Computation offloading with ICN

Michał Król
UCL
m.krol@ucl.ac.uk

Adrian-Cristian Nicolaescu
UCL
a.nicolaescu@ucl.ac.uk

Sergi Reñé
UCL
s.renel@ucl.ac.uk

Onur Ascigil
UCL
o.ascigil@ucl.ac.uk

Ioannis Psaras
UCL
i.psaras@ucl.ac.uk

David Oran
Network Systems Research & Design
daveoran@orandom.net

Dirk Kutscher
Huawei
dirk.kutscher@huawei.com

ABSTRACT

This demo shows an implementation of a computation-centric architecture over NDN. The system is able to perform in-network load balancing of incoming computation requests, reliably authenticate consumers and allow them to submit large payloads without routable prefixes. The system is able to migrate requested function in a form of unikernels where they are needed, follows ICN pull-based model and introduces only minimal changes to the NDN stack.

CCS CONCEPTS

• **Networks → In-network processing**; *Naming and addressing*; *Network architectures*; Session protocols;

KEYWORDS

Information Centric Networks, Named Data Networking, in-network processing, naming, thunks

ACM Reference Format:
Michał Król, Adrian-Cristian Nicolaescu, Sergi Reñé, Onur Ascigil, Ioannis Psaras, David Oran, and Dirk Kutscher. 2018. Computation offloading with ICN. In *5th ACM Conference on Information-Centric Networking (ICN '18), September 21–23, 2018, Boston, MA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3267955.3269009

1 INTRODUCTION

During the past two decades, we have been witnessing a continuous trend towards centralising Internet content delivery and application-oriented computation. Centralisation led to the development of huge-scale data-centres (commonly referred to as the cloud), which is where 90% of users' requests end up being executed [1]. Although this trend served well the purposes of the Internet as we know it today and achieves impressive economies of scale, it is certainly not a good fit for a number of future applications.

ICN '18, September 21–23, 2018, Boston, MA, USA

ACM ISBN 978-1-4503-5959-7/18/09.
https://doi.org/10.1145/3267955.3269009

Edge-/Fog-computing has been proposed recently as a complementary paradigm to the cloud. The main driver behind the edge-/fog-computing trend is the de-centralisation of the cloud into multiple smaller scale computing devices ranging from mini-data centres to server racks, WiFi APs and Raspberry Pis. The need for a shift from the traditional/current host-centric paradigm to a more flexible information and computation-centric environment is becoming clear. The current, IP-based routing, forwarding and especially the name resolution model is brought to its knees if applied to an environment where computation resources need to be chosen and invoked at millisecond timescales.

Based on a loosely coupled communication model, Information-Centric Networks (ICN) eschews a host-centric communication model. The paradigm uses content-identifiers directly as network names which simplifies discovery, permits direct access to data, and offers mobility support inherently [2]. Therefore, the paradigm can be a great fit for provider-agnostic distributed clouds: it does not matter where and by whom an application/function is executed, as long as the result is correct, valid, verified and trustworthy. This is in stark contrast to a host-centric edge-computing environment where edge devices need to connect to some specific IP address operated by a trusted entity (e.g., redirected from the cloud).

Despite its conceptual fit, the vast majority of ICN approaches to date focus on naming, routing/forwarding and distribution of static content. In view of these limitations, multiple works have recently tried to extend ICN's capabilities to deal with dynamic content. Notable among these efforts, Named Function Networking (NFN) [3] and Named Function as a Service (NFaaS) [4] extend ICN's named data access model to a remote function invocation capability, enabling consumers to request the network to execute functions remotely. In NFN [3], for instance, function invocation corresponds to independent computational processes, evaluated as expressions in a functional programming model, while NFaaS exploits unikernels migration to satisfy users' requests.

In addition to NFaaS/NFN, there have been several other approaches for integrating computation with ICN. However, when using them to realize real-world applications like web-style interactions, several additional aspects beyond the fundamental Named Function invocation concept need to be addressed: consumer authentication and authorization, parameter passing and accommodating non-trivial computations

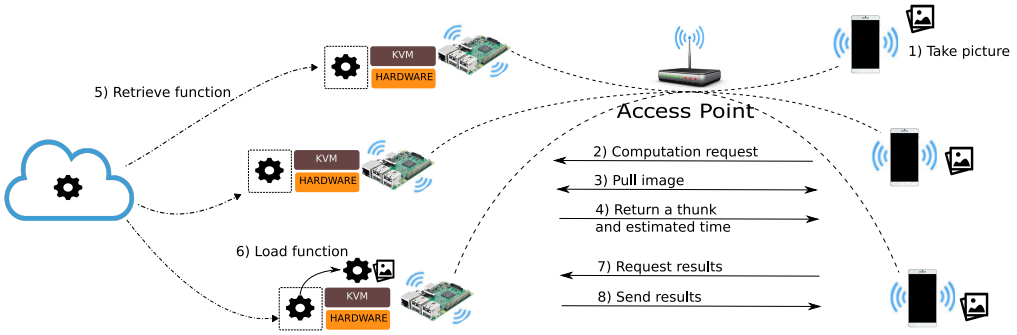

Figure 1: Demo description

To deal with those problems we recently proposed RICE - a remote method invocation framework for ICN [5]. RICE introduces a secure, 4-way handshake for ICN in order to achieve shared secret derivation, consumer authentication and parameter passing. It also employs the concept of *thunks*[1] from the programming language literature to decouple method invocation from the return of results to enable long-running computations. The *thunk* is used to name the results for retrieval purposes.

The goal of the demo is to prove the feasibility of in-network function execution with client authentication and non-trivial parameter passing, to support cases where computation takes longer than PIT expiry time. In our system, users can take pictures and submit them to the network requesting image processing. The requests are dispatched to several Raspberry PIs acting as workers and hosting functions in a form of unikernels. When the processing is done, users can retrieve the final result using thunks.

2 DEMO DESCRIPTION

Our demonstration showcases a computation-centric architecture using Android phones (Fig. 1). The system consists of the following actors:

- Users - taking pictures and sending processing requests using Android phones.
- Workers - Raspberry Pis receiving and processing requests.
- Access Point - setting up a wireless network and performing load balancing among workers.

Users take pictures using their phones and send them to the network for processing. The requests are automatically dispatched by the Access Point to multiple deployed workers to equally share the computation load. Raspberry Pis, acting as workers, implement an optical character recognition (OCR) algorithm and detect text included in the pictures. After, processing workers return results to the requesting users.

2.1 Implementation

We implement our system on top of NDN [6]. Each system component contains an NFD forwarder. The devices are connected in a star topology with the AP in the center. The users have to first take a picture of a text (step 1). Requests in the form of Interest messages are forwarded from phones to the AP that implements a round-robin forwarding strategy to perform load balancing among workers (step 2). The Interests contain a handshake TLV field set up in order to initiate the 4-way handshake presented in [5]. Upon reception of a request, workers pull images from phones using the handshake procedure (step 3) and return a *thunk* name and an estimated time for processing the image (step 4).

The image processing function represented as a unikernel (as described in [4]) is retrieved from the network (step 5) instantiated on the worker and fed with an image to process (step 6). The image fetching process is an optional step, if the file to be processed is not on the worker already, or provided by the Android phone(s) at the start of or before the initialization of the computation offloading communication process.

Users wait for the time returned by workers and use the *thunks* to request the final result (step 7). The corresponding worker returns the result as a regular Data message (step 8). For a detailed description of *thunks* and payload submission we refer readers to [5].

3 CONCLUSIONS AND FUTURE WORK

In our demo, we have successfully showcased a practical example of how to distribute computation at the edge of the network using NDN, outsourcing computation from users' Android smartphones, to a network of Raspberry PIs for an image recognition application. The same example can apply to other similar trendy usecases, such as augmented reality applications, where we think the mix of edge-computing and ICN networking is a perfect candidate to fulfill its (i.e., low latency) requirements.

REFERENCES

[1] Cisco Global Cloud Index. Forecast and methodology, 2016-2021 white paper. *Retrieved 1st June*, 2017.
[2] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.
[3] Christian Tschudin and Manolis Sifalakis. Named functions and cached computations. In *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, pages 851–857. IEEE, 2014.
[4] Michał Król and Ioannis Psaras. Nfaas: named function as a service. In *Proceedings of the 4th ACM Conference on Information-Centric Networking*, pages 134–144. ACM, 2017.
[5] Michał Król, Karim Habak, David Oran, Dirk Kutscher, and Ioannis Psaras. Rice: Remote method invocation in icn. https://www.ee.ucl.ac.uk/~ipsaras/files/remote-method-invocation-thunks.pdf, 2018.
[6] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, et al. Named data networking (ndn) project. *Relatório Técnico NDN-0001, Xerox Palo Alto Research Center-PARC*, 2010.

[1] https://en.wikipedia.org/wiki/Thunk