1. Installation von Prometheus

Zunächst wurde Prometheus aus dem offiziellen Release-Repository heruntergeladen und entpackt.

```
wget https://github.com/prometheus/release/download/v3.2.1/prometheus-3.2.1.linux-amd64.tar.gz
tar xvf prometheus-3.2.1.linux-amd64.tar.gz
```

2. Erstellung der Docker-Compose Datei

Für eine bessere Organisation wurde eine docker-compose.yml-Datei erstellt und als docker-compose-Prometheus.yml gespeichert.

3. Installation des Node Exporters

Der Node Exporter wurde sowohl auf dem Load Balancer als auch auf den anderen Raspberry Pi's installiert.

Die zugehörige docker-compose.yml für den Node Exporter wurde unter dem Namen docker-compose-exporter.yml gespeichert.

4. Einrichtung von Grafana und Start der Container

Nach der Installation von Grafana wurden alle Vorbereitungen abgeschlossen, und die Container wurden gestartet:

```
docker-compose up -d
```

5. Anpassung der Exporter-Konfiguration

Nach ein wenig Recherche wurde entschieden, die exporter.yml durch ein Docker Swarm-Setup zu ersetzen. Damit ließ sich eine flüssige Eingliederung der einzelnen Nodes sicherstellen.

6. Erstellen des Grafana Dashboards

Wir haben Grafana über den Port 3000 aufgerufen und ein Dashboard, mit folgenden Panels: CPU-Auslastung, Netzwerk Auslastung, RAM-Auslastung, Anzahl der Laufenden Prozesse, Temperatur und Up-Time. Diese Diese wurden in 3 Tabs unterteilt, CPU-Auslastung, Netzwerk Auslastung und RAM-Auslastung sind unter dem Tab

„Auslastung“ sortiert, Laufzeit und Anzahl der Laufenden Prozesse ist unter „Laufzeit“ und Temperatur ist unter „Temperatur“ zu finden.

Die einzelnen Panels nutzen verschiedene Query’s zur abfrage bei Prometheus. Dabei wurden die Namen der Instanzen von den IP-Adressen zu den Namen der Raspberry Pi’s geändert.

Das Panel CPU-Auslastung zeigt 3 Halbrunde Grafiken, welche die momentane Belastung der CPU in Prozent anzeigt.

```
100* (1- avg(rate(node_cpu_seconds_total{mode="idle"}[5m])) by (instance))
```

- Das „node_cpu_seconds_total{mode=„idle“} misst die Zeit, in der die CPU im Leerlauf ist über den Zeitraum von5 Minuten.
- „rate([5m])“ misst wie sich dieser Wert in den Letzten 5 Minuten verändert hat.
- „avg () by (instance)“ ermittelt den Durchschnitt für die jeweiligen Instanzen.
- „1- avg“ gibt den wert aus der nicht im Leerlauf verbracht wurde.
- „100*“ wandelt es in Prozent um.

Auf der rechten Seite findet man die Einstellung, um die Einheit in Prozent zu zeigen, Den Namen des Panels, Die Schwellwerte, welche die Farben angeben und die Overrides, welche die Instanzen benennen.

Für Panel wie Laufzeit, kann man die Grafen nicht miteinander vergleichen, da die Felder für jede Instanz ein eigenes Minimum and Maximum haben. Deswegen wenn eine Instanz bei 100 kB/s ist, wird es auf ähnlicher höhe sein, wie eine Instanz die bei 50 kB/s ist.