The EdgeNet System

Timur Friedman
Department of Engineering
UPMC Sorbonne Universités
Paris, France
timur.friedman@sorbonne-universite.fr

Rick McGeer
US Ignite
Orinda, CA, USA
rick.mcgeer@us-ignite.org

Berat Can Senel
Department of Engineering
UPMC Sorbonne Universités
Paris, France
berat.senel@lip6.fr

Matt Hemmings
engageLively
Victoria, Canada
mhemmings@engageLively.com

Glenn Ricart
US Ignite
Salt Lake City, UT
glenn.ricart@us-ignite.org

***Abstract*—EdgeNet is the prototype of a scalable, sustainable general-purpose testbed for very wide area distributed systems and extremely low-latency distributed services. In this, it is aimed at the same experimenters and systems that formed the core usage of previous, highly-successful wide-area testbeds such as PlanetLab[1], G-Lab[2], V-Node[5], GENI[4], and SAVI[3], and it incorporates many of the features that characterized those previous testbeds. EdgeNet's goal is to achieve the usability and research value of the previous generations of wide area testbed, whilst offering radical improvements in the scalability and sustainability of those systems. It achieves this scalability and sustainability through a strategy of using industry-standard open-source software as the basis of its software stack, and by a strategy of hardware-free, bottom-up, site-driven deployment. EdgeNet follows the Seattle[6] and PlanetIgnite[9] strategy of permitting sites to join the testbed with purely local action.**

Keywords—distributed testbed, containers-as-a-service, wide-area systems

I. Introduction

The National Science Foundation and other CS research funding agencies have funded a number of wide-area distributed clouds over the years: PlanetLab and GENI in the US, FED4FIRE and PlanetLab Europe in the EU, SAVI in Canada, G-Lab in Germany, V-Node in Japan. Each of these testbeds hit scaling limits fairly rapidly, with PlanetLab still holding the record at 1353 nodes at 717 sites. *EdgeNet* is the prototype of a highly-scalable, sustainable, usable worldwide distributed cloud. EdgeNet achieved this by achieving three primary goals:

- *Scalability*: The ability to grow without limit implies that EdgeNet must spread without central action; any site is able to join EdgeNet and contribute worker nodes through local action only.
- *Sustainability* implies that both per-site and centralized costs must be low, with the *incremental* costs of each new EdgeNet site is near zero.
- *Usability* implies that the tools and technologies required to deploy an experiment or service on EdgeNet must be familiar and continuously improving without the involvement of EdgeNet. Further, contributions of EdgeNet to these tools and technologies will feed back into the broader open-source community, to better leverage the investment into these tools and to gain the benefit of community improvement and maintenance.

In sum, EdgeNet's purpose is to develop and demonstrate a scalable, sustainable, usable distributed worldwide cloud, which achieves its goals by consciously leveraging and acting in harmony with the global social and cyber environments. Just as kinetic technologies are ultimately limited by their resource footprints, cyber technologies are limited by their software, hardware, and administrative footprints. The EdgeNet project is ultimately about a quantum leap in the limits to distributed systems technologies; and therefore it shows a "live-off-the-land" strategy of an extremely light cyber footprint.

II. Living Off the Land

The live-off-the-land strategy was dependent on the adaptation of local hardware and software technologies to the wide-area use case, and so the primary goal in EdgeNet was achieving this adaptation. Usability required the adaptation of cluster technology to the wide area, and it was important to determine whether these technologies would work in a wide-area environment.

Sustainability and living off the land requires more than a lightweight software and hardware footprint; it also means full engagement with the broader cyber community, recruiting volunteers to help build EdgeNet. The way open-source projects attract volunteer participants is to make participation in the open-source project advance the volunteers' objectives. In order to do this, the project work must be as broadly applicable as possible; the use of open-source, broadly-applicable software as a project base greatly aide in this.

The Internet of Things and Smart Cities applications require low latencies between the edge cloud and the sensors and actuators which are deployed at the edge. This was recognized and is a focus of the *NSF/VMware Partnership on Edge Computing Data Infrastructure*. This is also the focus of a number of commercial cloud efforts, notably Azure IoT Edge.

III. Software Architecture

We chose Kubernetes (K8s) as the control framework. We considered two alternative architectures: the first was to deploy a K8s cluster of VMs at each site, with a VM as the local headnode and one or more VMs per worker node; the second was a single, worldwide cluster with a single head node and several worker nodes per site. We chose the latter, because:

1. A primary goal of EdgeNet is to determine how *little* we must require from sites and still have a valuable, useful cluster. PlanetLab and GENI

978-1-7281-2700-2/19/$31.00 ©2019 IEEE

demonstrated each requirement for a site is both painful for some sites and a potential source of failure. Footprint minimization was thus a primary design goal of EdgeNet. The single lightest footprint possible was a single VM/site, with memory and disk minima and a routable IP address, and that drove EdgeNet's lightweight goal.

2. Deployment across a Kubernetes cluster is one of the most common operations in Cloud computing: thus, there are a wide variety of tools, technologies, and tutorials available to do just that, available on every platform. kubectl is the workhorse engine familiar to every K8s user. Our usability and sustainability goals implied that every EdgeNet user should be able to deploy a slice using kubectl *alone and unmodified*. Making EdgeNet a single worldwide cluster permitted that.

Once this architecture had been adopted, we set about the implementation. The first order of business was to determine whether the architecture was feasible. In particular, K8s is designed as a cluster solution, where node-node and headnode-node latencies are in the microsecond range. EdgeNet has node-node and headnode-node latencies in the *tens* of milliseconds, four orders of magnitude beyond the design point of K8s. Feasibility of this was not guaranteed.

We first ran experiments on CloudLab[7] and Emulab, setting up a K8s cluster and testing it under a wide variety of simulated latencies and network conditions, determining that the cluster was able to keep functioning even under condition of high latency and moderate loss. Once this was established, we implemented a portal and headnode to administer a worldwide cluster.

A. A Heterogeneous Kubernetes

A key distinction between EdgeNet and K8s' design point is that the nodes in K8s clusters are assumed to be homogenous and identical; thus, it doesn't matter *which* worker node runs a particular Pod (the unit of deployment in K8s; equivalent to a *sliver* on PlanetLab or GENI), merely the number of Pods of each kind running. EdgeNet nodes are heterogeneous in a number of dimensions, the most important of which is location. As a result, the experimenter/developer must be able to control on which nodes his experiment/application runs.

The solution to this is labeling, a K8s feature to control a number of aspects of K8s, notably including Pods, deployments, and nodes. We designed a global labeling scheme for nodes, with labels at the city, region, nation, and continent level. This permits developers and experimenters fine control over pod deployment.

B. User Experience

An important mantra to users is that "EdgeNet is just Kubernetes". The user logs in to the EdgeNet portal using his Google account (other identity providers will be supported later), agrees to the AUP, and then is offered a certificate to download. Once he has that, standard Kubernetes tools such as kubectl can be used to deploy an experiment. Further, standard Kubernetes tools to maintain, contact, and sample data for an experiment are available. This is a significant upgrade in user experience over previous testbeds, where experimenters had to stand up their own messaging systems or distributed servers to take measurements from their systems.

C. Adding a Node

Adding a node to EdgeNet is simple. A local VM is allocated, using whatever local methods are in place. Any Linux distibution will work, though Debian, Ubuntu, or CentOS are recommended. The site then downloads an add-node script from the portal and runs it; the script downloads the Docker and Kubernetes software, installs it, and registers with the head node. Node addition has been measured as a one-minute operation

IV. Status

EdgeNet is up and running at over 30 sites worldwide, and has working partnerships with PlanetLab Europe and the NSF/VMWare Partnership on Distributed Computing.

V. Acknowledgements

This research was supported by the National Science Foundation under Award 1820901. The authors thank our colleagues on that proposal, Justin Cappos and Albert Rafetseder of NYU, who contributed significantly to the system, our colleagues at GENI, SAVI, and PlanetLab Europe who havc contributed and maintained nodes, the reviewers who gave feedback on the submission of this paper, and our colleagues at MeasurementLab and the worldwide volunteer team maintaining EdgeNet.

References

[1] B. Chun, D. Culler, T. Roscoe, A. Bavie, L. Peterson, M. Wawrzoniak M, Bowma. "Planetlab: an overlay testbed for broad-coverage services". *ACM SIGCOMM Computer Communication Review*. 2003 Jul 1;33(3):3-12.

[2] D. Schwerdel, B. Reuther, T. Zinne, P. Mülle, P. Tran-Gia "Future Internet research and experimentation: The G-Lab approach". *Computer Networks*. 2014 Mar 14;61:102-17.

[3] JM Kang, T. Lin, H. Bannazadeh, A. Leon-Garcia "Software-defined infrastructure and the SAVI testbed". In *International Conference on Testbeds and Research Infrastructures* 2014 May 5 (pp. 3-13). Springer, Cham.

[4] R. McGeer, M. Berman, C. Elliott, R. Ricci R, editors. *The GENI Book*. Berlin, Germany: Springer; 2016 Aug 31.

[5] A. Nakao, K. Yamada. "Research and Development on Network Virtualization Technologies in Japan: VNode and FLARE Projects". In *The GENI Book* 2016 (pp. 563-588). Springer, Cham.

[6] J. Cappos, I. Beschastnikh, A. Krishnamurthy, T. Anderson. "Seattle: a platform for educational cloud computing." In *Acm SIGCSE bulletin* 2009 Mar 4 (Vol. 41, No. 1, pp. 111-115). ACM.

[7] R. Ricci, E. Eide, C. Team. "Introducing CloudLab: Scientific infrastructure for advancing cloud architectures and applications." ; *login:: the magazine of USENIX & SAGE*. 2014;39(6):36-8.

M. Berman, T Friedman, A. Gosain, K. Keahey, R. McGeer, I. Moerman,

[8] A. Nakao, L. Nussbaum, K. Rauschenbach, V. Syrotiuk, M. Veeraraghavan.."Report of the Third Global Experimentation for Future Internet (GEFI 2018) Workshop". *arXiv*, preprint arXiv:1901.02929. 20

[9] A. Bavier, R. McGeer, G. Ricart "Planetignite: A self-assembling, lightweight, infrastructure-as-a-service edge cloud." In 2016 *28th International Teletraffic Congress (ITC 28) 2016* Sep 12 (Vol. 1, pp. 130-138). IEEE.