Load Balancing on an IoT Fog

Sebastian Correa
Christopher Gonzales

CONCEPT OF OPERATIONS

REVISION - 2
25 February 2023

Concept of Operations
for
Load Balancing on an IoT Fog

Team <21>

Approved by:

Project Leader Date

Prof. Kalafatis Date

T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	2/10/2023	Sebastian Correa		Draft Release
	2/25/2023	Sebastian Correa		Revision 2
	12/3/2023	Sebastian Correa		Revision 3
	12/4/2023	Christopher Gonzales		Revision 4

Table of Contents

List of Figures

1. Executive Summary

An IoT Fog nodes can be overloaded and inefficient due to a high number of tasks and applications coming from the device that deployed the Fog. This can be detrimental to the Fog and device's performance. To prevent this, we will use Load Balancing on an IoT Fog, a technique using algorithms in a cloud to distribute applications and tasks equally among nodes in the Fog. In our project, we'll be transferring our application between a cloud and an edge device using K3s. Our application will be a website we created that can be hosted on every worker node. We'll be using K3s because it is light-weighted and easy to use while also containing all the utilities Kubernetes has to offer. K3s can also be used in the three environments we'll be using: clouds, IoT Fogs, and edge devices. We will create the cloud using Microsoft Azure and use a Raspberry Pi 4 as our edge device. The master node in the kubernetes cluster will take in all tasks or traffic coming in from the nodes in the IoT Fog, then sort and distribute them back to the end users who activated the tasks or traffic. To accomplish this process, we will choose one of the several techniques used for Load Balancing to distribute loads evenly in the cloud. This design can then help organizations manage their devices to reduce energy consumption, increase performance efficiency, and prevent any device from being overloaded.

2. Introduction

2.1. *Background*

As technology becomes more apparent in everyday life, more devices can collect data and connect to the internet. As these devices are more common, more servers are required to handle the traffic and input requests. One solution to control this increase is to utilize IoT fog computing. Fog computing allows edge nodes, also called fog nodes, to perform computations closer to the data; rather than in the cloud.

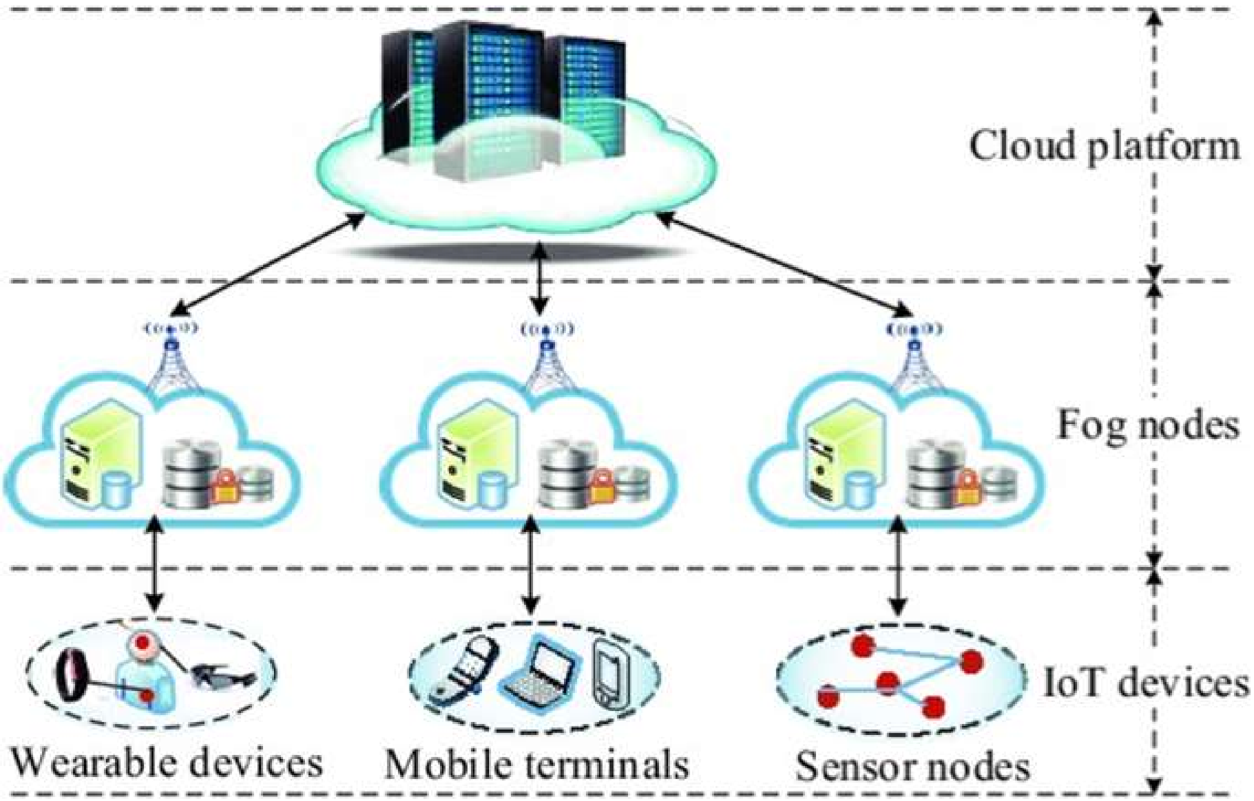

Figure 1: Computing Diagram
(https://link.springer.com/article/10.1007/s00607-022-01055-8

As seen in figure 1, when utilizing fog computing the cloud servers can work more efficiently and lower network latency. The proposition can work more efficiently because fog nodes or the worker nodes in the system are physically closer to the devices. The computations done by fog nodes are quicker than in the Cloud (https://www.fortinet.com/resources/cyberglossary/iot-edge).

Therefore, by adding a load-balancing algorithm to help distribute traffic among the worker nodes, the network can run more securely and smoothly. Overall, the system will perform better than if performing computations remotely. Comparably, the company ScaleArc has applied a load-balancing algorithm to its cloud database. The organization has been able to increase revenue by improving the reliability and performance of its database. Adding database failover, zero downtime maintenance, and instant scalability, the company has been at the forefront of the load-balancing industry (https://ignitetech.com/application/files/5916/1610/7337/IgniteTech_Solution-Overview_ScaleArc_Jan21.pdf). By applying these concepts to an IoT fog the system can run faster and smoother than solely using a load Balancing algorithm.

2.2. Overview

K3s is a highly available Kubernetes distribution. The project will use K3s because it is lightweight and easy to use while containing all the utilities Kubernetes has. K3s is designed for production workloads in remote locations or IoT appliances (https://k3s.io/). The proposed project will allow the created IoT fog to load-balance its traffic efficiently. The proposed system will apply k3s on the fog. The load balancer on an IoT fog will require worker nodes. Furthermore, a cloud and microprocessing unit will both install k3s. The cloud component running in Azure will have the master node and one worker node. Similarly, the microprocessor will be a raspberry pi component acting as a edge device but also as a worker node.

2.3. Referenced Documents and Standards

https://k3s.io/
https://www.fortinet.com/resources/cyberglossary/iot-edge
https://ignitetech.com/application/files/5916/1610/7337/IgniteTech_Solution-Overview_ScaleArc_Jan21.pdf
https://kemptechnologies.com/load-balancer/load-balancing-algorithms-techniques
https://www.sciencedirect.com/science/article/abs/pii/S0045790618305056
Article: A Fog-Cluster Based Load Balancing Technique by Prabhdeep Singh, Rajbir Kaur, Junaid Rashid, Sapna Juneja, Gaurav Dhiman, Jungeun Kim, and Mariya Ouaissa
https://link.springer.com/article/10.1007/s00607-022-01055-8

3. Operating Concept

3.1. Scope

The project involves the design and implementation of a load-balancing service. The algorithm will then be integrated within the IoT fog and tested using a website being hosted on every worker node. The simulation that will be conducted involves sending a number of requests to the website then having the load-balancing service balance the traffic being sent to the website.

3.2. Operational Description and Constraints

The project will be accomplished by creating an IoT fog with a cloud and microcontroller component. The cloud element will mainly be used for the master node for the kubernetes cluster. Also, a microprocessor will act as the worker node. More worker nodes can be used to increase the performance of the system. Next, the ingress controller distributes workloads among the nodes. The project is constrained to limited manpower as the group consists of two students. In addition, the project is limited in its budget of $200. Lastly, ports need to be configured on the edge device in order to see metrics and establish better communication between the pi and the cloud. This limitation lowers the range of efficiency of the proposed system.

3.3. System Description

The proposed system will improve the efficiency and reliability of an IoT fog.
Overall the system will include:

- **The load-balancing algorithm**
  - Load balancing on our IoT fog system is done using our custom kubernetes service and the Nginx Ingress Controller service. Our custom service uses a docker image by the Raspberry Pi to display our website and service type NodePort to open a port on every node in the kubernetes cluster. There's also a custom deployment resource to set the number of pods to be distributed evenly among worker nodes. When using the node's IP and assigned port as a URL, it takes the user to our website. The ingress controller is how our system load balances. The controller exposes every worker node's IP address and communicates with our custom service so whenever an end user accesses a node's IP, they'll be directed to our website inside a pod in one of our two worker nodes.

- (https://www.weave.works/blog/nginx-ingress-controller-101)

- **Data monitoring system**

  The system can be monitored using Kubernetes Dashboard, another service in our kubernetes cluster. The dashboard shows metrics such as memory, workloads, and status for every node in the cluster. Pods taking in requests will have a custom message in their logs stating a request was received. The dashboard can only be accessed by a local computer connected to the common network the Pi and Cloud are connected to. To log into the dashboard the user needs a token for a service account with admin access. If an admin user would like to see more details, they can access any node in the cluster and use k3s commands to see more information about every node.

- (https://kubernetes.io/docs/tasks/access-application-cluster/web-ui-dashboard/)

- **IoT fog network integration**

- Cloud component

  The cloud component acts as an overall view of our Load Balancing system. Data and applications can be transferred here to be analyzed and given to admin users if requested. While the data is in the cloud, we can use K3s to manage and update data and tasks along with the balancer if needed.

- Microprocessing Component

  The raspberry pi will act as the edge node. It will handle all of the edge computing from IoT devices. It will install K3s on the RPI to ensure the load balancer is running on the microprocessor and can connect with the cloud.

Finally, when both components are complete they will be connected to communicate and determine how the system can run most efficiently. The algorithm will take into consideration the cloud and edge nodes.

The use of k3s on the RPi edge nodes can provide several benefits, including increased reliability, scalability, and management capabilities. It can also simplify the integration of the load-balancing solution with the existing network architecture, making it easier to manage and monitor the network.

- Improved Reliability: k3s provides built-in high availability features, such as automatic failover, that can help improve the reliability of the system. In the event of a failure, k3s can automatically reroute workloads to other nodes in the network, ensuring that the system remains operational.

- Enhanced Scalability: k3s can automatically scale the number of pods in the network as needed, making it easier to accommodate increased workloads and ensure optimal performance. This can be especially important for the RPi edge nodes, which have limited resources.

- Improved Management: k3s provides a centralized management system that makes it easier to manage and monitor the RPi edge nodes. This can simplify the administration of the network and make it easier to identify and resolve issues.

- Simplified Network Integration: k3s provides a standardized platform for managing workloads on the RPi edge nodes. This can simplify the integration of the load-balancing solution with the existing network architecture by providing a common interface for managing the RPi devices and accessing data and resources stored on them.

The Raspberry Pi will also include docker software to containerize the demo application. It will package all of the necessary files to build the website and deploy it to end users when applied to the system.

3.4. Modes of Operations

Normal Mode: The load balancing system actively distributes incoming data and requests to the available fog nodes based on real-time performance metrics and resource utilization.

Maintenance Mode: During maintenance or upgrades, the system may enter this mode to temporarily redirect traffic to other nodes, or take specific fog nodes offline for maintenance while still functioning.

Emergency Mode: In emergencies where fog nodes are facing high levels of traffic or stress, the load-balancing system enters this mode to dynamically allocate additional resources and redistribute incoming data and requests to ensure that necessary services are maintained.

Disaster Recovery Mode: In the event of any disaster or system failure, the load-balancing system enters this mode to redirect traffic to working nodes and ensure that critical services function while the primary nodes are being repaired or restored.

3.5. Users

Consumers: consumers can use load-balancing algorithms in their IoT fog systems to improve the performance of their connected devices.

Companies: organizations can use load-balancing algorithms in their IoT fog systems to improve the performance and reliability of their IoT solutions, such as connected devices and sensor networks.

Government Agencies: can use load-balancing algorithms to support critical infrastructure, such as in smart cities and public safety networks.

Research institutions: these institutions can use load-balancing algorithms in their IoT fog systems to support scientific experiments and data collection.

3.6. Support

Care for the load balancing system on an IoT fog will be provided in edge node support. Meaning maintenance and necessary fixes to the microcontrollers will be available. In addition, a detailed user manual describing the setup, at-home maintenance, and usage. The manual will also explain the user interface and how to use it.

4. Scenario(s)

4.1. Traffic Management

Sometimes IoT Fog deployments take in heavy amounts of traffic (data requests, commands, communication) that is coming from many devices. We can use Load Balancing to distribute the traffic evenly among all the devices or to any available sources. This can help prevent overloading on IoT devices and increase the efficiency of performing IoT Fog deployments.

4.2. Fault Tolerance

If one device were to fail in a system, we could use Load Balancing to redirect the traffic that was flowing into the failed device to other available devices in the same system. Our technique of using different algorithms will make sure the system will function as efficiently as it did before a device failed. We can also use the same process when devices need maintenance while the system is operating.

4.3. Energy Efficiency

While Load Balancing can distribute data and tasks evenly among devices, it can also save energy for each device while completing this process. The load balancer can monitor every device and make sure a device is limiting its traffic intake to preserve its energy to make it more efficient. Benefits of saving energy include better battery life in devices, reduced energy costs, and making systems environmentally friendly.

5. Analysis

5.1. Summary of Proposed Improvements

- Load testing can be done before the loads go into the kubernetes cluster to find any performance issues and we can work on these problems before the applications and data are distributed evenly to nodes in the IoT Fog.
- Include algorithms that adjust to changes in traffic flow so the system still functions without taking a break.
- We could monitor using Grafana and Prometheus services to see more metrics for all the nodes in the kubernetes cluster

5.2. Disadvantages and Limitations

- Wi-Fi connections to the Pi may disrupt the connection to the cloud, preventing the ingress controller from sending loads to the Raspberry Pi.
- During the building process, we found that some services on the kubernetes cluster needed specific ports to be opened on the Raspberry Pi. Meaning

without access to the router the Pi is connected to, it would be impossible to port forward for the Pi unless you use a paid service such as Ngrok.

- Include algorithms that adjust to changes in traffic flow so the system still functions without taking a break.

5.3. Alternatives

- You can connect the cloud and edge device directly to reduce loads.
- To speed the process of data quickly, you can utilize edge computing to process loads into their destination without the use of an IoT Fog.
- Using API Gateways can let applications redirect and distribute the traffic on their own.

5.4. Impact

- Load Balancing involves the transferring of data and that data can be very sensitive or private so when accomplishing the task at hand, it's important to protect the clusters of data to follow any laws and regulations.
- With the right technique, Load Balancing on an IoT Fog reduces energy and power consumption among devices.
- Some of the resources that a Load Balancer could be transferring is network bandwidth. This can affect other systems in the local area so it's important for our algorithms to adjust for such scenarios and not affect other devices in the same vicinity.