Unlocking the compute continuum: scaling out from cloud to HPC and HTC resources

Diego Ciangottini[1,*], *Daniele* Spiga[1], *Ahmed Shiraz* Memon[3], *Andrea* Manzi[2], *Andrej* Filipcic[4], *Antonino* Troja[6], *Federica* Fanzago[6], *Giulio* Bianchini[1], *Massimo* Sgaravatto[6], *Teo* Prica[5], *Tommaso* Boccali[7], and *Tommaso* Tedeschi[1]

[1]INFN Sezione di Perugia, Via A. Pascoli 23c, 06123 Perugia, Italy
[2]EGI Foundation, Amsterdam, Netherlands
[3]Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Jülich, Germany
[4]Jozef Stefan Institute (JSI), Jamova cesta 39, 1000 Ljubljana, Slovenija
[5]Institute of Information Science (IZUM), Prešernova ulica 17, 2000 Maribor, Slovenia
[6]INFN Sezione di Padova, via Marzolo 8, 35137 Padova, Italy
[7]INFN Sezione di Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy

Abstract. In a geo-distributed computing infrastructure with heterogeneous resources (HPC and HTC and possibly cloud), a key to unlock an efficient and user-friendly access to the resources is being able to offload each specific task to the best suited location. One of the most critical problems involves the logistics of wide-area, multi-stage workflows that move back and forth between multiple resource providers. We envision a model where such a challenge can be addressed enabling a "transparent offloading" of containerized payloads using the Kubernetes API primitives creating a common cloud-native interface to access any number of external hardware machines and type of backends. Thus we created the interLink project, an open source extension to the concept of Virtual-Kubelet with a design that aims for a common abstraction over heterogeneous and distributed backends. interLink is developed by INFN in the context of interTwin, an EU funded project that aims to build a digital-twin platform (Digital Twin Engine) for sciences, and the ICSC National Research Center for High Performance Computing, Big Data and Quantum Computing in Italy. In this talk we first provide a comprehensive overview of the key features and the technical implementation. We showcase our major case studies, such as the scale-out of an analysis facility, and the distribution of ML training processes. We focus on the impacts of being able to seamlessly exploit world-class EuroHPC supercomputers with such a technology.

1 Introduction

The need for computing capacity for scientific computing has been continuously growing, and inclusion of High-Performance Computing (HPC) providers represent a unique opportunity to satisfy those needs. This also brings the complexity of efficiently exploiting an increasingly heterogeneous scenario. In fact, to the usual way of managing distributed and independent (embarrassing parallel) data analysis workflows with the highest throughput (HTC), there

*e-mail: ciangottini@infn.it

are other additions driven by the ML/AI-on-cloud interface paradigm and both need to be enabled to effectively exploit the world-class HPC resources. One of the pivotal keys to efficiently accessing diverse resources lies in the ability to offload each specific task to the most suitable location. This process, however, is a challenge due to the different access patterns and interfaces and wide-area network policies.

As a path forward, we propose a model that enables "transparent offloading" of containerized payloads. In the context of computing, one way to define such a scenario is often with the name of "continuum". We choose to leverage Kubernetes API primitives to create a common cloud-native interface that facilitates the access to a variety of external hardware machines and backend types. This vision brought to the development of the interLink project [1], an open-source extension to the Kubernetes Virtual Kubelet interface. The design of interLink strives to provide a common abstraction layer, simplifying the orchestration and management of computing resources from diverse providers types.

The interLink project has been bootstrapped by INFN within the framework of the interTwin project [2] initiative, an EU-funded project dedicated to constructing a digital-twin platform, known as the Digital Twin Engine, for scientific applications. Additionally, the project is supported by the ICSC National Research Center for High Performance Computing, Big Data, and Quantum Computing in Italy [3], and it is currently in evaluation in a number of scientific institutions, commercial and scientific projects.

2 The implementation strategy

interLink is designed not only to ensure seamless integration with both resource providers and user applications. It is based on a plugin model which aims to be easy to extend to "any" technical solution for resources provisioning. The core strategy revolves around the following key components:

- Kubernetes API Primitives: interLink provides a standardized interface for managing containerized workloads across diverse computing environments. This approach ensures compatibility and ease of integration with existing Kubernetes-based infrastructures and most of the ML/AI development frameworks. To this end, interLink extends the concept of Virtual-Kubelet already part of the Cloud-Native Computing Foundation (CNCF). The idea is to provide an interface that enables users to interact with a unified platform, without worrying about the way the resources should be accessed.
- Interoperability and Flexibility: interLink is designed to be highly interoperable, supporting a wide range of backend systems and hardware configurations. This flexibility ensures that the platform can adapt to various use cases and resource availability scenarios. Future specialized computing centers should be looked at as being as important as the current case studies.
- Case Studies and Practical Applications: The implementation strategy is being validated through several case studies, including the scale-out of the so-called "analysis facilities" (see later) and the distribution of ML training processes through cloud-native frameworks [4, 5]. These case studies demonstrate the practical benefits and real-world applications of interLink, and on the other hand, they provide crucial feedback for the future direction of the project. In the next session, we will show the most influencial early studies we performed on two of the HPC providers participating in the interTwin project.
- Collaboration and Support: The development of interLink is meant to be a collaborative effort involving multiple institutions and research centers. This collaboration ensures that the project benefits from a range of expertise and needs, driving the inclusion of new features to be integrated and put at the disposal of everyone.

By focusing on these key assets, we envision that we can provide a robust solution for managing geo-distributed computing resources through Kubernetes, the "de-facto" standard interface for cloud-native tools. The target is to ensure an efficient exploitation of resources for scientific computationally intensive domains through its interoperability and flexibility.

3 The interLink architecture

As anticipated, the architecture of interLink (fig. 1) is designed with plugin and flexibility as first class citizens. The key architectural components include:

- Virtual-Kubelet core component: The core of interLink's architecture is the Virtual-Kubelet, which acts as a bridge between Kubernetes and the various backend systems. Essentially, it is a pod inside the cluster that communicates with the Kubernetes API server as if it were a real physical node.
- Kubernetes API Server: The Kubernetes API server is responsible for interacting with the Virtual-Kubelet to schedule and manage workloads on the appropriate backend system. It is where the handshake between the cloud world and remote providers occurs. Technically speaking, it consists in a REST API interface that is instrumented to translate a pod request from the Kubernetes cluster into a OpenAPI spec for the remote backend.
- Backend Systems: interLink supports native integrations with HPC and HTC clusters, enabling the offloading of computationally intensive tasks to these resources. It is not limited to it though, in fact the backend implementation is a simple REST-ful service that follows a CRUD-like spec. Therefore, it is straightforward to create and maintain its own backend plugin for any provider administrator. As a matter of fact, there are already cases of integration with a remote kubernetes cluster and cloud providers with GPU-equipped VMs allowing for the dynamic allocation of resources based on workload requirements.

3.1 Authentication flow

Since communication from a local Kubernetes cluster to a remote provider is involved, the authentication and authorization flows cover a crucial role in the model we envision to serve with the interLink implementation. The current state of the art relies on what we called "edge-node" deployment strategy. It is strongly driven by the current priority, which is the seamless integration of HPC centers. This is representing only one of the possible solutions. To see other options, you can refer to the roadmap in sec. 5.

In the edge-mode scenario, the handshake is supposed to happen between the Kubernetes cluster administrator and the resource providers. In this case, we envision a Virtual Kubelet service talking to a remote interLink API server through a token-based OAuth2 authentication [6]. All the rest of communications (b/w interLink API server and the plugin) happens inside an edge node provided by the remote infrastructure through unix sockets (by default). In this way, any payload landing on the remote provider is owned by a service account, to which only a Virtual Kubelet presenting the right token claims can access. As a matter of fact, this means that the policy for authorizing a payload to be offloaded is delegated to the Kubernetes administrator.

The presented flow enables the possibility to support fine-grained authorizations both at the level of Kubernetes, but also at the level of interLink itself. This in turn means that in addition to the service model presented above, the providers can decide to do further enforcements defining policies at the level of the edge node.

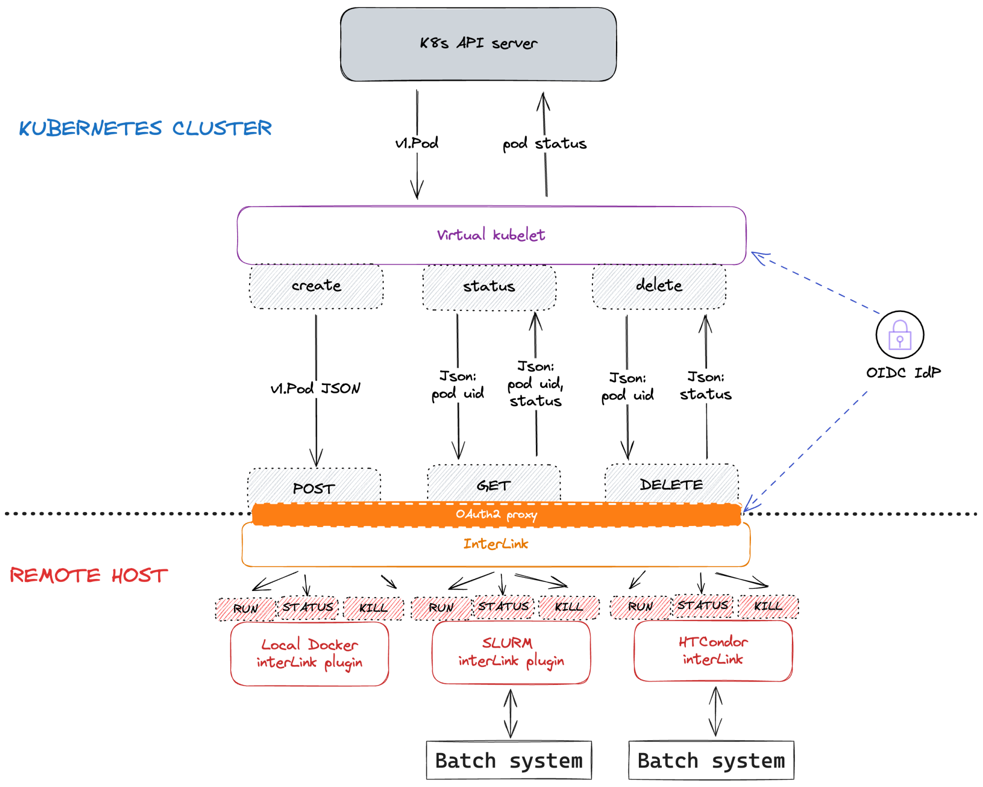

Figure 1. In this schema there are all the core components of the interLink design, in the configuration called "edge node". From top to bottom: Kubernetes API server schedule a pod on the virtual node, the interLink Virtual Kubelet pod takes all the information about the payload, then it sends them to the interLink API server through a REST interfaced authenticated through OAuth2 flow. After some preliminary checks, the interLink server delegates the execution to the plugin chosen for the deployment. Depending on which kind of backend is available, the only changing part of the deployment is the CRUD-like RESTful service of the plugin part at the bottom.

4 Case studies

The implementations we have been testing along with real-world case studies will be now presented briefly. Unless specified otherwise, we identified a set of benchmarking use cases that have been successfully replicated at the participating HPC centers.

- 3D GAN for LHC Particle Jet simulations: in this case study brought by CERN [7], a set of ML trainings of a GAN network have been done on HPC resources while tracking each experiment with metrics of interest with an MLFlow instance hosted on the local Kubernetes cluster. This is the perfect example of offloading heavy GPU tasks to the remote center while developing and hosting cloud services on cloud resources.
- Event-driven analyses with OSCAR [8] framework: we demonstrated an ML model evaluation of interTwin DT use cases to be possible in an event-driven fashion where a storage event (e.g. a new dataset is available) triggers several tasks. Part of those tasks (the heaviest ones) have been successfully offloaded to HPC centers through interLink abstraction.

4.1 interLink at Vega HPC center

HPC Vega is the first EuroHPC JU supercomputer hosted at the Institute of Information Science in Maribor, Slovenia. It was an essential piece of the development phase for interLink, enabling early prototyping of the whole chain and providing support for the first interLink edge service deployment. The plugin installed is capable of submitting SLURM jobs configured to execute the pod containers in a seamless way through Apptainer [9].

The authN/Z model (fig. 2) is what we described above, where the Kubernetes cluster requests are validated through EGI Check-in [10] released tokens, and the site delegates any responsibility to the owner of the cluster. We did demonstrate also the capability of interLink to support multiple Kubernetes clusters connected to the same interLink edge service.

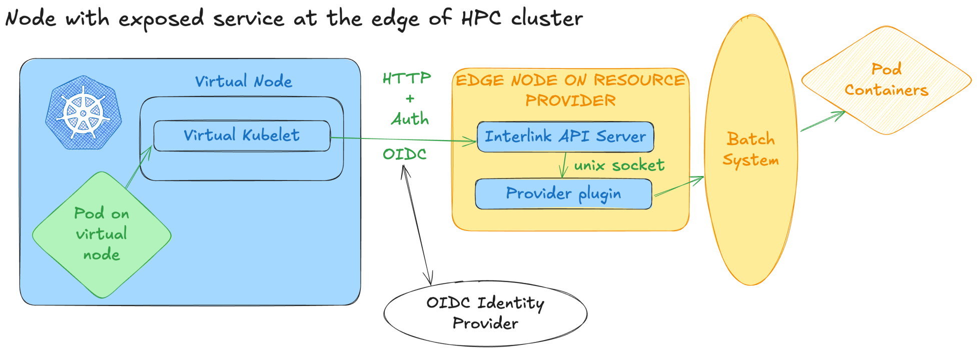

Figure 2. A simplified schema highlighting the structure of the deployment at HPC Vega. From left to right: the kubernetes cluster is instrumented to offload pod execution contacting the interLink edge service, if the cluster shows a valid token released by the expected identity provider the payload is accepted and transformed into a SLURM job executing Apptainer containers

4.2 interLink at Julich HPC center

The Julich Supercomputing Centre operates one of the most powerful supercomputers in Europe, JUWELS, and JUNIQ the first European infrastructure for quantum computing. The peculiarity of the interLink implementation is on the plugin adopted. In fact, at Julich, there is already an authenticated API available at for users to run containers on the HPC infrastructure (UNICORE [11]). This feature allowed for a demonstration of the ease of development for new provider-specific plugins along with a more fine-grained authN/Z models (fig. 3).

A UNICORE plugin has been successfully developed by Julich administrators and with that it was possible to offload tasks from the Kubernetes cluster to the UNICORE container execution infrastructure. The authentication and authorization of the offloaded payloads have been delegated to the UNICORE interface, requiring each pod to present, as an annotation, valid credentials. This way, a one-to-one mapping between pod owners and SLURM job owner is enforced, offering a fine-grained alternative to the edge service delegation model.

4.3 interLink at WLGC grid-sites and cloud at INFN resources

The case studies in this section aim to be complementary to what has been achieved in the context of the interTwin project. In fact, at INFN a series of interLink demonstrators have been included in several R&D initiatives.

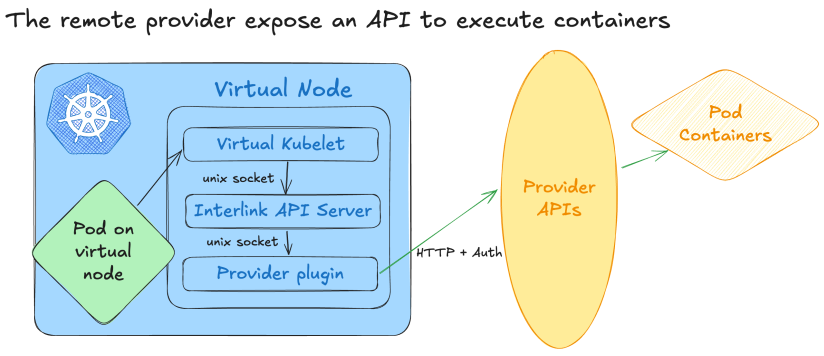

Figure 3. A simplified schema highlighting the structure of the deployment at HPC Julich. Thanks to the UNICORE interface allowing to submit container execution on the SLURM cluster, all the interLink components can be managed in the so called "in-cluster" topology. The UNICORE plugin accepts pod submission only if a particular annotation is shown, containing the user credentials needed to submit UNICORE containers.

- INFN "Analysis Facility" for the CMS experiment at LHC: this infrastructure has been developed to host the interactive data analysis ecosystem for the next-generation of HEP data analysis frameworks [12]. In this context interLink allowed for seamlessly spawning, from a cloud hosted Jupyter notebook, Kubernetes pods (Dask clusters in this case [13]) on Italian WLCG Grid-sites [14] with an HTCondor interLink plugin. For this use case, the offloading was also demonstrated towards CloudVeneto [15], a distributed private cloud infrastructure based on the OpenStack middleware, which also offers a centralized multi-tenant Container-as-a-Service (CaaS) service for container orchestration and management [16]. To enable the offloading to nodes instantiated via the CaaS service, a dedicated plugin which translates requests into Kubernetes commands and executes them on the CaaS nodes was developed.
- AI_INFN initiative: a prototype of a fully fledged AI development platform has been built adopting interLink as a solution to integrate external resources. In this context, a plugin to exploit "beefy" GPU VMs that are geo-distributed has been developed and successfully integrated. The integration with the HPC supercomputer Leonardo at CINECA is on going, see next. Also for this use case, the offloading towards CloudVeneto (on CPU-only nodes) was tested [17].
- ICSC cloud integration with Leonardo HPC: the current activity aims to mimic the edge-node setup demonstrated at Vega HPC to seamless integrate cloud resources exposing end-user services for the ICSC use cases, with the specialized hardware offered by a supercomputer like Leonardo. The peculiarity of this initiative is in the number of frameworks that can be demonstrated to work in synergy with the offloading mechanism. At the time of writing, the integration is functionally tested and the case studies measurements are ongoing.

5 Summary

We described how we are testing a new resource exploitation model that is fully cloud-native compatible and that can be extended to any backend capable of managing container lifecycles. We demonstrated the SLURM cluster integration first, as it represents one of the major gap in terms of accessibility to the specialized hardware hosted in supercomputers. We then extended the same approach to WLCG grid sites and beefy cloud VMs with possibly GPUs, leveraging the same interface provided by interLink. Following the Julich HPC example, providers can now offer an effortless Kubernetes interface for their resources without touching the way they manage their batch systems.

As we are approaching a broader audience and adoption, we have a roadmap to lead interLink future developments:

- Network Abstraction: the architecture is currently missing a network abstraction layer that ensures seamless communication between containers and backend systems, regardless of their physical location. The initial focus was on standalone payload to be offloaded. From the first round of feedback, it started to be evident that indeed network overlay is what can be a real game changer in terms of cloud-native feature parity. Therefore, we have drafted a version of interLink that is capable of bringing a user-level network overlay, again with minimal requirements from the provider perspective.
- Streamlining communities onboarding: documentation and real-world examples are the key to nurture an ecosystem driven by community use cases. This is from now on the major priority in the roadmap.
- Monitoring Tools: The architecture already includes monitoring tools (mainly from in-code telemetry) to track the performance and health of the infrastructure. Many metrics and dashboard can use a better description and an improved visualization.
- Fine-grained authN/Z: this is essentially about pushing the limit on which kind of policy engines can be attached to the offloaded mechanism. Advanced policy tool integration is on the list for making authN/Z granularity as fine as needed.

Aknowledgements

This work has been supported by the Fondazione ICSC, Spoke 3 Astrophysics and Cosmos Observations. National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR). Project ID CN_00000013 "Italian Research Center on High-Performance Computing, Big Data and Quantum Computing" funded by MUR Missione 4 Componente 2 Investimento 1.4: Potenziamento strutture di ricerca e creazione di "campioni nazionali di R&S (M4C2-19)" - Next Generation EU (NGEU).

This work was supported by the project 'An interdisciplinary Digital Twin Engine for science' (interTwin) that has received funding from the European Union's Horizon Europe Programme under Grant 101058386

References

[1] interLink home page, https://intertwin-eu.github.io/interLink/
[2] interTwin EU project, https://www.intertwin.eu/
[3] ICSC National Research Center for High Performance Computing, Big Data, and Quantum Computing, https://www.supercomputing-icsc.it/en/icsc-home/
[4] Tracking ML experiments with MLFlow, https://mlflow.org/
[5] The cloud-native ML platform KubeFlow, https://www.kubeflow.org/
[6] Daniel Fett and Ralf Kuesters and Guido Schmitz - A Comprehensive Formal Security Analysis of OAuth 2.0 - 2016 pre-print arxiv - doi.org/10.48550/arXiv.1601.01229

[7] G. R. Khattak, S. Vallecorsa, F. Carminati and G. M. Khan, "Fast simulation of a high granularity calorimeter by generative adversarial networks," Eur. Phys. J. C **82** (2022) no.4, 386 doi:10.1140/epjc/s10052-022-10258-4 [arXiv:2109.07388 [physics.ins-det]].

[8] Pérez, Alfonso and Risco, Sebastián and Naranjo, Diana María and Caballer, Miguel and Moltó, Germán - On-Premises Serverless Computing for Event-Driven Data Processing Applications - 2019 IEEE 12th International Conference on Cloud Computing (CLOUD) - doi.org/10.1109/CLOUD.2019.00073

[9] Apptainer, https://apptainer.org/

[10] EGI Checkin, https://aai.egi.eu/registry/

[11] A. Streit, D. Erwin, T. Lippert, D. Mallmann, R. Menday, M. Rambadt, M. Riedel, M. Romberg, B. Schuller and P. Wieder, - "UNICORE: From project results to production Grids," - arXiv:cs/0502090.

[12] D. Ciangottini, A. Forti, L. Heinrich, N. Skidmore, C. Alpigiani, M. Aly, D. Benjamin, B. Bockelman, L. Bryant and J. Catmore, *et al.* "Analysis Facilities for the HL-LHC White Paper" Computing and Software for Big Science - 2510-2036 - DOI: 10.1007/s41781-025-00133-8

[13] D. Ciangottini, T. Boccali, A. Ceccanti, D. Spiga, D. Salomoni, T. Tedeschi and M. Tracolli - "First experiences with a portable analysis infrastructure for LHC at INFN," - EPJ Web Conf. **251** (2021), 02045 doi:10.1051/epjconf/202125102045

[14] T. Tedeschi, V. E. Padulano, D. Spiga, D. Ciangottini, M. Tracolli, E. Tejedor Saavedra, E. Guiraud and M. Biasotto, "Prototyping a ROOT-based distributed analysis workflow for HL-LHC: The CMS use case," Comput. Phys. Commun. **295** (2024), 108965 doi:10.1016/j.cpc.2023.108965 [arXiv:2307.12579 [cs.DC]].

[15] P. Andreetto et al., "Merging OpenStack based private clouds: the case of CloudVeneto.it", Published in: EPJ Web Conf. 214 (2019) 07010, DOI: 10.1051/epjconf/201921407010

[16] F. Fanzago et al. - The CloudVeneto's Container-as-a-Service ecosystem. - To appear in proceedings of CHEP2024

[17] L. Anderlini, M. Barbetti, G. Bianchini, D. Ciangottini, S. Dal Pra, D. Michelotto, C. Pellegrino, R. Petrini, A. Pascolini and D. Spiga - "Supporting the development of Machine Learning for fundamental science in a federated Cloud with the AI_INFN platform," - arXiv:2502.21266.