Computer Networks 68 (2014) 187–198

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Internet factories: Creating application-specific networks on-demand

Rudolf Strijkers [a,b,*], Marc X. Makkes [a,b], Cees de Laat [a], Robert Meijer [a,b]

[a] University of Amsterdam, Netherlands
[b] Netherlands Organisation for Applied Scientific Research TNO, Netherlands

ARTICLE INFO

Article history:
Received 15 May 2013
Received in revised form 24 December 2013
Accepted 28 January 2014
Available online 12 February 2014

Keywords:
Computer networks
Programmable networks
Distributed computing
Network management
Network architecture

ABSTRACT

We introduce the concept of Internet factories. Internet factories structure the task of creating and managing application-specific overlay networks using infrastructure-as-a-service clouds. We describe the Internet factory architecture and report on a proof of concept with three examples that progressively illustrate its working. In one of these examples, we demonstrate the creation of a 163-node IPv6 network over 18 cloud locations around the world. Internet factories include the use of libraries that capture years of experience and knowledge in network and systems engineering. Consequently, Internet factories solve the problem of creating and managing on-demand application-specific overlay networks without exposing all their intricacies to the application developer.

1. Introduction

Infrastructure-as-a-service clouds [1] have the potential to become an elementary part of large and complex systems, such as cyber physical systems [2] and ICT systems to support smart cities [3,4]. Nowadays, we all know Internet applications and services, such as Spotify [5] and Youtube [6], that have resulted in dedicated overlay infrastructures (servers, networks, storage facilities) on top of the Internet. Instead of creating dedicated overlay infrastructures for large and complex systems, they can now be implemented entirely in the application domain with infrastructure-as-a-service clouds. Consequently, developers of Internet applications and services are confronted with an enormous increase in complexity. They must deal with network domain complexity and life cycle management of the overlay infrastructure, in addition to the application domain complexity. To make this approach feasible, we need to develop a software architecture in which domain experts – application developers and network engineers – can apply and reuse knowledge in the design, implementation, and management of overlay infrastructures. Consequently, a new paradigm to create large and complex overlay infrastructures emerges in which developers build on experience and achievements of domain experts.

In essence, overlay infrastructures on top of the Internet add capabilities for controlling software and resources at specific locations. A generic way to provide these capabilities is to use virtual machines. Not surprisingly, network research test beds use virtual machines to experiment with new networking and distributed computing technologies [7–10]. Even the telecom industry is adopting the use of virtual machines for encapsulating network element functions (such as firewalls, residential gateways) [11]. Still, the emergence of infrastructure-as-a-service clouds marks a significant step from state of the art, because developers gain programmatic control over the overlay infrastructure and its application services. In the end, application developers only want to deal with the infrastructure details

* Corresponding author at: University of Amsterdam, Netherlands. Tel.: +31 622901201.
E-mail address: rudolf@strijkers.eu (R. Strijkers).

http://dx.doi.org/10.1016/j.comnet.2014.01.009
1389-1286/

relevant to their application. So, the problem is how to create and manage overlay infrastructures without exposing all their intricacies to the application developer. We introduce the Internet factory concept as a solution to this problem.

The Internet factories concept is based on the broader concept of software factories in software engineering [12]. The rationale of software factories is that the majority of application development can be captured by standardized components and common patterns. Instead of reinventing the wheel for problems they encounter, developers re-use existing components and customize, adapt, and extend them for their needs.

An Internet factory is a software architecture in which domain experts implement libraries and design patterns to simplify the development of applications that interwork with an overlay infrastructure. It builds on the User Programmable Virtualized Network (UPVN) architectural framework, which models the general programming problem for interworking between application domain programs and the network domain [13,14]. In this article we refer to these application domain programs as Netapps – Networked Applications, to distinguish from other application programs that do not interwork with the network domain. So, an Internet factory provides the libraries and common functions with which developers can implement Netapps, such as a cyber physical system, new networking concept [15], or application-specific overlay [16].

The contribution of this article is the design and implementation of an Internet factory proof of concept to support the development of Netapps. We developed three Netapps that progressively illustrate the deployment stages in the life cycle of an overlay infrastructure: creation, configuration, and management. The proof of concept exploits resources of infrastructure-as-a-service cloud (from now on referred to simply as clouds) providers whose interfaces are accessible via in the Internet. We also show how libraries, tools, and frameworks can be applied to simplify Netapp development. For instance, we describe a library in which the creation of an IPv6 overlay infrastructure is fully automated and how third party applications can be used to adapt networks. In fact, the presented work shows the potential of Internet factories to create large and complex overlay infrastructures.

Related work and the Internet factories concept are presented in respectively Sections 2 and 3. In Section 4, the design and implementation of an Internet factory, called Sarastro, is presented. Sarastro is evaluated by illustrating three typical Internet factory patterns in Section 5. The implications of Internet factories are discussed in Section 6 and the paper ends with the conclusion and future work in Section 7.

2. Related work

In the past, programmable networking concepts have been proposed to give application developers more control over the network domain. In its simplest form, a programmable network consists of a collection of programmable network elements, such as commodity PCs, which can simply be loaded with software providing new functionality for processing network traffic. The basic concepts to program a collection of programmable network elements have been developed and validated in active network research [17] and in subsequent research [9,18–20] of which Software Defined Networks (SDNs) is currently the most popular [21]. The concepts allow a programmable network to become an active component of distributed applications by exposing APIs to such applications or by allowing applications to embed network instructions in the packets they send over the network. Network operators use the same interfaces to create network services for specific applications or to optimize network services for their own operations. In practice, this leads to an increase in complexity for both network operators and application developers.

For example, application developers typically use sockets to implement communication over the Internet without knowing details of intermediate networks. When using programmable network interfaces, they must also deal with details of the networks, such as routing and topology. Choices about where to place application code in a network, filtering and redirecting traffic to the code, and interventions on failures all require in-depth networking expertise. For any service that reaches beyond a single ISP, the complexity to setup and manage a network service over multiple domains quickly increases.

Network engineers who understand the network requirements of application developers can implement a solution in the network on their behalf. Then the roles are turned; the network engineer must understand the intricacies of the application domain and anticipate on the network functions and services it requires at what moment. With traffic engineering and control plane programmability [21,22], solutions can be implemented strictly in the network domain. In practice, due to the scale, complexity, and diversity of the application domain, the best that network engineers can do is to over provision and prioritize certain traffic classes.

The only alternative is that application developers create their own services and interventions as an overlay on top of the Internet. On one hand, overlay networks such as Planetlab [7] and content delivery networks [23] both exploit the Internet and implement solutions to reach past its limitations. For example, Planetlab and similar initiatives [8,9,24] allow researchers to develop and test new network and distributed computing services, which cannot be done on the Internet. On the other hand, the creation of an overlay networks result in an additional, often dedicated, infrastructure layer on top of the Internet. Thus, developers of these overlay infrastructures require expertise of both the application and network domain and need to deal with their interworking. Internet factories help in compartmentalizing the complexity of common patterns, so the developer can focus on the application-specific issues.

Two additional research domains are related: peer-to-peer networks [25] and autonomic networks [26]. In contrast with the overlay networks previously mentioned, peer-to-peer networks exist exclusively on end-user hosts. Consequently, peer-to-peer technologies focus on delivering a service despite intermittent connectivity, for exam-