

Nestor Saavedra

3-20-2023

Observing Cloud Resources

SRE Assessment Template

Categorize Responsibilities

Prometheus and Grafana Screenshots

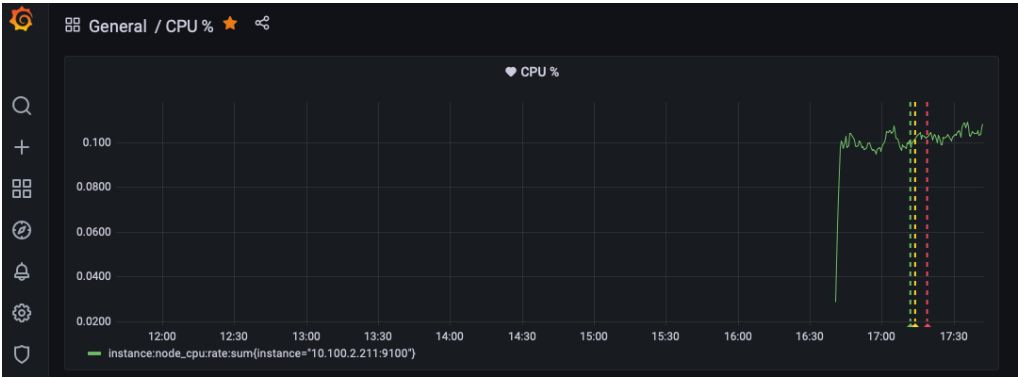
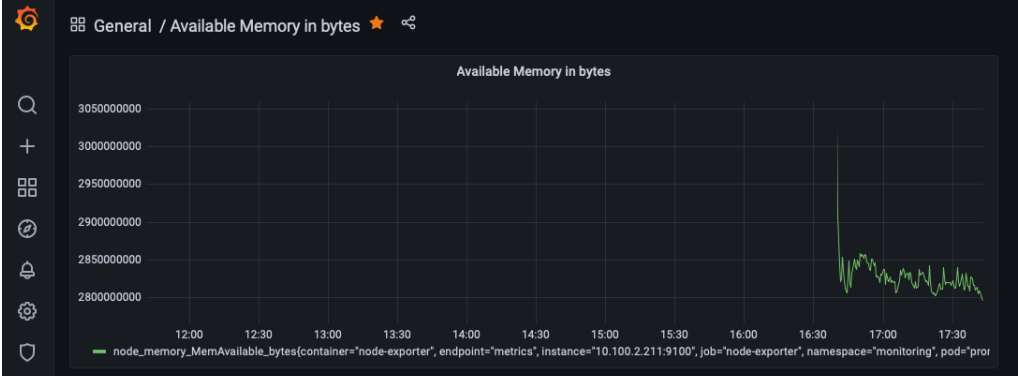

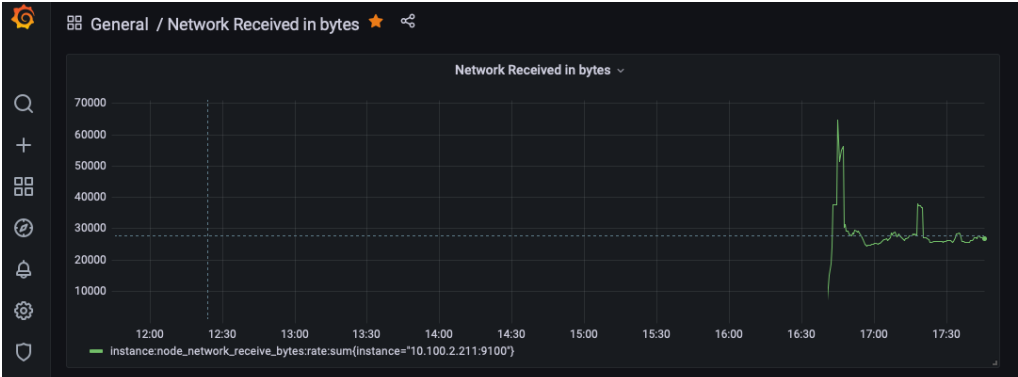
Provide a screenshot of the Prometheus node_exporter service running on the EC2 instance. Use the following command to show that the system is running: `sudo systemctl status node_exporter`

```
ubuntu@ip-172-31-47-137:~$ sudo systemctl status node_exporter
● node_exporter.service - Node Exporter
   Loaded: loaded (/etc/systemd/system/node_exporter.service; enabled; vendor pr
   Active: active (running) since Mon 2023-03-20 21:50:04 UTC; 2h 45min ago
     Main PID: 784 (node_exporter)
        Tasks: 4 (limit: 1104)
       CGroup: /system.slice/node_exporter.service
              └─784 /usr/local/bin/node_exporter

Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
Mar 20 21:50:04 ip-172-31-47-137 node_exporter[784]: level=info ts=2023-03-20T21
```

Host Metric
(CPU, RAM, Disk,
Network)

Dashboard

<p><i>CPU %</i></p>	 <p>General / CPU %</p> <p>0.100 0.0800 0.0600 0.0400 0.0200</p> <p>12:00 12:30 13:00 13:30 14:00 14:30 15:00 15:30 16:00 16:30 17:00 17:30</p> <p>instance:node_cpu:rate:sum(instance="10.100.2.211:9100")</p>
<p><i>Available memory in bytes</i></p>	 <p>General / Available Memory in bytes</p> <p>3050000000 3000000000 2950000000 2900000000 2850000000 2800000000</p> <p>12:00 12:30 13:00 13:30 14:00 14:30 15:00 15:30 16:00 16:30 17:00 17:30</p> <p>node_memory_MemAvailable_bytes(container="node-exporter", endpoint="metrics", instance="10.100.2.211:9100", job="node-exporter", namespace="monitoring", pod="pror")</p>
<p><i>Disk I/O</i></p>	 <p>General / Disk I/O</p> <p>100 80 60 40 20 0</p> <p>12:00 12:30 13:00 13:30 14:00 14:30 15:00 15:30 16:00 16:30 17:00 17:30</p> <p>node_disk_io_now(container="node-exporter", device="nvme0n1", endpoint="metrics", instance="10.100.2.211:9100", job="node-exporter", namespace="monitoring", pod="pror")</p>
<p><i>Network received in bytes</i></p>	 <p>General / Network Received in bytes</p> <p>70000 60000 50000 40000 30000 20000 10000</p> <p>12:00 12:30 13:00 13:30 14:00 14:30 15:00 15:30 16:00 16:30 17:00 17:30</p> <p>instance:node_network_receive_bytes:rate:sum(instance="10.100.2.211:9100")</p>

Responsibilities

1. The development team wants to release an emergency hotfix to production. Identify two roles of the SRE team who would be involved in this and why.

- **Monitoring Engineer:** The Monitoring Engineer is a key player in ensuring that an emergency hotfix is successfully deployed to production. They would oversee creating a thorough dashboard that precisely displays the functionality and stability of the deployed application. The Monitoring Engineer can swiftly identify any issues that crop up throughout the deployment process by continuously monitoring the system. By acting swiftly to address these issues, the Monitoring Engineer can guarantee a smooth and trouble-free rollout of the hotfix.
- **Release Manager:** The Release Manager is another crucial SRE team member who would be involved in the deployment of an emergency hotfix. Their main duty would be to discover and evaluate the risks connected to the new hotfix and choose the best course of action for its secure and safe deployment. This would entail working together with other developers, carrying out exhaustive testing and validation procedures, and making any required adjustments to the production environment. The Release Manager may ensure that the hotfix is effectively released and that any possible issues are immediately found and fixed by carefully supervising the release process.

2. The development team is in the early stages of planning to build a new product. Identify two roles of the SRE team that should be invited to the meeting and why.

- **SRE Team Lead:** The SRE Team Lead ought to be invited to the new product planning meeting since they can offer insightful opinions and suggestions regarding the IT infrastructure required to support the project. In order to ensure that all stakeholders are in agreement with the project's goals, they can also suggest techniques for incorporating each team member's function into the project. The IT infrastructure needs to be scalable, dependable, and secure for the project to succeed, and the SRE Team Lead can help make sure that the development team is aware of these needs.
- **SRE System Architect:** The SRE System Architect is a crucial role that needs to be discussed throughout the planning meeting. They can offer suggestions and direction on the IT setup and tools required to support the new product. The technological stack chosen for the project can be made to be reliable, secure, and scalable by utilizing their experience and knowledge of best practices, allowing the product to fulfill the intended performance and functionality requirements. Moreover, the SRE System Architect can assist in early infrastructure discovery of potential bottlenecks or obstacles, assisting in preventing delays and unneeded costs in the future.

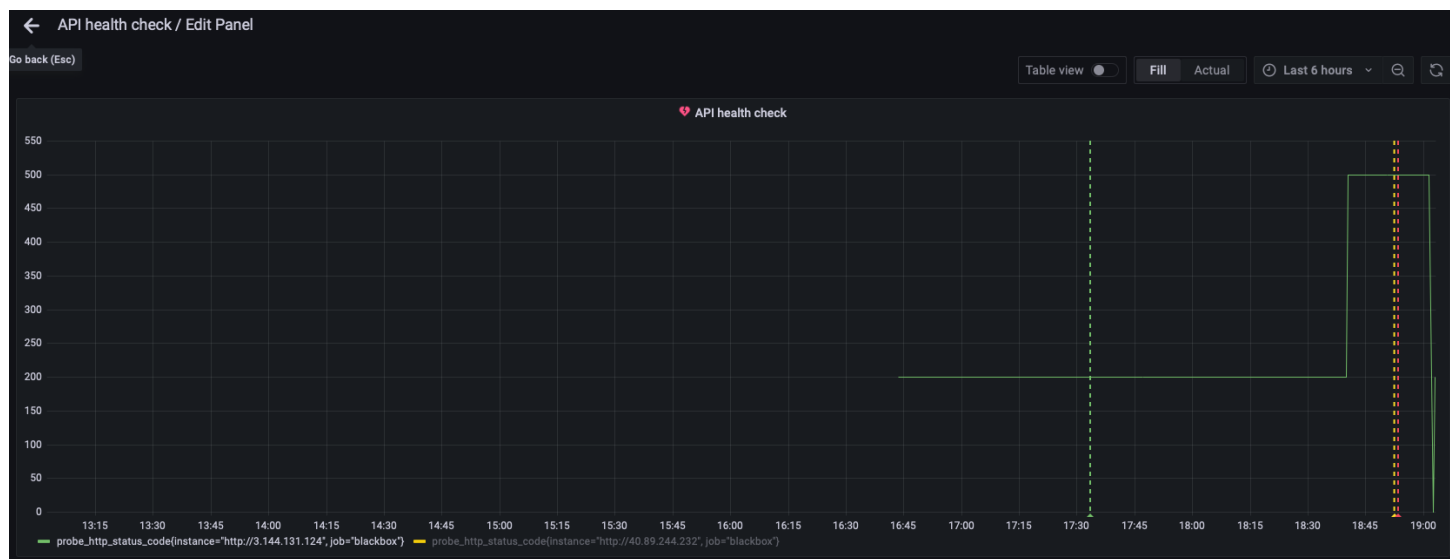
3. The emergency hotfix from question 1 was applied and is causing major issues in production. Which SRE role would primarily be involved in mitigating these issues?

- **The Release Manager** would be primarily liable for addressing these difficulties if an emergency hotfix results in significant problems in production. The Release Manager oversees ensuring that the deployment and rollback procedures are well-defined, tested, and carried out correctly because they are the gatekeepers of the release management lifecycle. They would collaborate closely with the development team to pinpoint the underlying causes of any problems, start the necessary corrective action, and handle stakeholder communication, including with top management and clients.
- The Release Manager would also make sure that the incident is completely documented and that any lessons gained are used to the planning and execution of upcoming releases. By using their knowledge and experience, the Release Manager can lessen the effects of any production-related problems and guarantee that the system is rapidly and effectively brought back up and running.

Team Formation and Workflow Identification

API Monitoring and Notifications

Display the status of an API endpoint: Provide a screenshot of the Grafana dashboard that will show at which point the API is unhealthy (non-200 HTTP code), and when it becomes healthy again (200 HTTP code).




Create a notification channel: Provide a screenshot of the Grafana notification which shows the summary of the issue and when it occurred.

The screenshot shows a Grafana notification channel titled "# grafana-sre-alert-notifications". It displays a list of incoming webhooks. The first two alerts are from the "incoming-webhook" app and are labeled "[Alerting] CPU % alert" and "[Alerting] API health check alert". The third alert is labeled "[OK] API health check alert". Each alert includes a summary of the issue, the time it occurred, and the Grafana version (v8.1.2).

- [Alerting] CPU % alert**
CPU is above 0.078
instance:node_cpu:rate:sum{instance="10.100.2.211:9100"}
0.1029454545454545
Grafana v8.1.2 Today at 5:19 PM
- [Alerting] API health check alert**
probe_http_status_code(instance="http://3.144.131.124", job="blackbox")
500
Grafana v8.1.2 Today at 6:53 PM
- [OK] API health check alert**
Grafana v8.1.2 Today at 7:07 PM

Configure alert rules: Provide a screenshot of the alert rules list in Grafana.

List of alert rules:

**Alerting**
Alert rules and notifications

Alert rules


Notification channels

Search alerts

States

All


How to add an alert



API health check alert
OK for 22 minutes

Pause

Edit alert

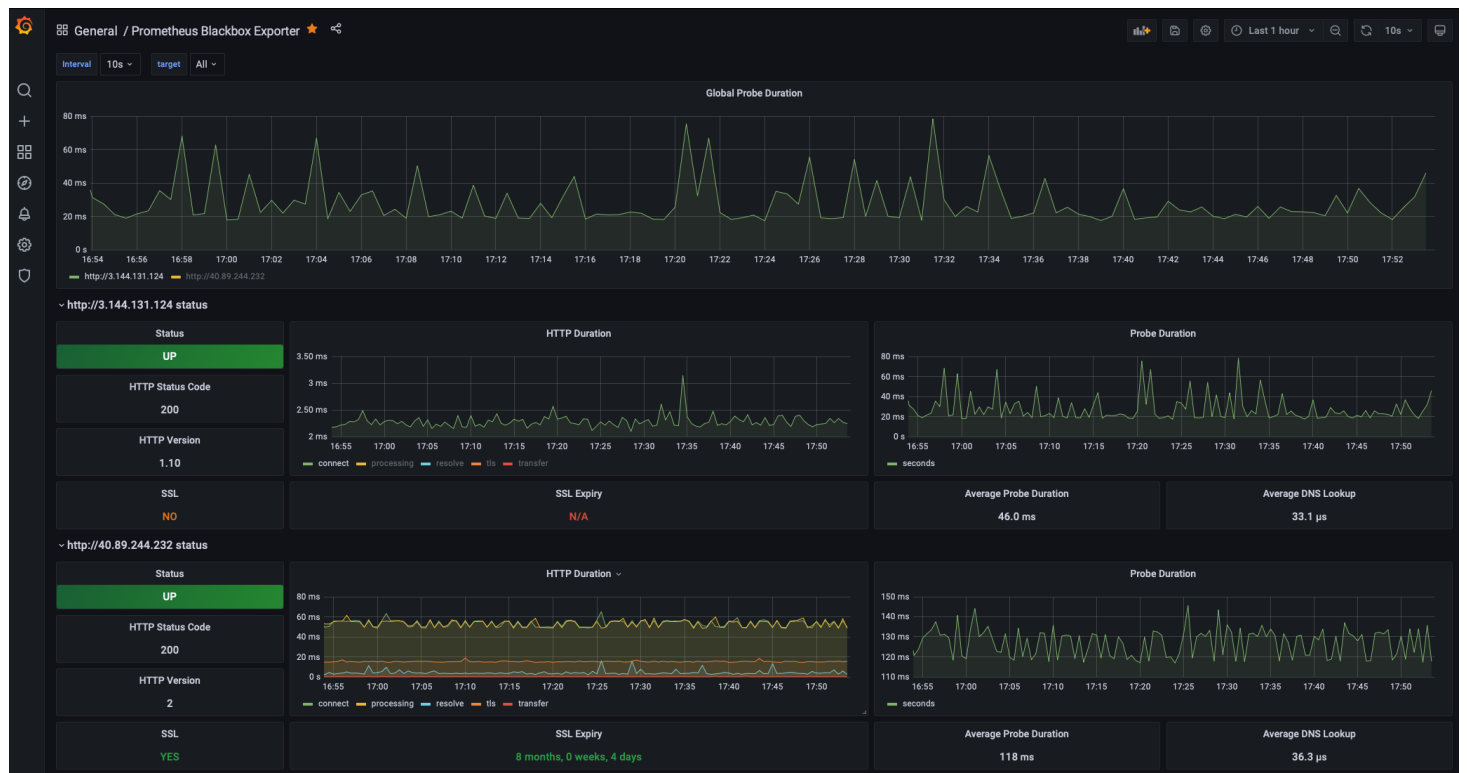


CPU % alert
PAUSED for 36 minutes

Resume

Edit alert

Blackbox export dashboard:



CPU Dashboard with alert:



Slack CPU Alert:

incoming-webhook APP 5:19 PM

[Alerting] CPU % alert

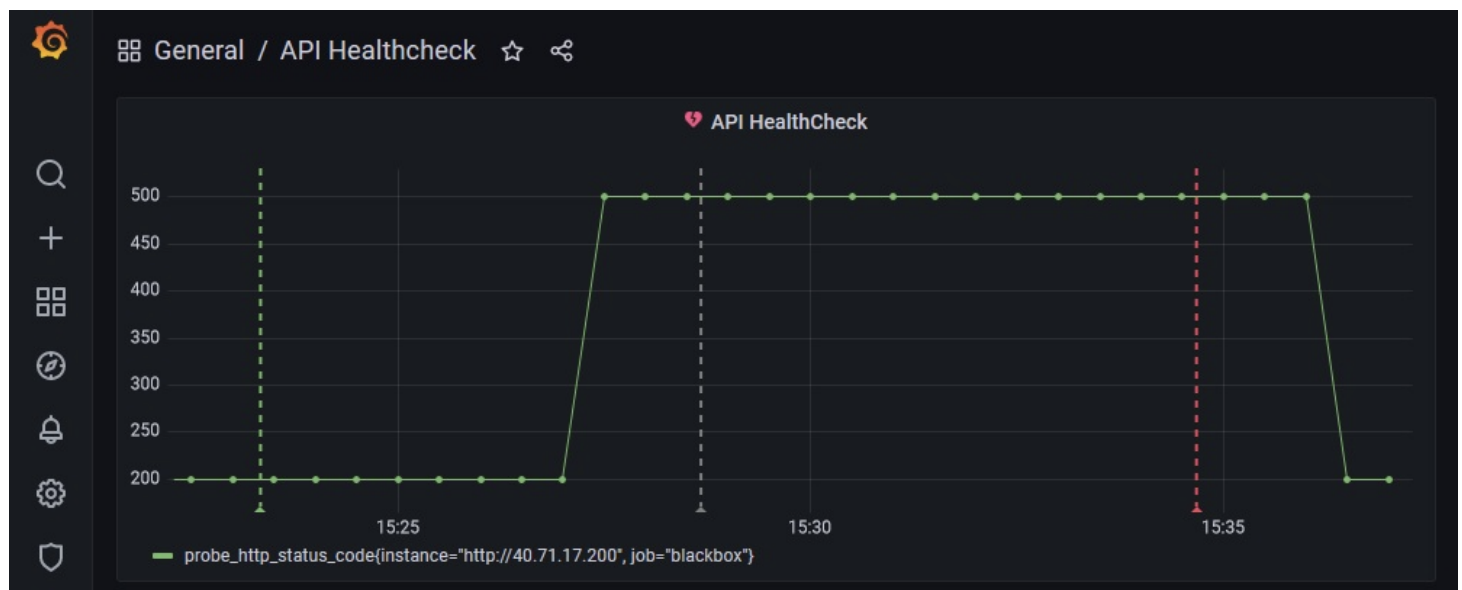
CPU is above 0.078

instance:node_cpu:rate:sum(instance="10.100.2.211:9100")

0.10294545454545

Grafana v8.1.2 Today at 5:19 PM

Graph 1



4a. Given the above graph, where does it show that the API endpoint is down? Where on the graph does this show that the API is healthy again?

- The presented graph makes it evident that there is an outage at the API endpoint from 15:27 to 15:36. The data points on the graph, which demonstrate that the API endpoint is unavailable currently, make this clear. A pending alert from the API HealthCheck at 15:28 is also present and is consistent with the outage. The fact that the alert is still active at 15:35 shows that the problem is still there at that time.
- From the graph, it shows that the API is healthy again at **15:36**.

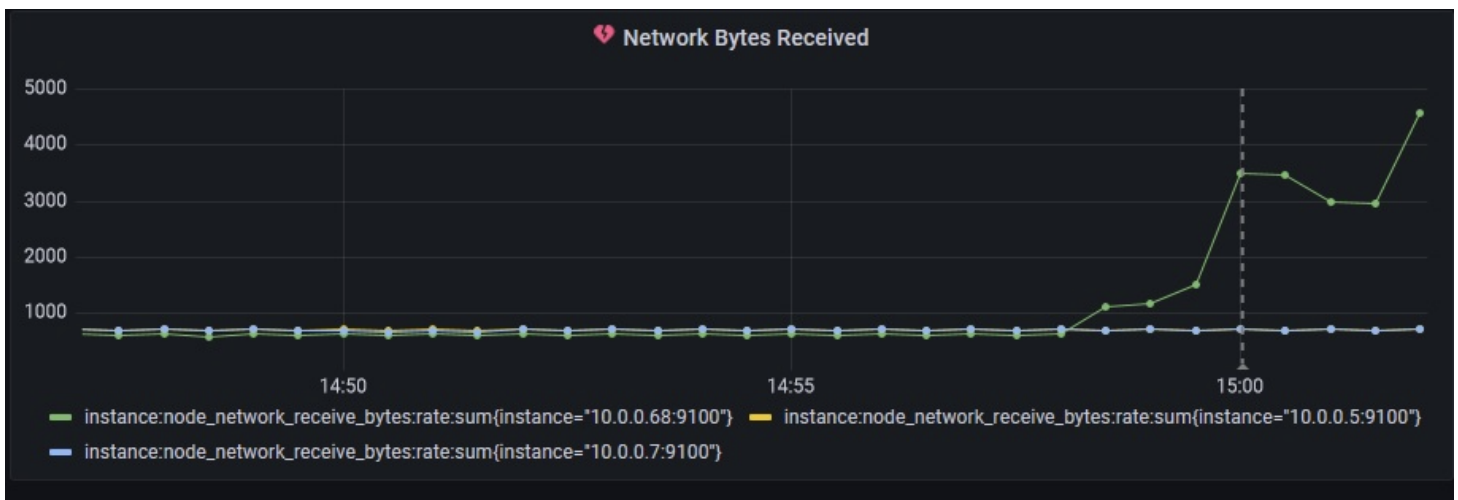
4b. If there was no SRE team, how would this outage affect customers?

- Without an SRE team to keep an eye on and maintain the application, any outages or service problems would have a big impact on the users. Without an SRE team, no one would be available to quickly identify and address the issue, which could result in a lengthy period of service unavailability. The poor user experience would irritate the customers, who might need to get in touch with the development team to report the problem. Yet, it can take longer to fix the issue without an SRE team, which might upset customers even more. Without an SRE team, the application's overall performance and dependability would likely deteriorate, which would lower customer satisfaction and could result in long-term damage to the business.

4c. What could be put in place so that the SRE team could know of the outage before the customer does?

- There are numerous measures that might be implemented to guarantee that the SRE team is informed about disruptions before customers. Creating a dashboard with host metrics (such CPU%, Memory, Network I/O, and Disk I/O) and the health check API endpoint is one practical option. The SRE team can identify any irregularities or problems that could potentially result in an outage by monitoring these indicators in real-time before consumers are harmed. This strategy minimizes the impact on end users by enabling the SRE team to act quickly to address the issue before it has a chance to worsen. Additionally, by taking a proactive approach, the SRE team can offer consumers a more dependable and responsive service, enhancing their overall pleasure and trust.

Graph 2



5a. Given the above graph, which instance had the increase in traffic, and approximately how many bytes did it receive (feel free to round)?

- It is clear from the supplied graph that traffic to the instance with IP address 10.0.0.68 at port 9100 significantly increased. More specifically, this instance received about 3500 bytes at around 15:00. Following this, the instance experienced a rise in the number of bytes received, indicating a continuous increase in traffic. Although the graph makes it impossible to establish the precise number of bytes received, it is obvious that there was a noticeable and continuous increase, which might have been caused by an increase in demand or other external factors affecting the traffic flow.

5b. Which team members on the SRE team would be interested in this graph and why?

The presented graph would be of interest to several SRE team members, each for a different reason:

- Release Manager:** The Release Manager would be curious about this graph because they could compare it to recent releases to see whether any changes to the infrastructure or deployment procedures are to blame for the reported rise in traffic. To ensure the service stays dependable, they would also need to evaluate how the increased traffic will affect the service level objectives (SLOs).
- Cloud Architect:** The Cloud Architect would be interested in the graph as they could use the data to make informed decisions about modifying the infrastructure. For example, they may decide to add a load balancer to distribute traffic between nodes, thereby reducing the load on any one instance. By implementing this change, they could improve the scalability and reliability of the system.

- Team Leader: The graph would be of interest to the team leader since they could work with the development team to resolve the underlying problem that is causing one node to receive all the traffic. To make the system more resistant to unanticipated traffic surges, they would investigate automation techniques to help distribute the load between various nodes. Together, they might make adjustments that would raise the system's overall effectiveness and dependability.