NetGraf: A Collaborative Network Monitoring Stack for Network Experimental Testbeds

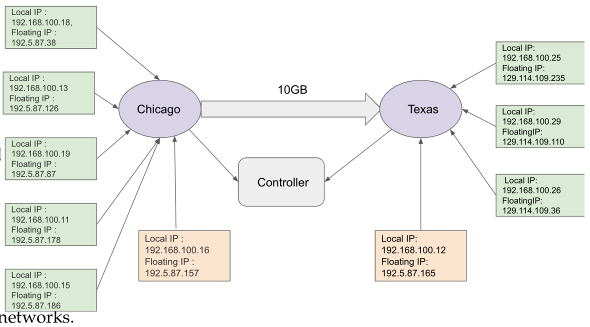
Divneet Kaur[1], Bashir Mohammed (advisor)[2], Mariam Kiran (advisor)[2]
[1]University of California San Diego, [2]Lawrence Berkeley National Laboratory

Introduction

Motivation : Network performance monitoring (NPM) is the process of visualizing, monitoring, optimizing, troubleshooting and reporting the service quality of your network as experienced by your users [1]. NPM tools collect data such as network flow data to monitor a network's performance. Commonly, many NPM tools are used to get a holistic view of the network infrastructure. However, multiple dashboards have to be used to visualize network statistics from several NPM tools.

Goal : NetGraf is a collaborative cloud network monitoring stack which collects, analyzes and aggregates relevant network measurement data and extracts relevant information which is and visualized in a single Grafana dashboard to provide a holistic view of the network system in order to obtain valuable insights in order to identify abnormal behavior in network system and improve it.

Methodology


Local IP : 192.168.100.18, Floating IP : 192.5.87.38
Local IP : 192.168.100.13 Floating IP : 192.5.87.126
Local IP : 192.168.100.19 Floating IP : 192.5.87.87
Local IP : 192.168.100.11 Floating IP : 192.5.87.178
Local IP : 192.168.100.15 Floating IP : 192.5.87.186
Local IP : 192.168.100.16 Floating IP : 192.5.87.157
Chicago
10GB
Texas
Controller
Local IP: 192.168.100.12 Floating IP: 192.5.87.165
Local IP: 192.168.100.25 Floating IP: 129.114.109.235
Local IP: 192.168.100.29 FloatingIP: 129.114.109.110
Local IP: 192.168.100.26 FloatingIP: 129.114.109.36

The architecture consists of three modules:

a) Network and Application Module : Shows network topology deployed on the Chameleon testbed

b) Collector and Aggregator Module :

Monitoring Tools :

- ***ntopng*** and ***netdata*** - installed on all nod in our network to get a global view.
- ***Prometheus*** - installed on one node as it can scrape metrics from multiple sources
- ***Zabbix*** - installed on one node as it collects server related metrics.
- ***perfSONAR*** - installed once on Texas and Chicago site as it works on end-to-end networks.

Storing Collected Data :

- **ntopng and netdata** - connected to InfluxDB, a database optimized for storing time-series data.
- **Prometheus and Zabbix** - have an inbuilt database, they were directly connected to Grafana.
- **perfSONAR** - collected results are archived in a relational database, postgreSQL.

c) Monitoring and Visualization Module : To generate visualizations from the metrics stored in our central database, we created an Application Programming Interface between the databases and Grafana. This API was established by adding different databases present in Influxdb and postgreSQL as datasource in Grafana. Desirable metrics were then queried from different monitoring tools in order to get all the network performance statistics in one dashboard.

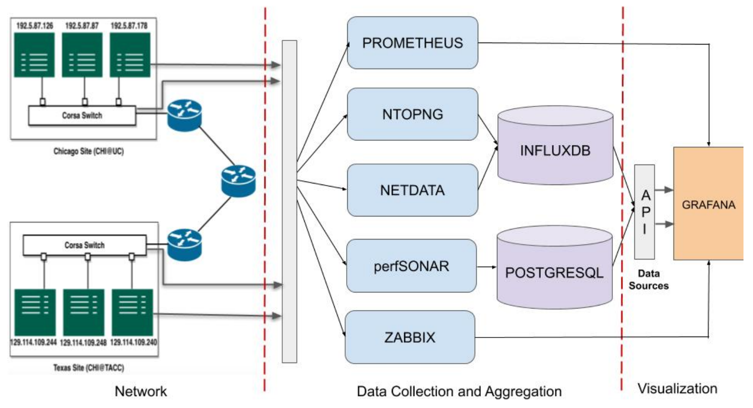
Elimination Process : Due to a large number of metrics collected, the elimination process helped us to select metrics related to network like traffic, throughput and loss. This helped create an efficient dashboard.

What did not work

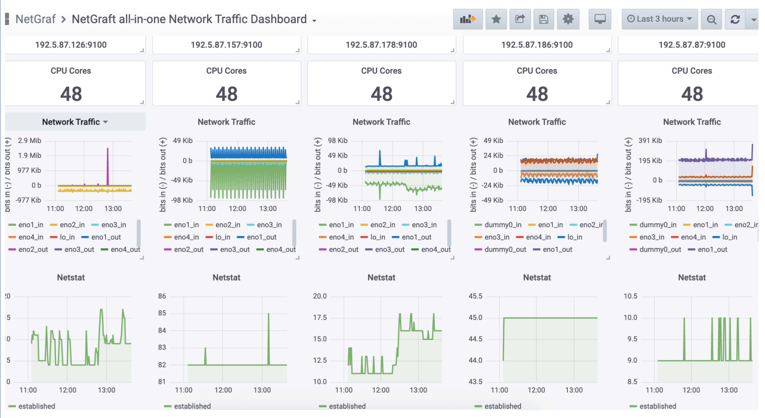
To connect the NPM tools to Grafana in order to generate visualizations we used two other approaches :-

- Connected the NPM tools to Prometheus which was in turn connected to Grafana. We received node metrics such as CPU storage which didn't fulfill our purpose of getting network data.
- We feeded the data directly to Grafana using Grafana plugins. This approach was not ideal as not all tools have direct plugins. Also, due to lack of a central database, the collected data would not be accessible in the long run.

Architecture



Results



A snapshot of network metrics data collected, aggregated and visualized in real time in Grafana by five nodes from Chameleon testbed located at Chicago

Dashboard

Try me!

Username : viewer

Password : viewer

NetGraf all-in-one Network Traffic Dashboard

Conclusion and Further Work

We have presented a unique monitoring approach which is able to collect, store, monitor and identify a networks performance by solving the heterogeneity of diverse network monitoring tools mainly in terms of resource relationship and sub-system levels and visualizing them all in a single dashboard. We created 2 users - admin and viewer to allow many people to view dashboard. In future works, we will develop a pipeline and apply machine learning algorithms to the data collected to give us more insights in terms of network performance and availability analysis

Acknowledgments

We would like to thank Paul Ruth of the Chameleon project for his technical support in setting up the environment.

References

[1] S. Narayana, A. Sivaraman, V. Nathan, P. Goyal, V. Arun, M. Alizadeh, V. Jeyakumar, and C. Kim, "Language-directed hardware design for network performance monitoring," in Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 85–98, 2017