GLACIATION

Green responsibLe privACy preservIng dAta operaTIONs

Deliverable D5.1 – Glaciation Power Measurement Framework Software

GRANT AGREEMENT NUMBER: 101070141

GLACIATION

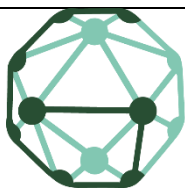|	
Project acronym:	GLACIATION
Project full title:	Green Responsible Privacy Preserving Data Operations
Call identifier:	HORIZON-CL4-2021-DATA-01-01
Type of action:	RIA
Start date:	01/05/2022
End date:	30/11/2025
Grant agreement no:	101070141

D5.1 – Glaciation Power Measurement Framework Software			
Executive Summary:	This document reports on the work done in Work Package 5 "Energy Management Techniques" which was carried out from month 7 to month 14 and is aligned to T5.1 "Performance Measurement Framework"		
WP:	WP5		
Author(s):	Cem Tanrikut, Aidan OMahony, Pournima Sonawane, Paolo Tartara, Raja Appuswamy, Flavio Scaccia, Maciej Besta		
Editor:	Cem Tanrikut		
Leading Partner:	HIRO-MICRODATACENTERS B.V.		
Participating Partners:	DELL, MEF, HIRO, SOGEI, ECOM, ETH, ENG		
Version:	1.0	**Status:**	Final
Deliverable Type:	Other	**Dissemination Level:**	PU - Public
Official Submission Date:	30/11/2023	**Actual Submission Date:**	30/11/2023

Disclaimer

This document contains material, which is the copyright of certain GLACIATION contractors, and may not be reproduced or copied without permission. All GLACIATION consortium partners have agreed to the full publication of this document if not declared "Confidential". The commercial use of any information contained in this document may require a license from the proprietor of that information. The reproduction of this document or of parts of it requires an agreement with the proprietor of that information.

The GLACIATION consortium consists of the following partners:

No.	Partner Organisation Name	Partner Organisation Short Name	Country
1	MINISTERO DELL'ECONOMIA E DELLE FINANZE	MEF	IT
2	EMC INFORMATION SYSTEMS INTERNATIONAL UNLIMITED COMPANY	EISI	IE
3	HIRO MICRODATACENTERS B.V.	HIRO	NL
4	GOTTFRIED WILHELM LEIBNIZ UNIVERSITAET HANNOVER	LUH	DE
5	THE LISBON COUNCIL FOR ECONOMIC COMPETITIVENESS ASBL	LC	BE
6	UNIVERSITA DEGLI STUDI DI MILANO	UNIMI	IT
7	UNIVERSITA DEGLI STUDI DI BERGAMO	UNIBG	IT
8	GEIE ERCIM	ERCIM	FR
9	EURECOM	EURECOM	FR
10	SAP SE	SAP SE	DE
11	UNIVERSITY COLLEGE CORK - NATIONAL UNIVERSITY OF IRELAND, CORK	UCC	IE
12	SOGEI-SOCIETA GENERALE D'INFORMATICA SPA	SOGEI	IT
13	LAKESIDE LABS GMBH	LAKE	AT
14	ENGINEERING - INGEGNERIA INFORMATICA SPA	ENG	IT
15	EIDGENOESSISCHE TECHNISCHE HOCHSCHULE ZUERICH	ETH	CH

Document Revision History

Version	Description	Contributions
0.1	Table of Content	HIRO
0.9	Ready for internal review	DELL, ECOM, ETH, MEF
1.0	Ready for submission	HIRO, MEF

Authors

Authors	Partner
Aidan O Mahony	DELL
Pournima Sonawane	DELL
Paolo Tartara	SOGEI
Raja Appuswamy	ECOM
Flavio Scaccia	ENG
Cem Tanrikut	HIRO
Maciej Besta	ETH

Reviewers

Name	Organisation
Aidan O Mahony	DELL
Raja Appuswamy	ECOM
Maciej Besta	ETH
Alessio Chellini	MEF

Table of Contents

List of Figures

List of Tables

List of Terms and Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
APC	American Power Conversion
AWS	Amazon Web Services
BIOS	Basic Input/Output System
CNCF	Cloud Native Computing Foundation
CPU	Central Processing Unit
DB	Database
DKG	Distributed Knowledge Graph
FIPS	Federal Information Processing Standard
FQDN	Fully Qualified Domain Name
FRU	Field Replaceable Unit
GPU	Graphics Processing Unit
HPE	Hewlett Packard Enterprise
iDRAC	Integrated Dell Remote Access Controller
iLO	Integrated Lights-Out
IoT	Internet of Things
IP	Internet Protocol
IPMI	Intelligent Platform Management Interface
IPMVP	International Performance Measurement and Verification Protocol
KVM	Kernel-based Virtual Machine
MIB	Management Information Base
ML	Machine Learning
NIST	National Institute of Standards and Technology
NIX	Network Information eXtension
NVLM	NVIDIA Management Library
OID	Object Identifier
OS	Operation System
PCI	Peripheral Component Interconnect
PDU	Power Distribution Unit
PMF	Power Measurement Framework
RAC	Remote Access Controller
RAPL	Running Average Power Limit
RBAC	Role-Based Access Control
SBOMs	Software Bills of Materials
SNMP	Simple Network Management Protocol
SPDX	Software Package Data Exchange
TSDB	Time Series Database
UI	User Interface
WP	Work Package
YAML	Yet Another Markup Language

Executive Summary

GLACIATION aims to leverage new technologies to enhance the energy, carbon, and material footprint of data operations throughout the data life cycle in data spaces. The data life cycle assists citizens, companies, and public organizations in meeting privacy and commercial requirements.

This document will focus on providing the research solution for the modern computing systems in collecting the different metrics of the power consumption in the GLACIATION platform.

Within the Work Package (WP), our research solution revolves around the development of a robust framework. This framework seamlessly integrates with key technologies such as Kepler, Kubernetes, and Prometheus, aiming to provide accurate energy measurements and detailed reporting of power consumption at the pod level. Additionally, integration with iDRAC and Smart PDUs ensures secure and comprehensive server management, facilitating deployment, updates, and monitoring of PowerEdge servers.

Moreover, the framework extends its compatibility to Grafana, a powerful tool for generating charts, graphs, and alerts from connected data sources. This integration with Grafana not only enhances visualization but also contributes crucial data to the GLACIATION platform. The collected metrics, power consumption measurements, and monitoring functions supported by iDRAC, Kepler, and Grafana become integral components feeding into the Power Measurement Framework (PMF) detailed in the Work Package.

By providing diverse power metrics, measurements, and visualization reports, this framework serves as a cornerstone within the GLACIATION platform, offering indispensable source data for subsequent functions and processes. Through this seamless integration of technologies, GLACIATION aims to set new standards for sustainable and efficient data operations, meeting both privacy and commercial requirements throughout the data life cycle.

1 Introduction

In the dynamic landscape of edge-to-cloud computing, where devices range from resource-constrained edge nodes to expansive cloud servers, the need for a unified power measurement framework has become paramount. Enter GLACIATION—an innovative paradigm designed to address the unique challenges of power monitoring and optimization in the continuum.

1.1 The Challenge of Unified Power Measurement

In the vast expanse between edge and cloud, diverse devices contribute to a complex ecosystem. Each node, be it at the edge or within the cloud infrastructure, demands precise power measurement for efficiency and sustainability. GLACIATION steps into this arena as a framework specifically engineered to integrate and unify power metrics across the entire spectrum.

1.2 Why GLACIATION?

Diverging from generic performance measurement frameworks, GLACIATION homes in on a pivotal facet of modern computing—power consumption. In this interconnected landscape, power efficiency transcends mere operational cost; it emerges as a strategic imperative for environmental sustainability. GLACIATION addresses this imperative by furnishing a centralized, adaptable, and comprehensive solution for the meticulous gathering, analysis, and optimization of power consumption at every tier of the edge-to-cloud continuum.

1.3 Navigating the Power Landscape

From individual edge devices to expansive cloud servers, GLACIATION adopts a multi-level approach to power measurement. This encompasses server-level metrics for granular insights into machine performance, rack-level data aggregation for collective energy usage, and pod-level measurements within the microservices architecture. This nuanced approach positions GLACIATION to optimize power consumption seamlessly across the entire ecosystem.

1.4 GLACIATION's Uniqueness

Distinguished from conventional frameworks, GLACIATION acknowledges the absence of a centralized solution for power measurement in the edge-to-cloud continuum. Its architecture is bespoke to this challenge, presenting not just a measurement tool but a holistic approach to power optimization. By harnessing GLACIATION, organizations gain a strategic advantage, achieving both operational efficiency and environmental responsibility in their computing infrastructure.

1.5 The Journey Ahead

In the subsequent sections, we delve into the intricacies of the GLACIATION Power Measurement Framework. From the pivotal role of Kubernetes as the central orchestrator to the challenges and opportunities entailed in integrating power measurement at various levels, this exploration aims to arm organizations with the knowledge and tools essential to navigate the power landscape in the era of distributed computing.

1.6 Deliverable Structure

This document reports on the work done in Work Package 5 "Energy Management Techniques" which was carried out from month 7 (April '23) to month 14 (November '23) of the project, and is aligned to T5.1 "Performance Measurement Framework". Work Package 5 is organized in 6 tasks and there is a chapter dedicated to each of the tasks, after a short introduction in Chapter 1.

Chapter 2 describes the section "GLACIATION Architecture". The GLACIATION framework is a comprehensive system that seamlessly connects edge to cloud. At its foundation, Kubernetes orchestrates microservices and scripts, shaping the GLACIATION experience. This section delves into the architecture, spotlighting the intricate integration of power measurement. The approach involves gathering power metrics at various levels, spanning servers, racks, and pods.

Chapter 3 describes the section "State of the art on Power Measurement Frameworks". The frameworks collectively address challenges in energy-efficient computing, offering tools for monitoring, analyzing, and optimizing power consumption. However, their suitability for GLACIATION depends on understanding its specific use cases and power measurement requirements.

Chapter 4 describes the section "Background". The Power Measurement Framework is a vital element in the domain of data operations and computing systems. Its purpose is to methodically evaluate and measure the efficiency, effectiveness, and overall performance of processes, systems, or platforms. The PMF offers a structured method to collect, analyze, and interpret performance metrics, empowering organizations to make informed decisions, streamline operations, and achieve specific objectives. In the context of GLACIATION, the PMF plays a crucial role in measuring and assessing power consumption metrics across the entire edge-to-cloud continuum. This data is instrumental for ongoing improvements and optimizations within the GLACIATION platform.

Chapter 5 describes the section "Performance/Power Measurement Framework Requirements". PMF is a pivotal element in the pursuit of efficiency and sustainability in modern computing. To harness its full capabilities, meeting specific requirements is essential to ensure the accuracy, comprehensiveness, and real-time availability of power metrics. These requirements serve as guiding principles, establishing the foundation for a robust framework that empowers organizations to optimize energy consumption throughout the edge-to-cloud continuum.

Chapter 6 describes the section "Software for Power Measurement Framework". The growing focus on energy conservation and efficient computing underscores the importance of comprehensive power measurement frameworks. For the GLACIATION project, understanding and optimizing power consumption are not just beneficial but could be crucial to achieving its objectives.

Chapter 7 describes the section "Preliminary Evaluation". Within the realm of GLACIATION, the primary goal was to showcase the seamless integration with validation cluster and the efficacy of our in-house power measurement framework. This framework was crafted to offer detailed insights into the energy consumption of the cluster, covering both edge gateways and servers.

2 GLACIATION Architecture

Within the complex fabric of the GLACIATION framework, the architecture serves as the foundational blueprint for a seamlessly integrated continuum spanning from edge to cloud. At its core, Kubernetes emerges as the linchpin, orchestrating the ballet of microservices and scripts that define the GLACIATION experience. In this section, we present the architecture of GLACIATION as seen in Figure 1 and also explore the intricacies of power measurement integration, showcasing the multi-level approach to gathering power metrics across servers, racks, and pods.
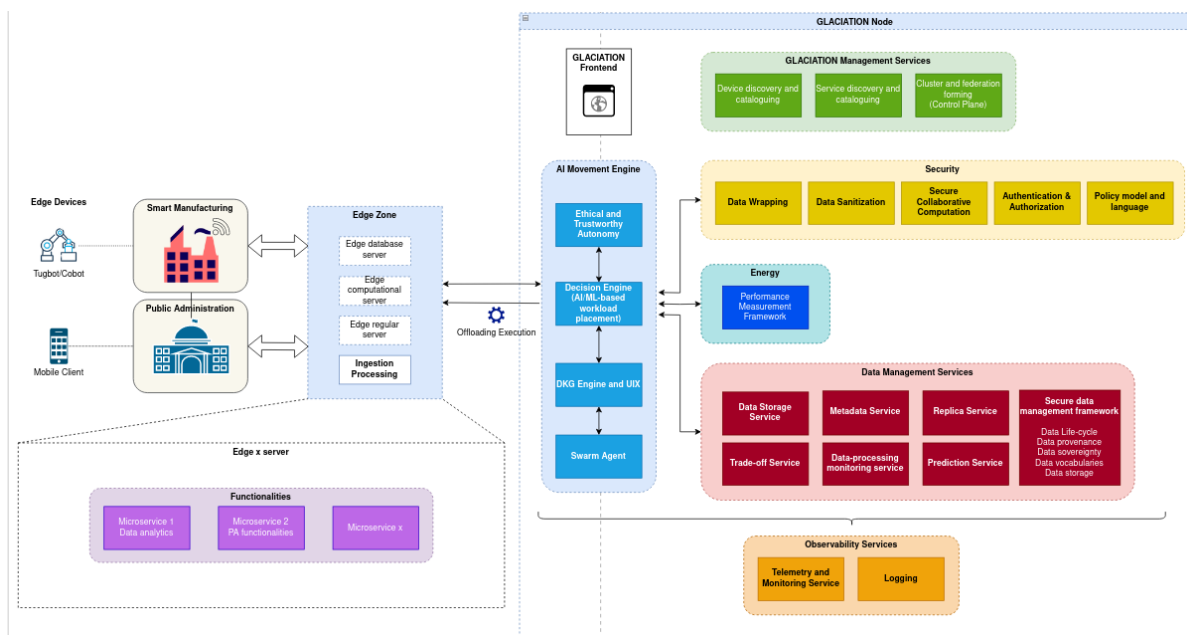
Figure 1: GLACIATION Architecture

2.1 Kubernetes as the Central Orchestrator

In the dance of distributed systems, Kubernetes takes center stage as the orchestrator extraordinaire. This open-source container orchestration platform provides the scaffolding upon which the GLACIATION framework is built. With Kubernetes, GLACIATION gains the ability to deploy, scale, and manage containerized applications seamlessly across diverse environments—from the edge devices to the expansive cloud infrastructure.

Kubernetes brings a level of abstraction that harmonizes the complexity of microservices. It ensures that GLACIATION can scale horizontally, distributing workloads efficiently, and enabling the framework to respond dynamically to the ebb and flow of demands in the edge-to-cloud continuum.

2.2 **Challenges and the Quest for Continuum Integration**

As we navigate the intricacies of power measurement integration within GLACIATION, it becomes evident that there is no singular, centralized framework capable of seamlessly gathering and integrating power metrics across the edge-to-cloud continuum. The decentralized nature of microservices, combined with the diverse infrastructure spanning edge devices to cloud servers, necessitates a tailored approach.

GLACIATION's architecture embodies the philosophy of adaptability and extensibility. It acknowledges the absence of a one-size-fits-all solution and instead embraces the challenge of integrating diverse technologies, sensors, and metrics across multiple levels.

In the chapters to follow, we delve deeper into the practical implementation of this architecture. From configuring Kubernetes for optimal power measurement integration to developing custom scripts for gathering and analyzing power metrics, we embark on a journey to empower GLACIATION with the insights needed to optimize not just performance but also the environmental impact of the entire edge-to-cloud ecosystem.

3 State of the art on Power Measurement Frameworks

Power measurement frameworks play a pivotal role in contemporary computing ecosystems, providing the essential tools and methodologies for quantifying and optimizing energy consumption. In the face of growing environmental concerns and the increasing demand for energy-efficient technologies, the need to monitor and manage power usage has become paramount. These frameworks encompass a spectrum of solutions spanning hardware, on-chip, and software domains, catering to diverse computing architectures.

Hardware solutions for power measurement frameworks involve the integration of specialized devices to accurately measure and monitor power consumption within computing systems. Notable examples include devices such as WattsUp, which offers real-time power monitoring capabilities, enabling users to gain insights into energy usage patterns. PowerInsight provides detailed power consumption data for various system components, offering granularity in power analysis to identify and target power-hungry elements for optimization. GreenMiner is specifically designed for energy-efficient cryptocurrency mining, reflecting a niche focus in the hardware-based power management landscape.

Modern chip power sensors are designed with high precision, beyond those of hardware solutions, offering real-time monitoring capabilities to accurately measure power consumption at the level of individual components or circuits within integrated circuits. These sensors often leverage advanced semiconductor manufacturing processes and innovative materials to achieve minimal intrusion on the chip's functionality while providing detailed insights into power usage patterns. RAPL is one such solution for Intel CPUs, while the power consumption of NVIDIA GPUs can be accessed through NVML.

Software solutions encompass a diverse array of tools designed to enhance energy efficiency in computing systems. PowerAPI provides a versatile software framework, offering an API for accessing and managing power-related information from various hardware components, enabling developers to integrate power awareness into applications. SmartWatts focuses on dynamic adjustments to CPU frequency and voltage in server environments, striking a balance between performance and energy efficiency. Tools like Joulemeter estimate power consumption in Windows-based systems, providing valuable insights for optimizing energy usage. Additionally, frameworks like JRAPL, Joliner, and Jalen showcase a commitment to power-aware runtime environments for Java applications and Linux-based systems, offering flexibility and extensibility for diverse system configurations. WattWatcher estimates power consumption of live systems by collecting performance events and feeding them as input into power models.

Power measurement frameworks for GPUs represent a critical facet in optimizing energy efficiency for artificial intelligence workloads. Tools like AccelWattch focus on profiling and analyzing power consumption in GPU-accelerated applications, providing insights into energy usage patterns to guide optimizations. GPUWattch, as a modeling and analysis framework, enables researchers and developers to gain a deep understanding of power consumption dynamics within GPU architectures. GPUSimPow, a simulation framework, allows for in-depth exploration of power-related aspects in GPU design, facilitating research and development in power-aware GPU architectures.

These frameworks collectively address the unique challenges associated with energy-efficient computing, offering researchers, developers, and system architects the tools needed to monitor, analyze, and optimize power consumption in specific elements of the computing stack. However, in order to assess whether these tools are suitable for GLACIATION, we need a clear understanding of the requirements of each GLACIATION use case with respect to power measurement.

4 Background

The Power Measurement Framework serves as a crucial component in the realm of data operations and computing systems. This framework is designed to systematically assess and quantify the efficiency, effectiveness, and overall performance of various processes, systems, or platforms. It provides a structured approach to gather, analyse, and interpret performance metrics, enabling organizations to make informed decisions, optimize operations, and meet specific goals. In the context of GLACIATION, the PMF plays a pivotal role in measuring and evaluating the power consumption metrics across the entire edge-to-cloud continuum, contributing essential data for further enhancements and optimizations within the platform.

4.1 Integration of the Power Measurement Framework

At the heart of GLACIATION lies the integration of the Power Measurement Framework—a critical component in the pursuit of efficiency and sustainability. By integrating power measurement at multiple levels, GLACIATION takes a granular approach to understanding and optimizing energy consumption. This integration spans the server level, where individual machines are monitored, the rack level, capturing the collective power usage, and the pod level, encompassing the microservices environment.

4.1.1 Server-Level Power Measurement

At the server level, GLACIATION employs power measurement sensors to gather real-time data on individual machine performance. These sensors capture metrics such as CPU utilization, memory consumption, and energy consumption. Kubernetes, acting as the orchestrator, ensures that these measurements are seamlessly incorporated into the overall monitoring and decision-making processes.

4.1.2 Rack-Level Power Measurement

Scaling up, GLACIATION extends its power measurement capabilities to the rack level. By aggregating the power metrics from individual servers, the framework gains insights into the collective energy usage within a rack. This holistic view enables efficient resource allocation, optimizing the power footprint of the entire rack.

4.1.3 Pod-Level Power Measurement

The microservices architecture, orchestrated by Kubernetes, operates at the pod level. GLACIATION embraces this modularity, measuring power consumption at the granular level of individual pods. This fine-grained approach allows for precise identification of power-intensive microservices, facilitating targeted optimization efforts.

4.2 Hardware based Power Measurement Technologies

There are several technologies and devices available for real-time power measurement, offering remote monitoring via network or serial connection login. The most accurate strategy to measure energy consumption is using power monitors – hardware tools that connect to the power source of your device or component and measure the actual power leveraged at any instant of time.

4.2.1 Smart PDUs

Power Distribution Units (PDUs) are devices used to supply power to devices like server, storage and network equipment mounted on a datacentre rack. Smart PDUs are power distribution units which are having remote management capabilities. There are two types of PDUs, the basic type and the smart type. While both can provide reliable power distribution to critical IT equipment within a rack or cabinet, smart PDUs offer several intelligent features to help data centre managers understand their power infrastructure. With data centres becoming more dynamic and complex, smart PDUs have become more prevalent.

A smart PDU, also known as intelligent PDU, is capable of monitoring, managing, and controlling power consumption to multiple devices. The smart PDU provides remote network access to real-time critical infrastructure data to help drive informed decision making to ensure maximum availability and to meet important efficiency requirements [1] [2].

Key Features:

- IP Aggregation: The smart PDUs can be deployed by utilizing units with IP aggregation capabilities. IP aggregation with self-configuration of downstream devices can significantly reduce deployment time and costs.
- Environmental Monitoring: Smart PDUs can incorporate environmental sensors to proactively monitor environmental conditions within the rack to ensure optimal operating conditions.
- Remote Connectivity: Smart PDUs provides the ability to access the PDU remotely through the network interface or serial connection to monitor power consumption and configure user-defined alert notifications to prevent downtime.
- Out-of-Band Communication: If the primary network to the PDU goes down, Smart PDUs provide redundant communications through integration with out of band management devices, such as serial consoles or KVM switches.
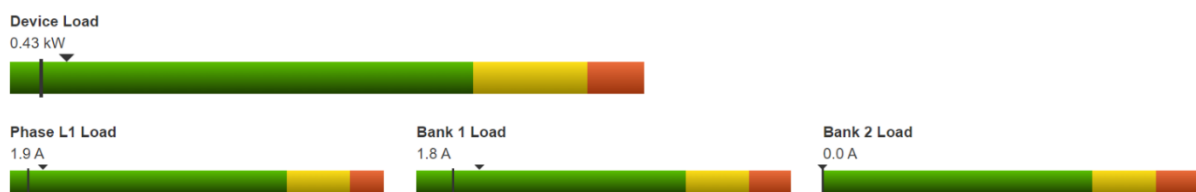
Figure 2: Smart PDU, below shows one of the APC Metered Rack PDU with 42 output connectors connected to Gateways as a part of GLACIATION Platform
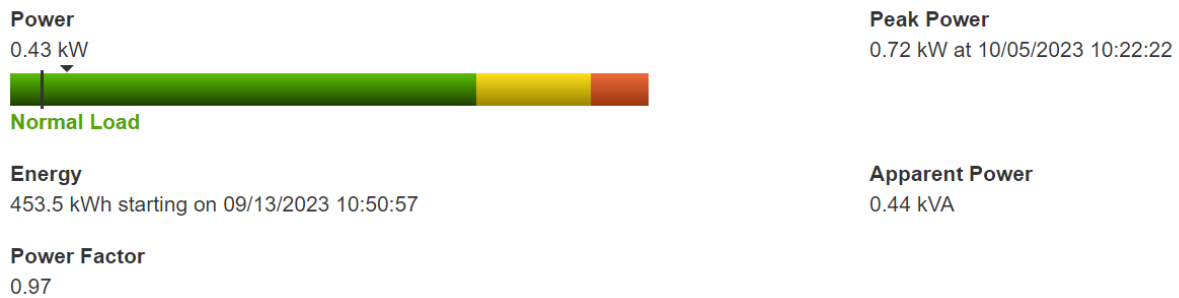
Figure 2: Smart PDU

Figure 3: Smart PDU Load Status and Figure 4: Smart PDU Device Status below shows the Graphical User Interface view of Smart PDU status.

Figure 3: Smart PDU Load Status

Power
0.43 kW

Peak Power
0.72 kW at 10/05/2023 10:22:22

Normal Load

Energy
453.5 kWh starting on 09/13/2023 10:50:57

Apparent Power
0.44 kVA

Power Factor
0.97

Figure 4: Smart PDU Device Status

4.2.2 iDRAC

Embedded with every Dell PowerEdge server, the integrated Dell Remote Access Controller (iDRAC) enables secure and remote server access for out-of-band and agent-free server management tasks. Features include BIOS configuration, OS deployment, firmware updates, health monitoring, and maintenance. One key set of data that iDRAC provides is power usage.

iDRAC is a hardware-based management and monitoring tool developed by Dell for its PowerEdge server line and other Dell EMC server products. iDRAC is designed for secure local and remote server management and helps IT administrators deploy, update, and monitor PowerEdge servers anywhere, anytime. iDRAC provides the rich server power usage data available from Dell PowerEdge servers and the various methods to collect, report, analyse, and act upon it. It also simplifies server management, reduces downtime, and aids in troubleshooting and maintenance tasks [3].

iDRAC monitors the power consumption in the system continuously and displays the following power values [4]:

- Power consumption warning and critical thresholds.
- Cumulative power, peak power, and peak amperage values.
- Power consumption over the last hour, last day or last week.
- Average, minimum, and maximum power consumption.
- Historical peak values and peak timestamps.

The histogram for the system power consumption trend (hourly, daily, weekly) is maintained only while iDRAC is running. If iDRAC is restarted, the existing power consumption data is lost, and the histogram is restarted [5].

Key Features: -

Remote Access: iDRAC provides secure remote access to the server's hardware, including remote console access, keyboard, video, and mouse (KVM), and virtual media access. This allows administrators to troubleshoot, configure, and manage servers remotely as if they were physically present at the server location.
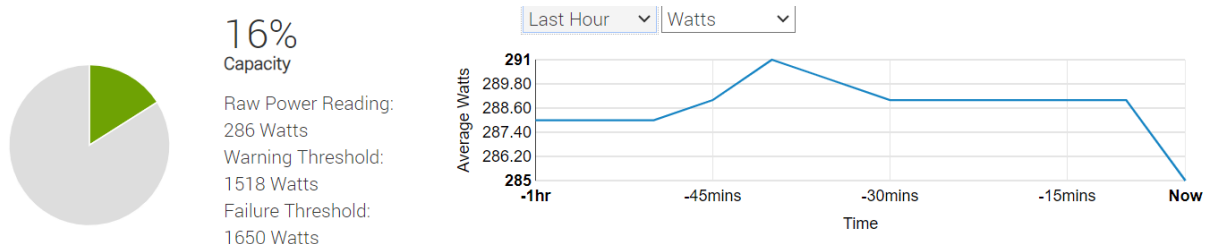
- System Monitoring: iDRAC monitors server's all hardware components, including CPU, memory, storage, fans, fan speed, temperature sensors, chassis alarms, power supply, individual disk status, RAID status and more.

- Notification alerts: iDRAC can provide email alert notifications in addition to IPMI (Intelligent Platform Management Interface) alerts, SNMP traps, and other management notifications when the status of a system component is greater than the pre-defined condition, for system events such as reboot, power cycle, high temperatures, power issues or any hardware component failures or that requires attention.
- Lifecycle Controller: iDRAC with Lifecycle Controller technology in the server's embedded management allows to perform tasks such as configuring BIOS and hardware settings, firmware updates, OS deployment, updating drivers, RAID settings from a remote repository, simplifying maintenance tasks. Together, iDRAC and Lifecycle Controller provide a robust set of management functions that can be used throughout the entire server lifecycle.
- Secured Connectivity: iDRAC offers industry-leading security features that adhere to and are certified against well-known NIST (National Institute of Standards and Technology), Common Criteria, and FIPS-140-2 (Federal Information Processing Standards). iDRAC provides secure remote access using encryption, user authentication, offering simple two-factor authentication option to enhance login security for local users.
- Scalable data analytics with telemetry streaming: iDRAC allows to proactively manage systems by analysing trends and discovering relationships between seemingly unrelated events and operations using analytics tools. iDRAC9 telemetry streaming with over 180 metrics/sensors can provide data on server status with no performance impact on the main server.

iDRAC is available in various versions and with different features depending on the specific Dell server model. The iDRAC 9 provides a graphical view of these power metrics such as the power consumption as shown in the Figures below (Figure 5: Power Trends in Last One Hour, Figure 6: Power Trends in Last One Week, Figure 7: Power Reading Details - I and Figure 8: Power Reading Details - II).

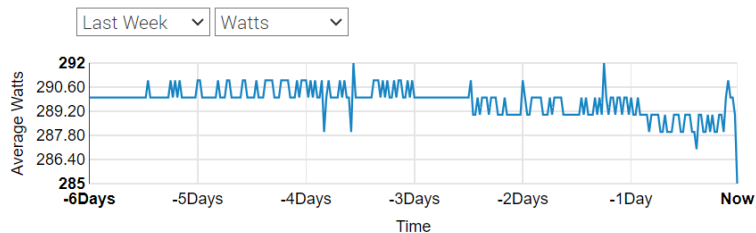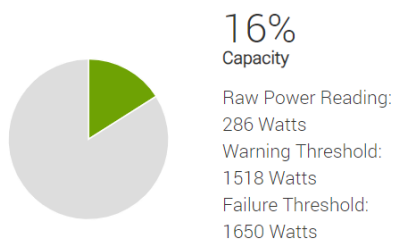**Figure 5: Power Trends in Last One Hour**

16%
Capacity

Raw Power Reading:
286 Watts
Warning Threshold:
1518 Watts
Failure Threshold:
1650 Watts

Figure 6: Power Trends in Last One Week

Present Power Reading and Thresholds

Probe Status	Probe Name	Present Reading	Warning Threshold	Failure Threshold
✅	System Board Pwr Consumption	264 Watts 901 BTU/hr	1518 Watts 5181 BTU/hr	1650 Watts 5631 BTU/hr

Edit Warning Threshold

Power Supply Unit Readings

Name	Amps	Volts	Watts
PS1	1.2	232	286
PS2	0.2	236	5

Figure 7: Power Reading Details - I

Raw Power Consumption

Raw Power Consumption	286 Watts | 976 BTU/hr

Cumulative Reading

Time	Since Sun Sep 20 05:44:40 2020
Total Usage	288.975 kWh

Historical Peaks

Time	Since Sun Sep 20 05:44:41 2020
Peak Watts	666 Watts
Peak Watts Time	Mon Sep 25 13:55:57 2023
Peak Amps	2.9 Amps
Peak Amps Time	Tue Oct 24 11:48:28 2023

System Headroom

Instantaneous	1114 Watts | 3802 BTU/hr
Peak	734 Watts | 2505 BTU/hr

Figure 8: Power Reading Details - II

The advanced capabilities of the IDRAC offers an extensive amount of data about power consumption from Dell PowerEdge servers. This power information is available on the iDRAC UI, as is telemetry information ready to be consumed by analytic solutions such as Splunk, RACADM CLI and RESTful API [6].

5 Performance/Power Measurement Framework Requirements

In the quest for efficiency and sustainability in modern computing, the Power Measurement Framework stands as a crucial component. To unlock its full potential, specific requirements must be met to ensure accurate, comprehensive, and real-time power metrics. These requirements serve as the guiding principles, laying the groundwork for a robust framework that empowers organizations to optimize energy consumption across the edge-to-cloud continuum. From hardware-level precision to seamless integration with monitoring tools, the Power Measurement Framework Requirements pave the way for informed decision-making and strategic resource allocation.

5.1 Use Case Specific Requirements

In tailoring a Power Measurement Framework to specific use cases, precision is paramount. Each scenario demands nuanced requirements to capture, analyse, and optimize power consumption effectively. Whether it's the dynamic demands of edge computing, the scalability needs of cloud environments, or the intricacies of distributed microservices, the Use Case-Specific Requirements for the Power Measurement Framework become the guiding parameters. These requirements ensure that the framework aligns seamlessly with the unique characteristics of each use case, empowering organizations to make informed decisions and drive energy efficiency with targeted precision.

5.1.1 Use Case 1

The Actual tool used in the MEF/Sogei Datacentre is the Schneider Electric StruxureWare Data Centre Expert. This tool is integrated with APC Power Distribution Units installed on every Rack of the Datacentre. This tool is used to measure the instant power consumption of every Rack.

The expected Outcome and KPI of Use Case 1 include reduction of energy consumption, optimization of data movement and saving in central processing time, as shown in Table 1: Use Case 1 Requirements. As stated in the glaciation requirements document "GLACIATION Use case requirements" delivered in MS3 - Project Baseline have been defined this specific UC energy requirements:

Table 1: Use Case 1 Requirements

User Story ID#	I want «some goal»	… so that «some reason»
US1_005	Use AI and ML technologies to identify areas of energy waste and central computation overhead and solutions to reduce it through a workload placement	The platform could gain energy optimization features addressing the data movement bottleneck

The actual tool present in the MEF/Sogei datacentre doesn't gather the specific energy consumption of the single application process or measurement of energy consumption due to data movement. So, by integrating Glaciation platform and using its tools for energy measurement the NoiPA service aim to optimize distributed vs central calculation and data movement by adopting AI/ML engine to analyse data on energy consumption.

5.1.2 Use Case 2

The Smart Factory introduces changes to the current factors and elements of traditional manufacturing facilities and incorporates multiple requirements for Smart Factory use case to be fulfilled as a crucial step to define for new architectural developments and to optimize the manufacturing process, as seen in Figure 9: Smart Factory Requirements below. There are several key functional and non-functional requirements to be considered to harness the full potential of smart manufacturing, to enhance efficiency, productivity, and competitiveness.
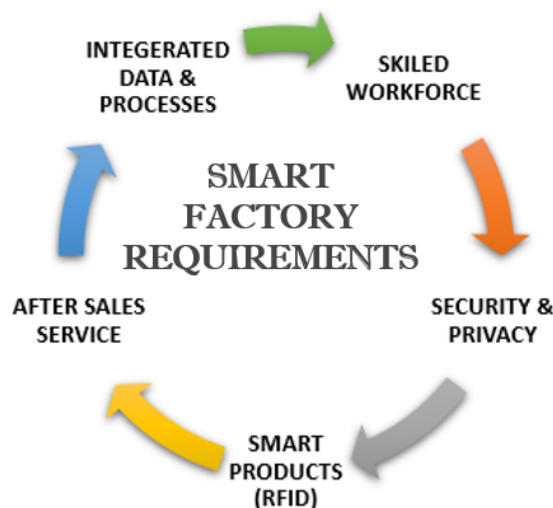

Figure 9: Smart Factory Requirements

FUNCTIONAL REQUIREMENTS

Functional requirements define the system behaviour and features that allow the system to function as it was intended. They are what the system does or must not do in terms of how the system responds to inputs and include calculations, data input, and business processes. Essentially if the functional requirements are not met, the system will not work. Functional requirements are product features and focus on user requirements.

Table 2 below defines the functional requirements, what a system is supposed to do to achieve the goal of energy efficiency.

Table 2: Use Case 2 Functional Requirements

User Story ID#	I want «some goal»	… so that «some reason»
US2_001	To deploy a wide range of machine learning (ML) applications efficiently through a workload placement: Predictive maintenance in manufacturing, Object recognition in autonomous vehicles, Image and video processing for surveillance	The platform could gain energy optimization features addressing the data movement and central computation
US2_002	That large amounts of data generated could be collected, stored, and analysed optimizing the use of renewable resources	Energy consumption minimization
US2_003	To improve energy efficiency through Energy Data Management plan and hardware optimization	The factory can reduce energy consumption and participate in the RE100 initiative and the International Performance Measurement and Verification Protocol (IPMVP)
US2_004	To measure and verify energy performance using Energy Management Performance Indicators and advanced metering	To ensure and improve energy performance
US2_005	The use case to be designed for easy integration with existing green energy infrastructure	To incorporate renewable energy sources
US2_006	To leverage existing technologies and best practices	To efficiently incorporate reusable components into the use case and efficient technologies such as: Apache Cassandra for distributed data storage, Apache NiFi for data flow processing, Kubernetes
US2_007	To use swarm intelligence and secure collaborative computation to increase overall efficiency	The system can work together effectively and efficiently
US2_008	Monitoring through real-time energy consumption analysis and reporting	Energy management

US2_009	Use AI and ML technologies to identify areas of energy waste and solutions to reduce it.	Reduce energy waste.
US2_023	To process data at the edge, near where it is generated, employing necessary dynamic data movement mechanisms	To reduce the amount of data that needs to be transmitted over the network

5.1.3 Use Case 3

There are no specific energy requirements defined for this use case.

5.2 Non-functional Requirements

Based on the use case requirements described earlier, the following items have been identified as the non-functional requirements of the Power Measurement Framework for successful configuration and setup in an environment suitable for NoiPa Platform inside the MEF infrastructure and in DELL manufacturing infrastructure:

1. The PMF must have Restful APIs to integrate with other software
2. The PMF must be integrated with Prometheus Server
3. The PMF should permit to create, save and schedule user-defined reports for ease of data collection, distribution and analysis.
4. The PMF should have centralized management by using a centralized repository accessible from anywhere on the network through a console application.
5. The PMF should have functions to define user access and viewing capabilities to individual groups.

While Smart Factory can still work if non-functional requirements are not met, it may not meet the expectations or the needs but serves the attributes such as affordable, easy to use, and accessible and keep functional requirements in line. Table 3 shows the Non-Functional Requirements for Use Case 2.

Table 3: Use Case 2 Non-Functional Requirements

User Story ID#	I want «some goal»	… so that «some reason»
US2_010	To be scalable	To accommodate future increases in renewable energy generation
US2_011	To use a demand-response systems that can adjust energy consumption patterns	To accommodate changes in the availability of green energy

US2_012	To gather specific metrics (Section 10 of UC2 questionnaire)	To measure the green energy consumption performance
US2_015	Data minimization	Collect only necessary personal data and avoid collecting unnecessary data

As the manufacturing industries adopting smart manufacturing approach, understanding these functional and non-functional requirements is vital. These requirements not only enhance efficiency and productivity but also lay the foundation for sustainable, competitive, and innovative manufacturing operations.

6 Software for Power Measurement Framework

In today's era of smart electronics and digital evolution, understanding and optimizing power consumption has become crucial. Software for power measurement frameworks play a pivotal role in aiding developers, engineers, and researchers in monitoring and managing energy utilization in various devices, ensuring efficiency, longevity, and sustainability.

Power measurement frameworks are indispensable tools in the modern electronic landscape. As devices continue to become smarter and more integrated into our daily lives, the importance of efficient power consumption grows. With the right software tools, we can ensure that our devices not only deliver optimal performance but also consume energy responsibly, paving the way for a sustainable tech-driven future.

The increasing emphasis on energy conservation and efficient computing has heightened the need for comprehensive power measurement frameworks. For the GLACIATION project, understanding and optimizing power consumption is not just beneficial but could be crucial to its objectives.

Integration with GLACIATION:

- **Component-Specific Metrics:** Depending on the hardware and software components used in the GLACIATION project, the framework can be tailored to measure power consumption for each element distinctly.
- **Scalability Analysis:** If the GLACIATION project involves scalable systems or processes, the framework will highlight how power consumption varies with scaling, aiding in efficient resource allocation.
- **Operational Insights:** The framework can provide insights into which operations or tasks within the GLACIATION project are the most power-intensive, guiding potential refactoring or optimization efforts.

Benefits to GLACIATION:

- **Operational Efficiency:** By understanding power consumption patterns, GLACIATION can optimize its operations to minimize energy wastage.
- **Cost Savings:** Efficient power usage can translate to reduced operational costs, especially if the project involves large-scale or continuous operations.
- **Environmental Responsibility:** For projects committed to sustainability, optimizing power consumption aligns with eco-friendly objectives.

The PMF software is an invaluable tool for GLACIATION project. It not only provides granular insights into power consumption patterns but also sets the stage for informed, data-driven decisions. As the world moves towards sustainable solutions and efficient operations, such frameworks become central in balancing performance with power efficiency.

6.1 Overview of the Kubernetes Cluster

The Kubernetes cluster operates as a distributed system spanning cloud and edge computing layers. The architecture is divided into three main components: the cloud, near edge, and far edge. Within the cloud, the Kubernetes control plane manages the orchestration of containers,
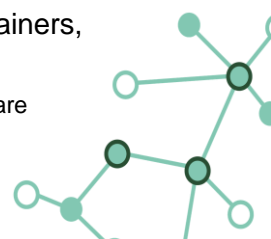
including scheduling, communication, and resource allocation. This allows for centralized management and coordination of complex workloads.

At the near edge, which is in closer proximity to the cloud infrastructure, Kubernetes nodes are deployed to facilitate lower-latency processing and regional data handling. These nodes serve as an intermediary processing layer, offering a balance between computational power and proximity to data sources.

The far edge, located in close vicinity to the data-generating devices, consists of Kubernetes nodes that operate in a constrained resource environment. These nodes are optimized for real-time data processing and immediate action, enabling localized decision-making and actions where latency is a critical factor.

The cluster employs Kubernetes' native tools such as Kustomize for resource configuration. This allows for consistent application management across the distributed architecture by handling application resources through Kubernetes manifests. The integration of specialized frameworks indicates the cluster's capability to support domain-specific applications, such as power measurement and management, essential for the monitoring and optimization of infrastructure performance.

The described setup exemplifies a scalable approach to infrastructure management, capable of accommodating a range of computational demands from centralized cloud services to decentralized edge functions.

6.2 Visualisation of GLACIATION Power Metrics

There are various ways to ensure green energy data operations, tracking its energy consumption. This section provides the essential details about the tools to gather system information and different ways of measuring energy consumptions approach on GLACIATION platform.

6.2.1 Grafana

Grafana is an open-source metrics visualization tool for advanced monitoring for latency, errors, transactions, health, and other process. It can be installed on any OS, and with Grafana data is accessible by everyone in the organization. With Grafana, anyone can create and share dynamic dashboards to foster collaboration and transparency. Grafana can take existing data- be it from Kubernetes cluster, raspberry pi, different cloud services, or even Google Sheets and visualize it from a single dashboard [7] [8].

Key Features:

- Dashboard templating: One of the key features in Grafana, the dashboard templating helps to build dashboards that can be reused for different purposes and shared across teams within the organization, can also contribute it to the whole community to customize and use, for example performance dashboard as seen in Figure 10.
- Panel editors: The panel editor is where user can update the elements of visualization, making easy to configure, customize and explore panels with a consistent UI, in a single pane and possibility to search each panel options.

- Custom Plugins: Plugins allows to extend Grafana and integrate it with other tools, visualizations, such as Zabbix, Splunk, Datadog, New Relic and others.
- Notification Alerts: Grafana Alerting allows to create, manage, and silence all alerts within one simple UI, to easily consolidate and centralize all the alerts. These events can be reported through Slack, SMS, email, or other communication channels. For example,one server down alert in Figure 11.
- Annotations: This feature shows up as a graph marker in Grafana, is useful for correlating data in case something goes wrong and allows to manually generate annotate graphs with rich events from different data sources or fetch data from any data source.
- Data sources and Real-time streaming: Grafana can view data stored in multiple different types of data sources such as Prometheus, MySQL, elasticsearch, etc, to have dashboards with real-time updates, for example read-write input-output requests, performance latency real-time streaming as seen in Figure 12.
- Authentication: Grafana supports a variety of authentication styles, including LDAP and OAuth, and allows to map users to organizations. Grafana supports user authentication and role-based access control (RBAC), ensuring only authorized individuals can access and modify dashboards.
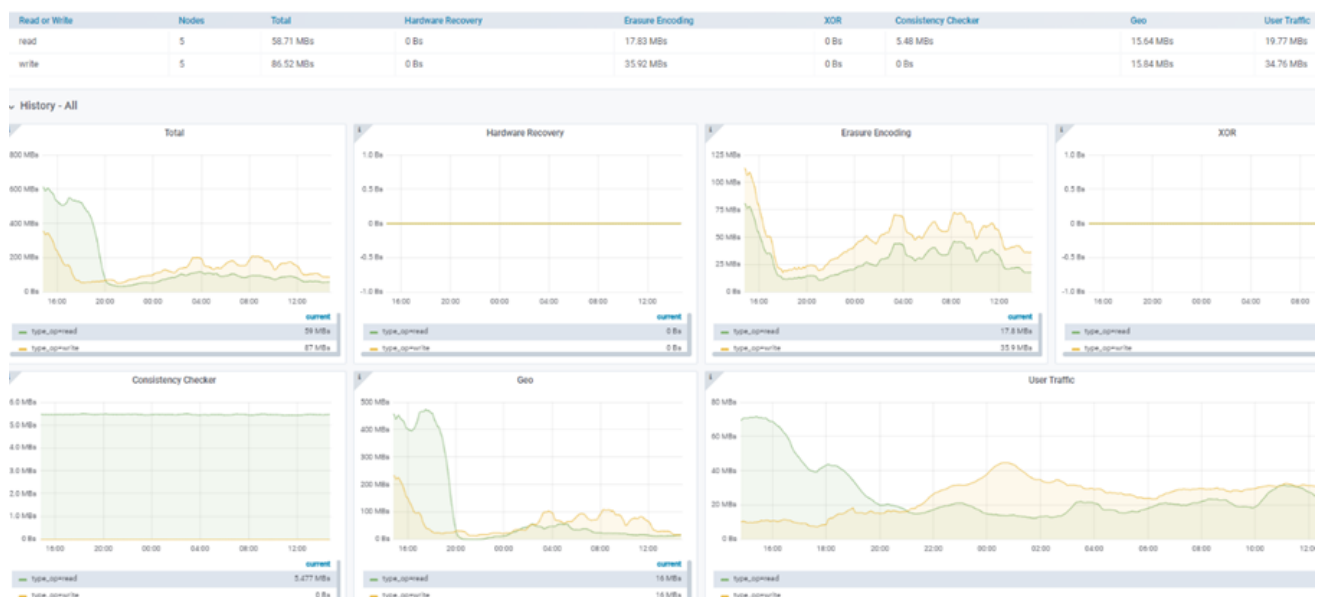
Figure 10: Performance History - All

Figure 11: 1 x Server Down / Services Unready[1]
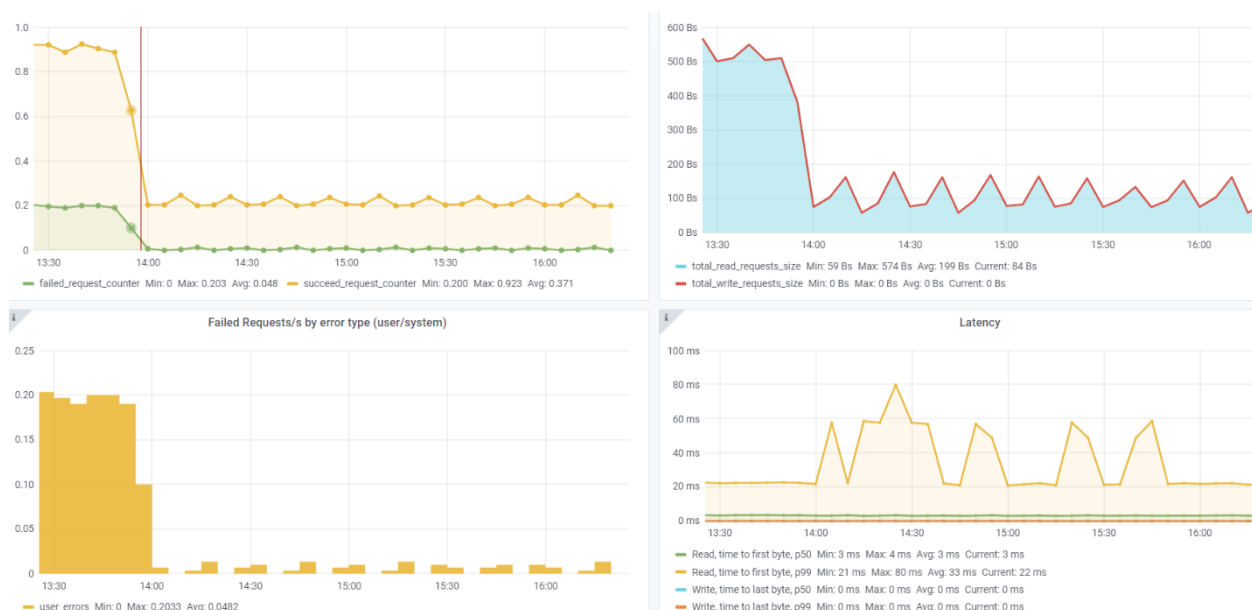
Figure 12: Data Access Performance Overview of 1 x Server

Grafana is often employed in conjunction with time-series databases like Prometheus, InfluxDB to create interactive and customizable dashboards for monitoring and observability solutions, performance data was streamed to InfluxDB for the data analysis, and Grafana then used for the visualization.

---

[1] Unready situation can cause due to services restarted or not running on a server

6.2.2 Prometheus

Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud in 2012 and later became a part of the Cloud Native Computing Foundation (CNCF), which houses various cloud-native technologies. Prometheus collects and stores its metrics as time series data, i.e., metrics information is stored with the timestamp at which it was recorded, alongside optional key-value pairs called labels. Prometheus, with a dimensional data model, flexible query language, efficient time series databases, supports Grafana integration and modern alerting approach, as Figure 13 shows the Prometheus deployment. Prometheus collects rich metrics and provides a flexible querying language; Grafana transforms metrics into meaningful visualizations. Both are compatible with most, data source types and is common for DevOps teams to run Grafana on top of Prometheus [9] [10] [11].
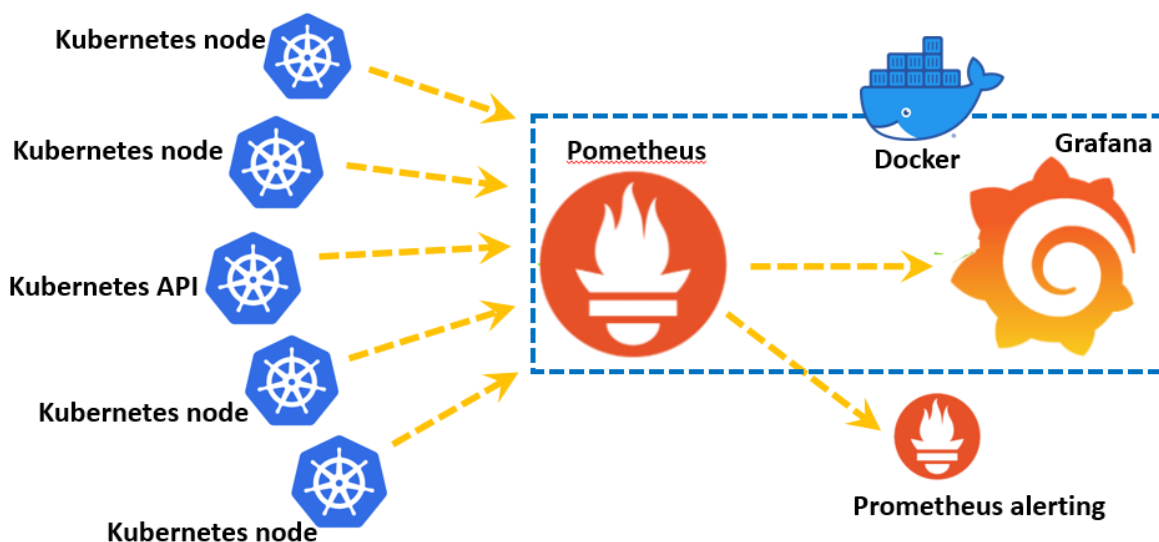
Figure 13: Prometheus Deployment

Key Features:

- Multidimensional Data Model: Prometheus uses a multi-dimensional data model, featuring time-series data with metric names and key-value pairs as identifiers.
- PromQL (Prometheus Query Language): PromQL is a robust and flexible query language which allows users to retrieve and process metrics data in real-time to support the multidimensionality of the data model.
- Pull Model: By actively "grabbing" data through HTTP, Prometheus may collect time-series data via a pull model over HTTP.
- No Reliance on Distributed Storage: No dependency on distributed storage, all single server nodes are self-contained.
- Pushing Time-series Data: This service is available through the usage of an intermediary gateway.
- Monitoring Target Discovery: The targets are discovered via both options service discovery and static configuration.

- Visualization: Prometheus has a variety of graphs and dashboards to choose from featuring various modes of graphing and Dashboard support.
- Notification Alerts: Prometheus is a flexible metrics collection and alerting tool that works with its companion Alertmanager service for alerts, which will convert them into pages, emails, and other notifications.
- Service Discovery: This component is used to discover targets automatically and monitor new service instances. Prometheus relies extensively on several service discovery techniques, and has integrations with many common service discovery mechanisms, such as Kubernetes, EC2, and Consul.
- Integration: Prometheus supports range of visualization options, including a built-in web UI and integration with third-party tools such as Grafana. The web UI lets users explore and visualize metrics data using a variety of chart types. In the meantime, Grafana provides advanced visualization and dashboarding capabilities.

Prometheus is a fundamental tool for monitoring the health and performance of services and applications and is often used alongside other cloud-native technologies such as Kubernetes.

6.3 **Prometheus Exporters**

There are several libraries and servers which help in exporting existing metrics from third-party systems as Prometheus metrics. This is useful for cases where it is not feasible to instrument a given system with Prometheus metrics directly (for example, HAProxy or Linux system stats) [12].

6.3.1 MIB Files

The Management Information Base (MIB) is a collection of Object Identifier (OID) definitions, which define the properties of the managed objects within the device in the Simple Network Management Protocol (SNMP). MIB is a structure that describes all objects a device can report on, such as CPU, fan, or temperature. MIB is a hierarchical structure, displayed as a navigation tree. Every entry in the MIB tree is a value for a specific component on a specific device [13].

MIBs are collections of definitions which define the properties of the managed object within the device to be managed and are accessed using a SNMP protocol.

PDUs utilize a separate MIB subtree for querying values via SNMP. These devices can report voltage, amps, volts, kW and kWh. Typically, these values are organized in tables in the MIB. For example, a hierarchical MIB for a rack PDU typically has separate tables for elements such as inlets, circuit breakers and outlets.

Every managed device keeps a database of values for each of the definitions written in the MIB. The MIB is not the actual database itself as it is implementation dependent. The MIBs are also available for Dell iDRAC, MIB provides management data that allows to monitor devices and software on a system via an out-of-band connection. DELL server MIBs can be found under the systems management section of the support page for the server, for example the MIB file for Dell PowerEdge R760, can be located from https://www.dell.com/ as seen in Figure 14.

- Enter the Service Tag of the DELL server or select the product type manually.
- Select an operating system and enter "MIB" in the Keyword field.

- The corresponding MIB package is shown as below as a .zip file.

Figure 14: MIB for Dell PowerEdge R760

There are two types of MIBs: scalar and tabular. Scalar objects define a single object instance whereas tabular objects define multiple related object instances grouped in MIB tables. These MIB files are versioned independently of product and can be used to identifies event types and event data related information [14].

6.3.2 SNMP Exporter

SNMP stands for Simple Network Monitoring Protocol. It is a protocol for management information transfer in networks, for use in LANs especially, depending on the chosen version. SNMP allows information about network-connected devices to be collected in a standardized way across a large variety of hardware and software types. Almost all kinds of devices from many different manufacturers support SNMP technology, which helps them achieve comprehensive monitoring [15] [16].

Figure 15: SNMP message flow

The transfer of SNMP messages is the typical communication between a client and a server, offering both pull and push technologies. The pull (or poll) technology is the most common communication type where a client, like the network management software on the managing entity, sends out a request to solicit a response from a server, or managed device. Its counterpart, the push technology, allows the managed device to "speak" and send out an SNMP message upon an event.

In SNMP terminology, for example, a GET request from an SNMP manager (client) follows the pull model, whereas an SNMP trap is "pushed out" by an SNMP agent (server) without any previous request, as Figure 15 shows.

SNMP can be used for network management. It gathers all the data from many devices and allows to put this data into context, which again allows to track issues, to make decisions based on real data, and to take control wherever necessary.

SNMP Exporter uses MIB files to interpret the messages sent by the managed devices, which is a critical step in network monitoring. MIB files provides a hierarchical database that contains configuration and other vital management information of SNMP devices in the form of data objects, describes managed device parameters such as port status, throughput etc. An SNMP management system uses these database files to request the agent for specific information and further translates the information as needed for the Network Management System (NMS). The MIB also serves as a best guide to the real capabilities of an SNMP device and gives a good idea of assets in place.

6.3.3 iDRAC Exporter

The idrac_exporter [17] is a simple tool designed for use with Prometheus to expose metrics from iDRAC (Dell), iLO (HPE), and XClarity (Lenovo) management controllers. It employs the Redfish API to communicate with these devices and exposes metrics through a /metrics endpoint, which Prometheus can scrape. The system supports various hardware systems adhering to the Redfish standard, and the exporter has been confirmed to work with HPE iLO 4/5, Dell iDRAC 9, and Lenovo XClarity.

The software is written in Go and can be downloaded and compiled or used within a Docker container. Configuration options allow specifying the metrics to be exposed, authentication details, and the address and port for binding the exporter. The metrics covered include system power, health, temperature, fan speeds, power supply readings, and system event logs, among others. These metrics are collected on-demand, which may take several minutes depending on the configuration, necessitating a sufficient Prometheus scrape timeout setting.

6.3.4 iDRAC SNMP Dashboard

A Dashboard for DELL iDRAC based on SNMP_Exporter and Prometheus shows Hardware status for: Storage, Disks, memory, controller, BIOS, Network, PCI, Power and iDRAC. Also shows some Hardware information such as: FQDN, ServiceTag, ServiceCode, FRUs, as seen in Figure 16 and Figure 17.

Dashboard can be found at https://grafana.com/grafana/dashboards/14395-idrac-snmp-dashboard/ and can also find the snmp_exporter settings used for this dashboard from git repository https://github.com/zorrzoor/grafana-idrac-dashboard.git.
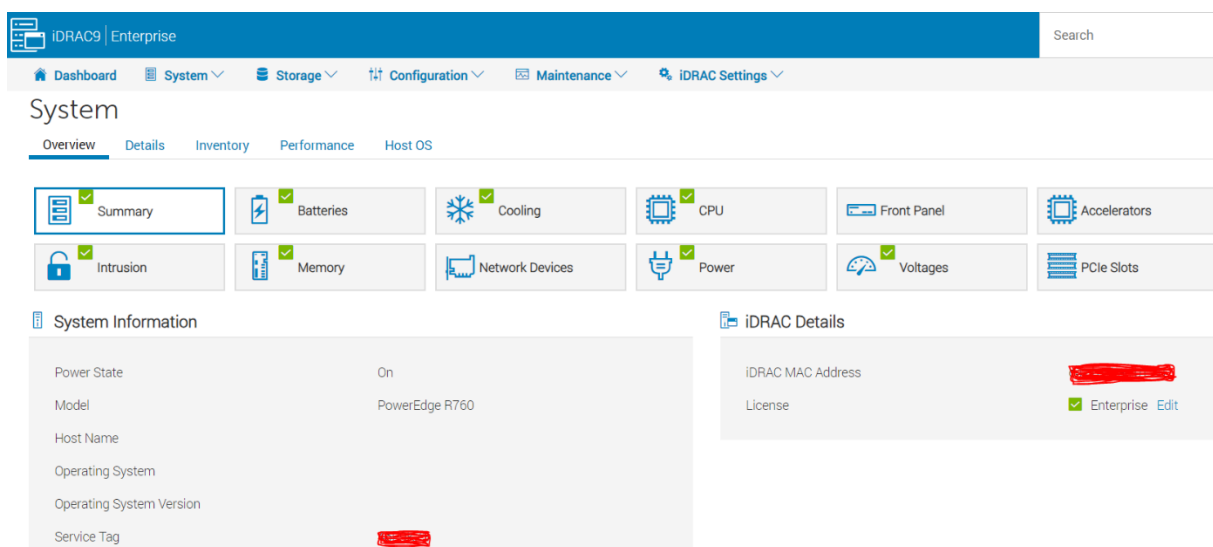
Figure 16: iDRAC Dashboard Summary
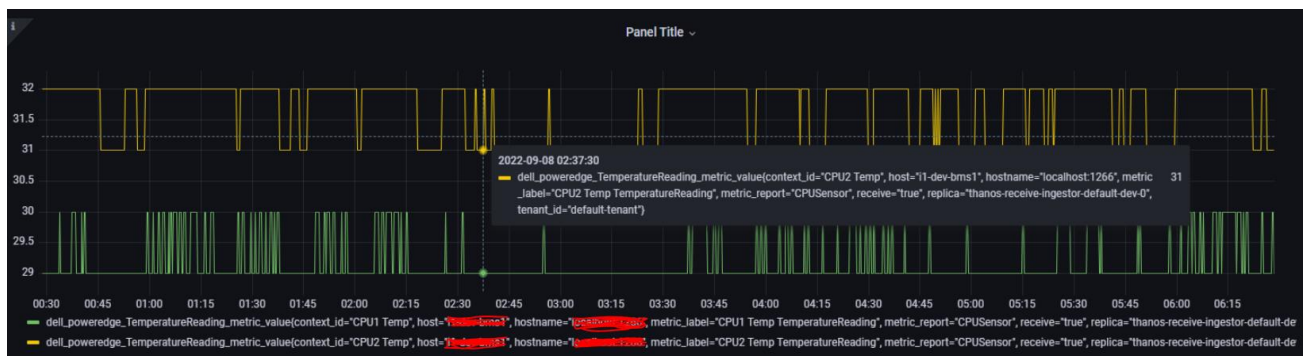
Figure 17: PowerEdge Server Temp Reading

Using the information presented above, the selection and connection of the correct instruments and tools can be made for various power measurement applications. Information gathered from these instruments can then be used to optimize designs, comply with standards, and provide nameplate information.

6.4 Representation of the Energy Metrics Lifecycle

Monitoring and displaying energy metrics are necessary for effective energy management in distributed systems in order to maximize resource utilization, identify performance problems, and guarantee system efficiency. This is a multi-step process that requires a combination of tools and components to complete from data extraction to visualization.
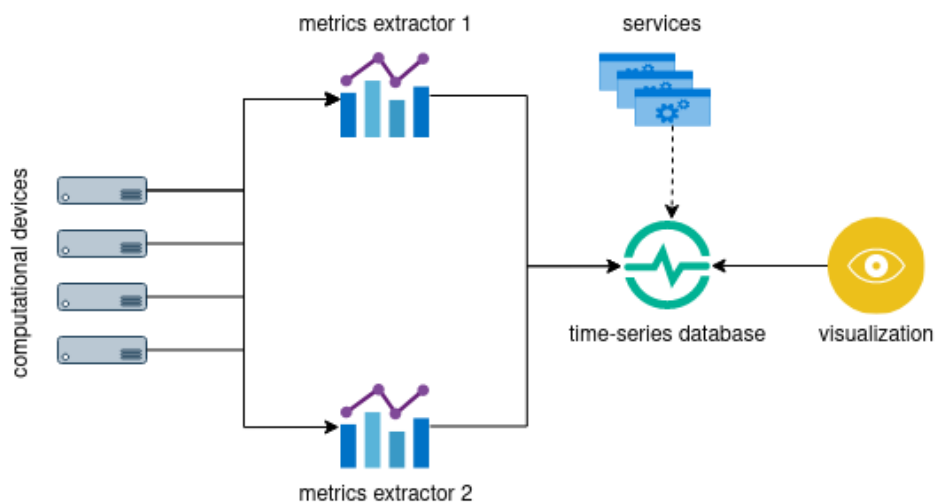
Figure 18: Metrics Collector Stack

A detailed description of the complete flow is visualized in **Error! Reference source not found.** and described below.

The first step in monitoring energy consumption is to retrieve relevant data from the distributed system. This information could include things like network traffic, CPU and memory utilization, rather, as this chapter focuses, energy metrics. Metrics are typically gathered by designated metric collectors or extractors who then periodically make them available through exporters. The diagram depicts the process, which starts at the end devices and proceeds to the metrics extractors. It is possible to employ multiple metrics extractors to gain the benefits of different frameworks that can work at different granularities of metrics extrapolation (node level, container level, etc.). Indeed, we employ iDRAC and Kepler, as explained in another section.

The measurements are sent then to a time series database (TSDB) for persistence once exporters have extracted and normalized them. Time series data can be stored using Prometheus and InfluxDB, two popular solutions. These TSDBs are excellent options for handling energy measures since they are specifically designed to store and retrieve time series data efficiently. For instance, Prometheus employs a pull-based paradigm. It can effectively store and handle time series data since it constantly scrapes information from exporters that are exposed at predetermined endpoints. Conversely, InfluxDB can operate in both pull and push modes, which enables it to adjust to various architectural configurations. These databases hold the metrics in a format that makes efficient aggregation and querying possible.
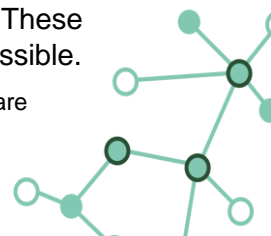
Finally, effective visualization of the stored energy measurements is necessary in order to extract valuable insights. Grafana is a popular platform that offers an easy-to-use interface for building personalized dashboards and visualizations, and it works easily with Prometheus and InfluxDB. Grafana can refresh the visualizations in real time by retrieving data from various databases via service monitors. Grafana's service monitors act as a link between the visualization layer and time series databases. They specify which metrics to show, how to deliver the data to users, and how to retrieve data from the databases. In order to guarantee that the visualizations are always current with the most recent metrics, service monitors can be set up to retrieve data at regular intervals.

Energy measurements collected from a distributed system are useful resources that go beyond monitoring and visualization. Although system administrators and operators can gain insights from visualization using tools such as Grafana, energy measurements can also be an important resource for other services and applications in the ecosystem. For instance, services in charge of scaling and resource distribution inside the distributed system can use energy data to help them decide how best to share computing resources. They can dynamically modify resource provisioning to maintain an ideal balance between performance and energy efficiency by examining the trends of energy consumption. They are also a useful tool for advanced analytics services to identify system anomalies and performance irregularities. These services could improve the reliability and stability of the system by proactively identifying possible problems, scheduling maintenance by utilizing machine learning and predictive modelling. These models can predict energy consumption trends, identify patterns, and optimize system behaviour, contributing to reduced environmental impact.

6.5 Metrics of System for Collecting Framework

In the realm of software and systems development, particularly for ambitious ventures like the GLACIATION project, monitoring and understanding operational metrics is of paramount importance. A framework that includes a metric gathering system offers an organized and efficient way to collect, analyse, and interpret these metrics.

6.5.1 Core Components of the Metric Gathering System:

- Data Collection Modules: These are specialized components designed to actively capture real-time metrics related to the operations of the GLACIATION project. This could range from system performance metrics to user interactions or any other quantifiable data.
    - o Database Integration Modules:
        - Modules that facilitate the connection to and extraction of data from databases, allowing scientists to integrate existing datasets into their workflows.
    - o API Integration Modules:
        - Modules that interact with external APIs (Application Programming Interfaces) to fetch data from online services or other software systems.
    - o Metadata Handling Modules:
        - Modules that manage and extract metadata associated with datasets, helping scientists understand the context and characteristics of the data they are working with.

.

- Data Processing Units: Once raw metrics are obtained, they're fed into processing units that filter, aggregate, and transform this data into meaningful statistics.
  - Data Transformation Units:
    - Modules that perform operations to transform data from one format to another. This might include scaling, normalization, or conversion of data types.
  - Analysis Units:
    - Modules designed for conducting specific analyses on the input data. This could involve statistical analyses, pattern recognition, or other computational procedures.
  - Visualization Units:
    - Modules responsible for creating visual representations of data. Visualization units help scientists interpret and communicate their findings effectively.
  - Machine Learning Units:
    - Modules that leverage machine learning algorithms for tasks such as classification, regression, clustering, or feature extraction.
  - Integration Units:
    - Modules that facilitate the integration of different datasets or the combination of data from various sources.
  - Parallel Processing Units:
    - Modules designed for parallel computing to enhance the efficiency of data processing, particularly when dealing with large-scale datasets.
- Visualization Interfaces: For any metric to be actionable, it must be interpretable. Visualization tools within the framework present the metrics in forms like graphs, charts, or dashboards, making it easier for stakeholders to understand and make informed decisions.
  - Expose Data from Kepler:
    - Ensure that relevant metrics or data generated by Kepler workflows are exposed or accessible in a format that Grafana can consume. This might involve writing scripts or modules to export data to a format compatible with Grafana's data sources.
  - Scripting or API Integration with iDRAC:
    - Develop scripts or modules within Kepler workflows that interact with iDRAC using its API. This could involve querying iDRAC for server health metrics, power consumption, or other relevant data.
  - Grafana Configuration:
    - Configure Grafana to connect to the data sources exposed by Kepler and, if applicable, the data retrieved from iDRAC. Create custom dashboards to visualize the desired information.
  - Automation and Alerts:
    - Implement automation within Kepler workflows to trigger alerts or actions based on the data retrieved from iDRAC. This might involve setting up alerts in Grafana or using other monitoring solutions.

Integration with GLACIATION:

- Custom Metrics: Depending on the unique needs and goals of the GLACIATION project, the metric gathering system can be tailored to collect project-specific metrics. This ensures that insights are directly aligned with project objectives.
  - Workflow Execution Metrics:
    - These could include metrics related to the time it takes for a workflow to execute, resource utilization, and efficiency measures.
  - Data Processing Metrics:
    - Metrics associated with the processing of data within the workflow, such as throughput, error rates, or the success/failure of individual processing steps.
  - Resource Utilization Metrics:
    - Metrics related to the utilization of computing resources, memory, and storage during the execution of the workflow.
  - Scientific Analysis Metrics:
    - Metrics specific to the scientific analyses performed by the workflow, which could include accuracy rates, precision, recall, or other domain-specific indicators.
  - Data Quality Metrics:
    - Metrics assessing the quality of input and output data, including completeness, accuracy, and consistency.
  - Custom Logging and Monitoring Metrics:
    - Workflows may include custom logging and monitoring mechanisms that generate metrics for specific events, warnings, or errors.
  - User Interaction Metrics:
    - If the workflow involves user interaction or decision points, metrics related to user engagement, response times, or decision-making could be relevant.

Benefits to GLACIATION:

- Informed Decision Making: Having a comprehensive view of operational metrics allows the GLACIATION team to make decisions based on hard data, minimizing risks and optimizing outcomes.
- Performance Monitoring: Continuously monitoring metrics ensures that the project operates at peak performance, and any deviations can be promptly addressed.
- Resource Optimization: By understanding resource utilization metrics, the GLACIATION project can ensure efficient use of resources, whether they be computational, human, or financial.

6.5.2 Runtime System Information Exporter

The Node Exporter is a Prometheus Exporter developed by the Prometheus project, is designed to expose hardware and OS metrics from *NIX based Kernels. The project can be found on GitHub, https://github.com/prometheus/node_exporter

The node exporter enables to measure various machine resources such as memory, disk and CPU utilization for each node running in a Kubernetes cluster [18] [19].

Deployment

The Node Exporter needs to run on each node in the Kubernetes cluster. The YAML creates a DaemonSet that launches the Node Exporter on each node in the Kubernetes cluster, includes a Kubernetes Service and ServiceMonitor to scrape metrics from all instances of Node Exporter, as seen in Figure 19.

Figure 19: Node Exporter Instances

Node Exporter has numerous collectors designed to gather OS and hardware metrics from various sources on a node. Figure below shows the output from a Node Exported Pod which shows the collectors that are active, as seen in Figure 20.

Figure 20: Node Exporter Collectors

A collector is a part of an exporter, is the code written to collect data of a metric or set of metrics such as CPU collector. Figure 21 shows its relation to nodes, Prometheus Node Exporter and Prometheus.
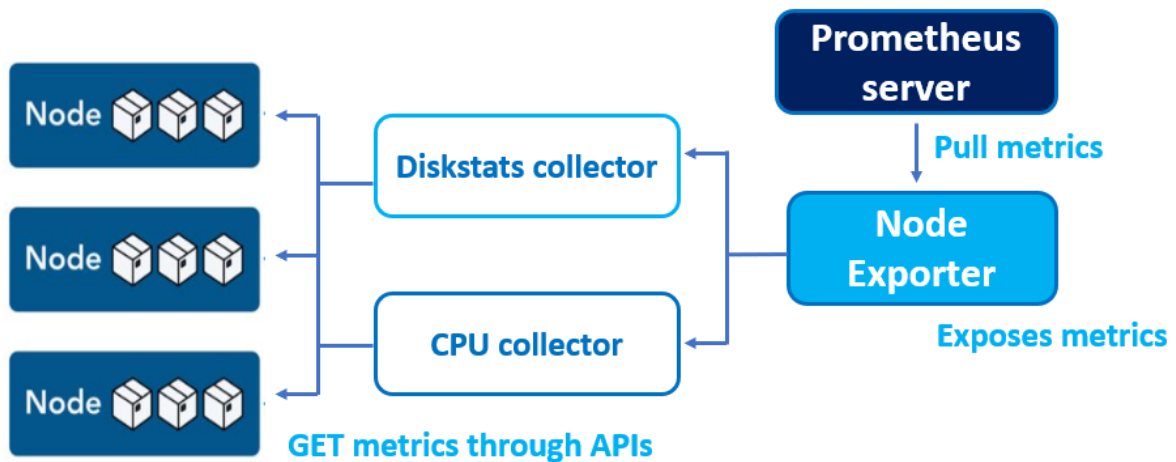
Figure 21: Metrics Collectors

Prometheus, Grafana, and Node Exporters are commonly used together in Kubernetes to monitor system-level application insights. These tools specifically provide node and container statistics, helps to analyse real-time metrics of containers and nodes. Prometheus Node Exporter can more specifically be used to get node metrics and system-level insights.

Prometheus Node Exporter is an essential part of any Kubernetes cluster deployment. As an environment scales, accurately monitoring nodes with each cluster becomes important to avoid high CPU, memory usage, network traffic, and disk IOPS. Avoiding bottlenecks in the virtual or physical nodes helps avoid slow-down and outages that are difficult to diagnose at a pod or container level [20].

6.5.3 Performance Monitoring Library

In the modern software development lifecycle, understanding how applications perform in real-time is pivotal. A performance monitoring library designed to record statistical information during run-time provides invaluable insights. When integrated with visualization tools like Grafana, it allows for easy interpretation and actionable insights from these statistics.

Performance Monitoring Library: Core Concepts

- Data Collection: This involves gathering real-time metrics from the application, such as CPU usage, memory consumption, response times, and error rates.
- Statistical Analysis: Raw metrics are processed to generate statistics like averages, percentiles, and standard deviations to better understand the application's performance nuances.
- Data Storage: The processed statistics are stored in a time-series database, ready for retrieval and visualization.

Integration with Grafana:

- Data Source Configuration: Grafana retrieves performance data from databases. When a performance monitoring library stores data, say in Prometheus, Grafana needs to be configured to fetch data from there.
- Dashboard Design: Once integrated, developers or system administrators can design Grafana dashboards tailored to visualize the recorded statistical information. This can include charts for average response times, histograms for request distributions, or heatmaps for error occurrences.
- Alerting: Based on the insights from the performance monitoring library, Grafana can be set up to send alerts when certain performance thresholds are breached.

Benefits of Using Grafana for Visualization:

- Intuitive Understanding: Graphical representation of statistics offers an intuitive understanding of how the application performs, making it easier to spot anomalies or trends.
- Customizability: Grafana dashboards are highly customizable, allowing teams to focus on metrics that matter most to them.
- Collaboration: Grafana's shared dashboards enable teams to collaborate, ensuring that everyone has access to real-time performance insights.

A performance monitoring library that records statistical information of run-time performance is a potent tool in the software optimization arsenal. When integrated with Grafana, it not only provides a granular look into application health and efficiency but also translates complex metrics into digestible, actionable insights. As applications grow and user expectations rise, such integrative tools will play a pivotal role in ensuring software robustness and reliability.

6.5.4 The Trace Information of Code for Both Serial and Parallel Environments

In software development and performance analysis, tracing code execution is a vital technique that allows developers to understand the flow and behaviour of their code. When dealing with both serial (single-threaded) and parallel (multi-threaded or distributed) environments, the complexity, and the importance of tracing increases exponentially.

Tracing is a dynamic analysis technique used to capture information about a program's execution. In complex orchestrated environments like Kubernetes, tracing provides clarity on how different microservices interact, especially when considering both serial and parallel execution patterns. The "Efficient Power Level Exporter" project, focusing on optimizing power consumption in Kubernetes, would benefit immensely from understanding these interactions.

What is Trace Information?

Tracing involves recording information about the execution of a program, capturing details such as function calls, variable values, or execution times. This trace information can be used for debugging, performance optimization, and understanding the code's overall flow.

Serial vs. Parallel Environments in Kubernetes:

- Serial Execution: This refers to a sequence where tasks are executed one after the other. In a Kubernetes context, it might refer to a series of dependent jobs or pods that need to run in a specific order.

- Parallel Execution: Here, multiple tasks execute concurrently. In Kubernetes, this could be multiple pods or containers running simultaneously, handling different aspects of an application.

Trace Information and Its Relevance:

- Pod Interactions: In the complex web of a Kubernetes cluster, understanding how pods interact is essential. Tracing can highlight call sequences, dependencies, and communication patterns.

- Resource Consumption: For a project like the "Efficient Power Level Exporter," understanding which pods or services consume the most power, especially during parallel execution, is crucial. Tracing can provide granular insights into CPU, memory, and I/O usage patterns.

- Parallelism Overheads: While parallel execution can speed up tasks, it can also introduce overheads, especially in synchronization or communication between pods. Tracing can help identify these bottlenecks.

Efficient Power Level Exporter and Tracing:

- Optimization Insights: By tracing the flow of execution, especially in parallel environments, the project can identify where power consumption inefficiencies arise.

- Anomaly Detection: Sudden spikes in power usage can be correlated with specific pods or services through tracing, aiding in early anomaly detection.

- Improved Scheduling: Insights from tracing can guide the Kubernetes scheduler to make power-efficient decisions, placing workloads on nodes in a manner that optimizes power consumption.

Tracing code execution in both serial and parallel environments provides a holistic view of interactions within a Kubernetes cluster. For a project like the "Efficient Power Level Exporter," which aims to optimize power consumption, such insights are invaluable. They not only shed light on current consumption patterns but also pave the way for predictive optimizations, ensuring Kubernetes operations that are both power-efficient and performant.

Kepler and Parallelism:

Kepler itself provides a graphical interface for designing workflows, and it supports the execution of workflows in various computing environments, including both serial and parallel settings. The choice of whether a workflow is executed in serial or parallel often depends on the nature of the scientific tasks and the available computational resources.

Kepler can be used in conjunction with parallel computing frameworks or job schedulers to execute workflows in parallel environments. Users can design workflows that take advantage of parallelism when it is beneficial for the tasks at hand.

In a sequential processing scenario, Kepler would go through the dataset one record at a time, performing the necessary computations. However, to expedite this process and handle the voluminous data more efficiently, Kepler can employ parallelism.

Parallelism in Action:

Data Partitioning: Kepler can break down the dataset into smaller partitions, each containing a subset of the records. For example, the dataset could be divided based on user sessions or time intervals.

Parallel Execution: With parallelism, Kepler can then distribute these partitions across multiple processing units or cores. Each core independently processes its assigned partition simultaneously.

Aggregation: Once the parallel processing is complete, Kepler can aggregate the results from each partition to generate a comprehensive analysis of user behaviour across the entire dataset.

By incorporating parallelism, Kepler significantly accelerates the data processing task, making it well-suited for handling large-scale datasets efficiently. This example illustrates how Kepler harnesses parallel computing to enhance its performance, making it a powerful tool for data analytics in scenarios where speed and scalability are critical.

6.6 Software Release Description

Software Bill of Materials

The Software Package Data Exchange (SPDX) format serves as a structured way of documenting software components and their metadata in our energy measurement project. It is a standardized format that helps in keeping track of the licenses, origins, and dependencies of software like Prometheus, Kubernetes, Docker, Grafana, and other related tools we employ. By using SPDX, we ensure that all elements of our framework are well-documented, which is essential for maintaining compliance with open-source licenses and managing software bills of materials (SBOMs) effectively. This practice of documentation aligns with the project's goals to measure software energy consumption transparently and responsibly, especially when it involves multiple software layers and interactions. A snippet of the SBOM for the PFM is shown in Figure 22.

```
      ---
134   ##### Package Information for Prometheus
135   PackageName: Prometheus
136   PackageVersion: 2.40.0
137   SPDXID: SPDXRef-Package-Prometheus-2.40.0
138   PackageFileName: prometheus-2.40.0.linux-amd64.tar.gz
139   PackageSupplier: Organization: Prometheus Authors
140   PackageOriginator: Organization: Prometheus Authors
141   PackageDownloadLocation: https://prometheus.io/download/#prometheus-2.40.0
142   FilesAnalyzed: false
143   PackageVerificationCode: d41d8cd98f00b204e9800998ecf8427e
144   PackageLicenseConcluded: NOASSERTION
145   PackageLicenseDeclared: Apache-2.0
146   PackageLicenseInfoFromFiles: Apache-2.0
147   PackageLicenseComments: None
148   PackageDescription: Prometheus, an open-source systems monitoring and alerting toolkit. Version 2.40.0 includes several improvements and bug fixes.
149   PackageComment: None
150
151   ##### Relationships
152   Relationship: SPDXRef-Package-GLACIATION-PowerMeasurementFramework MONITORS SPDXRef-Package-Prometheus-2.40.0
153   Relationship: SPDXRef-Package-GLACIATION-PowerMeasurementFramework VISUALIZES SPDXRef-Package-Grafana-9.1.5
154
155   ##### Package Information for Node Exporter
156   PackageName: Node Exporter
157   PackageVersion: 1.4.0
158   SPDXID: SPDXRef-Package-NodeExporter-1.4.0
159   PackageFileName: node_exporter-1.4.0.linux-amd64.tar.gz
160   PackageSupplier: Organization: Prometheus Authors
161   PackageOriginator: Organization: Prometheus Authors
162   PackageDownloadLocation: https://prometheus.io/download/#node_exporter
163   FilesAnalyzed: false
164   PackageVerificationCode: d41d8cd98f00b204e9800998ecf8427e
165   PackageLicenseConcluded: NOASSERTION
166   PackageLicenseDeclared: Apache-2.0
167   PackageLicenseInfoFromFiles: Apache-2.0
168   PackageLicenseComments: None
169   PackageDescription: Node Exporter, a Prometheus exporter for hardware and OS metrics exposed by *NIX kernels.
170   PackageComment: None
171
```

Figure 22: Snipped of SBOM for PFM

Deployment Script

The script serves as an orchestration tool to initialize and manage a suite of monitoring components collectively known as the Power Measurement Framework within a Kubernetes environment. The process begins with the deployment of various Prometheus exporters, each tailored for monitoring different aspects of system and hardware performance. These include the SNMP Exporter, iDRAC Exporter, and Node Exporter, which are launched in sequence to gather a comprehensive set of metrics. Once these exporters are operational, the script confirms the completion of this phase, ensuring that the foundational monitoring infrastructure is in place and actively collecting data, as seen in Figure 23 and Figure 24.

Subsequently, the script proceeds to set up Prometheus itself, which aggregates and stores the metrics collected by the aforementioned exporters. Prometheus is a vital component of the framework, serving as the central repository for time-series data that can be queried and analysed. After Prometheus is started, the script then deploys Grafana, a powerful visualization platform that connects to Prometheus to create informative dashboards. This enables users to visualize the metrics in an accessible and interactive manner. Finally, the script initializes Kepler, a specialized Prometheus exporter that uses eBPF to gather advanced performance metrics, providing insights into the power consumption of Kubernetes Pods. The execution of

this script results in a comprehensive monitoring framework that not only tracks system performance but also provides estimations of energy usage, catering to the growing need for energy-efficient operations in computing infrastructures.
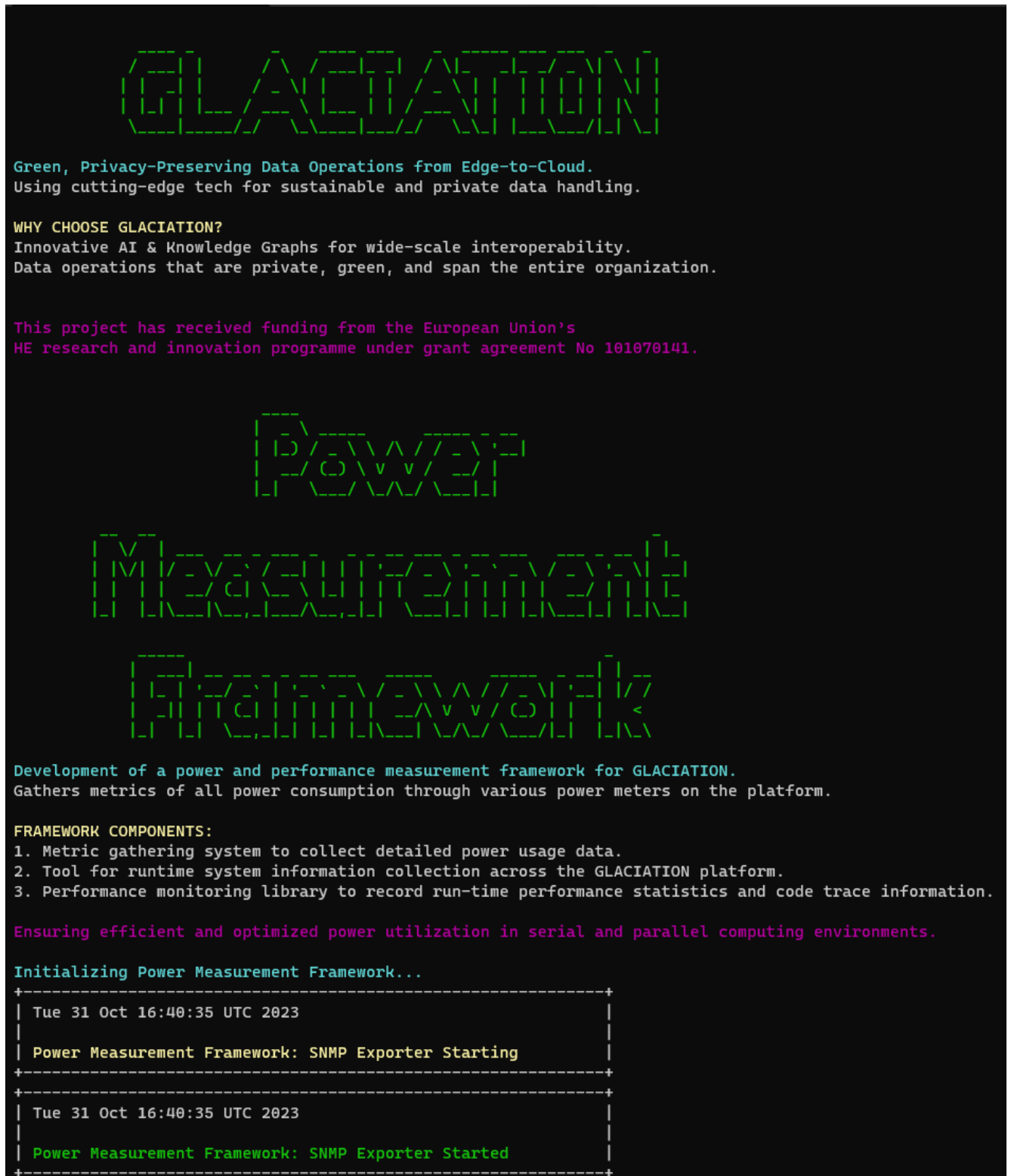
Figure 23: First Half of Startup Script

```
+-----------------------------------------------------+
| Tue 31 Oct 16:40:35 UTC 2023                        |
|                                                     |
| Power Measurement Framework: iDRAC Exporter Starting|
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:36 UTC 2023                        |
|                                                     |
| Power Measurement Framework: iDRAC Exporter Started |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:36 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Node Exporter Starting |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:36 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Node Exporter Started  |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:36 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Prometheus Exporters Complete |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:36 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Prometheus Starting    |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:37 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Prometheus Starting    |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:37 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Grafana Starting       |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:38 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Grafana Started        |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:38 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Kepler Starting        |
+-----------------------------------------------------+
+-----------------------------------------------------+
| Tue 31 Oct 16:40:38 UTC 2023                        |
|                                                     |
| Power Measurement Framework: Kepler Started         |
+-----------------------------------------------------+
ubuntu@GLACIATION-Mast01:~/energy-measurement$
```

Figure 24: Second Half of Startup Script

Software release version and location

The software for the PFM is on the GLACIATION GitHub repository, illustrated in Figure 25, will be included in the final version of the GLACIATION platform.

Figure 25: PFM Software Release

7 Preliminary Evaluation

In the context of the GLACIATION, the primary objective was to demonstrate the successful integration with the Use Case 2 validation cluster, and effectiveness of our internally developed power measurement framework. This framework was designed to provide granular insights into the cluster's energy consumption, encompassing both edge gateways and servers. To validate the framework's integration and utility, we employed K-Bench as our benchmarking tool.

7.1 Deployment of Power Measurement Framework with K-Bench

With the GLACIATION cluster operational, our first task was to ensure that our power measurement framework was seamlessly integrated with the existing Kubernetes infrastructure. The framework's sensors and monitoring tools were calibrated to capture power data in real-time across the cluster's diverse components. K-Bench was then installed and configured to execute a series of workloads that would engage the cluster in a manner reflective of its typical operational load.

7.2 Benchmarking Execution and Data Capture

The execution of K-Bench workloads was closely monitored, with a dual focus on performance metrics and power consumption. Our framework recorded the energy metrics in tandem with the workload performance, enabling us to establish a direct correlation between resource utilization and power usage. This data was pivotal in assessing the precision and reliability of our power measurement framework.

7.3 Analysis of Power Consumption Metrics

Upon completing the benchmarking procedures, we analysed the collected data to validate the accuracy and usefulness of the power measurement framework. The analysis confirmed that the framework was correctly integrated, capturing detailed consumption metrics that aligned with expected power usage patterns under the simulated workloads.

7.4 Optimization and Future Directions

The benchmarking exercise not only confirmed the successful integration of our power measurement framework but also provided actionable insights. These insights were used to fine-tune the GLACIATION cluster's energy consumption, leading to optimized power usage that did not compromise the cluster's performance or reliability.

The data derived from this benchmarking phase is critical in guiding our ongoing efforts to enhance the energy efficiency of the GLACIATION cluster. It demonstrates the value of our power measurement framework as a tool for continuous improvement and sets a strong foundation for its application in broader sustainability initiatives. The PMF will be used in the remaining tasks of WP5 and more immediately in T5.2 "Workload-driven evaluation of energy efficiency".

8 Conclusions

In our pursuit of a Performance Measurement Framework, we embarked on a journey to meet specific needs, construct a tailored solution, and ultimately, to leverage its capabilities. We identified the need for a unified system capable of gauging and optimizing performance across the intricate edge-to-cloud continuum.

What We Needed:

Our objective was clear—to bridge the gaps in conventional measurement systems and create a framework that not only captured the nuances of performance but did so seamlessly in the diverse landscape spanning edge devices to servers.

What We Did:

We crafted a comprehensive Performance Measurement Framework, intricately woven with components such as Kubernetes, Docker, and a nuanced approach to power measurement. The integration of iDRAC, Kepler, and Grafana further solidified our foundation, providing a robust infrastructure for data gathering, analytics, and visualization.

The Use and Impact:

By focusing on power measurement and integrating it seamlessly into our architecture, we laid the groundwork for not only better performance but also heightened environmental sustainability.

Interactions within WP5:

Crucially, the genesis of the Performance Measurement Framework aligns with the objectives of other tasks in WP5. This deliberate integration ensures that our research endeavours are synergistic, contributing to a cohesive and comprehensive approach within the Work Package. The framework serves as a unifying element, providing essential metrics that feed into the broader objectives of optimizing energy, carbon, and material footprints in data operations.

Fundamentally, the Performance Measurement Framework is not just a tool for data accumulation; it stands as a potent catalyst driving intelligent decision-making, enabling strategic resource allocation, and cultivating a steadfast commitment to excellence within the dynamic landscape of contemporary computing.

Fundamentally, the genesis of the Performance Measurement Framework was a deliberate endeavour, conceived as a tailored solution to address the intricacies of the current computing milieu. Its utility transcends the mere accumulation of data; rather, it functions as a potent catalyst, propelling intelligent decision-making, facilitating strategic allocation of resources, and fostering an unwavering commitment to excellence in the ever-evolving realm of dynamic computing.

Bibliography

[1]   VERTIV, "What is a Smart PDU?," 2023. [Online]. Available: https://www.vertiv.com/en-emea/about/news-and-insights/articles/educational-articles/what-is-a-smart-pdu/.

[2]   APC, "Rack Power Distribution Unit," 2023. [Online]. Available: https://download.schneider-electric.com/files?p_Doc_Ref=990-5570_EN&p_enDocType=User+guide&p_File_Name=990-5570O_Rack+PDU_MBOS.pdf.

[3]   Dell Technologies, "Getting Started with Integrated Dell Remote Access Controller," 2023. [Online]. Available: https://infohub.delltechnologies.com/p/getting-started-with-integrated-dell-remote-access-controller-idrac/.

[4]   Dell Technologies, "Integrated Dell Remote Access Controller," 2023. [Online]. Available: https://www.dell.com/en-us/lp/dt/open-manage-idrac.

[5]   D. Technologies, "iDRAC9 User's Guide - Monitoring Power," 2023. [Online]. Available: https://www.dell.com/support/manuals/en-us/idrac9-lifecycle-controller-v3.0-series/idrac_3.00.00.00_ug/monitoring-power?guid=guid-17c21f6e-db64-4bba-9977-3b335f822c41&lang=en-us.

[6]   Dell Technologies, "Transform Datacenter Analytics with iDRAC9 Telemetry Streaming," 2023. [Online]. Available: https://www.delltechnologies.com/asset/en-au/products/unified-workspace/industry-market/direct-from-development-datacenter-telemetry.pdf.

[7]   Grafana, "Grafana Labs," 2023. [Online]. Available: https://grafana.com/grafana.

[8]   Grafana, "What is Grafana," 2022. [Online]. Available: https://www.redhat.com/en/topics/data-services/what-is-grafana.

[9]   Prometheus, "Prometheus Overview," 2023. [Online]. Available: https://prometheus.io/docs/introduction/overview.

[10] Prometheus, "Prometheus Getting Started.," 2023. [Online]. Available: https://prometheus.io/docs/prometheus/latest/getting_started/.

[11] Prometheus, "Prometheus: Key Features, Components & Use Cases of the Open Source Monitoring Tool.," 2023. [Online]. Available: https://www.splunk.com/en_us/blog/learn/prometheus.html.

[12] Prometheus, "EXPORTERS AND INTEGRATIONS.," 2023. [Online]. Available: https://prometheus.io/docs/instrumenting/exporters/.

[13] Solarwinds, "Management Information Base (MIB) in the SolarWinds Platform.," 2023. [Online]. Available:

https://documentation.solarwinds.com/en/success_center/orionplatform/content/core-management-information-base--mib--sw1730.htm.

[14] VMware, "VMware MIB Files.," [Online]. Available: https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.monitoring.doc/GUID-A7CB813A-EE36-4408-8935-A299C967AB15.html.

[15] Paessler, "IT Explained: SNMP," 2023. [Online]. Available: https://www.paessler.com/it-explained/snmp.

[16] Paessler, "PRTG Manual: Monitoring via SNMP," 2023. [Online]. Available: https://www.paessler.com/manuals/prtg/monitoring_via_snmp.

[17] M. R. Hansen, "idrac_exporter," 2023. [Online]. Available: https://github.com/mrlhansen/idrac_exporter.

[18] OpsRamp, "Guide to The Prometheus Node Exporter," 2023. [Online]. Available: https://www.opsramp.com/guides/prometheus-monitoring/prometheus-node-exporter/.

[19] Github, "Prometheus Node exporter github page," 2023. [Online]. Available: https://github.com/prometheus/node_exporter.

[20] Prometheus, "Monitoring Linux host metrics with the Node Exporter," 2023. [Online]. Available: https://prometheus.io/docs/guides/node-exporter/.

[21] Redhat, "What is Grafana?," 2022. [Online]. Available: https://www.redhat.com/en/topics/data-services/what-is-grafana.