml
  
sudo systemctl enable /usr/lib/systemd/system/prometheus.service
  
service prometheus start

- Web based interface: [http://${IP}:9090](http://${IP}:9090)
  - Access internals: [http://${IP}:9090/metrics](http://${IP}:9090/metrics)
Configuring Prometheus node-exporter

- node-exporter collects information from the server
  - Newer: [https://launchpad.net/ubuntu/+source/prometheus-node-exporter](https://launchpad.net/ubuntu/+source/prometheus-node-exporter)

```bash
sudo dpkg -i ~/lactld/prometheus-node-exporter_0.18.*.deb

# vi /etc/default/prometheus-node-exporter
sudo service prometheus-node-exporter restart

curl -s http://${IP}:9100/metrics | egrep network.*_bytes.*
```

- Add the new exporter to `/etc/prometheus/prometheus.yml`

```yaml
- job_name: node
  static_configs:
    - targets: ['localhost:9100']

sudo service prometheus restart
```
Prometheus Expressions & Templates

- **Expression browser:**
  - [https://prometheus.io/docs/prometheus/latest/querying/basics/](https://prometheus.io/docs/prometheus/latest/querying/basics/)
  - Visit: [http://${IP}:9090/graph](http://${IP}:9090/graph)

```plaintext
node_network_receive_bytes_total
rate(node_network_receive_bytes_total[1m])
```

- **Console Templates**
  - [https://prometheus.io/docs/prometheus/latest/configuration/template_reference/](https://prometheus.io/docs/prometheus/latest/configuration/template_reference/)

```plaintext
# /usr/share/prometheus/console_libraries

service prometheus restart
```

- Visit: [http://${IP}:9090/consoles/index.html.example](http://${IP}:9090/consoles/index.html.example)
Homework: Getting to know Grafana

- What is the metric for:
  - Disk space usage on /
  - CPU usage
  - Memory consumption

- Make a graph with those metrics

- View/edit a template.
Grafana
Install & Usage
Installing Grafana

- [https://grafana.com/docs/v4.3/installation/debian/](https://grafana.com/docs/v4.3/installation/debian/)
- [https://grafana.com/docs/v4.3/installation/rpm/](https://grafana.com/docs/v4.3/installation/rpm/)

```bash
# curl -s https://packages.grafana.com/gpg.key | sudo apt-key add -
# sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"
# sudo apt-get update
sudo apt -y install grafana

sudo systemctl daemon-reload
sudo systemctl enable grafana-server
sudo systemctl start grafana-server
```

- Visit: [http://${IP}:3000](http://${IP}:3000)
  - User: admin
  - Password: admin
Configuring Grafana sources

- Add data source
  - Prometheus
    - URL: http://localhost:9090
    - Access: Server
    - HTTP Method: POST
    - -> Save & Test
Grafana Dashboard (part 1)

- New Dashboard
- Queries
  - Query A: `rate(node_network_receive_bytes_total[1m])`
    - Legend: Traffic IN
  - Query B: `rate(node_network_transmit_bytes_total[1m])`
    - Legend: Traffic OUT
  - Query C: `rate(node_network_receive_packets_total[1m])`
    - Legend: Packets IN
  - Query D: `rate(node_network_transmit_packets_total[1m])`
    - Legend: Packets OUT
Grafana Dashboard (part 2)

- Visualization

- Draw Modes
  - Alias or regex: /.*OUT.*/
    - Transform: negative-Y
  - Alias or regex: /.*Traffic.*/
    - Y-axis: 1
  - Alias or regex: /.*Packets.*/
    - Y-axis: 2
    - Points: true

- Axes
  - Adjust unit
  - Add Labels
Importing dashboards & Plugins

- Public repository for dashboards
  - [https://grafana.com/grafana/dashboards](https://grafana.com/grafana/dashboards)

- Dashboards (check the number id)
  - Manage
    - Import -> Add ID -> Load
    - Select DataSource
    - Example: Node Exporter Monitoring (ID 10645)

- Public repository for Plugins
  - [https://grafana.com/grafana/plugins](https://grafana.com/grafana/plugins)

# [https://grafana.com/grafana/plugins/grafana-piechart-panel/installation](https://grafana.com/grafana/plugins/grafana-piechart-panel/installation)
sudo grafana-cli plugins install grafana-piechart-panel
Adding DNS Monitoring
ISC BIND
Configuring BIND statistics channel

- Enable statistics for BIND
  - Review config in `/etc/bind/named.conf`
    ```
    statistics-channels {
      inet 0.0.0.0 port 8053 allow { 127.0.0.1; };
    };
    ```

- Check `named.pid` exists at run time
  ```
  # Check BIND PID location
  service bind9 restart
  ls -l /var/run/named/named.pid
  ```

- Check the bind channel interface: `http://${IP}:8053`
  ```
  curl -s http://localhost:8053/xml/v3/status
  ```
Installing Prometheus BIND exporter

```
sudo apt install prometheus-bind-exporter
```

- **Additional configurations:**
  - Edit `/etc/default/prometheus-bind-exporter`
    ```
    ARGS='--bind.stats-url http://localhost:8053/ --bind.stats-groups "server,view,tasks" --bind.pid-file "/var/run/named/named.pid" --bind.stats-version "xml.v3" --web.listen-address ":9153"
    ```

- Restart and check if running
  ```
  sudo service prometheus-bind-exporter restart
  sudo netstat -vatpnW | grep prometheus
  ```
Add the new exporter into Prometheus

- Add config to /etc/prometheus/prometheus.yml

```
- job_name: dns
  static_configs:
    - targets: ['localhost:9153']
```

- Restart prometheus and check

```
sudo service prometheus restart
sudo netstat -vatpnW |grep prometheus
```

- You can also check Prometheus config at:
  - `http://${IP}:9090/config`
Configuration for Grafana

- Add dashboard 10024
  - Select source Prometheus

- **CHALLENGE TIME**
  - In our custom dashboard, create a Pie Chart for
    - Query distribution type
    - Show a Legend with percentages
    - Group lesser known queries (below 3%)

- **Solution:**
  - Query: bind_incoming_queries_total
  - Legend: {{type}}
  - Visualization Pie Chart, Combine Threshold: 0.03
What about other DNS Servers?
Knot, NSD ...and resolvers
Other DNS Servers

- Knot has statistics channel:
  - Prometheus exporter made by Alessandro Ghedini (Cloudflare)
  - [https://github.com/ghedo/knot_exporter](https://github.com/ghedo/knot_exporter)

- For Knot-resolver:

- Unbound and NSD:
  - [https://github.com/Jean-Daniel/dns_exporter](https://github.com/Jean-Daniel/dns_exporter)
  - [https://github.com/kumina/unbound_exporter](https://github.com/kumina/unbound_exporter)
One World, One Internet

Visit us at icann.org and dns.icann.org

@icann
facebook.com/icannorg
youtube.com/icannnews
flickr.com/icann
linkedin/company/icann
slideshare/icannpresentations
soundcloud/icann