Demo: Niflheim - End-to-End Middleware for Applications Across all Tiers of the IoT

Sven Akkermans

imec-DistriNet, KU Leuven
sven.akkermans@cs.kuleuven.be

Wouter Joosen

imec-DistriNet, KU Leuven
wouter.joosen@cs.kuleuven.be

Nicolas Small

imec-DistriNet, KU Leuven
nicolas.small@cs.kuleuven.be

Danny Hughes

imec-DistriNet, KU Leuven
danny.hughes@cs.kuleuven.be

ABSTRACT

The state-of-practice for Internet of Things (IoT) applications is deployment on specialised networks of embedded devices connected to a cloud backend. This paradigm is limited by the high latency and bandwidth incurred by communications with remote data servers and the inability to share specialised IoT infrastructure across applications. Efficiency can be improved by re-imagining all resources of the IoT infrastructure as micro-service hosting platforms. Applications decomposed as a set of services can then share IoT resources and run communicating modules closer together, tightening control loops and reducing latency and communications. This demo showcases Niflheim, a generic end-to-end middleware that provides modular microservice-based orchestration of applications on all resources across the tiers of the IoT, from IoT end-devices through gateways to the cloud. We demonstrate that this enables increased flexibility in application deployment and operations, while remaining efficient in terms of hardware and software requirements.

CCS CONCEPTS

• **Computer systems organization** → *Distributed architectures*;
• **Hardware** → *Sensor applications and deployments*; • **Software and its engineering** → *Middleware*;

KEYWORDS

Middleware, Mist Computing, Internet of Things

ACM Reference Format:

Sven Akkermans, Nicolas Small, Wouter Joosen, and Danny Hughes. 2017. Demo: Niflheim - End-to-End Middleware for Applications Across all Tiers of the IoT. In *Proceedings of SenSys '17*. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3131672.3136975>

1 INTRODUCTION

The Internet of Things (IoT) is a new development where devices and services are being connected to change the way we perceive and interact with the physical world. It consists of large networks

of heterogeneous resources and technologies, from remote cloud servers to constrained battery-powered sensors and actuators, and a whole new spectrum of previously infeasible applications. Prominent examples of this evolution are smart buildings and cities which provide new interactions between residents and infrastructure. Today, the IoT mostly consists of specialised platforms made from dedicated devices supporting a single application and connected to a remote cloud backend which analyses and reacts to the data provided by these devices. The cloud has the benefit of being a well-developed model but places the intelligence that controls the IoT in remote servers which is inefficient in terms of latency and bandwidth. This also tightly couples resources and application logic thereby imposing a significant initial investment and preventing the reuse of IoT infrastructure across applications. This is both costly and wasteful, as IoT devices put into a specialised role are not utilised to their full potential. Prior research better fuses the cloud model with the IoT, by moving processing to the edges of the network, also known as ‘Edge Computing’ [2]. However, currently it provides mostly limited application development and deployment support up to the gateway edges of the IoT infrastructure, referred to as ‘Fog Computing’, and is less concerned with constrained edge devices such as sensors and actuators or how to distribute applications across a heterogeneous set of resources. We demonstrate Niflheim, a unifying middleware that elevates IoT edge devices to the status of microservice hosting platforms, known as ‘Mist Computing’, and provides end-to-end application orchestration across all tiers of the IoT infrastructure from IETF Class-1 IoT devices, through gateways to the cloud. The increased flexibility offered by Niflheim for applications fosters infrastructure (re-)use and reduces latency and communications.

2 SYSTEM OVERVIEW

Niflheim realises applications as compositions of microservices, mostly single-purpose units of software, that are bound together to create a distributed application. It deploys these services to a suitable infrastructure resource, based on the service’s requirements, and does so autonomously, relieving the user of deciding where to put these services. Niflheim optimises application performance and infrastructure resource utilisation by closely locating communicating services, co-locating where possible. To reach all tiers of the IoT, it incorporates and extends two other pieces of work: (1) virtualising IoT devices to support multiple applications using CerberOS [1], and (2) dynamic deployment of modular applications across a cloud infrastructure [3]. Figure 1 shows the Niflheim Architecture

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SenSys '17, November 6–8, 2017, Delft, Netherlands

© 2017 Association for Computing Machinery.

ACM ISBN 978-1-4503-5459-2/17/11...\$15.00

<https://doi.org/10.1145/3131672.3136975>

Figure 1: The Niflheim architecture with the Application Management System and the infrastructure pools.

and its two main components: the Application Management System (AMS) and the infrastructure pools. Niflheim enables transparent interactions across the IoT infrastructure through the use of infrastructure pools, which groups heterogeneous resources by type and makes them accessible through a standard REST API. To enable the full cross-tier exploitation of IoT infrastructures, Niflheim manages pools of constrained CerberOS IoT devices, gateways, and cloud assets. The AMS acts on service-composed applications, created through a GUI that allows choosing, configuring and binding services through a simple drag and drop interface, as illustrated in Figure 2. The AMS relies on three primary elements: (1) the configuration engine reads and configures compositions, (2) the deployment engine requests resources from the resource manager and deploys services to them, and (3) the resource manager is the interface between the AMS and the infrastructure pools and selects suitable resources for the services.

3 DEMONSTRATION

To demonstrate Niflheim, we will deploy an IoT infrastructure consisting of more than 50 IoT edge devices connected to a private cloud backend. The devices are a mix of CerberOS powered μ PnP motes [4] and DC9003A-B SmartMesh IP Evaluation/Development Mote Modules¹ in a SmartMesh IP network, and a Raspberry Pi gateway that is connected to a SmartMesh IP manager and allows for cloud communications to and from the IoT network. The μ PnP motes are representative IETF Class-1 IoT edge devices and can dynamically add sensing and actuating functionality through plugable peripherals. The DC9003A-B motes provide a larger scale network to demonstrate Niflheim on. Niflheim has been evaluated on infrastructures up to three times this size. To demonstrate the ease-of-use of Niflheim, and its ability to manage multiple applications on the same set of resources, we will invite visitors to create and deploy their own applications on this setup. They will

¹<http://www.linear.com/solutions/3106>

Composition Creator

Figure 2: Application composition GUI. Users add services to the composition by clicking the screen, selecting the type of service, and filling in parameters. Services are bound by dragging a line from one to the other.

Figure 3: Photograph of a sample of the IoT infrastructure deployment on which we will demonstrate Niflheim live. Video of the live demo: <https://youtu.be/xTtad1NDCao>

be able to plug in peripherals such as temperature sensors and switches into the μ PnP motes and, using the GUI of Niflheim, dynamically compose an application from a set of available services. Their applications will then be autonomously deployed across the infrastructure and perform their function. The demo will also show the difference in runtime network performance between a traditional IoT deployment and a Niflheim deployment. In the typical IoT application deployment, all communications go to and from the cloud and gateway, while in the Niflheim case services will be deployed at all tiers of the IoT, placed close to each other or even where possible co-located. This setup keeps network traffic as local as possible, tightening control loops, thus reducing latency and bandwidth consumption in the system.

REFERENCES

- [1] Sven Akkermans, Wilfried Daniels, Gowri Sankar Ramachandran, Bruno Crispo, and Danny Hughes. 2017. CerberOS: a resource-secure OS for sharing IoT devices. In *EWSN '17 Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks*. Junction Publishing, Canada 2017, 96–107.
- [2] Redowan Mahmud and Rajkumar Buyya. 2016. Fog computing: A taxonomy, survey and future directions. *arXiv preprint arXiv:1611.05539* (2016).
- [3] Klaas Thoenen, Nicolas Small, Sam Michiels, Wouter Joosen, and Danny Hughes. 2016. Component-based orchestration of dynamic IaaS applications. In *Proceedings of the 15th International Workshop on Adaptive and Reflective Middleware*. ACM, 1–6. <https://doi.org/10.1145/3008167.3008169>
- [4] Fan Yang, Nelson Matthys, Rafael Bachiller, Sam Michiels, Wouter Joosen, and Danny Hughes. 2015. μ PnP: plug and play peripherals for the internet of things. *Proceedings of the Tenth European Conference on Computer Systems - EuroSys '15* (2015), 1–14. <https://doi.org/10.1145/2741948.2741980>