ICN-based Edge Service Deployment in Challenged Networks

Christos-Alexandros Sarros*, Adisorn Lertsinsrubtavee†, Carlos Molina-Jimenez†, Konstantinos Prasopoulos‡, Sotiris Diamantopoulos*, Dimitris Vardalis*, Arjuna Sathiaseelan†
*Athena Research and Innovation Center, Greece
†Computer Laboratory, University of Cambridge, UK
‡Department of Electrical and Computer Engineering, Democritus University of Thrace, Greece
alexcsarros@ipet.gr,al773@cam.ac.uk,cm770@cam.ac.uk,konspras@ee.duth.gr,sdiaman@ipet.gr,dvardali@ee.duth.gr, as2330@cam.ac.uk

ABSTRACT

In this demo we present a NDN-based approach to deploy dockerised services closer to end-users when the network is impaired. We further increase resiliency, employing DTN to tunnel traffic between intermittently connected NDN nodes.

CCS CONCEPTS

• **Networks** → *Network architectures*; *Network layer protocols*;

KEYWORDS

Named Data Networking, Delay Tolerant Networking, Information-Centric Networking, Service-Centric Networking

1 INTRODUCTION

Service deployment in post-disaster scenarios and remote areas is a challenging task that has not been fully addressed yet, although offering emergency services to users can be vital. Information-Centric Networking (ICN) has been proposed as a potential solution to provide opportunistic connectivity in such scenarios [2].

Our contribution is two-fold: First, we apply Service-Centric Networking (SCN) [1] principles to deploy services (implemented as lightweight containers) at the edge of the network. Being based on ICN, services and content are decoupled from their physical locations. Following the fog computation paradigm, simple information processing can take place locally, (even in remote hotspots) significantly reducing data ferring to and from the core network.

Secondly, we depart from most of the existing work (e.g., [3]), which develops mechanisms for delay-/disruption-tolerant data transfers in ICN from scratch. Instead, we embrace a layered design leveraging the well-matured DTN mechanisms for data ferrying and routing in challenging environments, when such need arises.

Towards this end, we enhance the Named Data Networking (NDN) architecture [4] with a new DTN face that communicates with an underlying DTN implementation. This way, intermittent communication details are abstracted from NDN and handled transparently by the DTN layer, allowing for different DTN implementations to be integrated, depending on the deployment requirements. In this demo, we use the IBR-DTN [1] reference implementation.

ICN '17, September 26–28, 2017, Berlin, Germany

ACM ISBN 978-1-4503-5122-5/17/09.
https://doi.org/10.1145/3125719.3132096

2 SYSTEM ARCHITECTURE

Our solution is based on the UMOBILE architecture, which enhances NDN with delay-/disruption-tolerant and opportunistic communications support. We show the components related to edge service deployment in Figure 1 and describe the contributing mechanisms in the following sub-sections.

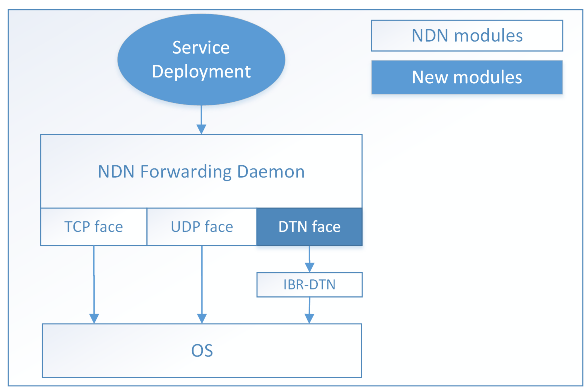

Figure 1: Architectural overview.

2.1 Service deployment

Our architecture is underpinned by lightweight dockerised services, which are treated as regular content by the network but can provide additional functionality as they are seamlessly deployed and executed across all compatible devices (e.g., a hotspot). This is achieved by building services as containers coupled with a semantic naming scheme. This way, we take advantage of caching and name-based routing to allow users to access nearby copies of services. Furthermore, thanks to the semantic naming scheme, a user can request a service that matches certain criteria. For instance, the location of users can be embedded into an Interest message to request customized services (e.g., information of local emergency authorities, local map).

[1]https://trac.ibr.cs.tu-bs.de/project-cm-2012-ibrdtn/wiki/docs

2.2 DTN forwarding

To facilitate intermittent communications in NDN, we have implemented a DTN face. This new face communicates with an underlying DTN implementation that handles intermittence by encapsulating Interest and Data packets in DTN bundles. Store-and-forward techniques are used by the underlying DTN implementation to reliably deliver the bundles utilizing standard DTN routing protocols. Upon reaching their destination, the bundles are decapsulated and the original Interest/Data messages are forwarded, resuming typical NDN communications. A possible deployment related to this demo includes a NDN-capable edge node (e.g., hotspot) that maintains a FIB entry towards the next edge node through the DTN face and one or more intermediate DTN mobile nodes that forward packets between the NDN nodes, essentially forming a tunnel.

By tunneling traffic through DTN, we create an alternative, reliable communication channel for NDN in situations where typical TCP and UDP faces would fail. We also enable data forwarding between two remote NDN nodes by using different upstream and downstream intermediate (DTN-capable) routes/nodes, without any modification of NDN semantics. This can alleviate the fact that the NDN breadcrumb routing approach requires the same node that forwards an Interest to return the Data as well, inhibiting data delivery capabilities in such environments. While the DTN face can be readily used by the existing NDN forwarding strategies, it is in our future plans to design strategies that fully leverage its potential.

2.3 Multi-Interest forwarding

NDN natively follows a synchronous communication model where a consumer sends one Interest message to retrieve one Data message in turn. This model is inadequate in scenarios where there is a single communication link between the consumer and producer, like in the disaster scenario depicted in Figure 2. In this example, the Android phone would need to travel several times between the main network and the disaster area to deliver all the data chunks.

As a solution, we implemented a *multi-Interest forwarding* model to support the NDN forwarding daemon: the idea is to allow the consumer to issue an aggregation (burst) of N Interests and retrieve multiple Data chunks at once. A consumer initially sends an Interest message requesting a service, which is forwarded towards the original content source or nearby cache. Along with the first Data message, the consumer receives information about the total content size and data chunk size. On this basis, it calculates the number of requests required to retrieve the whole content TNc. The consumer can request all remaining chunks within the second request by setting $N = TNc$ at the risk of creating network congestion if the content size is large. To avoid the problem, the value of N should be small. In this demo we statically set $N = 30$. Ideally, N should be adaptively calculated based on the current network condition and size of the content. We aim to investigate this issue in the future.

3 SYSTEM DEMONSTRATION

The demo scenario is depicted in Figure 2. We assume that HS_1 is at the edge of the main network and that the disaster area network is miles away and disconnected. The aim is to retrieve a service (S) from the *Main Network* and deploy it in the *Disaster area* (HS_2). We assume S to be a stateless service, e.g., a self-contained web server.

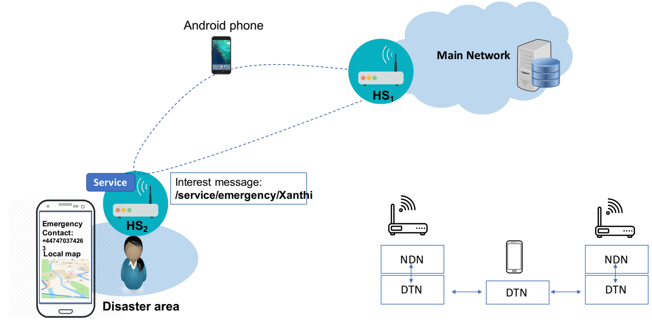

Figure 2: Service provisioning in challenging environments.

Both Wi-Fi hotspots HS_1 and HS_2 are equipped with the UMOBILE protocol stack of Figure 1. *Android phone* is a physically mobile, DTN-enabled node that can travel backwards and forwards between HS_2 and HS_1. In fact, any smart device attached to a UAV or any other means of transport can replace the Android phone, as long as it supports a DTN framework (e.g., IBR-DTN).

An Interest request is initially constructed by HS_2, including the name of the desired service (S). The location of the affected area is embedded into the Interest name (`/service/emergency/Xanthi`). This Interest is forwarded to the DTN face when the phone becomes wirelessly reachable to HS_2. The Interest is encapsulated in a DTN bundle and stored in the phone's persistent storage. The phone loses contact with HS_2 when it travels towards HS_1. When it reaches HS_1, the Interest is decapsulated and delivered to the NDN layer. It is then transferred by HS_1 through the main network, towards the service producer or the nearest cache. The latter —or another intermediate NDN node in possession of S— retrieves it and prepares a service image with customized information (i.e., emergency contact of local civil protection authority or map in `Xanthi`). The reply includes one or more Data messages sent along a reverse path using the DTN face. Communication follows the multiple-interest protocol of Section 2.3. When HS_2 receives all the chunks of S, it calls a service deployment function to execute S to provide the emergency service to local users via WiFi. They can access S from any standard web browser.

4 ACKNOWLEDGMENTS

This work has been supported by the EC H2020 UMOBILE project (GA no. 645124).

REFERENCES

[1] Arjuna Sathiaseelan, Liang Wang, Andrius Aucinas, Gareth Tyson, and Jon Crowcroft. 2015. SCANDEX: Service Centric Networking for Challenged Decentralised Networks. In *MobiSys-DIYNetworking*.

[2] Jan Seedorf, Mayutan Arumaithurai, Atsushi Tagami, K. Ramakrishnan, and Nicola Blefari-Melazzi. 2017. *Research Directions for Using ICN in Disaster Scenarios.* Internet-Draft draft-irtf-icnrg-disaster-02. Internet Engineering Task Force. Work in Progress.

[3] Atsushi Tagami, Tomohiko Yagyu, Kohei Sugiyama, Mayutan Arumaithurai, Kenichi Nakamura, Toru Hasegawa, Tohru Asami, and KK Ramakrishnan. 2016. Name-based push/pull message dissemination for disaster message board. In *IEEE LANMAN*.

[4] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, K. C. Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. 2014. Named Data Networking. *ACM SIGCOMM Computer Communication Review* 44, 3 (July 2014).