Systems Development Engineer PSE

Take-Home Assessment

Introduction

For this assessment, you will create a containerized service that provides network telemetry and analytics to a public endpoint of your choosing. The goal is to help diagnose and debug any issues over a public path between one address and the Fully Qualified Domain Name (FQDN) of your choosing. The destination should be an environment variable that is included in docker-compose file such that support teams would be able to modify that value then deploy this solution to help diagnose network latency, routing, or availability issues along the path those packets are being routed.

The outcome should include a dashboard, such as Grafana, displaying metrics from a data source like InfluxDB. This dashboard should include the route that packets are taking from the source to the destination to help diagnose any connectivity or latency issues.

Here are some things to think about before you get started:

- **Documentation**: Provide clear documentation explaining your system, decision-making process, and any other relevant information.
- **Quality**: Develop your service as if it were to be adopted by our team and put into production. Include tests or a way to validate that your service works properly, along with logging.
- **Tech Stack**: Please feel free to use the languages and tools you are most comfortable with. Our languages of choice are typically Python, Bash and JavaScript.
- **Timeline**: You have 72 hours to complete the project and submit your code. *Please submit your code at least 4 hours before your review.*

Requirements

- **Network Telemetry Collection**: Develop a script, service, or application that collects network telemetry data from the variablized source to the FQDN of your choosing. Capture metrics such as latency, packet loss, and hop count.
- **Data Storage**: Store the collected telemetry data in a database of your choosing. If you opt for a database service, like Mongo or SQL, please include a connector or means of getting that data into a time series database, like InfluxDB.

- **Data Visualization**: Set up a Grafana dashboard to visualize the telemetry data. Ensure the dashboard includes visualizations for latency, packet loss, and hop count over time.

- **Authentication**: Ensure your application includes necessary logic to authenticate requests to the database instance.

- **Docker Container**: Package your solution in a Docker container. Add your solution as a new service to a provided docker-compose configuration. This should allow us to bring up your service along with the InfluxDB and Grafana services using a single docker-compose up command. Include necessary scripts for building and running your service.

- **Review**: Be prepared to walk through your implementation and potentially extend it.

- **Submission**: Please submit your response via email to mattreid@netflix.com.

Bonus Points

- Consider scalability and build in features that could aid in scaling the solution. *Think about concurrency and high availability*.
- Implement two-factor authentication (2FA) by connecting it to an identity provider (IdP) of your choice, such as SAML or OAuth
- Integrate geolocation data to map the network flow, illustrating the connections between source and destination

Setup

InfluxDB & Grafana

To help you start your local InfluxDB and Grafana instance containers, we have provided a sample docker-compose.yml file below. Please feel free to use this configuration and add your implementation as an additional service(s) to this config.

```
None
# docker config for influxDB and grafana
services:
  influxdb2:
    image: influxdb:2
    container_name: influxdb2
    ports:
      - "8086:8086"
    volumes:
      - type: bind
        source: ./influxdb2_data
```

```
        target: /var/lib/influxdb2
      - type: bind
        source: ./influxdb2_config
        target: /etc/influxdb2
    environment:
      - DOCKER_INFLUXDB_INIT_MODE=setup
      - DOCKER_INFLUXDB_INIT_USERNAME=admin
      - DOCKER_INFLUXDB_INIT_PASSWORD=admin123!
      - DOCKER_INFLUXDB_INIT_ORG=nflx
      - DOCKER_INFLUXDB_INIT_BUCKET=default

  grafana:
    image: grafana/grafana
    container_name: grafana
    ports:
      - "3000:3000"
    depends_on:
      - influxdb2
```

Once the influx service has launched, `influxdb2_config/influx-configs` will contain the token for the admin user. You can use this token to read and write to influxDB.

Closing Thoughts

We want to see if you can build a simple but non-trivial system. We want to talk through the approach you've taken as a means to better understand you, your experience, and to use this example as a jumping off point into subject areas that might be relevant to this position.

We appreciate you taking the time to complete this assessment and will provide as much feedback on your submission as you'd like.

Lastly, **have fun with this project**! We recognize that interview-related tasks can be stressful, so we aim to make this as enjoyable as possible. Please do not hesitate to let us know how you felt about this process either before, during, and/or after this process.