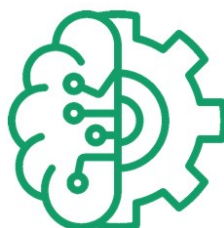


BRAINE



BRAINE - Big data Processing and Artificial Intelligence at the Network Edge

| | |
|--------------------------|---|
| Project Title: | BRAINE - Big data Processing and Artificial Intelligence at the Network Edge |
| Contract No: | 876967 – BRAINE |
| Instrument: | ECSEL Research and Innovation Action |
| Call: | H2020-ECSEL-2019-2-RIA |
| Start of project: | 1 May 2020 |
| Duration: | 42 months |

Deliverable D5.5

BRAINE test and validation Cycle 2

| | |
|---------------------------------|------------------|
| Due date of deliverable: | 30 November 2023 |
| Actual submission date: | 15 December 2023 |
| Version: | 1.0 |



Project funded by the European Community under the
H2020 Programme for Research and Innovation.



| | |
|----------------------------|--|
| Project ref. number | 876967 |
| Project title | BRAINE - Big data Processing and Artificial Intelligence at the Network Edge |

| | |
|-------------------------------------|--|
| Deliverable title | BRAINE test and validation Cycle 2 |
| Deliverable number | D5.5 |
| Deliverable version | Version 1.0 |
| Previous version(s) | - |
| Contractual date of delivery | 30 November 2023 |
| Actual date of delivery | 15 December 2023 |
| Deliverable filename | BRAINE_D5.5_Final |
| Nature of deliverable | Report |
| Dissemination level | PU |
| Number of pages | 82 |
| Work package | WP5 |
| Task(s) | T5.3, T5.5 |
| Partner responsible | DELL |
| Author(s) | Filippo Cugini (CNIT) Alessio Giorgetti (CNIT), Antonino Albanese (ITL), Patrick Moder (IFX), Sean Ahearne (DELL), Ahmed Khalid (DELL), Mustafa Al-Bado (DELL), Hemant Mehta (UCC), Javad Chamanara (LUH), Luca Valcarenghi (SSSA), Alessandro Pacini (SSSA), Justine Cris Borromeo (SSSA), Vojtěch Janů (CTU), JJ Vegas Olmos (MLNX), Koen Ouweltjes (HID), Nour Ramzy (IFX), Hans Ehm (IFX), Benat Alkorta (IFX), Federico Civerchia (SMA), Luca Maggiani (SMA), Philippe Nguyen (SIC), Elena Petrova (IMC), Janos Lazany (PCB), Remy Haynau (JJC), Gautier Rouaze (JJC), Sana Fateh (SYN), Thomas Gerner Nørgaard (COM), Bruno Cimoli (TUE), Simon Rommel (TUE), Adam Flizikowski (ISW), Md Munjure Mowla (ISW), Evgeniy Alkhovik (ISW), Péter Szántó (BME), Roberto Bifulco (NEC), Radoslav Gerganov (VMW), Edgard Marx (ECC), Tomi Niittumäki (MAI), Martin Ron (FS). |
| Editor | Sean Ahearne (DELL) |

| | |
|-----------------|--|
| Abstract | Report on the validation of the BRAINE 2.0 integrated solution |
| Keywords | HW components, SW components, testing, integration. |

Copyright

© Copyright 2020 BRAINE Consortium

This document may not be copied, reproduced, or modified in whole or in part for any purpose without written permission from the BRAINE Consortium. In addition to such written permission to copy, reproduce, or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

All rights reserved.

Deliverable history

| Version | Date | Reason | Revised by |
|---------|------------|------------------------|--------------|
| 0.1 | 04/09/2023 | ToC | Sean Ahearne |
| 0.2 | 20/10/2023 | HW Performance Report | Sean Ahearne |
| 0.3 | 24/11/2023 | Component Annex Update | All Authors |
| 0.4 | 11/12/2023 | Platform Guide | Sean Ahearne |
| 0.5 | 14/12/2023 | Final Version | Sean Ahearne |

List of abbreviations and Acronyms

| Abbreviation | Meaning |
|--------------|---|
| 5G | 5 th Generation |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| ASIC | Application Specific Integrated Circuit |
| BMC | Board Management Controller |
| CPU | Central Processing Unit |
| EU | European Union |
| FPGA | Field Programmable Gate Arrays |
| GDPR | General Data Protection Regulation |
| GPU | Graphics Processing Unit |
| IoT | Internet of Things |
| IP | Internet Protocol |
| IT | Information Technology |
| K8s | Kubernetes |
| KPI | Key Performance Indicator |
| OS | Operating System |
| PCB | Printed Circuit Board |
| SDN | Software Defined Networking |
| TCP | Transmission Control Protocol |
| MAAS | Metal as a Service |
| HA | High Availability |
| EMDC | Edge Micro Datacenter |
| DHCP | Dynamic Host Control Protocol |
| VLAN | Virtual Local Area Network |
| IPMI | Intelligent Platform Management Interface |
| PXE | Preboot eXecution Environment |
| SSH | Secure SHell |
| RBAC | Role Based Access Control |
| DNS | Domain Name System |

Table of Contents

| | | |
|-------|--|----|
| 1. | Introduction..... | 8 |
| 2. | BRAINE EMDC Performance Analysis | 9 |
| 3. | BRAINE Platform User Guide | 19 |
| 3.1 | Pre-requisites..... | 19 |
| 3.2 | Step 1: Setup and Installation | 19 |
| 3.3 | Step 2: Network Configuration and Disk Wiping | 20 |
| 3.4 | Step 3: Setting up MAAS..... | 25 |
| 3.5 | Step 4: Provisioning Nodes | 29 |
| 3.6 | Step 5: Setting Up Rancher..... | 35 |
| 3.7 | Step 6: Deploying Applications and Further Reading | 39 |
| 3.7.1 | Production Use | 40 |
| 4. | Conclusions..... | 43 |
| 5. | Appendix A – Component List | 44 |
| 5.1 | Platform components | 44 |
| 5.1.1 | WP2..... | 44 |
| 5.1.2 | WP3..... | 54 |
| 5.1.3 | WP4..... | 62 |
| 6. | Appendix B – Component Report | 71 |
| 6.1 | WP2..... | 71 |
| 6.2 | WP3..... | 75 |
| 6.3 | WP4..... | 79 |

List of Figures

| | | |
|-------------|---|----|
| Figure 3.1 | BRAINE Auto-install menu when booting from ISO/USB | 20 |
| Figure 3.2 | Network interface menu..... | 21 |
| Figure 3.3 | Network configuration..... | 22 |
| Figure 3.4 | Successful network connection established..... | 22 |
| Figure 3.5 | Confirm disk wiping for OS installation. | 23 |
| Figure 3.6 | Installation Complete | 24 |
| Figure 3.7 | Installation fully complete login screen | 25 |
| Figure 3.8 | MAAS Login | 26 |
| Figure 3.9 | MAAS Initial setup screen | 26 |
| Figure 3.10 | MAAS OS Image Selection..... | 27 |
| Figure 3.11 | MAAS Initial Setup Complete | 27 |
| Figure 3.12 | Subnet network selection | 28 |
| Figure 3.13 | MAAS IP Reservation..... | 28 |
| Figure 3.14 | VLAN DHCP Configuration..... | 29 |
| Figure 3.15 | Example of a simple network which can be used by MAAS | 30 |
| Figure 3.16 | Example of a boot menu showing network boot option | 30 |
| Figure 3.17 | MAAS detecting nodes | 31 |

| | |
|---|----|
| Figure 3.18 Node Power Configuration..... | 31 |
| Figure 3.19 Commissioning the nodes | 32 |
| Figure 3.20 MAAS Commissioning in Progress | 33 |
| Figure 3.21 Getting K8s token from head node | 33 |
| Figure 3.22 Deploying to nodes with cloud-init | 34 |
| Figure 3.23 Rancher bootstrap menu | 35 |
| Figure 3.24 Rancher K8s cluster dashboard | 36 |
| Figure 3.25 Rancher Applications Menu..... | 37 |
| Figure 3.26 Longhorn Installation | 37 |
| Figure 3.27 Longhorn Dashboard..... | 38 |
| Figure 3.28 Monitoring Installation..... | 38 |
| Figure 3.29 Grafana Dashboard | 39 |
| Figure 3.30 Pod Deployment Menu | 39 |
| Figure 3.31 Resources Section | 40 |
| Figure 4.1 BRAINE Public Release Reference Architecture | 43 |

List of Tables

| | |
|--|----|
| Table 1 EMDC CPU Node Power Consumption Information | 10 |
| Table 2 CPU Comparison Information..... | 10 |
| Table 3 EMDC CPU Nodes KPI Comparison | 10 |
| Table 4 Score-based performance results | 11 |
| Table 5 Comparison graph of score-based performance results..... | 12 |
| Table 6 Relative percentage performance table. | 13 |
| Table 7 Relative percentage performance graph..... | 13 |
| Table 8 4x ARM node estimated performance..... | 14 |
| Table 9 4x ARM node relative performance graph | 14 |
| Table 10 Score-based performance per watt graph..... | 15 |
| Table 11 Relative percentage performance per Watt table | 15 |
| Table 12 Relative percentage performance graph..... | 16 |
| Table 13 Power consumption and performance comparison with modern server | 17 |
| Table 14 Performance per watt EMDC vs server comparison table | 17 |
| Table 15 Performance per Watt EMDC vs Server graph | 18 |

1. Introduction

This document reports the final integration, testing, and performance results from the completed BRAINE platform and EMDC. It also included a user guide for the final public release of the platform and an updated component final report annex (component description is also included for completeness). Note that this document does not contain the final reports or updates for any use case as each use case was given its own final report:

- D5.6 – Use Case 1 Smart Hospital
- D5.7 – Use Case 2 Smart City
- D5.8 – Use Case 3 Smart Factory
- D5.9 – Use Case 4 industry 4.0.

Refer to the above deliverables for the updates and final reports related to the use cases. D5.5 instead contains:

- Performance report of BRAINE EMDC for real-world workloads
- User guide for deployment and administration of the platform.
- Component Annex

2. BRAINE EMDC Performance Analysis

In the realm of computing, especially in areas involving diverse CPU architectures, performance evaluation stands as a critical exercise. It is essential to understand not just how well a system performs under various workloads, but also how efficiently it uses power. This is particularly true in environments where energy efficiency and processing power are equally important.

The Importance of Performance Testing

Performance testing is indispensable for several reasons. Firstly, it provides quantitative data on the capability of different CPUs, offering insights into how well they can handle specific tasks. In the context of Edge Computing, where resources are often limited, knowing the capability of each component is crucial. Secondly, performance testing helps in identifying bottlenecks and inefficiencies, guiding future improvements and optimizations.

Geekbench: A Tool for Performance Benchmarking

Geekbench is a renowned benchmarking tool used to evaluate the performance of CPUs across different architectures. It operates by running a series of tests that mimic real-world scenarios and applications. These tests measure the processing power of the CPU. They include tasks like encryption, compression, and rendering, which are common in everyday computing tasks. Geekbench provides scores that make it easy to compare different CPUs on a uniform scale. Higher scores indicate better performance.

Upcoming Performance Tests and Measuring Performance per Watt

The forthcoming performance tests will utilize Geekbench to evaluate various CPU boards. The focus will be not only on the raw performance but also on the efficiency of power usage - performance per watt. This metric is increasingly important in modern computing, where energy efficiency can be as crucial as computational speed, especially in edge computing scenarios.

Results Presentation: KPIs, Score-Based Evaluations, and Comparative Tables

The results from these tests will be presented in several formats to provide a comprehensive understanding of the CPUs' performance:

1. **Key Performance Indicators (KPIs):** These will highlight critical aspects of CPU performance, such as processing speed and energy efficiency.
2. **Score-Based Evaluations:** Geekbench scores will be used to provide a standardized comparison of the CPUs' capabilities.
3. **Relative Comparison Tables:** These tables will offer a side-by-side comparison of the different CPUs, making it easier to see how they stack up against each other in various aspects.

The importance of this performance evaluation using Geekbench lies in its ability to provide clear, quantifiable metrics on CPU performance and efficiency. This will be instrumental in guiding the project's decisions regarding CPU selection and optimization, ensuring that the best possible balance between power and efficiency is achieved.

Before measurement of performance results, an average power consumption baseline was measured from each CPU type during execution. The result of this measurement is showing in Table 1 below.

| ARM Power (W) | AGX Power (W) | Intel Power (W) | AMD Power (W) | Dell R7625 Power |
|---------------|---------------|-----------------|---------------|------------------|
| 35 | 32.5 | 55 | 90 | 837 |

Table 1 EMDC CPU Node Power Consumption Information

Note that the inclusion of a new Dell R7625 server with the latest generation AMD Epyc CPUs was included as a comparison point, which will be referenced in later comparisons. Table 2 below shows a comparison of the CPU architecture, core count, and TDP.

| CPU Name | Core Count | Architecture | TDP |
|-------------------|------------|------------------------|---------|
| Intel Xeon D-1548 | 8 Cores | Broadwell DE (x86) | 35-45W |
| AMD EPYC 3451 | 16 Cores | Zen (Snowy Owl) (x86) | 80-100W |
| SolidRun LX2162A | 16 Cores | ARM Cortex-A72 | ~35W |
| Nvidia Jetson-AGX | 8 Cores | Nvidia Carmel ARM v8.2 | ~35W |

Table 2 CPU Comparison Information

The first performance result based on KPI's is shown in Table 3 Below

| Benchmark | ARM Board | Nvidia AGX | Intel Board | AMD board |
|-------------------------------------|-----------|------------|-------------|-----------|
| AES-XTS (GB/sec) | 2.88 | 14 | 4.99 | 5.44 |
| Text Compression (MB/sec) | 15.8 | 18.7 | 30.1 | 52.1 |
| Image Compression (Mpixels/sec) | 270.6 | 239.4 | 266.2 | 560.2 |
| Navigation (MTE/sec) | 3.68 | 11.5 | 6.13 | 10.4 |
| HTML5(MElements/sec) | 5.85 | 4.91 | 5.42 | 11.7 |
| SQLite (Mrows/sec) | 1.46 | 1.17 | 1.61 | 3.51 |
| PDF Rendering (Mpixels/sec) | 271.1 | 282 | 253.2 | 494.7 |
| Text Rendering (MB/sec) | 911.9KB | 1.23MB | 1.08MB | 1.63MB |
| Clang (Klines/sec) | 37.6 | 19.4 | 45 | 93.2 |
| Camera(images/sec) | 36.3 | 50.9 | 47.9 | 98.8 |
| N-Body Physics (Mpairs/sec) | 3.2 | 2.45 | 5.55 | 8.07 |
| Rigid Body Physics (FPS) | 39630.4 | 26440 | 40656 | 97006.1 |
| Gaussian Blur (Mpixels/sec) | 182.1 | 231 | 261.9 | 520.6 |
| Face Detection(images/sec) | 41.2 | 26.9 | 42.8 | 97.5 |
| Horizon Detection (Mpixels/sec) | 81.1 | 154.9 | 116.4 | 222.1 |
| Image Inpainting (Mpixels/sec) | 181.1 | 411.4 | 320.4 | 613.4 |
| HDR(Mpixels/sec) | 100.6 | 118.8 | 124.5 | 249.4 |
| Ray Tracing (Mpixels/sec) | 4.96 | 3.27 | 6.14 | 12.4 |
| Structure from Motion (Kpixels/sec) | 37.3 | 42.5 | 41.9 | 87.9 |
| Speech Recognition (Words/sec) | 30.1 | 96.4 | 77.1 | 134.9 |
| Machine Learning(images/sec) | 31.5 | 87.3 | 49.9 | 71.5 |

Table 3 EMDC CPU Nodes KPI Comparison

The comparative analysis of CPU boards within our EMDC using Geekbench benchmarks has yielded insightful data on the performance dynamics of ARM, Nvidia AGX, Intel, and AMD boards. The Nvidia AGX board's encryption performance, as indicated by AES-XTS,

is notably superior, emphasizing its robustness for security-centric applications. The Nvidia AGX nodes supports HW acceleration of AES-XTS, with that advantage clearly shown. The AMD board's significant lead in text and image compression benchmarks indicates a potential preference for data-intensive tasks. Furthermore, it demonstrates exceptional performance in HTML5 and SQLite processing, suggesting an advantage for web-based services and database management. In rendering tasks, such as PDF rendering, and computationally intensive simulations, including rigid body physics, the AMD board again shows a distinct performance edge. This would likely translate to enhanced capabilities for real-time data visualization and complex simulations. This however comes at the cost of a relatively high CPU power consumption.

In speech recognition and machine learning tasks, the Nvidia AGX and AMD boards show impressive results, with the Nvidia AGX board showing exceptional speech processing capabilities. Such performance highlights the suitability of these boards for AI-driven applications at the edge. This detailed performance data, coupled with energy efficiency measurements, will inform the selection and optimization of CPU boards for deployment in the EMDC, aligning with our project's objectives to maximize computational efficiency and energy utilization.

Geekbench also normalizes these results into a score-based system, with results shown in Table 4 below.

| Benchmark | ARM Board | Nvidia AGX | Intel Board | AMD board |
|-----------------------|-----------|------------|-------------|-----------|
| AES-XTS | 1691 | 8206 | 2925 | 3189 |
| Text Compression | 3124 | 3690 | 5945 | 10306 |
| Image Compression | 5720 | 5060 | 5627 | 11841 |
| Navigation | 1306 | 4071 | 2172 | 3693 |
| HTML5 | 4979 | 4179 | 4614 | 9990 |
| SQLite | 4667 | 3747 | 5137 | 11215 |
| PDF Rendering | 4995 | 5196 | 4665 | 9116 |
| Text Rendering | 2862 | 3947 | 3464 | 5249 |
| Clang | 4825 | 2495 | 5775 | 11961 |
| Camera | 3133 | 4388 | 4134 | 8523 |
| N-Body Physics | 2557 | 1957 | 4434 | 6454 |
| Rigid Body Physics | 6397 | 4268 | 6562 | 15658 |
| Gaussian Blur | 3313 | 4202 | 4765 | 9472 |
| Face Detection | 5345 | 3492 | 5564 | 12662 |
| Horizon Detection | 3291 | 6286 | 4721 | 9010 |
| Image Inpainting | 3691 | 8386 | 6532 | 12503 |
| HDR | 7385 | 8721 | 9134 | 18299 |
| Ray Tracing | 6181 | 4071 | 7642 | 15493 |
| Structure from Motion | 4167 | 4738 | 4676 | 9812 |
| Speech Recognition | 942 | 3017 | 2410 | 4219 |
| Machine Learning | 814 | 2259 | 1292 | 1851 |

Table 4 Score-based performance results

This produces a more visually legible performance table, shown in Table 5 below.

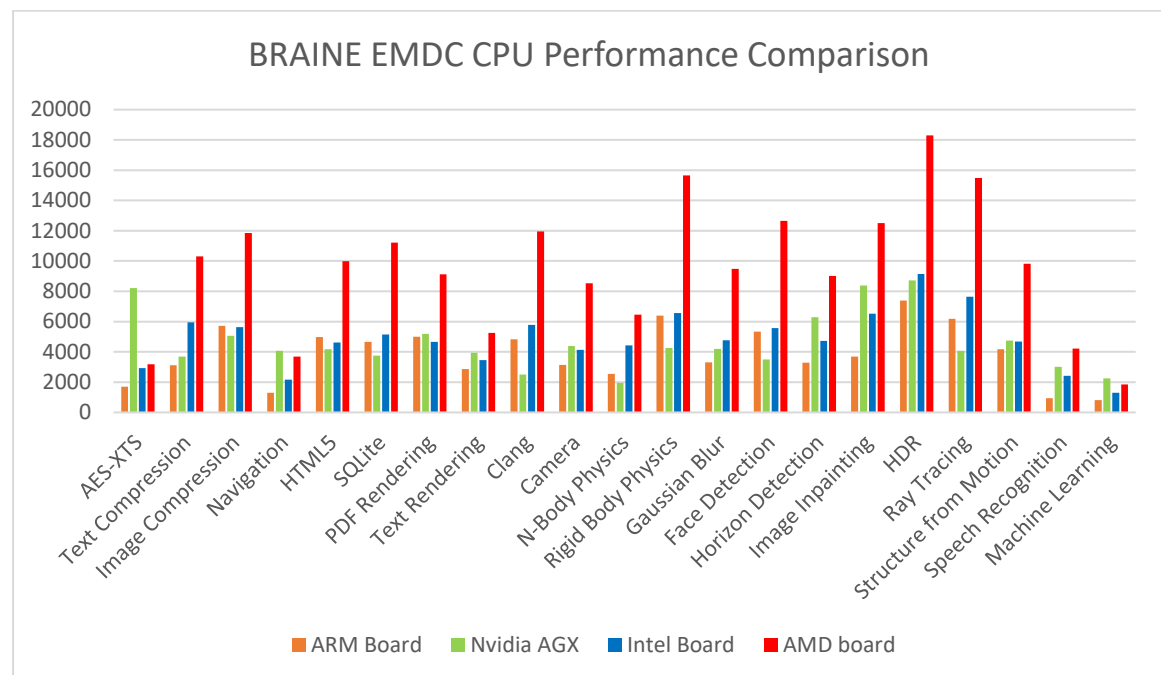
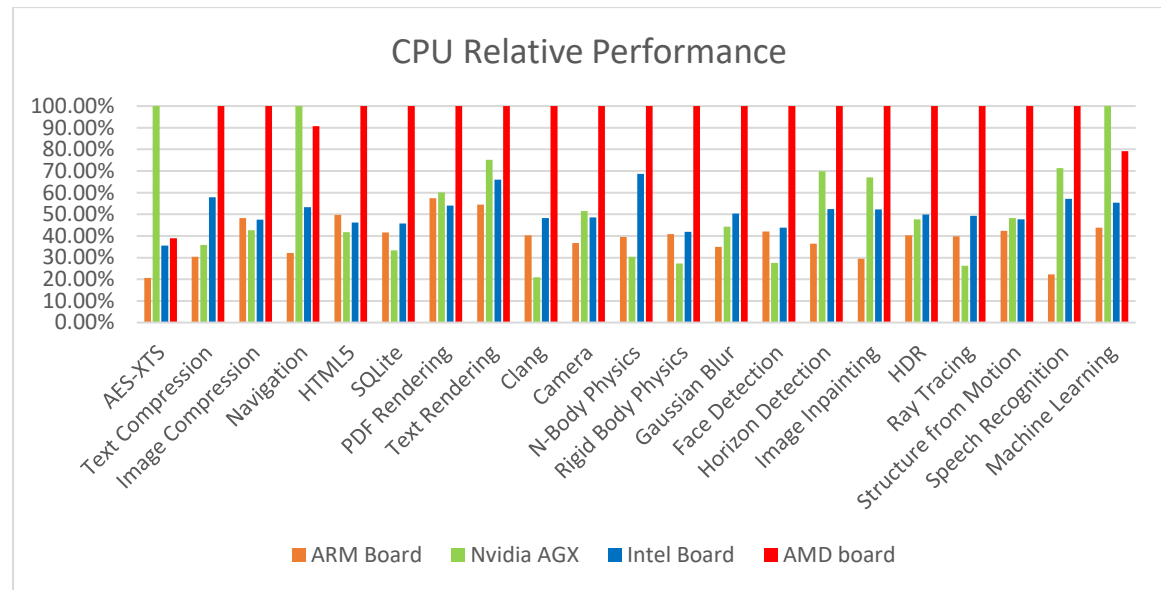


Table 5 Comparison graph of score-based performance results.

Tables 6 and 7 also show a relative percentage performance comparison, to more clearly show the difference between CPU's.

| Benchmark | ARM Board | Nvidia AGX | Intel Board | AMD board |
|-----------------------|-----------|------------|-------------|-----------|
| AES-XTS | 20.60% | 100% | 35.60% | 38.90% |
| Text Compression | 30.30% | 35.80% | 57.90% | 100% |
| Image Compression | 48.30% | 42.70% | 47.50% | 100% |
| Navigation | 32.10% | 100% | 53.30% | 90.70% |
| HTML5 | 49.80% | 41.80% | 46.20% | 100% |
| SQLite | 41.60% | 33.40% | 45.80% | 100% |
| PDF Rendering | 57.50% | 60.10% | 54.00% | 100% |
| Text Rendering | 54.50% | 75.20% | 66.00% | 100% |
| Clang | 40.30% | 20.90% | 48.30% | 100% |
| Camera | 36.80% | 51.50% | 48.50% | 100% |
| N-Body Physics | 39.60% | 30.30% | 68.60% | 100% |
| Rigid Body Physics | 40.90% | 27.20% | 41.90% | 100% |
| Gaussian Blur | 34.90% | 44.30% | 50.30% | 100% |
| Face Detection | 42.10% | 27.60% | 43.90% | 100% |
| Horizon Detection | 36.50% | 69.80% | 52.40% | 100% |
| Image Inpainting | 29.50% | 67.10% | 52.30% | 100% |
| HDR | 40.30% | 47.60% | 49.90% | 100% |
| Ray Tracing | 39.90% | 26.30% | 49.30% | 100% |
| Structure from Motion | 42.40% | 48.30% | 47.60% | 100% |
| Speech Recognition | 22.30% | 71.40% | 57.10% | 100% |

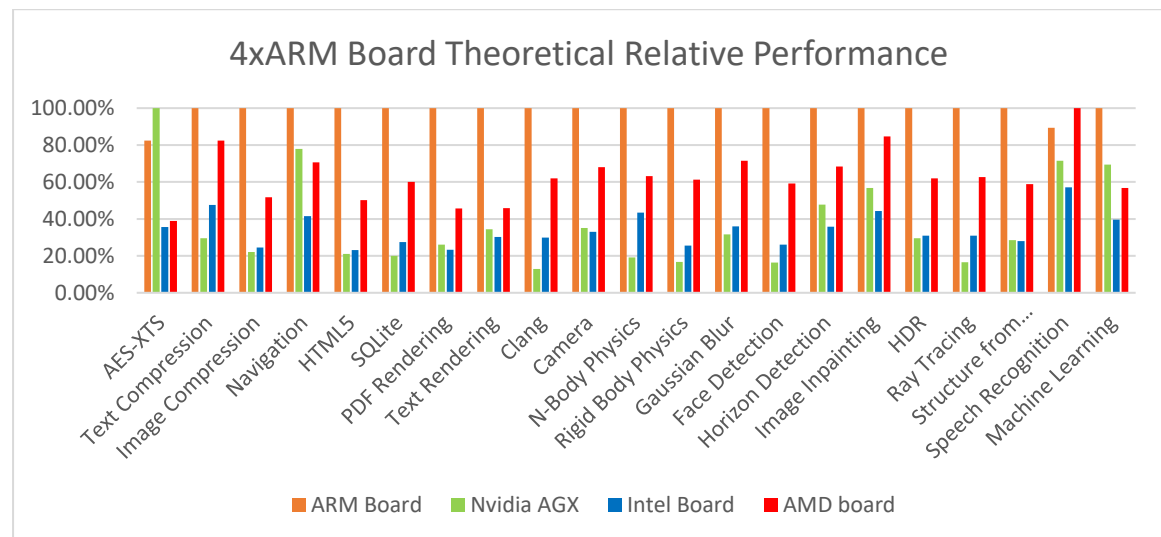
| | | | | |
|------------------|--------|------|--------|--------|
| Machine Learning | 43.90% | 100% | 55.30% | 79.20% |
|------------------|--------|------|--------|--------|

Table 6 Relative percentage performance table.**Table 7 Relative percentage performance graph**

It is worth noting that the ARM board consists of 4x16 core modules in a single EMDC slot. Thus, we can calculate the estimated performance of all 64 cores running compared to the other CPU solutions that require a full slot each. This comparison can be seen in Tables 8 and 9 below. Note that this was tested and resulted in a slot power consumption of 150W-160W, as shown in deliverable D2.4

| Benchmark | ARM Board | Nvidia AGX | Intel Board | AMD board |
|-----------------------|-----------|------------|-------------|-----------|
| AES-XTS | 82.43% | 100% | 35.64% | 38.86% |
| Text Compression | 100% | 29.53% | 47.58% | 82.47% |
| Image Compression | 100% | 22.12% | 24.59% | 51.75% |
| Navigation | 100% | 77.93% | 41.58% | 70.69% |
| HTML5 | 100% | 20.98% | 23.17% | 50.16% |
| SQLite | 100% | 20.07% | 27.52% | 60.08% |
| PDF Rendering | 100% | 26.01% | 23.35% | 45.63% |
| Text Rendering | 100% | 34.48% | 30.26% | 45.85% |
| Clang | 100% | 12.93% | 29.92% | 61.97% |
| Camera | 100% | 35.01% | 32.99% | 68.01% |
| N-Body Physics | 100% | 19.13% | 43.35% | 63.10% |
| Rigid Body Physics | 100% | 16.68% | 25.64% | 61.19% |
| Gaussian Blur | 100% | 31.71% | 35.96% | 71.48% |
| Face Detection | 100% | 16.33% | 26.02% | 59.22% |
| Horizon Detection | 100% | 47.75% | 35.86% | 68.44% |
| Image Inpainting | 100% | 56.80% | 44.24% | 84.69% |
| HDR | 100% | 29.52% | 30.92% | 61.95% |
| Ray Tracing | 100% | 16.47% | 30.91% | 62.66% |
| Structure from Motion | 100% | 28.43% | 28.05% | 58.87% |

| | | | | |
|--------------------|--------|--------|--------|--------|
| Speech Recognition | 89.31% | 71.51% | 57.12% | 100% |
| Machine Learning | 100% | 69.38% | 39.68% | 56.85% |

Table 8 4x ARM node estimated performance.**Table 9 4x ARM node relative performance graph**

For a performance per Watt comparison, the score-based system is calculated and visualized in Tables 9 and 10 below.

| Benchmark | ARM Board | Nvidia AGX | Intel Board | AMD board |
|-----------------------|-----------|------------|-------------|-----------|
| AES-XTS | 48.31 | 252.49 | 53.18 | 35.43 |
| Text Compression | 89.26 | 113.54 | 108.09 | 114.51 |
| Image Compression | 163.43 | 155.69 | 102.31 | 131.57 |
| Navigation | 37.31 | 125.26 | 39.49 | 41.03 |
| HTML5 | 142.26 | 128.58 | 90.25 | 111 |
| SQLite | 133.34 | 115.29 | 93.4 | 124.61 |
| PDF Rendering | 142.71 | 159.88 | 84.82 | 101.29 |
| Text Rendering | 81.77 | 121.45 | 62.98 | 58.32 |
| Clang | 137.86 | 76.77 | 105 | 132.9 |
| Camera | 89.51 | 135.02 | 75.16 | 94.7 |
| N-Body Physics | 73.06 | 60.22 | 80.62 | 71.71 |
| Rigid Body Physics | 182.77 | 131.32 | 119.31 | 173.98 |
| Gaussian Blur | 94.66 | 129.29 | 86.64 | 105.24 |
| Face Detection | 152.71 | 107.45 | 101.16 | 140.69 |
| Horizon Detection | 94.03 | 193.42 | 85.84 | 100.11 |
| Image Inpainting | 105.46 | 258.03 | 118.76 | 138.92 |
| HDR | 211 | 268.34 | 166.07 | 203.32 |
| Ray Tracing | 176.6 | 125.26 | 138.95 | 172.15 |
| Structure from Motion | 119.06 | 145.78 | 85.01 | 109.02 |
| Speech Recognition | 26.91 | 92.83 | 43.82 | 46.88 |
| Machine Learning | 23.26 | 69.51 | 23.49 | 20.57 |

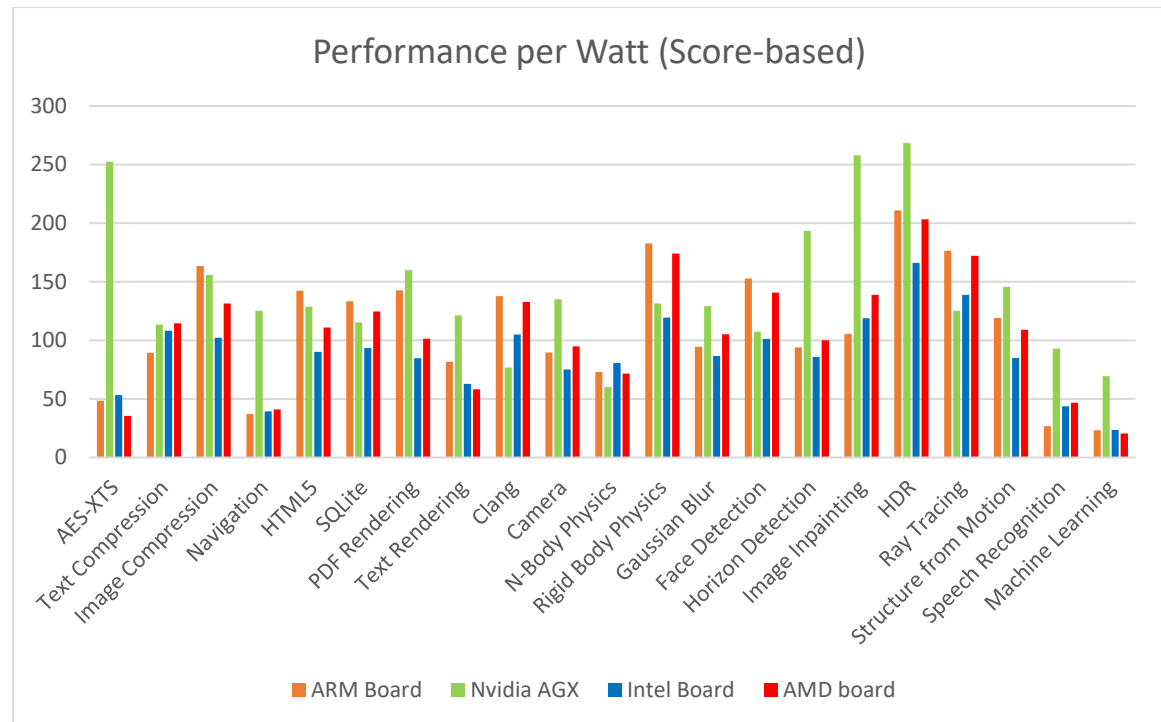


Table 10 Score-based performance per watt graph

Tables 11 and 12 provide the relative percentage measurement.

| Benchmark | ARM Board (%) | Nvidia AGX (%) | Intel Board (%) | AMD board (%) |
|-----------------------|---------------|----------------|-----------------|---------------|
| AES-XTS | 19.03 | 100 | 21.11 | 13.92 |
| Text Compression | 77.89 | 99.31 | 94.47 | 100 |
| Image Compression | 100 | 95.28 | 62.61 | 80.51 |
| Navigation | 29.66 | 100 | 31.36 | 32.77 |
| HTML5 | 100 | 90.42 | 59.28 | 77.84 |
| SQLite | 100 | 85.71 | 69.05 | 92.86 |
| PDF Rendering | 89.27 | 100 | 53.06 | 63.35 |
| Text Rendering | 68.42 | 100 | 52.63 | 47.37 |
| Clang | 100 | 55.59 | 76.16 | 96.46 |
| Camera | 66.22 | 100 | 55.62 | 70.11 |
| N-Body Physics | 90.1 | 74.26 | 100 | 89.11 |
| Rigid Body Physics | 100 | 71.85 | 65.28 | 95.19 |
| Gaussian Blur | 73.2 | 100 | 66.99 | 81.37 |
| Face Detection | 100 | 70.35 | 66.1 | 92.01 |
| Horizon Detection | 48.62 | 100 | 44.4 | 51.78 |
| Image Inpainting | 40.88 | 100 | 46.02 | 53.85 |
| HDR | 78.63 | 100 | 61.94 | 75.81 |
| Ray Tracing | 100 | 71.13 | 78.87 | 97.18 |
| Structure from Motion | 81.5 | 100 | 58.26 | 74.69 |
| Speech Recognition | 29 | 100 | 47.27 | 50.54 |
| Machine Learning | 33.51 | 100 | 33.77 | 29.56 |

Table 11 Relative percentage performance per Watt table

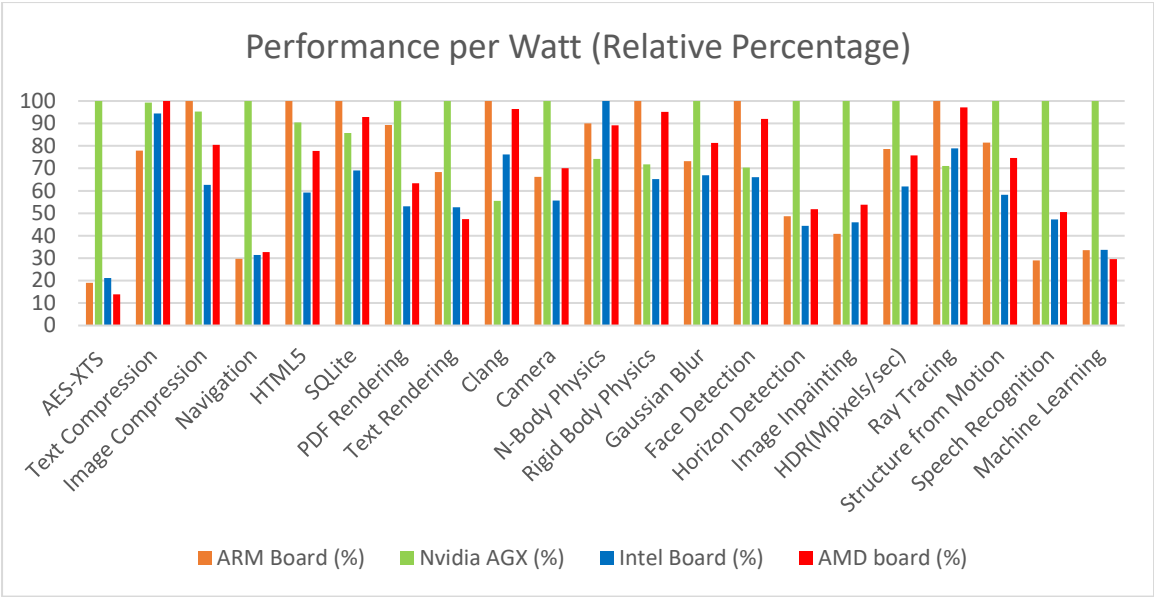


Table 12 Relative percentage performance graph

The efficiency of CPU boards across various benchmarks has been evaluated to ensure the optimization of energy consumption in relation to computational performance. The following observations have been noted and should be taken into consideration when selecting CPU boards for deployment within an EMDC for a specific use case.

The Nvidia AGX board demonstrates exceptional energy efficiency, achieving peak performance per watt ratings in numerous benchmarks including, but not limited to, Navigation, PDF Rendering, and Camera operation tasks. This exemplary performance indicates that Nvidia AGX is particularly adept at maximizing computational throughput while minimizing energy expenditure, an essential characteristic for sustained operations in power-conscious edge computing environments.

The AMD board presents a balanced profile of energy efficiency, excelling in Text Compression where it leads the table with the highest efficiency rating. Consistently high performance is also observed in the SQLite and Rigid Body Physics benchmarks, suggesting that the AMD board is a robust all-rounder capable of handling a diverse range of tasks without excessive power consumption.

The ARM Board achieves top efficiency in several key areas including Image Compression, HTML5 processing, and Face Detection. These results highlight the ARM Board as a potential frontrunner for scenarios where these tasks are prevalent, and power efficiency is a critical factor.

The Intel Board, while not consistently leading in efficiency, does show particular strength in the N-Body Physics benchmark. This specific efficiency indicates that the Intel Board may be best suited for specialized applications that require high computational precision and complex mathematical processing.

The presented performance per watt metrics offer invaluable insights into the operational efficiencies of the various CPU boards. The data suggests that while Nvidia AGX stands out as a general leader in efficiency, the NXP ARM Board's strong performance in many other tasks should not be underestimated. An EMDC consisting of 8x64 core ARM boards would exhibit both very high performance and power efficiency in many tasks.

The AMD board's well-rounded performance indicates its suitability for a wide array of applications, particularly given its x86 architecture.

For a final comparison, the overall system performance was compared to a modern Dell PowerEdge R7625 air cooled server, with 2x of the latest generation AMD Epyc 32-core processors. The total performance and system power consumption was measured monitored throughout the test with the results shown in Table 13

| Benchmark | Score | Power(W) | Cooling |
|-------------|-------|----------|---------|
| Dell R7625 | 57499 | 837 | 107 |
| BRAINE EMDC | 48064 | 552 | 8 |

Table 13 Power consumption and performance comparison with modern server

It can be seen here that the EMDC can still provide competitive efficiency performance despite less efficient CPU architectures which are now many years behind the state of the art. In this comparison it can be seen to be offering 83.5% of the performance for 63.5% of the power consumption. Of particular note is the difference in cooling system energy use, with a 10x reduction in total cooling system power reduction compared to traditional air cooling. For a closer comparison, the power draw of GPUs in the system was removed, and a score-based comparison table and graph created as shown in Tables 14 and 15 below.

| Task | ARM Board | Nvidia AGX | Intel Board | AMD board | Dell R7625 |
|-----------------------|------------|------------|-------------|-----------|------------|
| AES-XTS | 48.3142857 | 252.49 | 53.182 | 35.4333 | 16.7 |
| Text Compression | 89.2571429 | 113.54 | 108.09 | 114.511 | 113.25254 |
| Image Compression | 163.428571 | 155.69 | 102.31 | 131.567 | 131.36949 |
| Navigation | 37.3142857 | 125.26 | 39.491 | 41.0333 | 63.498305 |
| HTML5 | 142.257143 | 128.58 | 83.891 | 111 | 113.55763 |
| SQLite | 133.342857 | 115.29 | 93.4 | 124.611 | 130.95254 |
| PDF Rendering | 142.714286 | 159.88 | 84.818 | 101.289 | 114.0339 |
| Text Rendering | 81.7714286 | 121.45 | 62.982 | 58.3222 | 84.681356 |
| Clang | 137.857143 | 76.769 | 105 | 132.9 | 137.69492 |
| Camera | 89.5142857 | 135.02 | 75.164 | 94.7 | 90.779661 |
| N-Body Physics | 73.0571429 | 60.215 | 80.618 | 71.7111 | 20.166102 |
| Rigid Body Physics | 182.771429 | 131.32 | 119.31 | 173.978 | 147.98983 |
| Gaussian Blur | 94.6571429 | 129.29 | 86.636 | 105.244 | 51.672881 |
| Face Detection | 152.714286 | 107.45 | 101.16 | 140.689 | 120.95932 |
| Horizon Detection | 94.0285714 | 193.42 | 85.836 | 100.111 | 130.04576 |
| Image Inpainting | 105.457143 | 258.03 | 118.76 | 138.922 | 165.84407 |
| HDR | 211 | 268.34 | 166.07 | 203.322 | 239.89492 |
| Ray Tracing | 176.6 | 125.26 | 138.95 | 172.144 | 77.159322 |
| Structure from Motion | 119.057143 | 145.78 | 85.018 | 109.022 | 116.32203 |
| Speech Recognition | 26.9142857 | 92.831 | 43.818 | 46.8778 | 84.181356 |
| Machine Learning | 23.2571429 | 69.508 | 23.491 | 20.5667 | 56.874576 |

Table 14 Performance per watt EMDC vs server comparison table

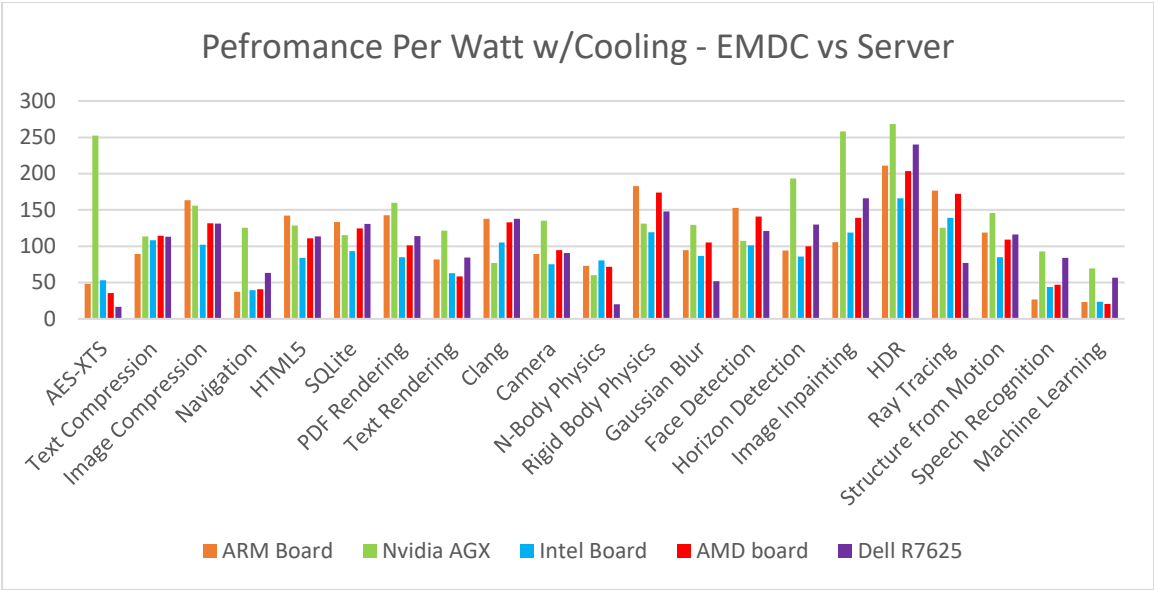


Table 15 Performance per Watt EMDC vs Server graph

It can be seen the the EMDC still remains competitive when adjusting for removal of GPU draw and other components, but including cooling requirements. The efficiency of the ARM CPU architecture can be seen here, still outperforming the latest generation AMD Epyc x86 CPU's in terms of efficiency in several tests despite their older ARM CPU architecture in 2023.

3. BRAINE Platform User Guide

Welcome to the BRAINE platform user guide. This guide is designed to simplify the deployment of a bare-metal cloud-native architecture at the edge, with as few pre-requisites and background knowledge required.

You can download the platform installer at https://gitlab.com/braine/braine_public_release

Look for **BRAINE-installer.iso**

Note: This user guide may continue to be updated within the GitLab documentation.

3.1 Pre-requisites

The BRAINE platform installer is designed to be as automated as possible, with the following noted pre-requisites:

Hardware

- **A single x86 CPU for the head node (any Intel or AMD processor)**
- **16GB+ RAM (32GB+ recommended)**
- **256GB+ of storage**

Note: While the system can run as a single node, it is recommended to have at least 3+ CPU nodes for High Availability (HA) (i.e. if one node fails the platform continues to function).

Note2: These system requirements only apply to the head node. The system requirements for other nodes can be as low as 4GB of RAM and 32GB of storage and can be ARM CPU's.

Network

- **Disable DHCP**
- **A functional internet gateway**

Dynamic Host Control Protocol (DHCP) is used to provide systems with IP addresses when joining a network (e.g. joining a WiFi network). You can proceed with installation without DHCP if you plan to only test on a single node for now, but this service needs to be disabled in order to utilize the built-in DHCP service offered by MAAS (Metal as A Service), which we will use for provisioning all of our other CPU nodes. For advanced users, a Virtual Local Area Network (VLAN) can be used for this purpose, or a configurable router such as OPNSense can be used to configure PXE booting.

A functional Internet connection is also required on the target machine in order to download and install the latest security patches and system software during the installation process. For this you will need to know the IP address of your networks default gateway, and any custom DNS servers used (if applicable). This will be described further in Step 2.

3.2 Step 1: Setup and Installation

To install the BRAINE platform you first need to set up a bootable USB to install the ISO file of the BRAINE installer. This process is similar to the installation of any Operating System (OS) on a machine. To do this, the ISO file needs to be flashed to a USB using a tool such as Rufus, BalenaEtcher, or Ventoy. For novice users, links to the Rufus software and a tutorial on how to use it are provided below.

<https://rufus.ie/en/>

<https://ubuntu.com/tutorials/create-a-usb-stick-on-windows>

Note: As the installer is an ISO file, you can also install this platform as a VM or set of VM's if preferred. All virtualization platforms should be compatible with this ISO. Installation via Intelligent Platform Management Interface (IPMI, e.g. Dell iDRAC) is also possible.

Once you have connected the ISO to your target machine via USB or another method, you must select the option to boot from it. When your device is turning on, look for some information relating to a "boot menu" or similar phrasing. On most devices, pressing Esc, F2, F12, or delete will load the system BIOS/UEFI or boot menu. Simply select boot from your USB device (or ISO).

Note: On some systems you may need to disable Secure Boot before booting a new OS. This can be done in your devices' UEFI menu, for a detailed tutorial see here:

<https://www.howtogeek.com/how-to-disable-secure-boot/>

The system has correctly booted if you see the BRAINE Auto install menu option.

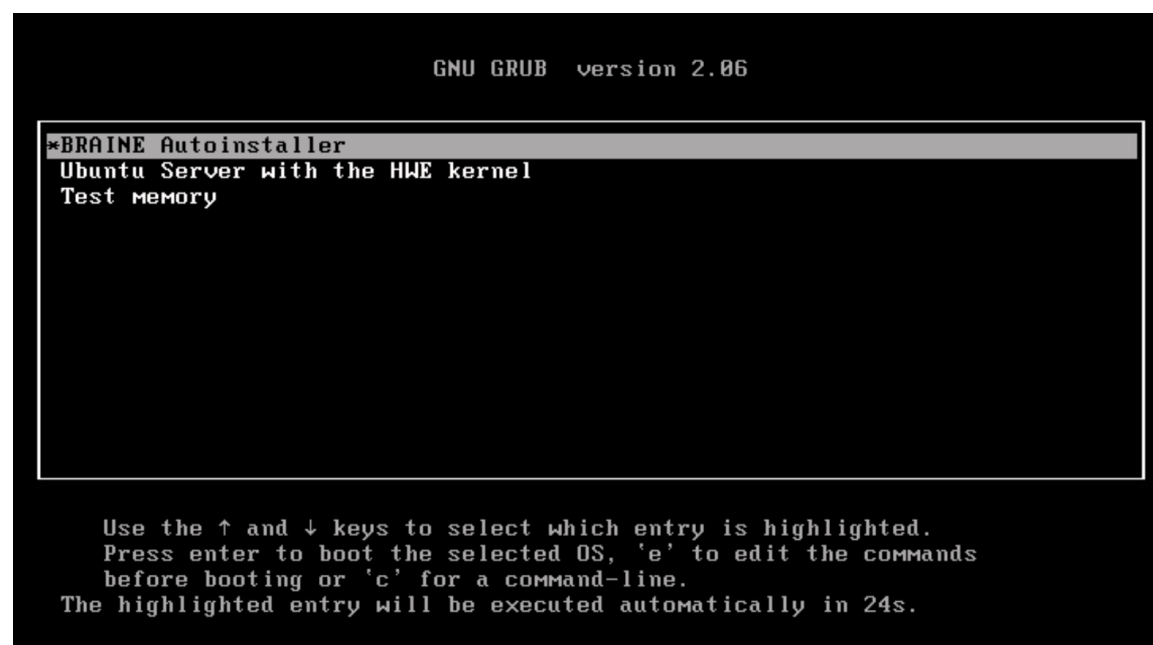


Figure 3.1 BRAINE Auto-install menu when booting from ISO/USB

The system will now automatically boot, or you can simply press Enter to skip the 30 second delay.

3.3 Step 2: Network Configuration and Disk Wiping

Once the installer has completed the boot process, it will ask for the target machines network configuration. If DHCP is still enabled on your network, the machine may have already been given an IP address, which you can continue with but remember that you will still have to disable it later if you want to use MAAS. Otherwise if DHCP is disabled or you already know the desired network configuration and want to set it manually, choose to edit the IPv4 information of your network interface.

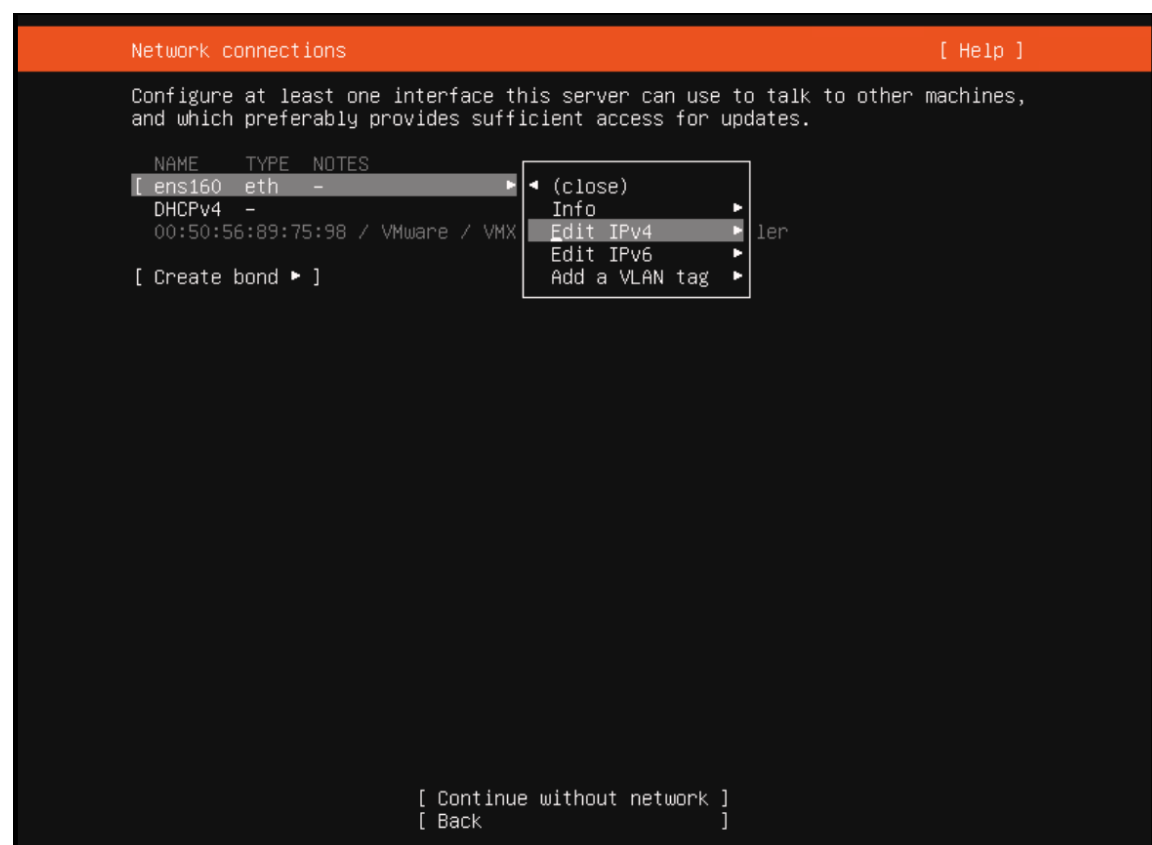


Figure 3.2 Network interface menu

Note: You may have several network interfaces on your device. The simplest option for configuration is to just connect a single network cable to the device for now. When your IPv4 information is correctly configured, you can move the network cable between ports until the OS reports it correctly has a network connection.

In the IPv4 menu, change the method from Automatic (DHCP) to Manual. You then need to provide your network information such as subnet range, device IP address, default gateway address, and DNS servers.

If testing on a home network this value is typically 192.168.xxx.yyy/24. You can determine this by checking the IP address information of another device already connected to your network.

(For novice users, a tutorial on IP addressing and subnetting [can be found here](#))

If you are on an enterprise network and are unsure, contact your local IT administrator and they should be able to provide you a network configuration. In this example, the subnet is 10.13.0.0/16, we will use a machine IP of 10.13.1.100 (make sure the IP is not already in use on the network), a default gateway of 10.13.1.1, and DNS information of 10.13.1.1, and 8.8.8.8 (Google public DNS). These values can be changed as desired. Search domains are an optional inclusion for advanced users.

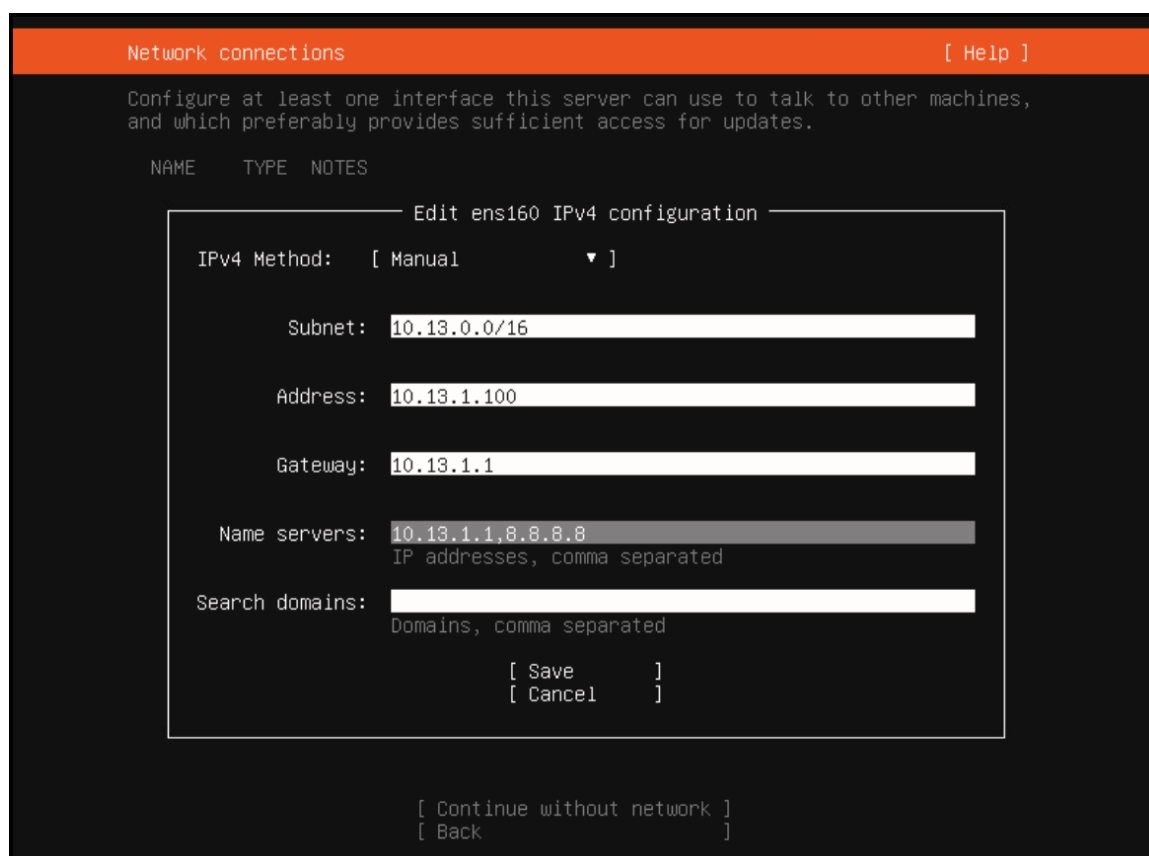


Figure 3.3 Network configuration

When the “Continue without network” option switches to “Done”, it means a successful network connection has been established and you can proceed with the installation.

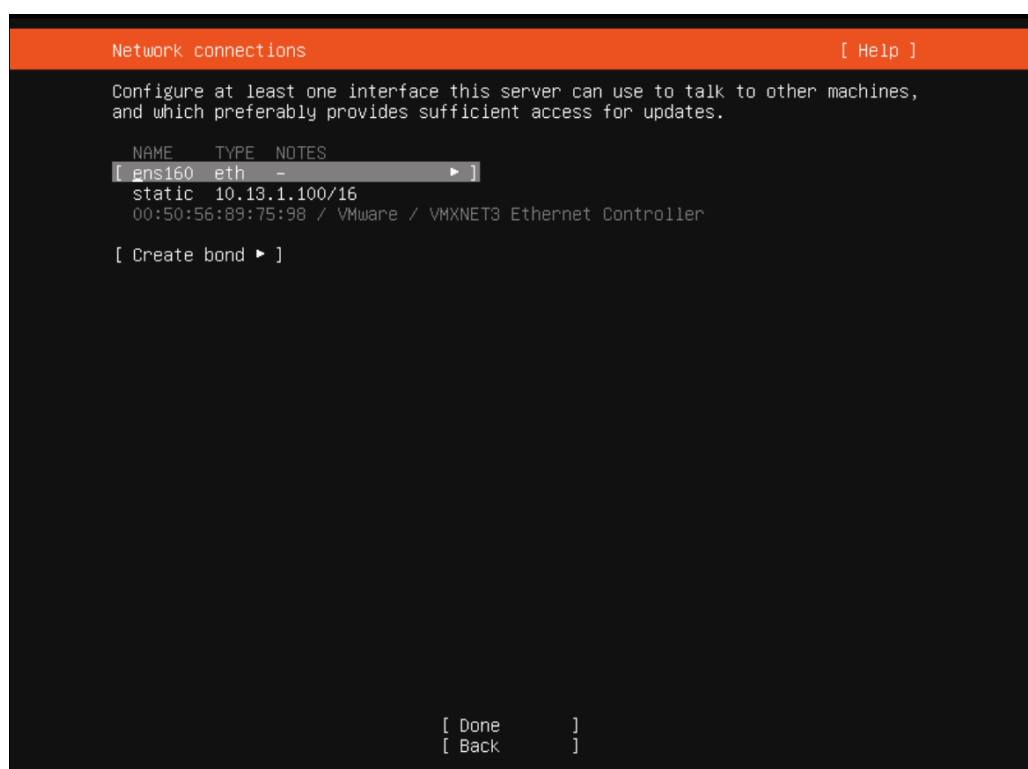


Figure 3.4 Successful network connection established.

The next step is to install the OS onto a local disk drive, for the simplification of installation, the installer:

- Will install onto the first disk drive it detects.
- Will completely wipe the disk during installation.

As such, ensure any important data from the target machine has been backed up prior to installation, and you are OK with the disk wiping process. If the system has multiple drives, the installer should normally install on the first indexed drive. Drives can be removed to ensure installation occurs on the correct drive and reinstalled after installation if needed. Simply press continue on the menu screen to confirm disk wiping and OS installation.

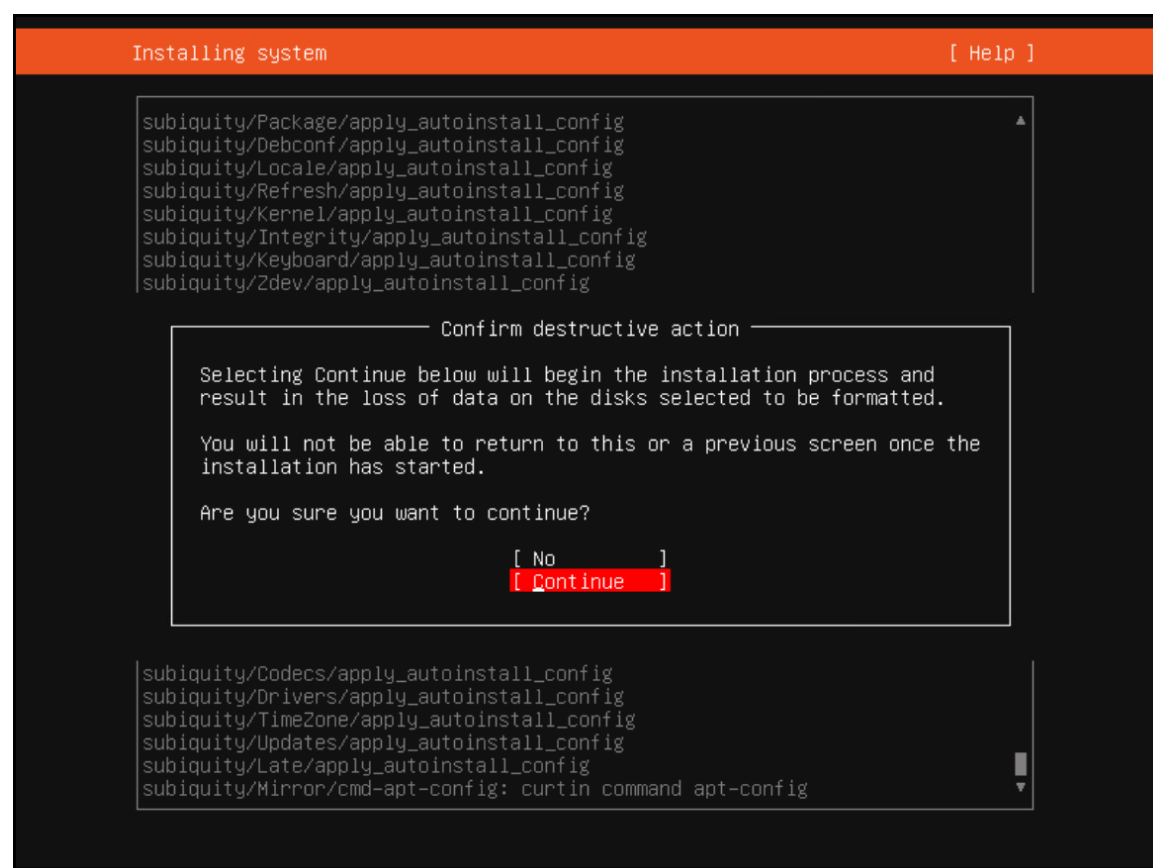
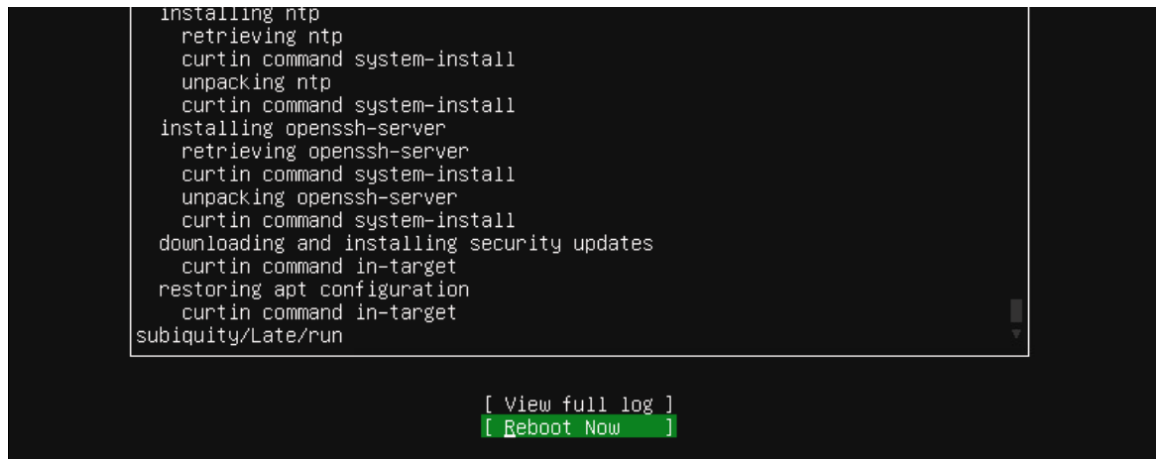


Figure 3.5 Confirm disk wiping for OS installation.

The system will then continue with installation automatically. It is safe to leave the machine during this install phase, and the process will take approximately 5-15 minutes depending on your machine's capabilities. During this phase the machine is:

- Installing Ubuntu 22.04 LTS as the base OS
- Downloading and installing the latest Linux kernel, firmware, and security patches, so that the machine is always up to date after installation.
- Installing additional standard Linux packages which are required for installing BRAINE platform components post-installation.

The machine has completed this phase of the installation when the final logged command is subiquity/Late/run, and you have the option to reboot now.

A terminal window with a black background and white text. The text lists a series of installation steps: installing ntp, retrieving ntp, curtin command system-install, unpacking ntp, curtin command system-install, installing openssh-server, retrieving openssh-server, curtin command system-install, unpacking openssh-server, curtin command system-install, downloading and installing security updates, curtin command in-target, restoring apt configuration, curtin command in-target, and subiquity/Late/run. At the bottom, there are two green buttons: '[View full log]' and '[Reboot Now]'.

```
installing ntp
retrieving ntp
curtin command system-install
unpacking ntp
curtin command system-install
installing openssh-server
retrieving openssh-server
curtin command system-install
unpacking openssh-server
curtin command system-install
downloading and installing security updates
curtin command in-target
restoring apt configuration
curtin command in-target
subiquity/Late/run

[ View full log ]
[ Reboot Now ]
```

Figure 3.6 Installation Complete

Simply remove the installer USB/ISO and press enter and let the device reboot.

Once rebooted, the node will continue to run the post-install process. This process:

- Downloads and configures Docker and Kubernetes (K8s)
- Downloads MAAS
- Starts a Kubernetes Cluster
- Starts the Rancher orchestrator, ONOS SDN controller, and a number of other additional services within K8s.
- Creates a private Docker Registry
- Starts MAAS

Simply leave the machine unattended, while the installer completes this post-install process. This may take another 5-15 minutes depending on your machine's capabilities,

and your internet connection. Once complete, you will see a login screen like this



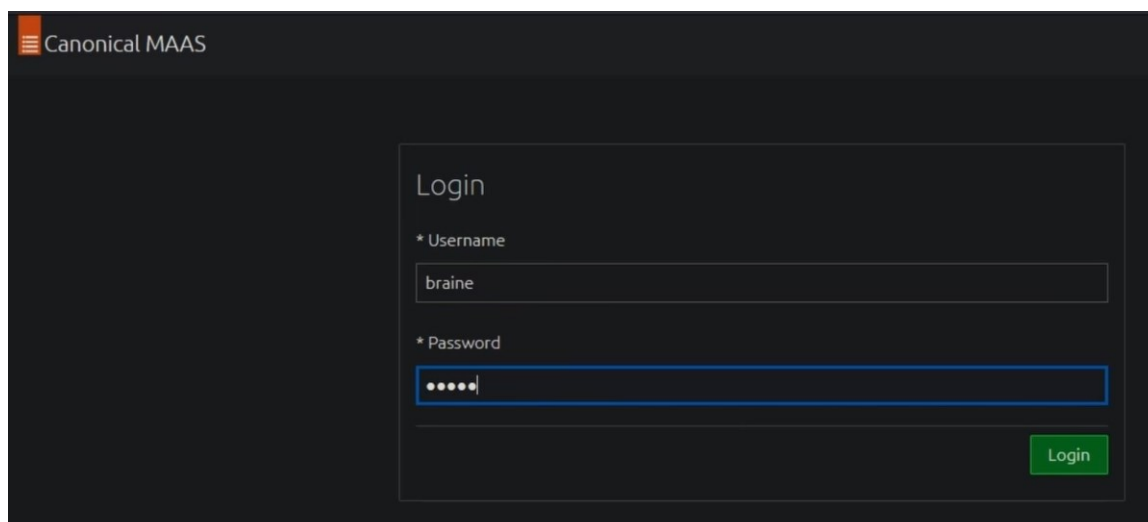
Figure 3.7 Installation fully complete login screen

3.4 Step 3: Setting up MAAS

Note: if you only want to test the system on a single node, or want to deploy k8s workloads immediately, you can skip to Step 5.

As can be seen in Figure 3.7, the system prompts the user to visit MAAS at <http://<your.ip.address>:5240>.

Simply open a web browser on another machine connected on the same network and enter that address. You may get a warning due to using HTTP, but this can be safely added as an exception.



Canonical MAAS

Login

* Username

braine

* Password

•••••

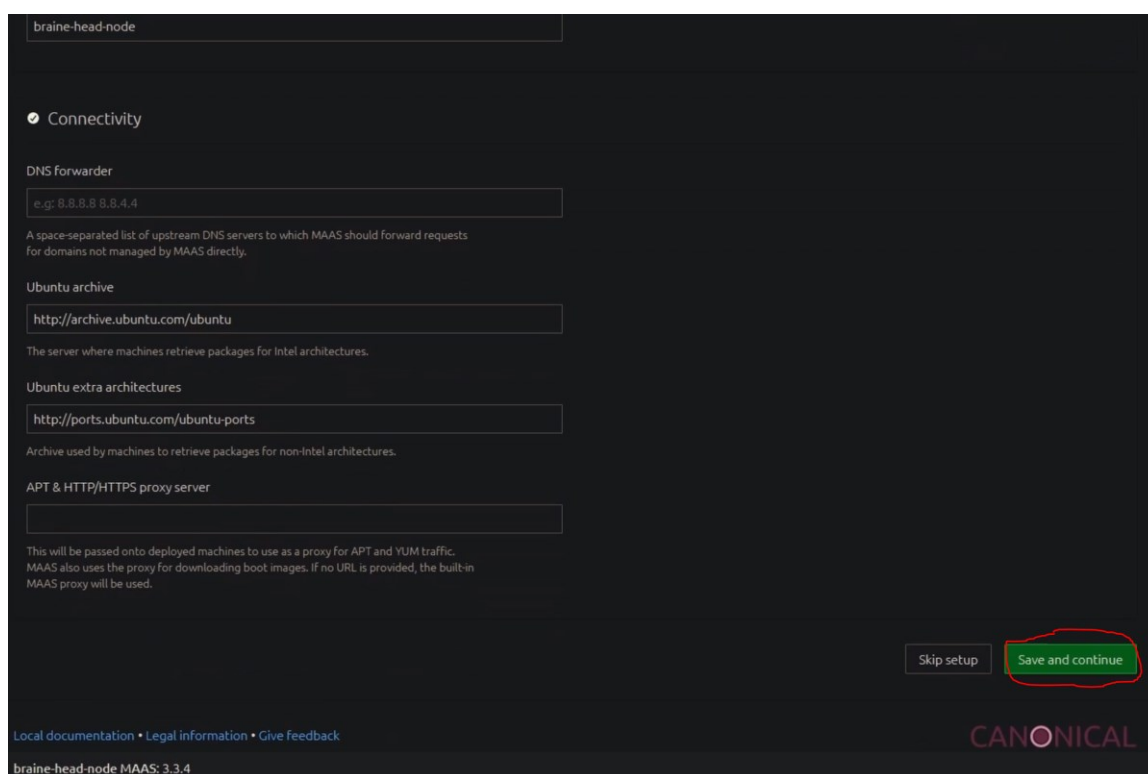
Login

Figure 3.8 MAAS Login

The default login provided for MAAS is:

- Username: braine
- Password: braine

Once logged in, MAAS will guide you through an initial setup. You can add a DNS forwarder of 8.8.8.8 (Google DNS) if desired, otherwise everything else can be left as default for now, and you can click Save and continue.



braine-head-node

Connectivity

DNS forwarder

e.g. 8.8.8.8 8.8.4.4

A space-separated list of upstream DNS servers to which MAAS should forward requests for domains not managed by MAAS directly.

Ubuntu archive

http://archive.ubuntu.com/ubuntu

The server where machines retrieve packages for Intel architectures.

Ubuntu extra architectures

http://ports.ubuntu.com/ubuntu-ports

Archive used by machines to retrieve packages for non-Intel architectures.

APT & HTTP/HTTPS proxy server

This will be passed onto deployed machines to use as a proxy for APT and YUM traffic. MAAS also uses the proxy for downloading boot images. If no URL is provided, the built-in MAAS proxy will be used.

Skip setup

Save and continue

Local documentation • Legal information • Give feedback

braine-head-node MAAS: 3.3.4

CANONICAL

Figure 3.9 MAAS Initial setup screen

The next installation screen for MAAS is choosing OS images. These images can be chosen as desired by the user, and additional OS images can be sourced by advanced users. The recommended OS for novice users at time of writing is Ubuntu 22.04 LTS, which is a Long-Term Service version of the Ubuntu OS that will receive security

updates from up to 4 years from release. As such, select this option and the desired architectures, in the context of BRAINE, we want amd64 (normal Intel or AMD CPU's) and arm64 (ARM CPU). So we select these options and click Update Selection, and you should see the selected images now preparing for download.

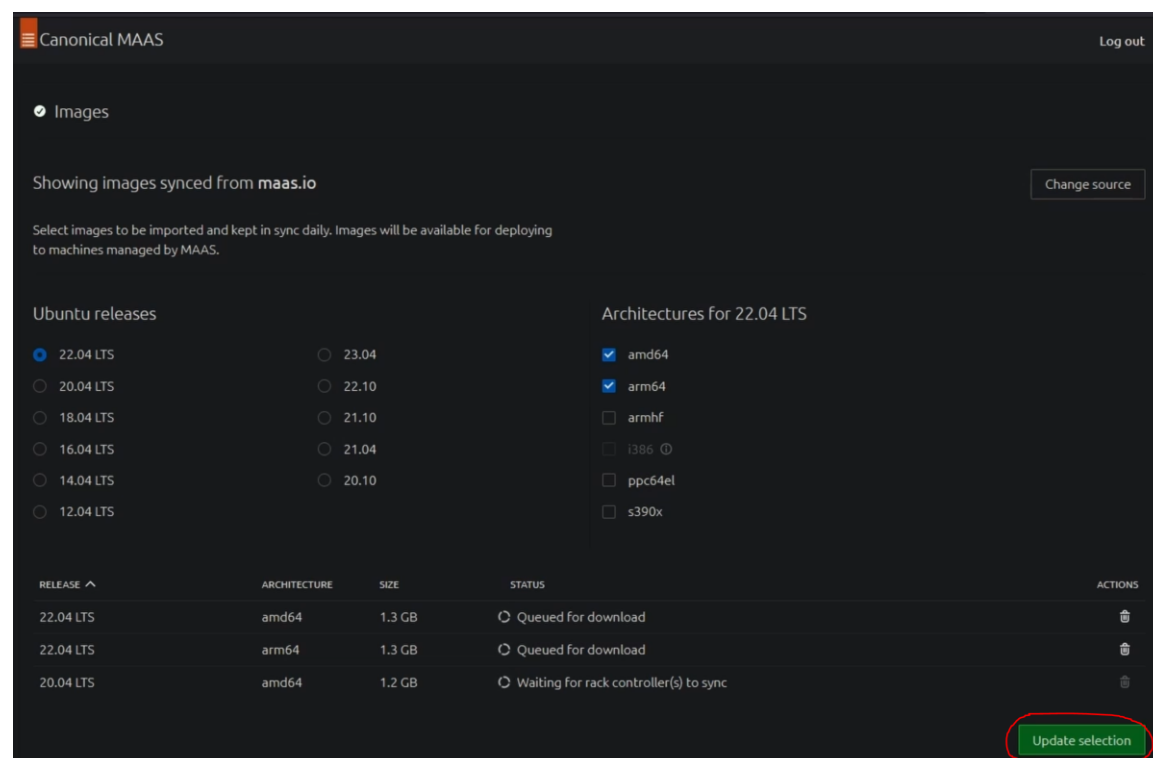


Figure 3.10 MAAS OS Image Selection

You can then scroll down, and select the green continue button.

At this point MAAS will say the setup has been completed, and you can click Finish Setup. Note that it mentions a warning that DHCP is not enabled which will prevent PXE booting, which we will resolve next.

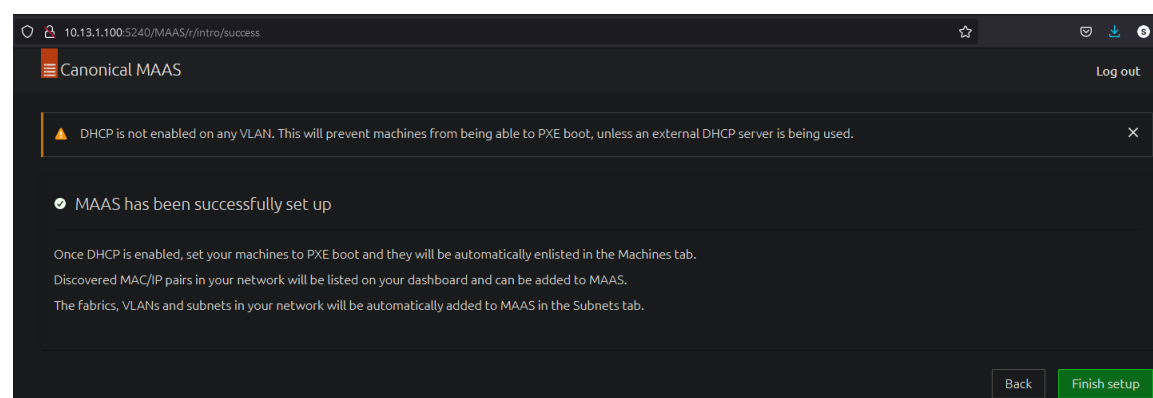


Figure 3.11 MAAS Initial Setup Complete

Note: MAAS will also prompt you to also setup SSH keys for secure access to the nodes provisioned by MAAS, but this step can be safely skipped in this guide, and SSH keys can be added later.

Once on the MAAS dashboard, click the Subnets button. Here you will see the networks that MAAS has detected. The ones it will typically display are:

- Your chosen network (in this case 10.13.0.0/16)
- Docker subsystem network (172.17.0.0/16)
- Kubernetes subsystem network (10.244.0.0/24)

Your chosen network is the one we need to configure, so select it by clicking on it.

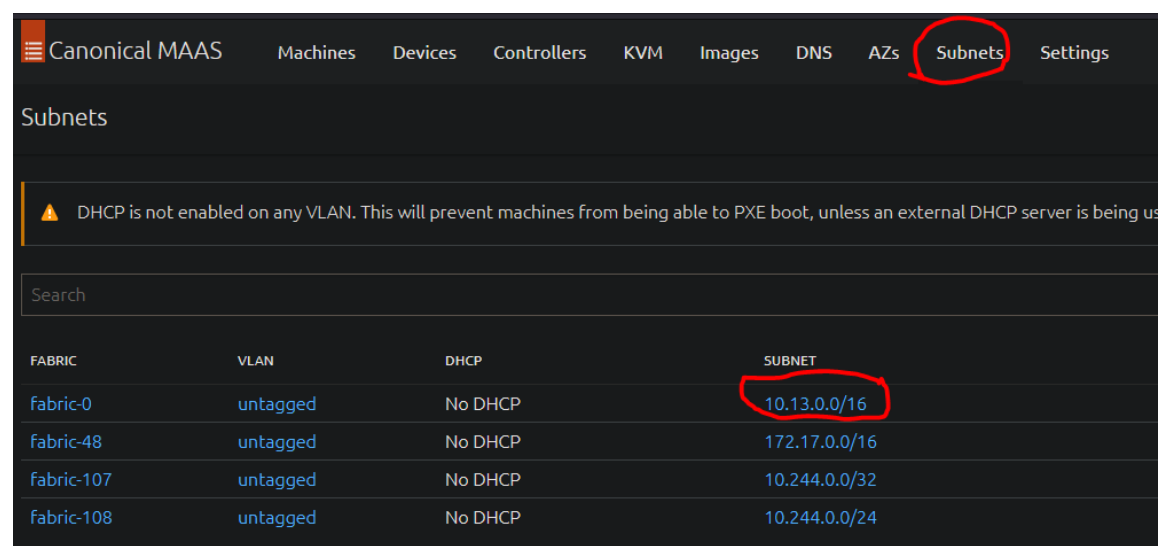


Figure 3.12 Subnet network selection

Scroll down until you see the green Reserve range button and select Reserve dynamic range. Then select an IP address range that you are aware of that is not in use by any other machines on the network. Contact your IT administrator if you are unsure of a range. In this example, we will use 10.13.1.101-10.13.1.200. This will enable MAAS to provision and control up to 99 nodes, but this can be increased to any desired number. Once the IP range is entered, click Reserve.

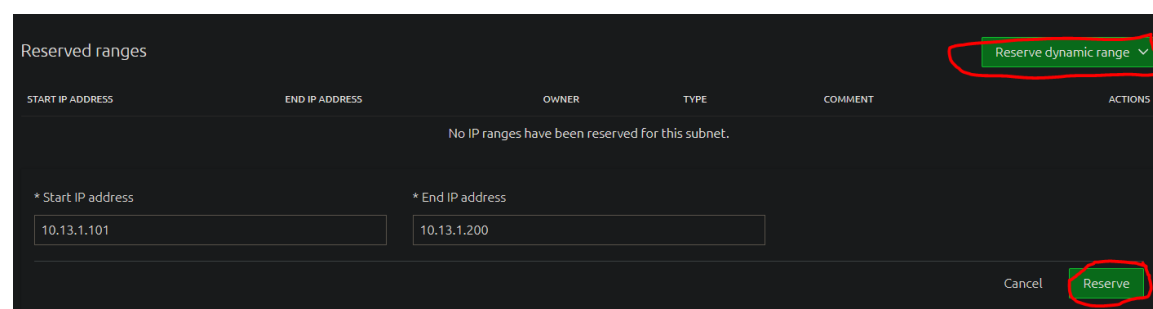


Figure 3.13 MAAS IP Reservation

Once complete, click back to the Subnet menu, and this time select the untagged VLAN. Once in the VLAN menu, click Configure DHCP.

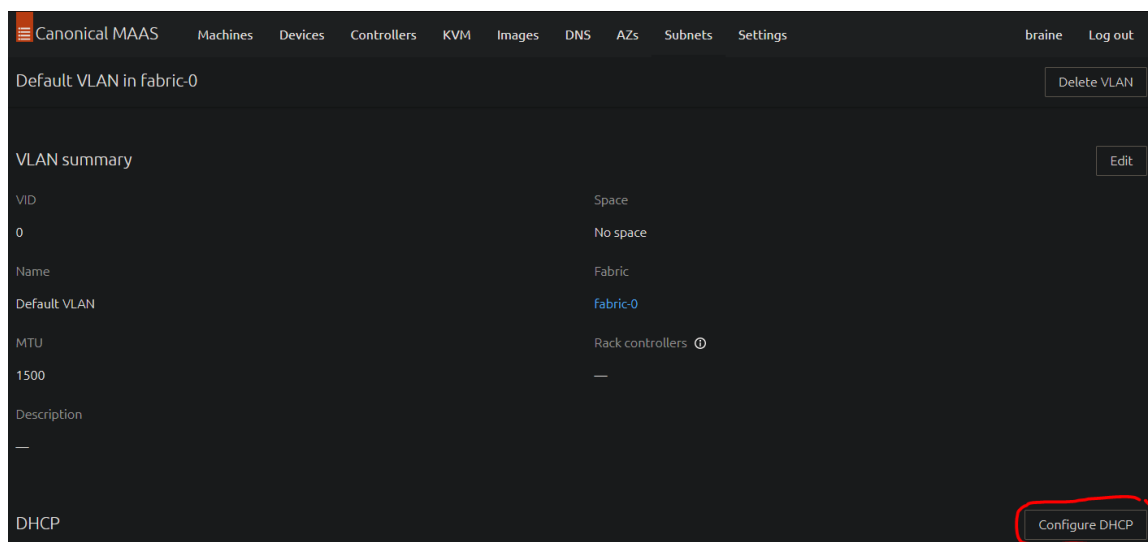
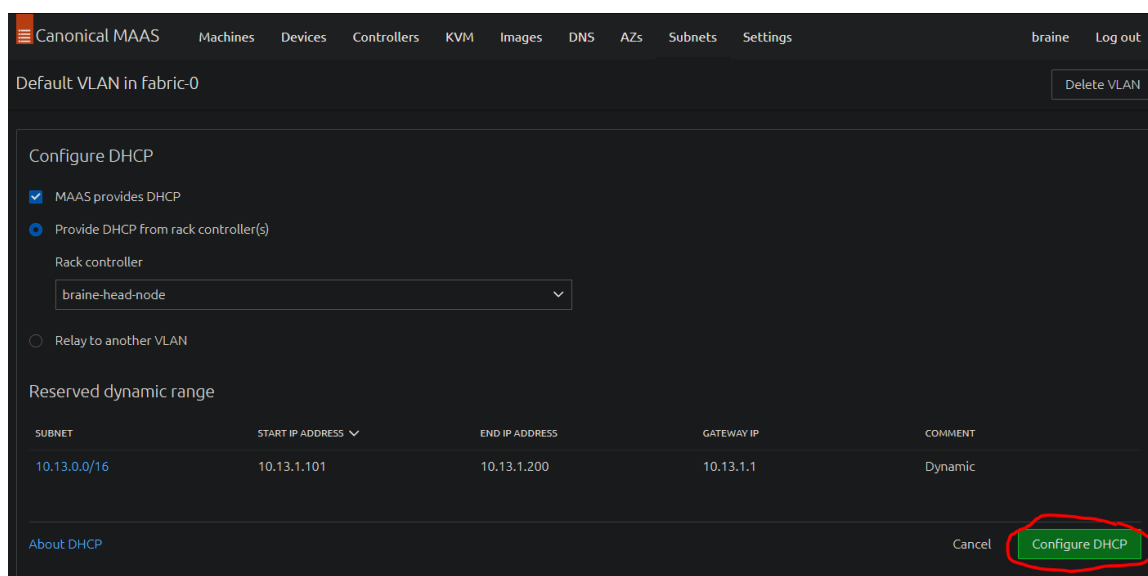


Figure 3.14 VLAN DHCP Configuration

In the DHCP menu, keep the option that MAAS provides DHCP selected, and in the rack controller list, select the braine-head-node, then click Configure DHCP.



MAAS Setup is now complete, and we can start provisioning nodes.

3.5 Step 4: Provisioning Nodes

Now that MAAS is ready, we will use MAAS to automate the set-up of all our other nodes for this BRAINE cluster. For this example we will follow a BRAINE EMDC-like design, where we have 8 nodes, one being the braine-head-node, and the other being cluster worker nodes. This is simply to show the process is parallelized and can work for any number of nodes. The only requirements are:

- All nodes are on the same network as the head-node.
- All nodes are capable of PXE booting.

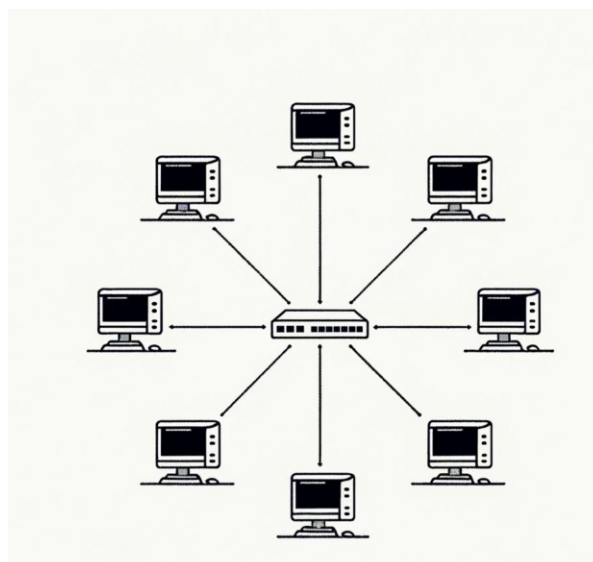


Figure 3.15 Example of a simple network which can be used by MAAS

To begin, simply power on another machine connected to the network and enter its boot-up menu by pressing F12, F2, Esc, or other possible keys. You should be presented with option to boot from the network. You may need to first enable this functionality in the systems BIOS/UEFI, but the majority of computing devices support this option so it should be available. You may also need to disable Secure Boot in some cases.



Figure 3.16 Example of a boot menu showing network boot option

If correctly done, after a brief moment of the device looking for an IP address, it should contact the MAAS server, and start booting an OS. Simply wait for this process to complete, and the machine will power off.

In MAAS, in the Machines tab, you will see how many new nodes MAAS has detected. MAAS will give them each a unique host name, which you can change if desired.



Figure 3.17 MAAS detecting nodes

The next step is to configure the power information for each node. If available, it is possible for MAAS to control the IPMI interface for each server under its control. If your device has IPMI available, you can set that up in the power configuration menu now. If you do not have IPMI available, you can set the power mode to Manual. This will simply require you to manually power on the node during the setup phase. Click Edit, choose manual (or your IPMI), and click save changes. Do this for all nodes.

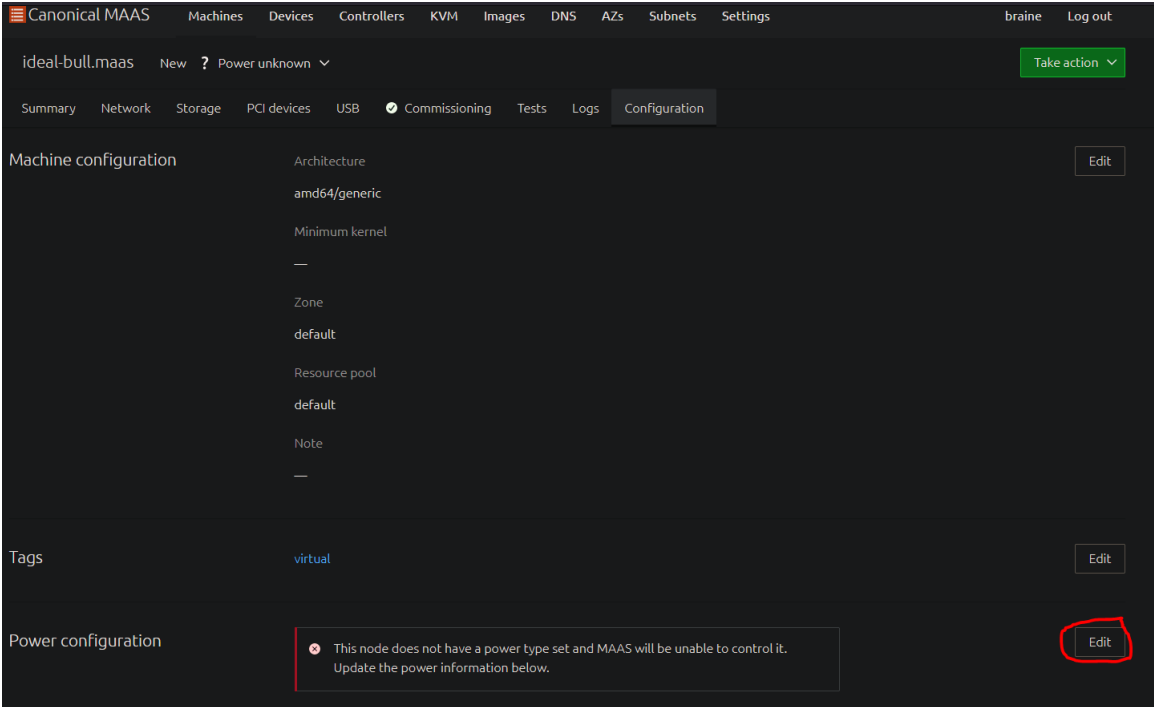


Figure 3.18 Node Power Configuration

Once all nodes have been detected and all power information has been given, a commissioning phase is required to register the devices fully on MAAS. To do this, when all nodes are in the “New” status, select all of them at once, and then Take Action>Commission.

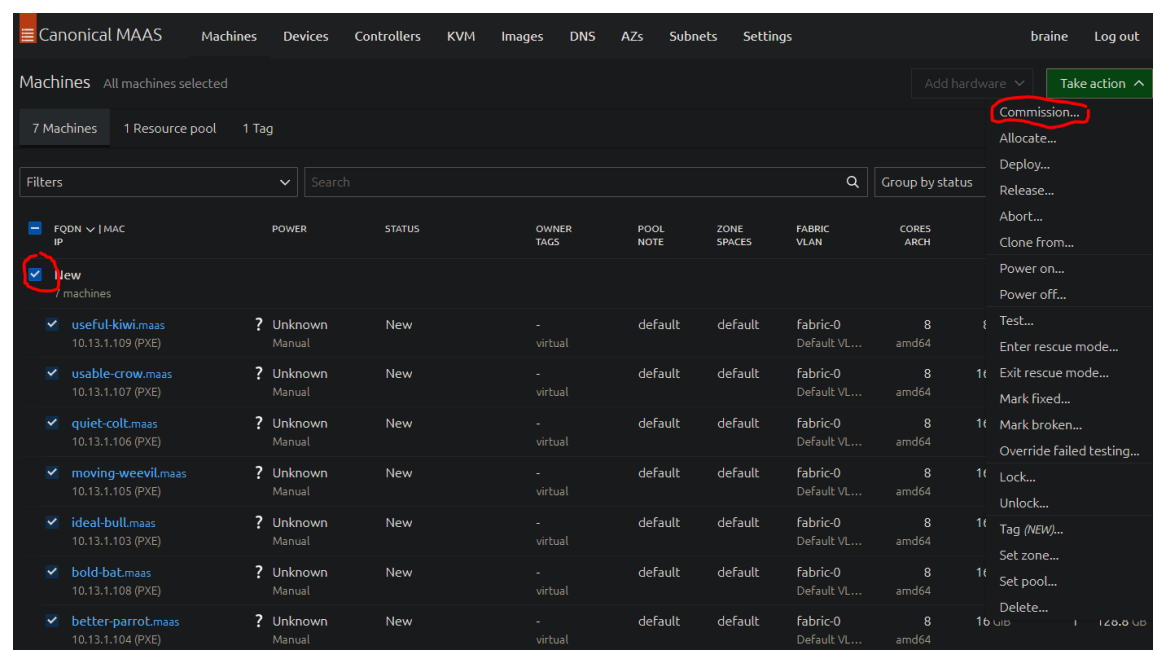
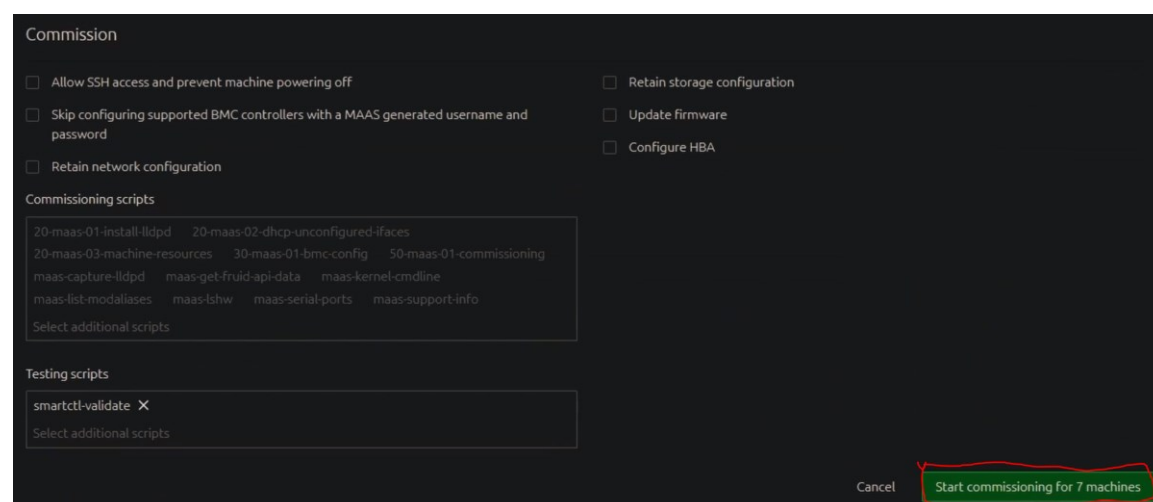


Figure 3.19 Commissioning the nodes

In the pop-up menu, there is no need to change any of the defaults, simply click Start commission for (your number of) machines.



MAAS will begin the commissioning phase. As we set the power mode to manual, this will require you to power up the nodes manually and select them to network boot again. MAAS will show the nodes moving from the Commissioning phase to the Testing phase, to the Ready phase, and the machine will power off once more. When all nodes are reported as ready by MAAS, we can move to the deployment phase.

| | | | |
|---|---------------------|--|--|
| <input type="checkbox"/> Commissioning 6 machines | | | |
| <input type="checkbox"/> usable-crow.maas 10.13.1.107 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> quiet-colt.maas 10.13.1.106 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> moving-weevil.maas 10.13.1.105 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> ideal-bull.maas 10.13.1.103 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> bold-bat.maas 10.13.1.108 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> better-parrot.maas 10.13.1.104 (PXE) | ? Unknown Manual | ⌚ Commissioning Gathering information | |
| <input type="checkbox"/> Ready 1 machine | | | |
| <input type="checkbox"/> useful-kiwi.maas 10.13.1.109 (PXE) | ? Unknown Manual | Ready | |

Figure 3.20 MAAS Commissioning in Progress

We are almost ready to move to the deployment phase. One thing we need before we start the deployment phase is a special key to join the new MAAS nodes to the same Kubernetes cluster as the head node. To retrieve this key, we need to Secure Shell (SSH) into the head node to generate it. To do this use an SSH client of your choice (e.g., PuTTY, WinSSHTerm), and login to the head node at your chosen IP address with:

- Username: braine
- Password: ubuntu

Once logged in, simply enter the command:

- ***sudo kubeadm token create --print-join-command***

This will output a kubeadm join command with a specific token ID that matches your head node. Copy this full command.

```

Using username "braine".
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-91-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Dec 12 09:37:19 PM UTC 2023

System load: 1.18798828125   Users logged in:      0
Usage of /:   30.1% of 117.55GB   IPv4 address for cnl0:  10.244.0.1
Memory usage: 18%             IPv4 address for docker0: 172.17.0.1
Swap usage:   0%               IPv4 address for ens160: 10.13.1.100
Processes:   566

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Tue Dec 12 19:59:17 2023 from 192.168.13.2
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

braine@braine-head-node:~$ sudo kubeadm token create --print-join-command
[sudo] password for braine:
kubeadm join 10.13.1.100:6443 --token najey8.soy2vrszksdieyld --discovery-token-ca-cert-hash sha256:d087fafb1582f7db7474727f23c50429f6b1e187556041f1d7ec02e5ce9c5172
braine@braine-head-node:~$

```

Figure 3.21 Getting K8s token from head node

Now that we have the token, we can start deployment on the MAAS nodes. Select all of the nodes in MAAS once more, and now click Take Action > Deploy.

Now select the Release as Ubuntu 22.04 LTS, and you can set a kernel type if required, otherwise leave the default No minimum level. Click the center box – Cloud-init user-data... and then paste in a cloud-init script from

- https://gitlab.com/braine/braine_public_release/-/tree/main/cloud-init%20scripts?ref_type=heads

Simply choose the correct script for your node (Note: x86 with GPU and ARM node will be added soon) and change the last line at the bottom of the script with the join command and token you copied earlier. Once that is done as shown in Figure 3.22 click start deployment, and power on and boot the machines from the network one final time.

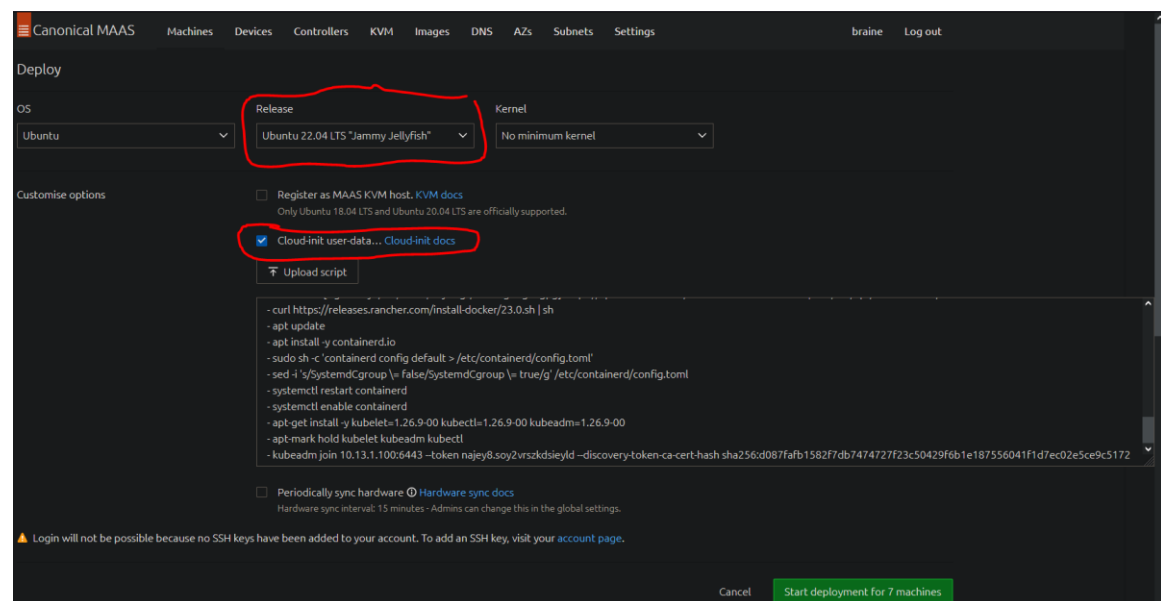


Figure 3.22 Deploying to nodes with cloud-init

This process does the following:

- It will wipe any disks on the node, so ensure all data is backed up prior to starting this process.
- It installed the Ubuntu OS on the nodes, further network boots will not be required.
- It uses cloud-init to:
 - Create a user called braine with password ubuntu (like the head node) and allow SSH access (the warning about SSH keys in Figure 3.22 does not apply to us as we are creating a user ion cloud-init)
 - Download and install Docker
 - Download and install Kubernetes
 - Join the head nodes Kubernetes cluster.

The process is fully automatic. Once complete, you should find your new worker nodes are part of you overall Kubernetes cluster in Rancher. Simply repeat step 4 again to add new nodes to you cluster as desired.

3.6 Step 5: Setting Up Rancher

With node setup complete, we can now access the Rancher UI via <https://<your.ip>:30443>

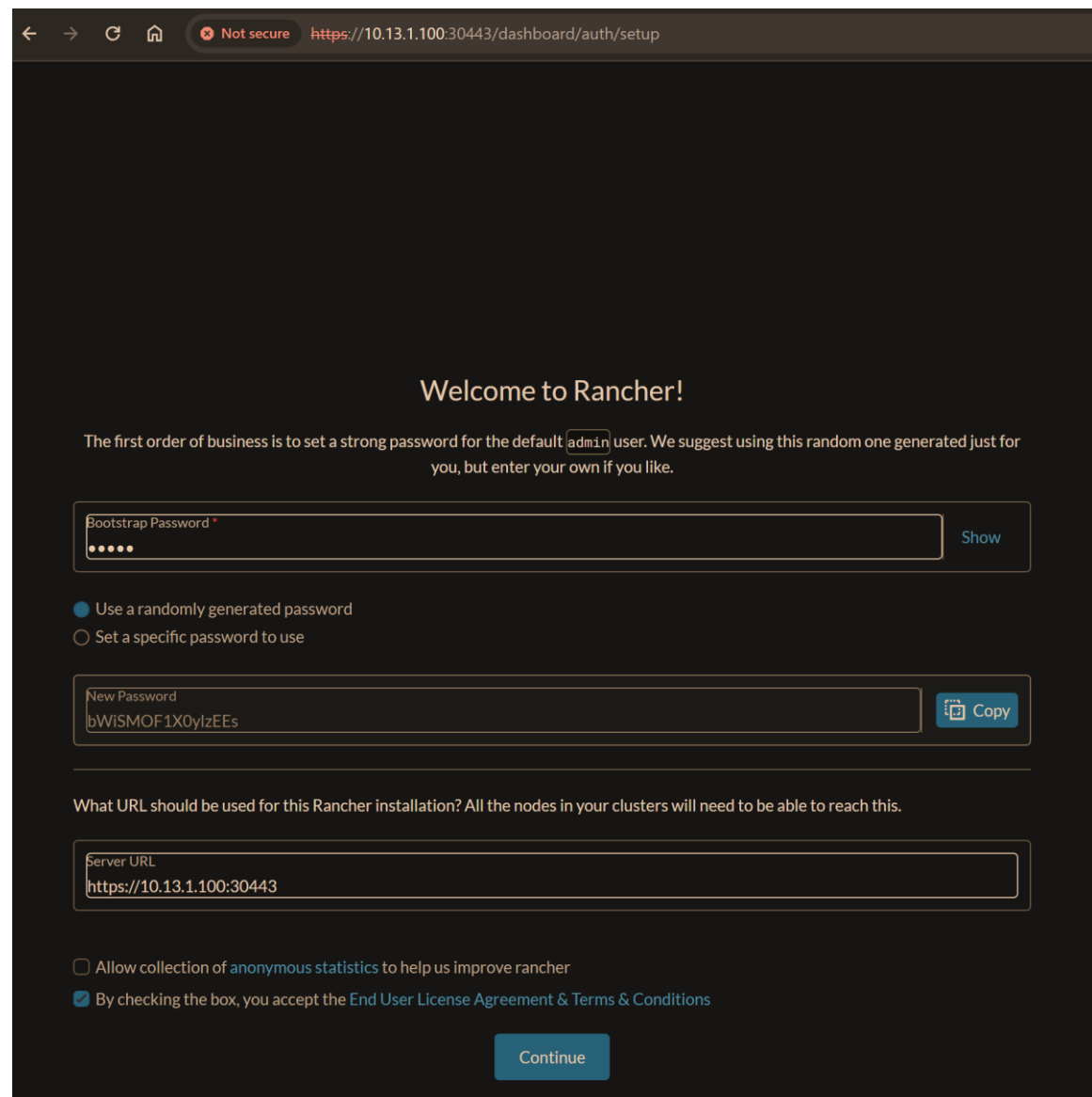
Note: You may need to explicitly specify https and accept the warning due to using a self-signed HTTPS certificate.

Note2: Rancher may occasionally exhibit minor errors with the Firefox browser, so a Chrome based browser is recommended.

On your first login, Rancher will show a bootstrap menu, where you must set a new administrator password and accept the conditions before using the software. A bootstrap password is required to complete this step:

- Bootstrap password: admin

Enter this and save Rancher's generated password or create your own. Leave the server URL as the default, unless you have a personal DNS service, and click continue.



← → ↻ 🏠 Not secure https://10.13.1.100:30443/dashboard/auth/setup

Welcome to Rancher!

The first order of business is to set a strong password for the default `admin` user. We suggest using this random one generated just for you, but enter your own if you like.

Show

☒ Use a randomly generated password
☐ Set a specific password to use

bWISMOF1X0ylzEEs Copy

What URL should be used for this Rancher installation? All the nodes in your clusters will need to be able to reach this.

https://10.13.1.100:30443

☐ Allow collection of [anonymous statistics](#) to help us improve rancher
☒ By checking the box, you accept the [End User License Agreement & Terms & Conditions](#)

Continue

Figure 3.23 Rancher bootstrap menu

Note: After bootstrap, your login details will be:

- Username: admin
- Password: (Generated by Rancher or chosen by user.)

After logging in once you should see a menu listing your Kubernetes clusters, of which you will have one right now called “local”. Note: If you do not see this and rancher shows a red warning message, simply refresh the browser tab. Click local, and you should now see your Kubernetes cluster dashboard.

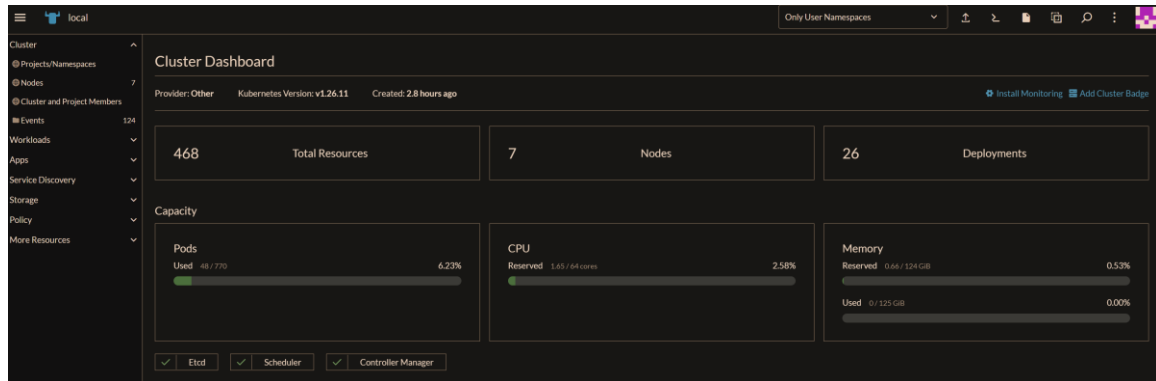


Figure 3.24 Rancher K8s cluster dashboard

Note that Rancher is showing 7 nodes available, meaning the nodes we added via the previous steps in MAAS have now been correctly added to the cluster (1 node remaining to be added).

Rancher contains predefined Helm charts which simplifies the integration of some of the same open-source components used by the BRAINE platform. Some of those components include:

- Istio: A service mesh to distribute K8s pods and services across several K8s clusters (i.e. EMDC’s in BRAINE)
- Longhorn: A distributed file system to enable K8s persistent volume storage necessary for some platform services and use case applications.
- Monitoring: A comprehensive set of monitoring and dashboard services including Prometheus, Grafana, and Alert Manager.
- Kafka: Messaging service used for DKB and other BRAINE components.

Other components pre-installed include:

- ONOS: SDN Controller
- InfluxDB: Time-series database
- Metal LB: BGP router and K8s Load Balancer

For a fully functional K8s platform, it is recommended to install Longhorn and Monitoring, followed by any other additional services required by your use case (e.g. Istio and Kafka if using multiple k8s clusters)

For distributed storage, this guide will focus on the installation of Longhorn, however other options are available for advanced users:

- [Apache Ozone](#) – for file and object storage and Ranger/DLCM integration
- [NFS Provisioner](#) – for those who want to use a large, centralized storage system instead of storage distributed to each node.
- [CEPH](#) – Powerful distributed file system for file and object storage with S3 API suited to production use.

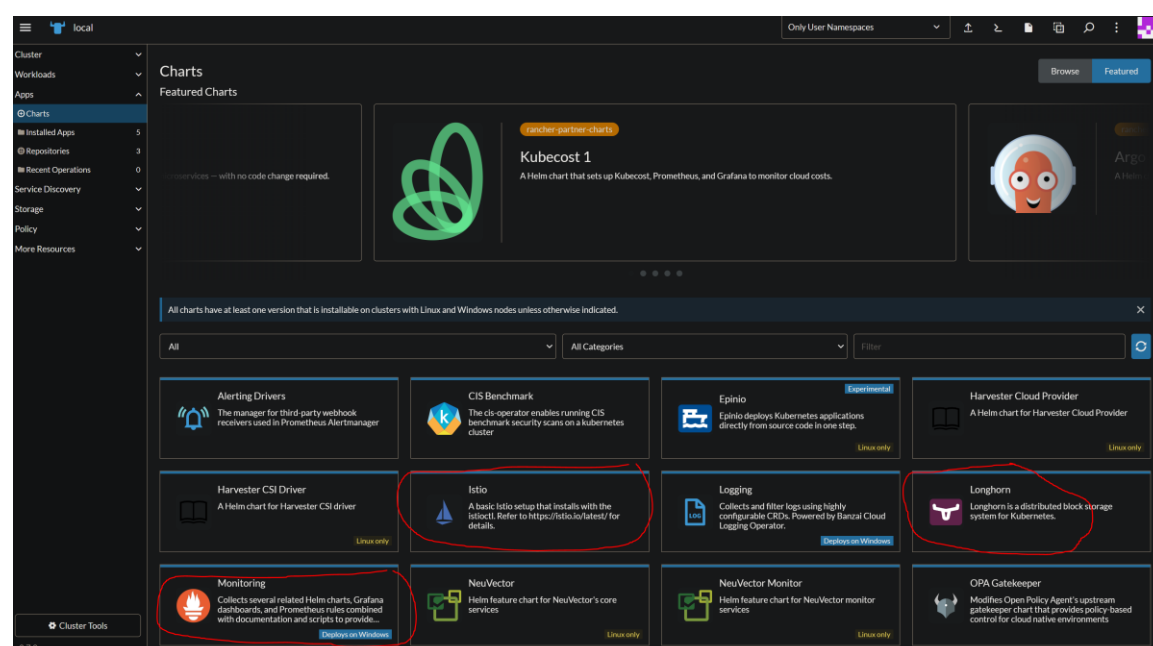


Figure 3.25 Rancher Applications Menu

To install Longhorn simply choose it from the Apps > Charts menu and then click the install button. Customization options are available, such as to choose custom private images of Longhorn, but for now the defaults are OK, so simply continue to install.

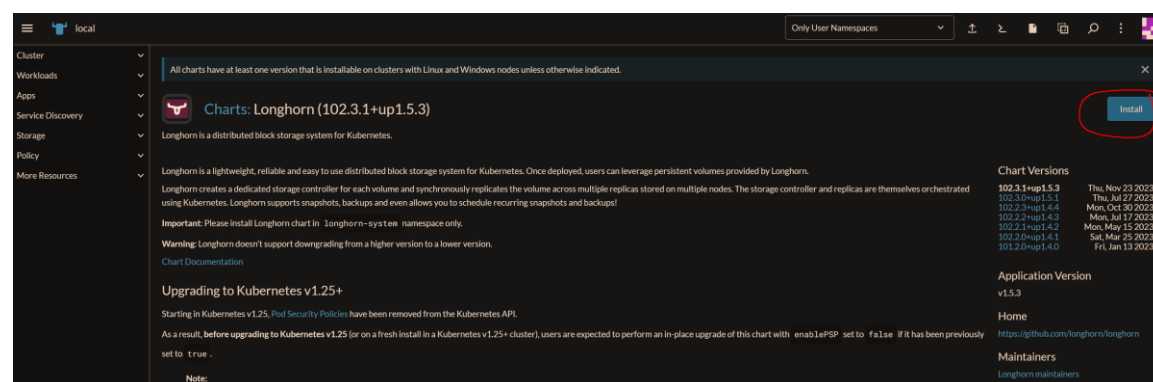


Figure 3.26 Longhorn Installation

The Longhorn installation will open a terminal window where it shows the installation in progress. Once complete, refresh your browser window until you see a new Longhorn menu on the left-hand side menu screen in the UI. Click that and you can now open a new Longhorn UI. Within it, you should see that Longhorn has correctly installed across all nodes in your K8s cluster and is ready to schedule storage to persistent volumes for any applications that need them.

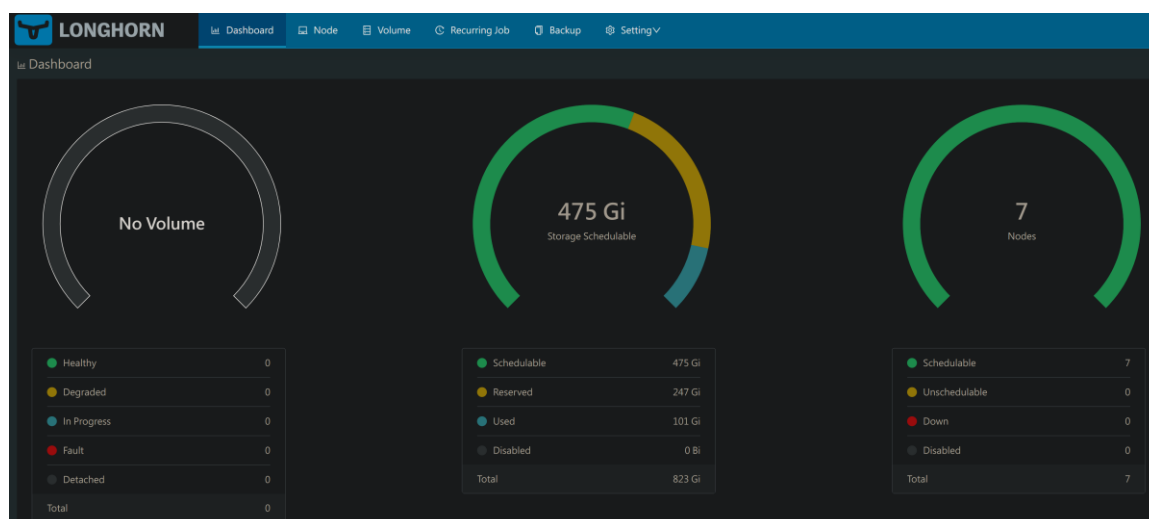


Figure 3.27 Longhorn Dashboard

With Longhorn installation completed, you can now return to the Apps > Charts menu and install Monitoring. As before, it will prompt you for some customizable configuration options, but for now the default configuration is OK so click Install.

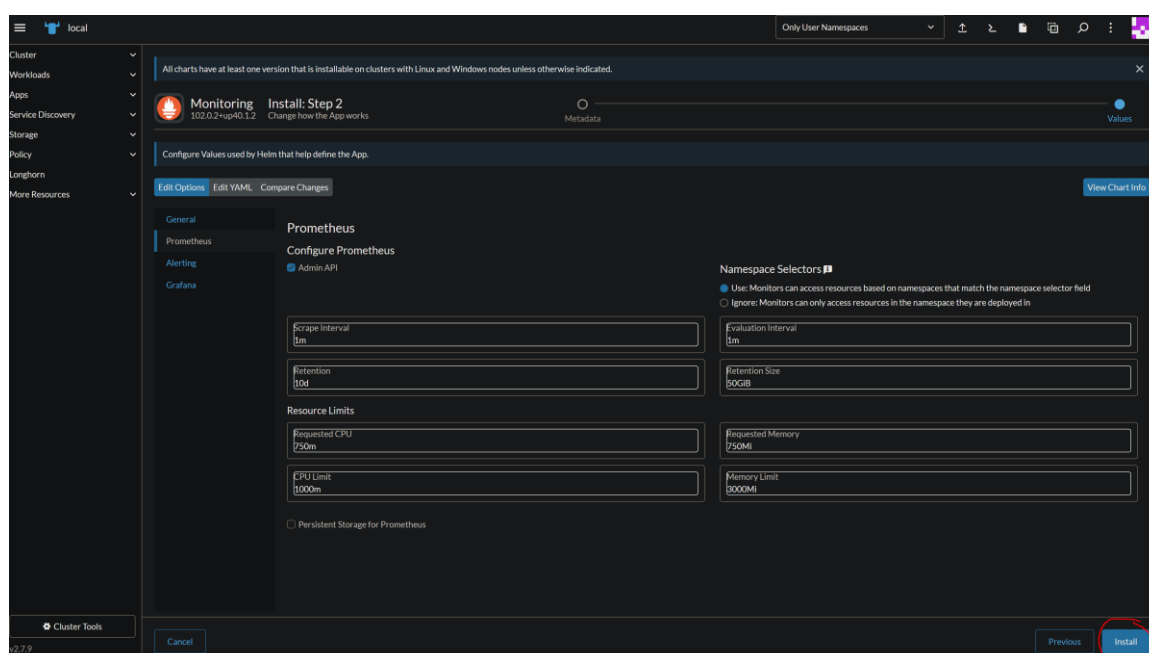


Figure 3.28 Monitoring Installation

After a few minutes, installation should fully complete. Once again refresh your browser window and a new menu called monitoring should appear in the left menu UI. Within it, you can choose to view the Prometheus and Grafana UI. If you click Grafana, you should correctly see the system dashboard showing your available CPU cores, RAM, and storage. This Grafana instance, InfluxDB, and Prometheus can also be used to integrate any other metrics required by your system or use case (e.g. system temperatures).

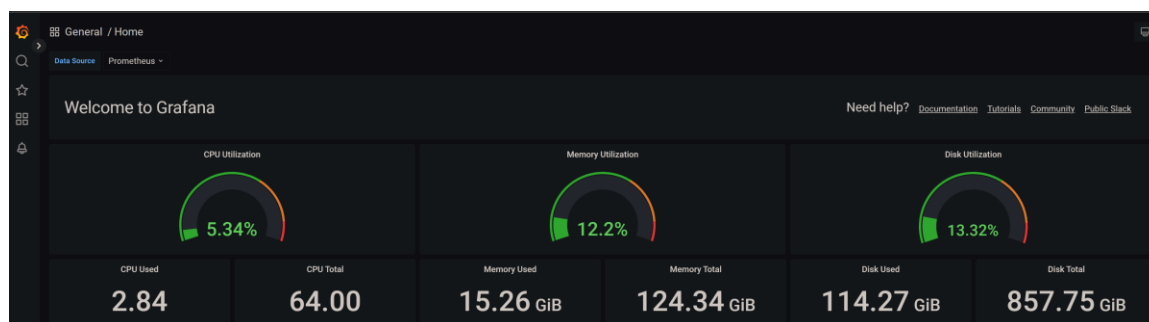


Figure 3.29 Grafana Dashboard

3.7 Step 6: Deploying Applications and Further Reading

With that, the installation of the BRAINE platform is complete! You are now ready to start deploying Docker containers as Kubernetes pods and services across your cluster. There are many further configuration options available in Rancher, for further reading on the capabilities of the platform, see:

- <https://ranchermanager.docs.rancher.com/>

To deploy your first use case/application workload, simply go to the Pod menu in the left-hand side UI and then click the deploy button. The pod deployment menu will then give you several options such as:

- The container name
- The namespace to deploy this container in
- The registry to pull the container from
 - This can be from a public Docker hub, or you can push your Docker images from a machine you own to a private internal registry, located at <your.headnode.ip>:5000 ([you will need to allow HTTP image pushing from your device](#))
- Network ports, or service ports to expose to the outside world for that application.

The image shows the 'Pod: Create' form in the Rancher UI. At the top, there are fields for 'Namespace' (set to 'default'), 'Name' (set to 'my pod'), and 'Description'. Below this is a tabbed interface with 'Pod' and 'container-0' tabs, and a '+ Add Container' button. The 'General' tab is active, showing a sidebar with 'General', 'Health Check', 'Resources', 'Security Context', and 'Storage'. The main area has sections for 'General' (with 'Container Name' set to 'container-0' and radio buttons for 'Init Container' and 'Standard Container'), 'Image' (with 'Container Image' set to 'Or from BRAINE Private registry' and 'Pull Policy' set to 'Always'), 'Networking' (with a description and an 'Add Port or Service' button), and 'Command' (with 'Command' set to 'e.g. /bin/sh' and 'Arguments' set to 'e.g. -kubernetes/httpd-f httpd.conf'). At the bottom right are 'Cancel', 'Edit as YAML', and 'Create' buttons.

Figure 3.30 Pod Deployment Menu

In the resources section, you can then set:

- Reserved and maximum CPU cores and RAM
- GPU reservations for AI workloads

Pod: Create

Namespace* default Name* my pod Description Any text you want that better describes this resource.

Pod container-0 + Add Container

General Health Check Resources Security Context Storage

Resources

CPU Reservation e.g. 1000 mCPUs Memory Reservation e.g. 128 MiB

CPU Limit e.g. 1000 mCPUs Memory Limit e.g. 128 MiB

NVIDIA GPU Limit/Reservation e.g. 1 GPUs

Cancel Edit as YAML Create

Figure 3.31 Resources Section

In the storage section, you can select the required storage for a persistent volume (if required for your application). You can also select the storage class (e.g. Longhorn or NFS if you have both installed).

Once you click deploy, Rancher will then automatically pull the Docker image you requested and deploy it on the cluster with the parameters you set. Any errors it encounters will be reported to you.

3.7.1 Production Use

NB: It is highly recommended you are experienced with cloud-native systems and architectures before you enter this type of system into production. Reading of all further documentation provided below as well as full security penetration testing is recommended.

To prepare the system for production use, a number of changes are required:

1. Change all default passwords, for reference these are:
 - a. Ubuntu SSH Username: braine
 - b. Ubuntu SSH Password: ubuntu
 - c. Note: It is recommended to change to using SSH keys for MAAS production use instead of a user as defined in cloud-init in this guide
 - d. MAAS Username: braine
 - e. MAAS Password: braine
 - f. Note: MAAS uses a PostgreSQL database, with user “maasuser” and password “maaspassword”, this should also be changed for added security, [and backed up](#).
 - g. Rancher changes the default password during the bootstrap process, but default username and password are “admin”

2. Define user accounts and strict Role Based Access Control (RBAC) policies, to ensure users only have access and control to specific pods/services. You may also have to do this from within your underlying application you are exposing to the Internet.
3. Set up for High Availability (HA), as previously mentioned a minimum of 3 nodes is recommended for this. Rancher automatically starts 3 instances of itself by default, once you have 3+ nodes you can safely delete 2 of the 3 Rancher pods and they will simply restart on a different node. You will also have to change the Rancher service exposure from NodePort to LoadBalancer when available. (This may be automated in a future update)
 - a. Note: [You should also set up MAAS in HA mode](#) just in case the head node fails and you want to provision more nodes.
4. Register a domain name, and HTTPS certificate for your cluster. This is required to replace the default self-signed certificate which will warn and may prevent users from accessing your cluster services.
5. [Set up Metal LB](#) to act as a load balancer for your cluster, this will require a public IP address from your Internet Service Provider

Further guidance and details on this process will become available on the GitLab in future, as this process is further tested and verified.

This guide is now complete! Further reading on system components and how you can use them to their best abilities are provided below.

Prometheus - Monitoring system

<https://prometheus.io/docs/introduction/overview/>

Longhorn - Distributed storage

<https://longhorn.io/docs/1.5.3/>

Apache Ozone - Distributed file system

<https://ozone.apache.org/docs/1.3.0/index.html>

Canonical Metal As A Service (MAAS)

<https://maas.io/docs>

Apache Ranger - Data policy manager

<https://ranger.apache.org/>

Apache Kafka - Messaging system

<https://kafka.apache.org/documentation/>

Docker – Containerizing your application

<https://docs.docker.com/get-started/>

Kubernetes – Concepts to understand how Kubernetes operates

<https://kubernetes.io/docs/concepts/>

InfluxDB – Time series database

<https://docs.influxdata.com/influxdb/v2/>

Helm – Manage applications as sets of Kubernetes pods

<https://helm.sh/>

Kubeapps – Large repository of Kubernetes applications

<https://kubeapps.dev/>

4. Conclusions

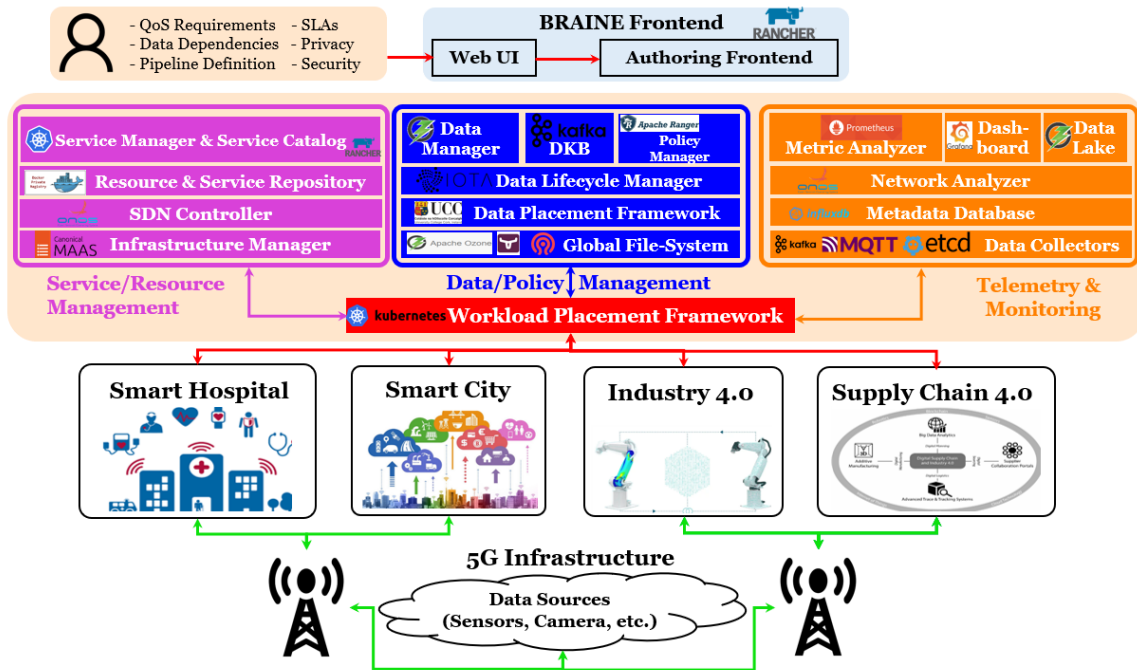


Figure 4.1 BRAINE Public Release Reference Architecture

Overall, the development, integration, and testing of all BRAINE platform components and use cases has been completed. All HW, SW, and use case developments ultimately completed successfully (with an issue and solution in place for the integrated network switch). All HW and SW components have been integrated and demonstrated, and successful demonstration of the use cases with the EMDC and platform has been showcased.

The public release of the BRAINE SW platform has been published. The development of the platform is expected to continue after the project, with both interest from parties within and external to the project. Note: The public version of the BRAINE platform uses only open-source components. Independent private software components which were primarily related to use case applications have been removed from the public release, however this does not impact platform functionality. The public release is intended to be stable and can be modified for production use without any major risk of failure due to using open-source and well-maintained components. Additional experimental features and functions developed within the project at the platform level have been released as open source, and their locations can be found in the Component Appendix B below. The public release software architecture can be seen in Figure 4.1.

The BRAINE platform is already in use in both project partner environments (e.g., Dell manufacturing), and is already continuing development in further EU projects (e.g. CLEVER). Future planned features include:

- Inclusion of a distributed AI platform for AI model training/inferencing at the Edge
- Direct Integration of an internal CI/CD environment, with private Git repository, automated unit testing, Docker image building, and automated production deployment staging.

These developments will be added to the public release over time.

5. Appendix A – Component List

Note: This is identical to D5.4, it is included here as a reference.

5.1 Platform components

5.1.1 WP2

| Component ID | Component Name | Use Cases | Owner |
|---|---------------------------------|-----------|-------|
| C2.1 | EMDC Backplane & CPU/GPU boards | ALL | PCB |
| <p>Component Overview</p> <p>BRAINE 1.0 includes the first release of EMDC hardware components.</p> <p>PCB Design is responsible for the PCB board development. Up till this moment 22 PCBs were defined, some of them will be directly used in the EMDC 2.0 systems, some others are being developed to support electrical /mechanical validation. Some cards are cancelled due to SOW change or component availability issues.</p> <p>The following PCBs were developed directly for EMDC 2.0 system.</p> <ul style="list-style-type: none"> • COMe CPU card, which support Intel and AMD COMe Type VII cards. • GPU Card which supports, Xavier AGX SOM • NVMe card, which support 4x M.2 SSDs • PCIe switch card. • 3kW peak power (4x 750 W) power supply board • BMC (Board Management Microcontroller) card • 2-slot bring-up backplane • Debug PCB (for processing cards) <p>Cards still under development:</p> <ul style="list-style-type: none"> • ARM card support was delayed because component obsolescence. • Ethernet Switch card, using Mellanox Spectrum1 switch, • 11 slot backplane cards <p>The above-mentioned cards were designed and prototyped accordingly in small 2-3 pcs qty. Hardware validation and software integration with BMC is currently ongoing. After successful validation the above PCBs will be produced in higher qty. for EMDC 2.0 systems.</p> <p>Several other PCBs were designed to support mechanical design/ validation.</p> <ul style="list-style-type: none"> • CPU Carrier mechanical dummy • EMDC 1 slot mechanical dummy backplane • Switch mechanical dummy (Examax) • EMDC switch mechanical dummy backplane (Examax) <p>PCBs were designed to support thermal design/ validation. (See C2.2)</p> <ul style="list-style-type: none"> • CPU Carrier thermal dummy • COM Express thermal dummy | | | |

- 4 slot thermal validation backplane

Cancelled PCBs: (partially designed)

- 24 slot backplanes (cancelled)
- 1 slot bringup backplane (replaced with 2 slot bringup backplane)
- BMC Carrier PCB (required only in 24 slot system)
- Xilinx Versal AI PCB, because component obsolescence

UC Relevance:

The above-mentioned PCBs are the electrical hardware building block for the EMDC system. Each testbed will be built from the mixture of these hardware components. All use cases will use the testbed to run selected algorithms.

| Component ID | Component Name | Use Cases | Owner |
|---|---------------------|-----------|-------|
| C2.2 | EMDC cooling system | ALL | JJC |
| <p>Component Overview</p> <p>BRAINE release 1.0 includes the design of a reduced 600W capacity cooling system to validate the working principle and the design choices for the large 1.5kW prototype. This reduced prototype is a 4 slots monobloc that hosts dummy heating modules. The AMD COMe Type7 module being the most critical one in simulations, the dummy heating module planned for the BRAINE 1.0 is mimicking its local power dissipation. Similar to the real AMD module, they are composed with the same form factor PCB as well as its carrier board. The aluminium heat spreader and inner cooler are also similar to the one planned for the real module to validate the thermal conduction path from the back carrier board to the bridge interface. Components with high power dissipation are replaced by power resistances properly chosen to get the corresponding proportional power from 0 to 150 W per module with only one power supply. The dummy CPU and the low power components (DDR4, NVMEs...) can be powered separately giving the possibility to overpower the dummy CPU and test the limits of the cooling system. Additionally, the 4 modules can be powered separately as well to test the ability of the loop thermosyphon to operate with non-uniform heat distribution between modules.</p> <p>The final EMDC cooling system is a gravity-driven loop thermosyphon system with a maximum capacity of 1.5kW and 150W/module. It is composed of the following HW components:</p> <ul style="list-style-type: none"> • 11-slot Monoblock evaporator • Low-pressure drop multi-port tube 19" x 2U condenser <p>The Monoblock evaporator fixed on the main PCB serves as both mechanical and cooling purposes. The heat spreading from the electronic case to the evaporator channels is made with an aluminium heat spreader. An additional inner cooler is needed for the CPU modules to cool the electronic components on their carrier board. The condenser of the loop-thermosyphon is cooled with air in a separate 2U enclosure where 5 fans are included.</p> <p>UC Relevance</p> <p>HW cooling components will be used for all BRAINE UCs.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|--|-----------|-------|
| C2.3 | Nanoparticle fluids for liquid cooling | ALL | SYN |
| <p>Component Overview</p> <p>To increase the heat transport across the solid-liquid interface of a heater, two different approaches are under investigation. The first approach is to use the nanofluids directly where suspended nanoparticles are utilized to intensify the boiling process, whereas the second approach is to use nanoparticles deposited at the heater surface to intensify the heat transport via the interface. In both approaches, the intention is to use the R1233zd(E) refrigerant as a base fluid. Graphene, alumina, Graphene Oxide nanoparticles are under investigation with R1233zd(E) refrigerant, to access the stability of the nanofluids, affinity to surfaces and the refrigerant for uniform deposition.</p> <p>UC Relevance</p> <p>The nano refrigerant or/and the deposition of nanoparticles can improve HW cooling performance in all BRAINE UCs.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|------------------------------|-----------|-------|
| C2.4 | EMDC board mounting assembly | ALL | HID |
| <p>Component Overview</p> <p>C2.4 includes a customized 3U 19" rack to accurately mount the EMDC board. The 3U rack supports the EMDC board to reduce stress on the EMDC board while assembling/connecting the various modules (e.g., CPU module, NVME module, etc.) to the EMDC board. Furthermore, it provides a sufficient EMI conductance on the connectors of the EMDC board and a user-interface to identify correct performance of the modules.</p> <p>The 3U rack is compatible with all 19" systems and it is expected to be integrated into TU/e testbed.</p> <p>Additionally, the thermal mechanics (heat sinks and inner coolers) for the finished PCB modules have been mechanically designed (Power, CPU and NVME).</p> <p>UC Relevance</p> <p>This HW component will be used for all UCs.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|---------------------|-----------|-------|
| C2.5 | 5G RRH Demonstrator | UC2 | COM |
| <p>Component Overview</p> <p>To enable 5G signalling at the edge a remote radio head demonstration platform has been developed that can transmit and receive traffic over the air interface. This enables testing with UE. The data are handed off to O-DU through high-speed Ethernet interfaces making use of the O-RAN standard. The radio is realized with Intel Arria10</p> | | | |

FPGA and Analog Devices Madura transceiver as main components on a proprietary developed PCB.

UC2 Relevance

The 5G RRH demonstrator platform will enable wireless connectivity to the network in both uplink and downlink direction. More specifically UE's will be able to connect to the edge through a 5G wireless link.

| Component ID | Component Name | Use Cases | Owner |
|--|-------------------------|-----------|-------|
| C2.6 | Quantum safe fiber link | UC1 UC3 | TUe |
| <p>Component Overview</p> <p>C2.6 includes the HW and SW to implement a quantum safe fiber link for high-capacity edge-to-edge communication; the component is expected to be integrated into TUe testbed. Security will be based on layer 2 symmetric encryption with the key exchange based on quantum key distribution (QKD).</p> <p>C2.6 will include the following HW components: two 256-AES encryptors capable to operate up to 100 GbE, which is compatible with BRAINE EMDCs, QKD Tx and Rx based on the coherent-one-way protocol, wavelength division multiplexing (WDM) components and an external server to run the key exchange management service, which establishes the connection between the encryptors and QKD modules.</p> <p>UC1 and UC3 Relevance</p> <p>UC1 requires the analysis of privacy sensitive information from the patients of the clinics or other healthcare related centres. This data is expected to be processed locally at the edge; however, this may still require the transport of the data from the clinic to the edge node via fiber. Moreover, multiple clinics are expected to share their data with a main academic medical centre (AMC) hub, where the data is stored and further processed. We QKD technology to provide an extra layer of security to the fibre links connecting the edge nodes in the clinics with their AMC hub. Compared to classic asymmetric encryption, such as RSA, QKD enables to share secret keys between two nodes without being vulnerable to quantum computer attacks. C2.16 emulates an edge-to-edge connection between an EMDC located in a clinic and another EMDC in an AMC.</p> <p>The same scenario can be applied to UC3 when multiple industrial facilities need to share data at the edge and sensitive information, such as design information, is shared.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-----------|
| C2.7 | P4 Program | UC2 | MLNX/CNIT |
| <p>Component Overview</p> <p>The BRAINE edge solution leverages on a P4-based programmable switch for advanced networking functionalities.</p> <p>The first version targets the Mellanox Spectrum1 ASIC that will be included in the BRAINE EMDC HW developed by Task T2.1. This version encompasses:</p> <ul style="list-style-type: none"> • P4-based layer 2-3-4 forwarding • Traffic steering <p>UC2 Relevance</p> | | | |

UC2 will implement the flows forwarding instantiation using the BRAINE SDN controller.

| Component ID | Component Name | Use Cases | Owner |
|---|----------------|-----------|-----------|
| C2.8 | Switch OS | UC2 | MLNX/CNIT |
| <p>Component Overview</p> <p>Proprietary EMDC Ethernet switch OS software with P4 programmability.</p> <p>The first version is designed for the Mellanox spectrum1 ASIC as standalone, off-the-shelf switch currently implemented in the BRAINE testbed at CNIT Lab. This version is based on ONYX over ONIE, augmented with a specifically designed P4 docker container released by Mellanox.</p> <p>The second version targets the Mellanox spectrum2 ASIC as standalone, off-the-shelf switch. This version is based on SONiC over ONIE, augmented with a different P4 docker container released by Mellanox. So far, this version has been tested on Spectrum 1 switch. Given the different underlying ASIC and APIs, only limited P4 capabilities and pluggable module configurations have been implemented over SONiC.</p> <p>UC2 Relevance</p> <p>It enables flow-matched routing, forwarding and steering automation the EMDC SDN switch with P4 data plane programmability, controlled by the ONOS Controller.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|---|-----------|-------|
| C2.9 | SW agent for transceiver configuration and monitoring | UC2 | CNIT |
| <p>Component Overview</p> <p>An open-source NETCONF Agent supporting OpenConfig YANG model has been developed to enable the instantiation of optical connectivity in metro networks connecting EMDCs. The agent may operate inside both fully optical and packet-optical nodes. The agent includes connectivity provisioning and adaptation functions triggered by the SDN controller. Moreover, it is extended with on-demand gRPC-based optical telemetry of selected KPI (e.g., coherent receiver OSNR and pre-FEC BER).</p> <p>UC2 Relevance</p> <p>Possible deployment of inter-EMDC smart-city connectivity resorting to SDN metro networks encompassing packet-optical switches. On-demand telemetry of the optical connection quality of transmission is also supported to guarantee seamless service continuity.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-------|
| C2.10 | 5G NR Model | UC2 | ISW |
| <p>Component Overview</p> <p>The 5G NR model targets to perform the high PHY layer blocks procedures for gNB transmitter (CRC addition, LDPC/Polar Channel Coding, Rate matching, scrambling, modulation mapper, layer mapper and resource mapper) and gNB receiver (synchronization, channel estimation & equalization, resource de-mapping, descrambling, modulation de-mapping, LDPC/Polar channel decoding, CRC detection).</p> | | | |

The output of this model in the DL should be delivered to low PHY where it needs to be processed and delivered to the user equipment's (UEs). In addition, the model should process the data delivered from the low PHY towards high PHY before delivering it to MAC layer.

UC2 Relevance

The model will perform the high PHY blocks processing where it is needed to process the data received from the UEs then deliver it to MAC layer in a non-real time manner (offline data exchange with low PHY and MAC).

| Component ID | Component Name | Use Cases | Owner |
|--------------|----------------|-----------|-------|
| C2.11 | vRAN Prototype | UC2 | ISW |

Component Overview

The vRAN prototype is the ORAN based set of radio protocols (O-RU, O-DU, O-CU) and the Core. All of the functions are virtualized using the K8, except the O-RU which now is dockerized. The components together provide 5G connectivity to external 5G UE, when attached to the USRP B210 radio with appropriate frequency license (or alternatively over cable). It is important to assume that at the current stage USRP B210 should be connected over USB cable, as Ethernet connectivity will be added later.

UC2 Relevance

VRAN prototype enables e2e connectivity for the UC2. It will be provided to be used in the CNIT testbed and deliver 5G service for the user terminals (e.g. UE with camera attached).

| Component ID | Component Name | Use Cases | Owner |
|--------------|----------------|-----------|-------|
| C2.12 | MBMC software | ALL | BME |

Component Overview

C2.12 component is a software code used in every MBMC module. This software component is based on the STM32L4R5VGT6 microcontroller. The software consists of the following main modules:

- Low level microcontroller abstraction layer: functions configuring the basic I/O-s and peripherals of the controller.
- Higher level board independent and dependent monitoring and supervisory functionality
- CDC-Ethernet communication through USB with the BMC controller. This communication is based on TCP sockets, one TCP socket is used for general purpose command type communication, and other sockets can be used for board specific purposes, like firmware upgrade.

HW Relevance

This software performs the core monitoring and controlling functionality of a BRAINE card. This functionality includes power supply control and monitoring, temperature monitoring, and another card dependent functionality (e.g. BIOS update). The MBMCs are supervised by the BMC.

| Component ID | Component Name | Use Cases | Owner |
|--------------|----------------|-----------|-------|
|--------------|----------------|-----------|-------|

| | | | |
|--|--------------|-----|-----|
| C2.13 | BMC SW, V1.0 | ALL | PCB |
| <p>Component Overview</p> <p>C2.13 Component is the first release of the embedded Linux image running on the BMC Card. This component plays critical role in the EMDC hardware while it acts as a system supervisor.</p> <p>The key hardware related functionalities:</p> <ul style="list-style-type: none"> • Card identification and management over I2C (and USB) <ul style="list-style-type: none"> ◦ FRU EEPROM identification ◦ Firmware upgrade on BMMC units. • Communication with BMMCs via USB after booting. <ul style="list-style-type: none"> ◦ The BMC is the USB host, each card is a USB Device (see C2.12) • KVM support (Keyboard Video Mouse) <ul style="list-style-type: none"> ◦ PCIe target within the BMC, can be assigned to each node via the PCIe network. Thus, the CPU root complex there sees BMC as a video tool. ◦ USB device emulation within the BMC, i.e. virtual mouse and keyboard. <ul style="list-style-type: none"> ▪ The BMC is the USB device, the cards are USB Hosts • ESPI / LPC support (for testing) <p>Key (higher level) software functionalities:</p> <ul style="list-style-type: none"> • The Linux built is based on OpenBMC, widely used in OCP projects as server management. • Custom build (kernel, device tree, filesystem, uboot) was built to support EMDC specific hardware. • iKVM functionality over VNC was tested on Congatec evaluation board. • SOL (Serial over Lan) was tested on Congatec evaluation board. • BMMC communication was successfully validated on BMC PCB • Card database was built using SQLite. • Redfish endpoints were tested. • Python based service to collect card telemetry was defined, and currently under development on the 2 slot bringup backplane. • InfluxDB telemetry interface under definition. <p>UC Relevance</p> <p>The BMC acts as a system supervisor, it identifies the populated cards, and manages power / sleep states. Moreover, hardware related telemetry data is gathered using this component. iKVM functionally is required to install and maintain low level firmware on CPU board.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|----------------|-----------|-------|
| C2.14 | FPGA Node | UC2 | BME |
| <p>Component Overview</p> <p>C2.14 is a firmware component consisting of the base hardware design and the SW components of the FPGA node:</p> <ul style="list-style-type: none"> • The base FPGA platform design which can be extended with application-specific acceleration functions. • The Linux operating system for the FPGA platform. | | | |

- Docker integration allowing containerized FPGA accelerated applications.
- Software interface for management and application deployment.

UC2 Relevance

The FPGA hardware platform and software components (firmware) will provide an efficient platform for accelerating various computations. UC2 requires processing a large number of video information. An AI inference accelerator application based on the FPGA node will provide high performance and energy efficiency for such use case.

| Component ID | Component Name | Use Cases | Owner |
|---|----------------------------|-----------|-------|
| C2.15 | OpenCL program for low PHY | UC2 | SSSA |
| Component Overview SSSA implemented the offloading of some virtual Distributed Unit (vDU) functions onto an FPGA. Specifically, the offloaded functions are the Inverse Fast Fourier Transform (IFFT) and the Cyclic Prefix (CP) addition in Orthogonal Frequency Division Multiplexing (OFDM) signals. FPGAs can be a very good accelerator for these functions since they offer almost deterministic latency and high processing capacity per Watt. The implementation is based on OpenCL to integrate it with other 5G functions in the vDU. | | | |
| UC2 Relevance The accelerated functions will be integrated with an open-source mobile stack to support the transmission of data from cameras to the EMDC. In this way the potential advantages of offloading some mobile stack function into programmable hardware will be shown, such as reduced latency and lower energy consumption. | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|--------------------------------|-----------|-------|
| C2.16 | VTU – Virtual transcoding unit | UC2 | ITL |
| Component Overview The VTU can convert audio and video streams from one format to another. The source stream can originate from a file within the local storage system, or maybe a packetized network stream. The requested transcoding service can be monodirectional, as in video streaming, or bi-directional, like in videoconferencing. The transcoding capabilities of the VTU are provided by Libav ^[1] . It is an open-source library, which can handle a wide variety of audio and video coding standards. For the most computationally intensive video encoding tasks, the VTU relies on GPU resources. | | | |
| UC2 Relevance Within UC2 the VTU is used for interfacing the cameras that acquire video with applications capable of performing video analysis using Deep Learning. The VTU has been extended to take advantage of available domain-specific hardware in the platform (e.g., GPUs), and to enable run-time reconfigurability of the transcoding configuration. This allows better performance scaling for parallel flows, as well on-the-fly format changes to reduce data transfer requirements. | | | |

| Component ID | Component Name | Use Cases | Owner |
|---------------------------|----------------|-----------|-------|
| C2.17 | N2Net | UC2 | NEC |
| Component Overview | | | |

N2Net provides the ability to automatically translate neural networks into network data plane programs to implement machine learning analysis directly on the network traffic, and within network devices. N2Net works as a compiler starting from a Binary Multi-Layer Perceptron (MLP) description, and generates code in data plane programming languages, such as P4, to implement such MLP. While the size of the MLP is generally limited given the constraints of the network data plane, this still gives enough flexibility to implement advanced analytics tasks.

UC2 Relevance

The analysis performed directly on the network traffic enables scalable handling of network traffic classification and monitoring tasks. For instance, N2Net can be applied to the scalable monitoring of security issues at network traffic level, which might affect the performance of the UC2 application-level data analysis.

| Component ID | Component Name | Use Cases | Owner |
|---|------------------------|-----------|-------|
| C2.18 | Quantum safe readiness | UC2 | SIC |
| <p>Component Overview</p> <p>Quantum computing is a new paradigm of computer implementation. It may enable to speed up some specific algorithm in a way that cryptanalysis, which was unfeasible with classic computer, can be computed in a reasonable amount of time, thus defeating currently, widely used, computationally secure ciphering such as RSA or elliptic curve cryptography. Today (official) quantum computer implementation state of art does not allow to perform such attack. However, the threat and the impact are so huge that countermeasure must be prepared from now on and has been taken into account worldwide, in the US, Europe, Japan, China.</p> <p>The Braine project has thus followed closely research, competition, and standardization process on Post Quantum Cryptography (PQC) which is the name for ciphering algorithms expected to be robust (computationally secure) against cryptanalysis on quantum computers. Outcomes from the worldwide NIST competition for PQC, the ETSI, European National Cyber Security agencies (ANSSI, BSI, ...) are used to prepare quantum safe readiness, that is to say identify and inventory weakness and changes prepare.</p> <p>This is done with ISW on the crypto library used and tailored by this partner and with TU/e Quantum Key Distribution prototype.</p> <p>UC2 Relevance</p> <p>Software libraries developed by partner ISW is widely used un UC2. The current work contributes to the cybersecurity of the whole software.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------------------------|-----------|-------|
| C2.19 | Low bitrate blockchain protocols | UC1 | IMC |
| <p>Component Overview</p> <p>C2.19 is a software and hardware component implementing two IoT-supporting blockchain protocols:</p> | | | |

1. Sequencer demonstrator (a sealed unit communicating via WiFi with the EDC and via LoRa broadcast with IoT "things"). It is based on the standard ESP32 evaluation board TTGO with firmware programmed as per point 2 below.
2. Sequencer firmware implementing the PLS protocol for posting blocks on the chain.
3. SLVP protocol implementation in Python for non-IoT clients
4. SLVP protocol implementation in C++ for IoT clients
5. Fog Server (running on the EDC) implementation in Python, including CAS, the Content Addressable Storage.

UC1 Relevance

The above will achieve non-repudiation of all transactions performed by both small embedded systems and external users. This is essential for the use case ("smart hospital" and "assisted living" scenarios, as it provides legal proofs of action by an identifiable actor referenced to an organisation-wide immutable sequence of records. The technology will be able to provide irrefutable proof of actions that embodies cause-and-effect relations between them as well as proof of provenance and originator. On the one hand, nonrepudiation will enable a much higher level of automation without risk of false claims of medical malpractice. On the other hand, personal data could still be protected despite the use of a blockchain with its attendant broadcasting of records, thanks to the permissioned nature of the blockchain architecture.

| Component ID | Component Name | Use Cases | Owner |
|--|---------------------------|-----------|-------|
| C2.20 | Smart sensors integration | UC4 | IFX |
| <p>We show smart sensors integration with two demonstrator components that include both hardware (the sensors), and intelligent data processing & communication on the edge (AI algorithms and cloud/edge communication). The initial component environment is composed of the following two demonstrators: (1) the "smart wristband for voting" platforms are physically distributed and represent the sensor nodes, while a display presentation of the wristband's height, retrieved from the barometric pressure, represents the actuator side. Beyond that, (2) we are investigating (in another set-up) possibilities to design a biometric identification device that captures Photoplethysmograph (PPG) signal via an infra-red sensitive PALS sensor (PALS2). Both demonstrators have in common that we are establishing wireless communication – first, on cloud level integrating the Arrowhead framework as service-oriented architecture; and second, by moving towards edge communication using EMDC in BRAINE. This way, by adding advanced data processing through AI algorithms on the edge, we demonstrate how an intermediate wireless communication gateway serves as the computing node.</p> <p>UC4 Relevance</p> <p>With this component, we contribute to the goal of UC 4 enabling a wireless equipment and process control for supply chain improvements for semiconductor manufacturing and supply chains containing semiconductors. We are approaching it by demonstrating Arrowhead Framework integration and advanced signal processing with AI algorithms on the edge. This way, we aim to improve automation capabilities in the factory and on supply chain levels, getting closer to an intelligent edge platform. We are specifically focusing on security and data plausibility aspects.</p> | | | |

5.1.2 WP3

| Component ID | Component Name | Use Cases | Owner |
|---|----------------|-----------|-------|
| C3.1 | Service Mesh | UC1, UC2 | VMW |
| Component Overview <p>The Service Mesh allows service discovery and cross EMDC communication between BRAINE components.</p> <p>UC1/2 Relevance</p> <p>For UC1 the Service Mesh can provide connectivity to centralized medical systems APIs with minimum efforts, costs, and timing.</p> <p>For UC2 the Service Mesh connects different services which process large volumes of audio and video information collected from different HW sources.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|----------------|-----------|-------|
| C3.2 | Image Registry | ALL | VMW |
| Component Overview <p>The Image Registry is a core part of the Kubernetes platform which orchestrate workloads in BRAINE. It contains docker images of all software components in BRAINE.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|---------------------------------------|-----------|-------|
| C3.3 | Telegraf agent for 5G data collection | UC2 | SSSA |
| Component Overview <p>A Telegraf agent has been implemented to integrate Mosaic 5G FlexRAN and Kafka. Mosaic5G FlexRAN is an SDN controller which implements RAN control interfaces on top of the OpenAirInterface code. Kafka is utilized to stream real time performance data that are used for mobile network optimization based also on forecasting. Telegraf is a plugin-driven server agent used to collect metrics and to report events from different sources to different destinations. It offers a very low memory footprint, and a very good deployment flexibility. Plugins are mainly used to gather metrics and send them to specific endpoints, but they can also aggregate or even pre-process them. For this specific case, the agent has been configured with an input plugin, called "execd", which periodically invokes a Python script. The script retrieves the mac statistics through an http GET request to the FlexRAN /stats endpoint. For each query, the script parses the returned JSON object and keeps only the information about each UE attached to the system. For each UE, it creates an ad hoc JSON metric, which inherits any significant information from the original structure. The output metrics follow the InfluxDB schema, using as tags the BS identifier and the UE identifier. Each of them is then sent, using a Kafka producer output plugin, over a dedicated Kafka topic.</p> <p>UC2 Relevance</p> <p>The data collected through the Telegraf plugin (e.g., the Channel Quality Indicator --- CQI) are utilised to implement a proactive Radio Resource Management (RRM) utilized by the Real Time RAN Intelligent Controller (RIC) implemented by Mosaic5G FlexRAN. When the CQI decreases, the mobile network PHY reacts by decreasing the modulation rate and increasing the code length to maintain a low bit error rate (i.e., adaptive modulation and coding). In this way however, if the camera UE is assigned the same number of Physical Resource Blocks (PRBs) will have less capacity to transmit the</p> | | | |

video. Thus, the video quality could be impacted. The RRM proactively assign more PRBs to the camera UE when the CQI is forecast to decrease to assign the UE the same capacity and not to impact the quality of the video transmission.

| Component ID | Component Name | Use Cases | Owner |
|--|---------------------------|-----------|-------|
| C3.4 | Distributed Knowledgebase | ALL | DELL |
| <p>Component Overview</p> <p>Distributed Knowledgebase (DKB) is a service that gathers information on resource availability from various nodes and entities in the system and makes them available to various controllers and orchestrators. The DKB is an integral part of edge resource collaboration and ensures that resource information acquired from one edge device is shared with other devices in its neighbourhood, creating resilient cells of edge devices. The DKB is designed to be reliable and fault-tolerant and scales well with the increasing number of edge nodes.</p> <p>Platform Relevance</p> <p>The DKB is one of the key components that forms the core functionality of the BRAINE platform. The use-case applications do not directly communicate with the DKB; however, they interact with components that utilize the information acquired from the DKB. Within an EMDC, the DKB integrates with various components such as Corporate Memory Tool (CMEM) and provides information on available resources and current resource utilization. Across EMDCs, the DKB distributes resource information gathered by one EMDC with other EMDCs. This distribution happens based on administrator policies and restrictions. The shared information is used by inter-EMDC components, such as cluster-level schedulers to compute efficient placement strategies.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|------------------------------|-----------|-------|
| C3.5 | Forecasting functional block | UC2 | SSSA |
| <p>Component Overview</p> <p>A forecasting module has been developed by SSSA for forecasting network parameters based on current and past measurement data. The forecasting module is based on both traditional statistical analysis techniques and AI/ML techniques. The current version of the module uses a LSTM neural network to provide a forecasted value of data related to the mobile network transmission parameters coming from the component C3.3. More specifically, the current wideband channel quality indicator (WCQI) for the downlink is read from a Kafka topic, and a forecasted value is produced in output to another Kafka topic.</p> <p>UC2 Relevance</p> <p>The forecasting module is used to produce in output the forecasted WCQI values. In this way, it is possible sending those data to the Mosaic5G FlexRAN RIC RRM function, so that resources can be increased or decreased according to future needs.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---------------------------|---------------------|-----------|-------|
| C3.6 | BRAINE RL Scheduler | ALL | LUH |
| Component Overview | | | |

This component is an ML-based scheduling plugin that integrates with the Kubernetes scheduling system in order to score and eventually select a proper worker node for each and every given workload that is submitted to the Kubernetes cluster. It works with a DRL-trained model via a REST-based inferencing API to predict the proper nodes. In order to obtain the available resources of the worker nodes in real-time, this component utilizes a data access agent that interacts with the cluster's central telemetry database.

Platform Relevance

This component is transparent to the applications yet important to the proper functioning of the system. It schedules the incoming workload in order to fulfil higher optimization objectives such as overall energy reduction or the reduction of the average waiting time of the applications.

| Component ID | Component Name | Use Cases | Owner |
|---|--------------------|-----------|-------|
| C3.7 | Telemetry Software | ALL | LUH |
| Component Overview <p>This is a compound module consisting of multiple components that together shape the telemetry infrastructure of the system. Its components are a set of off-the-shelf and in-house-built metric collectors, a metric harvester that ingests the metric data from the collectors, a database that stores and make available the current and historic metric data, and a dashboarding and visualization frontend.</p> <p>Platform Relevance</p> <p>The metric data consists of network telemetry data, system (worker nodes) resource status (availability, capacity) data, and application metrics (coming from use-cases). These data are maintained for different purposes including scheduling, SLA conformance evaluation, SDN rerouting, system health monitoring, and application-specific monitoring and analysis.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|-----------------------|-----------|-------|
| C3.8 | MOD – Learning module | UC3 | FS |
| Component Overview <p>The learning module of the MOD application is used for learning state-of-the-art machine learning models. The learning module requires access to the influx DB, where found motifs by the Discovery module (WP4-T4.2) are stored. On found motifs, the detection models are learned, and the dictionary of motifs is created. The dictionary of motifs is then stored in influx DB for use by the Detection module and is sent to the cloud for use by the Digital Twin to reconstruct the continuous data stream in the cloud.</p> <p>UC3 Relevance</p> <p>This component enables the MOD application to learn detection models for online detection in the UC3.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--------------|----------------|-----------|-------|
|--------------|----------------|-----------|-------|

| | | | |
|--|--------------------|-----|-----|
| C3.9 | Image Orchestrator | ALL | ECC |
| <p>Component Overview</p> <p>The image orchestrator register, update and remove images at the global image register according to the user's needs. When an image needs to be registered, the Image Orchestrator reads the image deployment constraints from the Service & Registry catalogue and perform the deployment on the global image registry updating the image status when needed.</p> <p>Platform Relevance</p> <p>The Image Orchestrator perform the deployment of an image in the Global Image Register making it available to be used by platform users when defining services and workflows.</p> | | | |

| | | | |
|--|----------------|-----------|-------|
| Component ID | Component Name | Use Cases | Owner |
| C3.10 | Metrics Relay | ALL | ECC |
| <p>Component Overview</p> <p>The Relay is responsible for extracting metrics from Kubernetes and instantiate respectively that information on the data model such as Computational Resource allocation and capacity from the Nodes (see T3.3) resources available on each Node as well as their condition. The Relay also makes use of the Kubernetes Metrics server that provides periodic updates on the Node Memory and CPU consumption.</p> <p>Platform Relevance</p> <p>The relay allows BRAINE platform partners to monitor resource consumption as while a service is being executed and to correctly identify the available computational resources.</p> | | | |

| | | | |
|---|-----------------|-----------|-------|
| Component ID | Component Name | Use Cases | Owner |
| C3.11 | BRAINE Ontology | ALL | ECC |
| <p>Component Overview</p> <p>The BRAINE Ontology consists in a vocabulary that is used to define the abstractions such as Nodes, Pods, Images and Services of the Resource & Service catalogue which facilitates the platform management.</p> <p>UC4 Relevance</p> <p>The BRAINE Ontology is used to store and define platform resources such as Nodes, Pods, Images, and services.</p> | | | |

| | | | |
|--|------------------------|-----------|-------|
| Component ID | Component Name | Use Cases | Owner |
| C3.12 | Knowledge Bootstrapper | ALL | ECC |
| <p>Component Overview</p> <p>The Knowledge Bootstrapper is responsible for populating the catalogue with information such as data types and data model (classes, properties). It also is responsible for transferring of information between the Kubernetes Cluster (Node) and the BRAINE Service & Resource Catalog. The bootstrap system reads information from</p> | | | |

the Kubernetes Cluster or from a Kafka topic in a specific address and pushes it to the Service & Resource Catalog.

Platform Relevance

The Knowledge Bootstrapper allows the platform to define available resources to be used when running services and to instantiate the Data Catalog with schema abstractions defined in the BRAINE ontology.

| Component ID | Component Name | Use Cases | Owner |
|---|----------------|-----------|-------|
| C3.13 | SDN Controller | UC2 | CNIT |
| Component Overview <p>The SDN controller will configure the network layer aiming at both enabling the traffic forwarding (with the required QoS) and the traffic monitoring toward the telemetry system. In its first version the BRAINE SDN controller was released with a northbound application (i.e., the BRAINE app) opening a REST APIs toward the K8s orchestrator and other BRAINE components. In this period a companion application has been developed, tested, and demonstrated in an international conference to enable the discovery of PODs deployed by K8s and the forwarding and monitoring of traffic exchanged among PODs.</p> <p>UC2 Relevance</p> <p>With this additional application the SDN controller enables the matching of the traffic at the PODs level (assuming K8s working with the Flannel tool in VXLAN mode) thus enables the specific monitoring of each traffic flow with the required granularity. This enables the detection of QoS degradation (e.g., latency degradation) by the telemetry system that can provide feedback to the SDN controller itself to take actions on the network aiming at recovering the QoS requirements satisfaction.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|--------------------------|-----------|-------|
| C3.14 | Audio Inferencing Engine | UC2 | MAI |
| Component Overview <p>The audio inferencing engine enables real time analysis of digital audio using deep neural network architectures. Computation of forward passes of the neural networks is accelerated using CUDA/CuDNN and can thus be processed efficiently on EMDC hardware. The engine is configured using JSON files representing computational graphs. Nodes in the graph represent stages of processing. Graphs can be easily extended / modified to suit a specific use case.</p> <p>UC2 Relevance</p> <p>The audio inferencing engine allows for audio classification and audio detection in various environments, such as cities and urban areas in UC2.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-------|
| C3.15 | SLA Broker | UC2/ALL | DELL |
| Component Overview <p>BRAINE introduced SLA Broker to collect deployment, services, resources, or traffic performance from the telemetry system to meet the SLA constraints. Then, the SLA</p> | | | |

Broker analyse the telemetry data and issues alarms to responsible actuators (e.g., the orchestration framework) in case of SLA violations to handle the violation based on the SLA agreement.

UC2 Relevance

SLA Broker will be used to notify the orchestration framework or the SDN controller in case of relative SLA violations.

| Component ID | Component Name | Use Cases | Owner |
|--------------|--------------------------------------|-----------|-------|
| C3.16 | Multi-access Edge Computing platform | UC2 | SMA |

Component Overview

5G network has introduced the ability to support low-latency services, in particular for those use cases that requires real-time response. To achieve this goal, the computation has to be moved closer to the end users (i.e., at the network edge). MEC technology represents the answer to these needs since the processing can be easily moved at the customer premise with the advantage of low latency communications, bandwidth reduction and more security.

For these reasons, BRAINE considers the implementation of a MEC platform capable of deploying the service computation from a centralised cloud to the network edge. MEC environment is also used for a performance comparison against the main BRAINE architecture.

UC2 Relevance

MEC platform will be exploited for the Smart City use case (i.e., UC2) where real-time notifications play a crucial role in preventing hazardous situations. Thus, moving processing towards the end users helps in reducing communication latency.

| Component ID | Component Name | Use Cases | Owner |
|--------------|----------------------------|-----------|-------|
| C3.17 | AI image processing engine | UC2 | SMA |

Component Overview

This component is responsible for AI processing of the video streams captured from cameras deployed around a municipality (e.g., traffic cameras). The objective of the AI image processing is to track the road users to foresee dangerous conditions. For this reason, the AI developed has to be fast and reliable to have an efficient scene reconstruction in real-time manner.

UC2 Relevance

Artificial Intelligence represents one of the key technologies for UC2 (Smart City scenario). Along with AI for audio processing, AI for video processing is essential for monitoring the city and preventing accidents.

| Component ID | Component Name | Use Cases | Owner |
|--------------|---------------------------------------|-----------|-------|
| C3.18 | Edge-to-edge multiagent communication | UC3 | CTU |

Component Overview

The multi-agent communication format is based on FIPA ACL structure. The message format is inspired by HTTP protocol where metadata and content are also separated. As a communication protocol is used AMQP that with use of RabbitMQ broker ensures low response time.

UC Relevance

Will be used to enable negotiation between agents. At the same time server can be utilized as a general MQTT, AMQP broker.

| Component ID | Component Name | Use Cases | Owner |
|--|-----------------|-----------|-------|
| C3.19 | Placement Agent | UC2 | ISW |
| Component Overview Placement agent supports operation of admission control inside the 5G radio stack. It utilizes the inputs from C33.07 (Parameter Predictor) which are the predicted SNR/SINR per user as well as CPU/memory of the underlying EMDC. It can support admission control, and interface with K8 scheduler (L1 or L2) in order to instruct it to provision additional resources were indicated by the C33.07. Placement Agent can also activate complementary functions (e.g. Traffic Steering) when admission control alone has no resources to assign to UE due to resource shortage (radio spectrum, CPU, memory, etc). | | | |
| UC Relevance Placement agent is complementary functionality that supports call admission control. So, in case of scenario where admission control will be used, or where the prediction capabilities of C33.07 (Parameter Predictor) are going to be showcased the Placement Agent can be used to provide additional capabilities of EMDC-driven vRAN scaling (either local L1 or federated L2). | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|---------------------|-----------|-------|
| C3.20 | Parameter Predictor | UC2 | ISW |
| Component Overview Parameter predictor offers set of prediction algorithms (NN, MDP, LSTM) which are used to learn prediction of vital radio or computing parameters. Such parameters can be SNR, SINR, CPU, memory usage or others. The various configurations of Parameter Predictor will assure flexibility of selection of best solution. This component is direct support for the admission control algorithms inside vRAN. | | | |
| UC Relevance This component together with Placement Agent and scheduler would be able to further optimize network behaviour for UC2 (e.g. maximize total amount of connections, minimize probability of blocking or dropping). | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|---|-----------|-------|
| C3.21 | BRAINE ML-based L1 Kubernetes Scheduler | UC2 | ISW |
| Component Overview ML-based L1 scheduler for vRAN is foreseen as preliminary solution in order to combine knowledge of radio resources and computing resources and assure smooth and in- | | | |

advance scaling of the vRAN networks inside EMDC servers. It enables smart scaling of the resources for vRAN while assuring reaching the QoS targets for the end users, as well as provide adjustment capabilities for operators of future ORAN based 5G vRAN networks.

UC Relevance

Will be used in order to support resource scaling for vRAN only for the preliminary phase of developing the target, which is the L2 scheduler model (C3.22).

| Component ID | Component Name | Use Cases | Owner |
|--|---|-----------|-------|
| C3.22 | BRAINE ML-based L2 Kubernetes Scheduler | UC2 | ISW |
| Component Overview ML-based L2 scheduler for vRAN is foreseen as target solution in order to combine knowledge of radio resources and computing resources and assure smooth and in-advance scaling of the vRAN networks deployed <u>between EMDC servers</u> . It enables smart scaling of the resources for vRAN while assuring reaching the QoS targets for the end users, as well as provide adjustment capabilities for operators of future ORAN based 5G vRAN networks. The L2 scheduler will benefit from disaggregated vRAN designs in order to allow distributed placement when current efficiency needs to be expanded. | | | |
| UC Relevance Will be used in order to validate resource scaling optimizations for vRAN (C2.11). | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|-------------------------------|-----------|-------|
| C3.17 | Telemetry monitors & exporter | UC2/ALL | MLNX |
| Component Overview Telemetry monitors are deployed on the network element and are responsible to extract telemetry data and stream it to telemetry adapter using gRPC protocol. Multiple telemetry adapters are supported simultaneously, enabling extraction of different data according operator requirements and network node capabilities. As a basic framework telemetry monitors can provide a data about BW consumption and port latency distribution. Another way to consume telemetry monitor is to deploy a dummy telemetry generator to test whole system end-to-end. | | | |
| UC Relevance Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|-------------------|-----------|-------|
| C3.17.1 | Telemetry adapter | UC2/ALL | MLNX |
| Component Overview Telemetry adapter is deployed in EMDC as a singleton and is responsible to collect streaming telemetry from all network nodes, pre-cache them and reformat to YANG | | | |

schema. Once collector will connect to adapter node, it will be capable to subscribe to telemetry data of interest and will start to receive it in real-time.

The data is formatted in well-known YANG schema so generic subscriber can ask to receive data or some filtered subset of data. Data is generic and every subscriber can receive needed portion.

UC Relevance

Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality

| Component ID | Component Name | Use Cases | Owner |
|--|--------------------|-----------|-------|
| C3.17.2 | Telemetry Ingester | UC2/ALL | MLNX |
| <p>Component Overview</p> <p>Telemetry ingester is implemented as a Telegraf agent that connects to YANG enabled telemetry adapter and injects telemetry to InfluxDB.</p> <p>Telegraf is a plugin-driven server agent used to collect metrics and to report events from different sources to different destinations. It offers a very low memory footprint, and a very good deployment flexibility. Plugins are mainly used to gather metrics and send them to specific endpoints, but they can also aggregate or even pre-process them.</p> <p>Each telemetry record is processed and divided to key/tags/attributes and exported to database. Selection of keys and attributes is telemetry specific and can be adapted for specific use.</p> <p>UC Relevance</p> <p>Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality.</p> | | | |

5.1.3 WP4

| Component ID | Component Name | Use Cases | Owner |
|---|------------------------|-----------|-------|
| C4.1 | Data lifecycle manager | ALL | DELL |
| <p>Component Overview</p> <p>The data lifecycle manager has three key tasks:</p> <ul style="list-style-type: none"> • Receive input from the policy manager (C4.2) on how to handle data and enforce it on data processing applications. • Tag user data with metadata as it changes state and progresses through its lifecycle. • Store this metadata in such a way that it cannot be modified or tampered with, to ensure verifiable auditability. <p>Platform Relevance</p> | | | |

The data lifecycle manager is a key component of the BRAINE platform and can be applied to each application that collects or processes data in any way. This also includes applications that provide the storage system for data (C4.3). To ensure that policies (C4.2) defined by data controllers are enforced by the lifecycle manager, an enforce plugin is applied on top of the application. This enforcer intercepts client requests and accepts or denies them based on the defined policies. For proof-of-concept, a plugin for the storage system will be demonstrated. All requests (whether accepted or denied) are persisted to the distributed ledger. This component is built on top IOTA blockchain and maintains an immutable record of events that occur at various stages of data lifecycle.

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-------|
| C4.2 | Policy Manager | ALL | DELL |
| <p>Component Overview</p> <p>Policy manager maintains a list of policies and regulations defined by data controllers. The operation of the data lifecycle manager (C4.1) and underlying file system (C4.3) will be governed by data policies set within the policy manager. Besides any data management policies set by the end user for their data, the platform as an entity must ensure compliance with other regulations, such as GDPR if operating within the EU. The principles of GDPR are applied at every stage of the lifecycle for any data produced, contained, or consumed by the platform. These data management principles can be defined as standard policies and applied to the data lifecycle system to ensure the behaviour of both the lifecycle management system and underlying data is compliant with all current data regulations.</p> <p>Platform Relevance</p> <p>Each UC that collects, processes, stores, or shares data can make use of the policy manager to provide a list of regulations, restrictions or policies associated with that data. A UC will provide a blueprint of their application and the associated requirements or constraints using the frontend and authoring tool (C4.6). The policy-related information will be sent to a policy manager that is built using Apache Ranger. The policy manager will then connect to the lifecycle manager (C4.1) and provide an up-to-date list of the policies to be enforced on the data of the user application.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|--------------------------|-----------|-------|
| C4.3 | Distributed Data Storage | ALL | DELL |
| <p>Component Overview:</p> <p>The global file system or object store is responsible for storing user data that is generated by various data sources and ingested into the platform. There are two key types of storage spaces needed by applications and services in the BRAINE platform:</p> <ul style="list-style-type: none"> • Long-term backend storage for storing historic data and to be accessed by internal or external entities. • Storage space required by stateful applications and containers in real-time. <p>To satisfy both these requirements and various other key features needed for the BRAINE use-cases, Apache Ozone was selected as the base to build the storage solution on. Ozone is a scalable, redundant, and distributed object store that aims to provide a resilient cost-efficient solution while addressing various problems that Apache</p> | | | |

Hadoop faces. Ozone achieves its aim of high scalability by separating the namespace and block-space management.

The distributed data storage in an EMDC is designed to run a single filesystem based on Apache Ozone and the user or system applications that run inside the EMDC use the Ozone storage class if they require persistent volumes. As the BRAINE platform is built on the containerized environment and Kubernetes for management and orchestration, the Ozone and all of its aforementioned components are also designed as containers.

Platform Relevance

Each UC that requires persistent storage space, may use the distributed data storage for that purpose. Within the Kubernetes-based testbeds, this is achieved by defining Ozone as the storage class for the persistent volumes. In response, the distributed data storage creates the required volumes based on data placement strategies using component C4.9 and attaches them to the UC applications.

| Component ID | Component Name | Use Cases | Owner |
|---|---------------------|-----------|-------|
| C4.4 | Active Data Product | ALL | LUH |
| <p>Component Overview:</p> <p>An active data product is indeed a dataset encapsulated in a secure container that allows access via a well-defined access point that conforms to the terms of an agreed-upon contract. A data product is said to be active as it can operate in an external environment. It is indeed a self-contained, secure executable package that must be run in order to allow for the utilization of the data it contains. When requested by external agents to access the data, it will ensure that the request and the response comply with contract terms, usage, sovereignty regulations, and boundaries defined. A contract definition language using YAML is under development, by which the data owner can define the terms and conditions for accessing the data. The contract is then enforced by the contract controller. Any legitimate access to the data will be recorded in a blockchain for contract term enforcement as well as for auditing and accounting. The ADP component is a prototype and may not be secure enough for production environments with high-sensitive data.</p> <p>UC Relevance</p> <p>Each UC that aims at sharing data with other UCs or external systems/parties may benefit from this component.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-------|
| C4.5 | Catalyzer Tool | UC2 | SIC |
| <p>Component Overview:</p> <p>The Catalyzer is a tool designed by Secure-IC to assess the security of software programs with regards to side channel attacks. The goal of the tool is to analyse software programs in order to detect, locate and characterize different types of vulnerabilities associated with those attacks.</p> <p>The Catalyzer can be used on any type of program, application, library, or firmware manipulating sensitive information: for example, software targets include cryptographic libraries, which use private keys, or applications manipulating passwords or pin codes.</p> | | | |

The tool operates in a semi-automated way, allowing the user to generate a vulnerability report given an input software target. Depending on the type of vulnerability, the Catalyzr performs the detection either by static analysis, by identifying leakages at the source code level, or by dynamic analysis, by executing the program binary and discovering vulnerabilities from its observable behaviour.

In the context of Braine, the targeted vulnerabilities are the so-called microarchitectural vulnerabilities, and are detected by static analysis.

Platform Relevance

The Catalyzr software will be used by the partner IWS (IS-Wireless) to build the robustness to malware attack of its own library written on the top of OpenSSL for the Braine Project.

| Component ID | Component Name | Use Cases | Owner |
|--|----------------|-----------|-------|
| C4.6 | Authoring Tool | ALL | ECC |
| <p>Component Overview:</p> <p>The Authoring tool is part of the Data Exploration module of the Eccenca Corporate Memory. It interacts with the EMDC and receives information from the registered Kubernetes clusters such as available nodes, memory, CPU, and pods capacity. It then populates the BRAINE knowledge graph with this information. Later on, it receives Node metrics information from a Metrics relay and is capable of storing this information together with the timestamp.</p> <p>With the knowledgebase populated (by WP3 services), users can use the Authoring Tool to define Services Profiles. The Services Profiles contain the Docker description of the service and will be further developed in other interactions. A Service Profile is used to define Service Deployment Specification that contains the available placements (Kubernetes Nodes) as well as the conditions.</p> <p>After defining the Service Deployment Specification, the user can create a Deployment whose states and metadata will be later managed by the Service Orchestrator. All information available in the interfaces comes from the BRAINE Knowledge Base, but from different components. For instance, the conditions for running the service defined in the Service Deployment Specification come from the BRAINE ontology while the Kubernetes Nodes available, as well as their setup (type of CPU, memory), comes from the EMDC.</p> <p>Platform Relevance</p> <p>The Authoring tool allows Use Case partners to create and deploy workflows using Argo (https://argoproj.github.io/).</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|----------------------|-----------|-------|
| C4.7 | Service Orchestrator | ALL | ECC |
| <p>Component Overview:</p> <p>The Service Orchestrator is responsible for service deployments, collecting service metadata and maintaining the information of the Deployments at the BRAINE Knowledge Base (BKB) synchronized with the information of the running service. It supports Service Deployment and Partial Status synchronization features.</p> <p>When a user defines a Deployment at the Authoring tool a new Deployment is instantiated at the BKB. The Service Orchestrator recognizes the new service through</p> | | | |

the state and reads its Service Deployment Specification containing information such as the Docker deployment, the user constraints, and the target Kubernetes Cluster (Node). Notice that a deployment may or not contain a Kubernetes Node or constraints as some attributes are not mandatory. In case the Node is not specified but there are constraints, it then verifies which Node can run checking the available constraints of the Node at the BKB, when not having, the Deployment will be in waiting state until a Node containing the constraints is found. In case the Service Deployment Specification has no constraints, the Service Orchestrator will use the first Node available. Otherwise, it will use the Node specified. When a Deployment is successful, the Service Orchestrator changes the Deployment state for running or stops with error otherwise. On its current version, the Service Orchestrator performs deployments directly at Kubernetes through the Kubernetes API, thus it does not communicate with the EMDC, nor report on metadata of services running such as Memory and CPU consumption.

Platform Relevance

The Service Orchestrator is crucial for synchronizing information among the Service and Resource catalogue and the platform components (EMDC) to offer service and system feedback as well as facilitate platform management.

| Component ID | Component Name | Use Cases | Owner |
|---|----------------------|-----------|-------|
| C4.8 | Monitoring Dashboard | ALL | LUH |
| <p>Component Overview:</p> <p>The monitoring dashboard is a visualization system for the time-series metric data that are stored in InfluxDB (and Prometheus). The dashboard comes with a pre-configured set of gauges and charts that display various system metrics scraped from node-exporter, including CPU, memory, disk I/O writing, and network traffic metrics. It is also able to visualize additional time-series data generated by the UC applications. The monitoring dashboard relies on the telemetry infrastructure components such as scraper and database from WP3.</p> <p>Platform Relevance</p> <p>Each UC may generate metrics and send them to the telemetry database for storage and processing. The monitoring dashboard can be tuned to extract UC-related metric data, filter and aggregate them and then display charts, gauges, or other visual forms of the data.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|--------------------------|-----------|-------|
| C4.9 | Data Placement Framework | ALL | UCC |
| <p>Component Overview:</p> <p>This is an external data placement framework, built as an API in Java. This API uses the available information for data placement. It keeps the record of the present state of data allocation on various data nodes. The API also has the details of the constraints on the data defined by the application. These constraints may be on node sharing, hardware (Intel, AMD, ARM), selected data nodes, or some specific location, etc. All these details are stored in config (XML and properties) files.</p> <p>The modified Ozone requests this API to get the data node list to store the input data. Based on the available information about this new data request (application, user, persistent volume, etc.) and the stored information about the current allocation the data</p> | | | |

placement API returns the list of data nodes to Ozone. Finally, Ozone uses these nodes to store the data.

Platform Relevance:

Each UC may have its own types of constraints on the data, the placement framework stores and uses these constraints during selecting the data nodes for the data store requests generated by the UC. The data placement framework works with modified Ozone.

| Component ID | Component Name | UC | Owner |
|---|---|-----|-------|
| C4.10 | Exporter for the metrics for the UC1 application 'AI-driven Digital Twin solution for new digital ecosystems enabling Smart Healthcare in Medical and Caregiving Centres' | UC1 | IMC |
| <p>Component Overview:</p> <p>Metrics Exporter is a component, written in Go, for the UC1 application. It provides an endpoint "/metrics" and sends GET metrics on request from the Prometheus server. Exporter will periodically (by default once every 15 seconds) go to the mentioned endpoint and receive metrics in a given format in response.</p> <p>For monitoring third party services that we use in our environment, there are already solutions available such as nginx, PostgreSQL, Kafka, Prometheus itself, node_exporter for pod node, etc.</p> <p>Metrics names and labels will be made accordingly to the official metric and label conventions provided by the https://prometheus.io/docs/practices/naming/.</p> <p>UC1 Relevance:</p> <p>The component creates an endpoint where metrics will be available to Prometheus for scraping. By creating it we can gather a particular set of metrics relevant to the UC1 application, helping us to get multiple metrics in one go.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|--|----------------------|-----------|-------|
| C4.11 | Motif Discovery Tool | UC3 | FS |
| <p>Component Overview:</p> <p>The Motif Discovery Module (MOD) enables the discovery of all repetitive patterns of any size in any time-series data. The time-series data from the perspective of BRAINE are sensory data from machines and devices on the shop floor. The discovered patterns represent unique operations of the machine.</p> <p>MOD is divided into several containerized modules, which can be deployed individually on different host machines. Within WP4, the Discovery module and the Detection module are being developed and implemented.</p> <p>UC3 Relevance</p> <p>This component enables the detection of motifs (patterns) from the time-series data. The online detection module uses the model learned from the Learning module (WP3, C3.8) for the online detection of motifs from the time-series data stream.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|----------------------|-----------|-------|
| C4.12 | VRAN with adjustment | UC2 | ISW |
| <p>Component Overview</p> <p>This component is the extension of the C2.11 component. It is assuming that the baseline vRAN operation has been extended with acceleration and telemetry and integrated with the architecture of the BRAINE. Key target here is the assessment of acceleration of a selected radio protocol (either PDCP or MAC). The availability of the solution depends on the availability of relevant acceleration card and successful porting of a PDCP (or MAC) layer, to the acceleration language compatible with the card.</p> <p>UC2 Relevance</p> <p>This extension of vRAN will be provided in case acceleration is successfully prototyped. The current experiences with the acceleration of the vRAN PDCP encryption algorithm, has not generated improvements in performance (actually it caused performance drop).</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|------------------------------|-----------|-------|
| C4.13 | AI Platform Profiling Engine | UC2 | NEC |
| <p>Component Overview:</p> <p>The execution of complex Deep Learning (neural network) algorithms on heterogeneous hardware configurations opens the field for several optimization opportunities. However, the combination of large number of algorithms and hardware configurations makes the development of one-size-fits-all heuristics complex. In order to explore the optimization space and ensure optimal hardware utilization, thereby reducing energy requirements and improving runtime performance, we designed a profiling tool that can reconstruct the dynamic direct acyclic execution graph of deep learning algorithms.</p> <p>The extracted profiles are then ready to be delivered to the remaining part of the tool stack to apply optimization rules generated by potentially different optimization technologies, including heuristics and data-driven approaches.</p> <p>The overall tool is integrated into applications by leveraging direct import in high-level domain-specific machine learning frameworks, such as PyTorch and TensorFlow, and therefore it can be shipped jointly with the application deployment units.</p> <p>UC2 Relevance</p> <p>This component will be used in UC2 to profile the other AI processes operating within the use case.</p> | | | |

| Component ID | Component Name | Use Cases | Owner |
|---|----------------------|-----------|-------|
| C4.14 | Flow telemetry agent | UC2/ALL | MLNX |
| <p>Component Overview</p> <p>The Flow Telemetry agent is responsible to initialize and configure the P4 program that was auto generated by MLNX P4 backend compiler, this agent's main user interface is a CLI running on SONIC.</p> <p>This agent also configures the needed HW option to enable Telemetry reporting (Mirror).</p> <p>An example configuration is a Flow (5 tuples) that should be monitored.</p> | | | |

UC2 Relevance

Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into the telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality.

| Component ID | Component Name | Use Cases | Owner |
|--------------|-----------------|-----------|-------|
| C4.14.1 | Flow P4 program | UC2/ALL | MLNX |

Component Overview

The Flow telemetry P4 program is responsible to configure the low-level HW to support the P4 program. In the flow telemetry case, this is a P4 table monitoring 5 tuples and mirroring the sampled traffic to the switch's CPU for reporting to the remote collector.

The P4 table (ACL) holds entries with 5 tuple keys and mirror actions, the Flow telemetry agent is responsible to add those entries.

This program is auto generated from the P4 code by MLNX backend P4 compiler.

The source P4 program is based on P4-lang to program the ASIC data plane of network elements and telemetry collector.

UC2 Relevance

Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into the telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality.

| Component ID | Component Name | Use Cases | Owner |
|--------------|-----------------------------------|-----------|-------|
| C4.14.2 | Flow telemetry monitor & exporter | UC2/ALL | MLNX |

Component Overview:

The telemetry monitor & exporter is responsible to collect and report telemetry data from network elements regarding network node behaviours and the traffic passing over the network node.

This component will wait for selected telemetry events from HW and will generate a report via gRPC to the Adapter (C13.17.1).

UC2 Relevance:

Network telemetry framework is used to populate network telemetry data extracted from the live network related to network elements and network flows observed in the network into the telemetry collector database. This telemetry information is later fetched by UC1 and UC2 partners to optimize workload placement and predict service operation quality.

| Component ID | Component Name | UC | Owner |
|--------------|---|-----|-------|
| C4.15 | Monitoring of the inter-agent data exchange | UC3 | CTU |

Component Overview:

The monitoring solution is custom made two components. The first part, sniffer, is built in java and works as an AMQP message harvester. The collected messages are then visualized in a thin visualization client build in Java FX.

UC3 Relevance:

The tool is used by use case to monitor and debug platform behaviour.

6. Appendix B – Component Report

6.1 WP2

Note: The first 10 components in D2.2 have been summarized in this document for brevity.

| Component ID | Component Name | Development | Owner |
|--|---------------------------------|-------------|-------|
| C2.1 | EMDC Backplane & CPU/GPU boards | 100% | PCB |
| <p>Technical progress summary</p> <p>Development is fully completed. All hardware components, including the replacement Ethernet switch and 11-slot backplane, are manufactured. Integration tests with BMDC and power supply characterizations have been successfully concluded. The 3U and 2U integration with the mechanical design is finalized.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------|-------------|-------|
| C2.2 | EMDC cooling system | 100% | JJC |
| <p>Technical progress summary</p> <p>The cooling system is fully operational. Thermal characterization of both 4-slot and 11-slot systems completed at JJC lab. All simulations for system optimization are concluded, and the 11-slot system has been fully tested and integrated with HID and PCB components.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|--|-------------|-------|
| C2.3 | Nanoparticle fluids for liquid cooling | 100% | SYN |
| <p>Development of nano-refrigerants is complete. Both suspension and deposition methods have been successfully implemented and tested. Experimental setups for pool and flow boiling have been finalized and shown effective heat transport enhancement.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|------------------------------|-------------|-------|
| C2.4 | EMDC board mounting assembly | 100% | HID |
| <p>The mounting assembly, including the 3U 19" rack, is fully developed and prototyped. All module assemblies, including CPU, NVME, and Power, have passed final testing. The integration of condenser components in the 2U rack and their performance testing with JJ Cooling and PCB Design have been successfully completed.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------|-------------|-------|
| C2.5 | 5G RRH Demonstrator | 100% | COM |
| <p>Technical progress summary</p> <p>The 5G RRH Demonstrator has achieved full capacity, successfully demonstrating O-RAN with robust transmit and receive capabilities for 5G RF traffic. All debugging phases are complete, ensuring a mature and reliable solution. Integration with various vendors' DU implementations is finalized.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|-------------------------|-------------|-------|
| C2.6 | Quantum safe fiber link | 100% | TUe |
| <p>Technical progress summary</p> <p>The QKD link is now fully operational, capable of encrypting data at 10GbE line-rate. The integration of WDM filters, enabling the multiplexing of channels into a single fiber, is completed. Commercial servers are replaced by BRAINE EMDC prototypes, seamlessly integrating within the project infrastructure.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|-------------------------|-------------|-----------|
| C2.7 | EMDC Switch P4 programs | 100% | MLNX/CNIT |
| <p>GitLab Repository:</p> <p>https://github.com/Dscano/Postcard-Telemetry-Braine;</p> <p>https://github.com/Dscano/GTPV1-P4</p> <p>Complete and fully functional, the P4-based programmable switch supports advanced networking functionalities, targeting Mellanox Spectrum1 ASIC. It features layer 2-3-4 forwarding and traffic steering. Fully integrated on the BRAINE platform, it is deployed and operational.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|----------------|-------------|-----------|
| C2.8 | Switch OS | 100% | MLNX/CNIT |
| <p>Technical progress summary</p> <p>The Switch OS, based on Onyx over ONIE, is fully implemented and operational in the BRAINE testbed. SONiC over ONIE, complemented by a P4 docker container, is successfully tested on Spectrum 1 with advanced P4 capabilities. Integration with ONOS controller API for standalone switch versions, including P4Runtime and NETCONF, is completed.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|-----------------------|-------------|-------|
| C2.9 | Optical node SW Agent | 100% | CNIT |
| <p>GitLab Repository:</p> <p>https://github.com/asgamb/OpenConfig</p> <p>The Optical Node SW Agent, an open-source NETCONF Agent supporting OpenConfig YANG, has been enhanced for coherent pluggable modules, updated to the 2021 OpenConfig YANG model, and implemented in a Docker container within SONiC OS. It features advanced access control and monitoring capabilities through SONiC, ensuring effective management of optical and packet parameters.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--------------|----------------|-------------|-------|
|--------------|----------------|-------------|-------|

| | | | |
|---|-------------|------|-----|
| C2.10 | 5G NR Model | 100% | ISW |
| <p>Technical progress summary</p> <p>The 5G NR Model's Downlink (DL) transmitter is fully developed and enhanced to align with the eCPRI compression specification. Integration with lowPHY via offline hooks and the eCPRI management plane is successfully completed, ensuring robust connectivity and communication.</p> | | | |

| | | | |
|--|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C2.11 | vRAN Prototype | 100% | ISW |
| <p>Technical progress summary</p> <p>The vRAN Prototype has completed final tuning for the RU component, with other components virtualized and packaged as Kubernetes containers. Integration with telemetry systems is finalized, capturing key metrics like CPU, memory, and storage for comprehensive performance assessment.</p> | | | |

| | | | |
|---|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C2.12 | MBMC software | 100% | BME |
| <p>Technical progress summary</p> <p>MBMC software now fully supports FreeRTOS and is ported to BRAINE hardware. Features like CDC-ECM communication, TCP stack, UART over TCP, and Board Level API are integrated and tested. The software also includes defined protocols for system tasks and successful board bring-up tests for CPU and GPU nodes.</p> | | | |

| | | | |
|--|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C2.13 | BMC SW, V1.0 | 100% | PCB |
| <p>Technical progress summary</p> <p>BMC software version 1.0 is complete with a custom Linux image, DDR, and uboot functionality. Full integration includes iKVM functionality, SOL, EMDC card identification, and BMC-BMMC communication, all thoroughly tested.</p> | | | |

| | | | |
|--|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C2.14 | FPGA Node | 100% | BME |
| <p>The FPGA Node is fully developed with upgraded FPA and firmware components. It features a base FPGA hardware for AI inference benchmarks, Docker runtime, and K3s integration. Ethernet interface and partial dynamic reconfiguration developments are completed, enhancing its integration capabilities.</p> | | | |

| | | | |
|--------------|----------------------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C2.15 | OpenCL program for low PHY | 100% | SSSA |

The OpenCL program for Low PHY is finalized, offering advanced 5G Low-PHY functions and efficient implementation on Altera DE10pro FPGA using OpenCL Framework. The comparison with CPU and GPU-based implementations shows improved processing time and energy efficiency.

GitLab Repository:

- https://gitlab.com/braine/braine_source_release/-/tree/main/wp2-opencl-sssa?ref_type=heads

| Component ID | Component Name | Development | Owner |
|---|--------------------------------|-------------|-------|
| C2.16 | VTU – Virtual transcoding unit | 90% | ITL |
| Both single and multi-stream environments of the VTU are complete, offering flexibility in camera management. The Docker containers, built on Alpine and Ubuntu, are optimized for x86 servers with NVIDIA GPUs and Jetson AGX Xavier Development Module, ensuring high performance and adaptability. | | | |
| GitLab Repository: https://gitlab.com/braine/vtu-itl (currently private) | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------|-------------|-------|
| C2.17 | N2Net | 100% | NEC |
| N2Net is fully implemented for various hardware targets, with a streamlined compiler tool. Applications for traffic classification and anomaly detection are developed, demonstrating the system's efficiency and innovation. Integration with network telemetry subsystems is in progress. | | | |

| Component ID | Component Name | Development | Owner |
|--|------------------------|-------------|-------|
| C2.18 | Quantum safe readiness | 100% | SIC |
| The cryptographic software analysis adheres to NIST and ETSI guidelines for quantum safety. The prototype of the quantum-safe TLS library is complete and successfully debugged. | | | |

| Component ID | Component Name | Development | Owner |
|--|----------------------------------|-------------|-------|
| C2.19 | Low bitrate blockchain protocols | 100% | IMC |
| Two protocols, PLS and SLV, for a permissioned blockchain suitable for low bit-rate communications are fully developed and published. The architecture units have been implemented in a testbed and was ported to an IoT platform (ESP32). | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------------|-------------|-------|
| C2.20 | Smart sensors integration | 100% | IFX |
| Component complete and integrated in UC4. | | | |

6.2 WP3

| Component ID | Component Name | Development | Owner |
|---|----------------|-------------|-------|
| C3.1 | Service Mesh | 100% | VMW |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-braine-mesh?ref_type=heads Summary: The Service Mesh is now fully operational on the BRAINE platform, with a configuration service that allows seamless registration and communication across EMDCs. This completion ensures robust and efficient communication within the service mesh network. | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------|-------------|-------|
| C3.2 | Image Registry | 100% | VMW |
| The Image Registry on the CNIT testbed is complete, with multiple components successfully registered. This development streamlines the management and deployment of images across the platform. | | | |

| Component ID | Component Name | Development | Owner |
|--|---------------------------------------|-------------|-------|
| C3.3 | Telegraf agent for 5G data collection | 100% | SSSA |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-5g-sssa?ref_type=heads The Telegraf agent is now fully functional for data collection, with optimized configuration for collection intervals and message formats. It is effectively integrated with the forecasting functional block. | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------------|-------------|-------|
| C3.4 | Distributed Knowledgebase | 90% | DELL |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-dkb?ref_type=heads The DKB is fully functional, collecting and sharing CPU and memory data across clusters. It is effectively integrated with controllers and orchestrators. | | | |

| Component ID | Component Name | Development | Owner |
|--|------------------------------|-------------|-------|
| C3.5 | Forecasting functional block | 100% | SSSA |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-ffb-5g-sssa?ref_type=heads The Forecasting Functional Block is complete, with datasets collected and the model trained for consistent result production. It integrates effectively with the Telegraf agent. | | | |

| Component ID | Component Name | Development | Owner |
|--------------|---------------------|-------------|-------|
| C3.6 | BRAINE RL Scheduler | 100% | LUH |

| | | | |
|--|-------------------|------|-----|
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-work_placement-luh?ref_type=heads | | | |
| The BRAINE RL Scheduler and its components (Trainer, Inferencer, DataAccessAgent) are fully developed. These components are integrated with the telemetry database and the ML-based inference engine, offering advanced scheduling functionalities. The system efficiently predicts worker node allocations and manages resource usages. | | | |
| C3.6.01 | BRAINE RL Trainer | 100% | LUH |
| This component generates a trained ML model which is then persisted and utilized by the Inferencer (BRAINE RL Inference Engine for Scheduling). | | | |
| C3.6.02 | Inferencer | 100% | LUH |
| This component utilizes the trained ML model and serves the scheduling mechanism via a REST API for the prediction of a proper worker node for a given workload and system state. | | | |
| C3.6.03 | DataAccessAgent | 100% | LUH |
| This component acts as a bridge between the scheduling plugin and the telemetry database. Its responsibility is to serve the plugin, via a REST API, with information about the resource usages/availability of each of the worker nodes of the cluster. | | | |

| Component ID | Component Name | Development | Owner |
|---|--------------------------|-------------|-------|
| C3.7 | Telemetry Infrastructure | 100% | LUH |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-telemetry-luh?ref_type=heads | | | |
| The Telemetry Infrastructure, encompassing multiple integrated components like a telemetry database, metric exporter, scraper, and alert manager, is fully operational. Deployed as a pod on the BRAINE platform, it effectively monitors and manages telemetry data. | | | |

| Component ID | Component Name | Development | Owner |
|--|-----------------------|-------------|-------|
| C3.8 | MOD – Learning module | 100% | FS |
| The Learning Module of MOD is now complete, successfully integrating with the Discovery and Detection Modules of the MOD application. This module enhances the overall learning capabilities within the BRAINE platform. | | | |

| Component ID | Component Name | Development | Owner |
|--|--------------------|-------------|-------|
| C3.9 | Image Orchestrator | 100% | ECC |
| GitLab Repository: https://github.com/eccenca/braine/tree/main/container-orchestrator | | | |
| The Image Orchestrator is fully functional and integrated and with the Global Image Registry. | | | |

| Component ID | Component Name | Development | Owner |
|--------------|----------------|-------------|-------|
|--------------|----------------|-------------|-------|

| | | | |
|---|---------------|------|-----|
| C3.10 | Metrics Relay | 100% | ECC |
| GitLab Repository: https://github.com/eccenca/braine/tree/main/relay The Metrics Relay, fully integrated with the SLA Broker, efficiently collects and relays node metadata, facilitating the management of available nodes and resources. | | | |

| | | | |
|--|-----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C3.11 | BRAINE Ontology | 100% | ECC |
| GitLab Repository: https://github.com/eccenca/braine-vocab The BRAINE Ontology is effectively instantiated for the BRAINE Resource & Service Catalog, enhancing data modelling and structuring. | | | |

| | | | |
|---|------------------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C3.12 | Knowledge Bootstrapper | 100% | ECC |
| GitLab Repository: https://github.com/eccenca/braine/tree/main/setup The Knowledge Bootstrapper is fully developed and operational, successfully bootstrapping the Resource & Service Catalog with information aligned to the BRAINE data model. | | | |

| | | | |
|---|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C3.13 | SDN Controller | 100% | CNIT |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/WP3-SDN-CONTROLLER?ref_type=heads The SDN Controller, based on the ONOS project, is fully integrated with the telemetry system and Kubernetes orchestrator. It efficiently manages network device configurations and telemetry data collection, enhancing network control and monitoring. | | | |

| | | | |
|--|--------------------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C3.14 | Audio Inferencing Engine | 100% | MAI |
| The engine, capable of neural network-based audio event classification and translation, supports audio recordings and real-time inputs. Component was successfully demonstrated with real-time camera audio on K8s as part of UC2. | | | |

| | | | |
|---|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
| C3.15 | SLA Broker | 100% | DELL |
| The SLA broker has been completed and integrated as part of the overall BRAINE platform architecture. It works with components such as the workload scheduler and SDN controller to monitor and enforce SLA policies. | | | |

| | | | |
|--------------|----------------|-------------|-------|
| Component ID | Component Name | Development | Owner |
|--------------|----------------|-------------|-------|

| | | | |
|--|--------------------------------------|------|-----|
| C3.16 | Multi-access Edge Computing platform | 100% | SMA |
| GitLab Repository: https://github.com/Sma-RTy/native-on-prem.git The MEC platform is online, ready to host third-party applications for UC2, offering an alternative platform for performance comparison. | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------------------|-------------|-------|
| C3.17 | AI image processing engine | 100% | SMA |
| GitLab Repository: https://github.com/Sma-RTy/deepsort The AI engine for tracking road users is developed and containerized. Deployment on the BRAINE platform with hardware acceleration support was successfully demonstrated in UC2 | | | |

| Component ID | Component Name | Development | Owner |
|--|---------------------------------------|-------------|-------|
| C3.18 | Edge-to-edge multiagent communication | 100% | CTU |
| The multiagent communication prototype, using RabbitMQ for messaging, is integrated into agents and was successfully tested with the multi-agent platform in UC3 | | | |

| Component ID | Component Name | Development | Owner |
|---|-----------------|-------------|-------|
| C3.19 | Placement Agent | 100% | ISW |
| The vRAN Placement Agent is now fully implemented, demonstrated vRAN scaling as part of BRAINE final system demo. | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------|-------------|-------|
| C3.20 | Parameter Predictor | 100% | ISW |
| GitLab Repository: Link: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-parameter-predictor-isw?ref_type=heads The LSTM-based parameter predictor was successfully completed and demonstrated as part of the SD-RAN scaling demonstration. | | | |

| Component ID | Component Name | Development | Owner |
|--|---|-------------|-------|
| C3.21 | BRAINE ML-based L1 Kubernetes Scheduler | 100% | ISW |
| L1 scheduling has been successfully completed, with all 5G SD-RAN pods capable of being placed on any node with in single EMDC with no side-effects. | | | |

| Component ID | Component Name | Development | Owner |
|--|---|-------------|-------|
| C3.22 | BRAINE ML-based L2 Kubernetes Scheduler | 100% | ISW |
| The L2 scheduler for 5G SD-RAN was successfully implemented in conjunction with the LSTM model. This successfully enabled the 5G CU to scale to a second EMDC when a defined resource threshold was met. | | | |

| Component ID | Component Name | Development | Owner |
|---|-------------------------------|-------------|-------|
| C3.17 | Telemetry monitors & exporter | 100% | MLNX |
| The postcard-based telemetry and monitoring system for P4 based switches is completed and integrated as part of the SDN controller. | | | |

| Component ID | Component Name | Development | Owner |
|---|-------------------|-------------|-------|
| C3.17.1 | Telemetry adapter | 100% | MLNX |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp3-telemetry-mlnx?ref_type=heads The telemetry adapter for Mellanox switches is now also complete, with full flow telemetry and latency information available. | | | |

| Component ID | Component Name | Development | Owner |
|--|--------------------|-------------|-------|
| C3.17.2 | Telemetry ingester | 100% | MLNX |
| Implemented with the Telegraf framework, the ingester is successfully integrated with BRAINE telemetry data and InfluxDB schema. | | | |

6.3 WP4

| Component ID | Component Name | Development | Owner |
|---|------------------------|-------------|-------|
| C4.1 | Data lifecycle manager | 100% | DELL |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp4-datalifecyclemanager?ref_type=heads Fully integrated into the BRAINE platform, the Data Lifecycle Manager effectively manages data through its entire lifecycle. It is seamlessly connected to the Apache Ozone global filesystem, enabling efficient handling of filesystem events. Its capability to containerize and process data into an immutable ledger enhances data integrity and security across the platform. | | | |

| Component ID | Component Name | Development | Owner |
|--------------|----------------|-------------|-------|
|--------------|----------------|-------------|-------|

| | | | |
|---|----------------|------|------|
| C4.2 | Policy Manager | 100% | DELL |
| <p>GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp4-dmf?ref_type=heads</p> <p>The Policy Manager, utilizing Apache Ranger, is fully operational and integrated with the Ozone filesystem and SLA Broker. This integration ensures robust monitoring and enforcement of data policies, playing a crucial role in data governance and security within the BRAINE platform.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|--------------------------|-------------|-------|
| C4.3 | Distributed Data Storage | 100% | DELL |
| <p>GitLab Repository: https://hub.docker.com/repository/docker/khalia6/ozone https://hub.docker.com/repository/docker/khalia6/csi-node-driver-registrar https://hub.docker.com/repository/docker/khalia6/csi-provisioner</p> <p>Apache Ozone-based Distributed Data Storage is now fully functional, providing versatile and reliable storage solutions across various CPU architectures. Its integration into the BRAINE platform equips all applications requiring persistent storage with efficient and scalable data storage capabilities.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|---------------------|-------------|-------|
| C4.4 | Active Data Product | 100% | LUH |
| <p>The Active Data Product is now a fully integrated component, focused on contract enforcement and consumer access. It has undergone successful lab tests, ensuring its effectiveness in managing and providing access to data products within the BRAINE ecosystem.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------|-------------|-------|
| C4.5 | Catalyzr Tool | 100% | SIC |
| <p>The Catalyzr Tool, specialized in detecting microarchitecture information leaks, is now fully operational. The tool's successful implementation of cache attack detection and its collaboration with ISW have strengthened its vulnerability hunting capabilities, particularly in virtual and cross-development environments.</p> | | | |

| Component ID | Component Name | Development | Owner |
|--|----------------|-------------|-------|
| C4.6 | Authoring Tool | 100% | ECC |
| <p>GitLab Repository: https://github.com/eccenca/braine/tree/main/webclient</p> <p>The Authoring Tool for service composition on the BRAINE platform is complete, featuring an intuitive user interface and a robust data model for persistence. This tool plays a vital role in facilitating service composition and ensuring seamless integration with the Resource & Service Catalog.</p> | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------------|-------------|-------|
| C4.7 | Service Orchestrator | 100% | ECC |
| GitLab Repository: https://github.com/eccenca/braine/tree/main/service-orchestrator The Service Orchestrator, essential for synchronizing service metadata between the Global Service Registry and the Resource & Service Catalog, is now fully functional and integrated. Its role is critical in maintaining consistency and up-to-date service information across the BRAINE platform. | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------------|-------------|-------|
| C4.8 | Monitoring Dashboard | 95% | LUH |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp4-monitoringsystem-luh?ref_type=heads The Monitoring Dashboard, integrated with InfluxDB, provides a comprehensive view of system metrics. Its default dashboard console with basic monitoring elements enhances the visibility and monitoring capabilities within the BRAINE platform. | | | |

| Component ID | Component Name | Development | Owner |
|---|--------------------------|-------------|-------|
| C4.9 | Data Placement Framework | 100% | UCC |
| GitLab Repository: https://gitlab.com/braine/braine_source_release/-/tree/main/wp4-DataPlacementFramework?ref_type=heads The Data Placement Framework is now complete, offering optimized data management and placement strategies. This tool enhances the efficiency and effectiveness of data handling within the BRAINE platform. | | | |

| Component ID | Component Name | Development | Owner |
|---|--|-------------|-------|
| C4.10 | Exporter for the metrics for the application Healthcare Digital Twin | 100% | IMC |
| The Exporter for Healthcare Digital Twin metrics is fully operational, enabling efficient tracking and management of healthcare-related data. It plays a vital role within the Healthcare Digital Twin system and was successfully demonstrated as part of UC1. | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------------|-------------|-------|
| C4.11 | Motif Discovery Tool | 100% | FS |
| The Motif Discovery Tool, designed to discover patterns and motifs in data, is fully integrated with the Learning and Detection Modules of the MOD application. This tool enhances the data analysis capabilities of UC3 and was successfully demonstrated as part of the use case. | | | |

| Component ID | Component Name | Development | Owner |
|--------------|----------------|-------------|-------|
| | | t | |

| | | | |
|---|-----------------------|------|-----|
| C4.12 | vRAN with adjustments | 100% | ISW |
| The vRAN with adjustments is complete, including validated PDCP acceleration using FPGA. However, latency issues have prompted the use of CPU-based solution for demonstration, and a re-evaluation and potential integration of SmartNIC or alternative solutions to optimize performance. | | | |

| Component ID | Component Name | Development | Owner |
|--|------------------------------|-------------|-------|
| C4.13 | AI Platform Profiling Engine | 100% | NEC |
| The AI Platform Profiling Engine, a key component of the Deep Learning toolchain, performs automatic software optimization for neural networks to reduce energy usage. It integrates static analysis and a training phase for implementation options evaluation, enhancing NEC's SOL compiler with auto-tuning features. This engine is instrumental in optimizing Use Case 2 applications for efficiency. | | | |

| Component ID | Component Name | Development | Owner |
|---|----------------------|-------------|-------|
| C4.14 | Flow telemetry agent | 100% | MLNX |
| The Flow Telemetry Agent, tested for adding/removing traffic flows, updates P4 tables and sends telemetry events for monitoring. It's integrated with the Flow P4 Program (C4.14.1) for effective telemetry management. | | | |

| Component ID | Component Name | Development | Owner |
|---|-----------------|-------------|-------|
| C4.14.1 | Flow P4 Program | 100% | MLNX |
| The Flow P4 Program is fully developed with hardware tables created for new flow additions. This program facilitates dynamic flow management and is integrated with Flow telemetry monitoring and exporter (C4.14.2) for efficient streaming of telemetry data. | | | |

| Component ID | Component Name | Development | Owner |
|---|-----------------------------------|-------------|-------|
| C4.14.2 | Flow telemetry monitor & exporter | 100% | MLNX |
| This component collects and exports hardware telemetry events to a remote collector via gRPC. Integrated with the Telemetry Adapter, it plays a crucial role in streaming and managing telemetry data within the BRAINE platform. | | | |