Device-to-Device Communication with Named Data Networking

Wentao Shang
UCLA
wentao@cs.ucla.edu

Alex Afanasyev
Florida International University
aa@cs.ucla.edu

Yanbiao Li
UCLA
lybmath@cs.ucla.edu

Jeff Burke
UCLA
jburke@remap.ucla.edu

Lixia Zhang
UCLA
lixia@cs.ucla.edu

ABSTRACT

Named Data Networking (NDN) architecture uses data-centric communication primitives that naturally support direct device-to-device (D2D) communications. To make NDN-enabled D2D communication a reality, this poster aims at two goals. First, we report our recent progress in enabling NDN connectivity over a number of popular D2D networking technologies. Second, we share with the broader community the roadblocks that we discovered in the process. Our experience suggests that launching a new network protocol stack for D2D communication on common platforms can be a daunting engineering challenge because of the lack of standard cross-platform APIs, limited documentation, and general platform restrictions to use L2 interfaces directly. Moreover, platforms are often equipped with different D2D networking technologies, forcing one to use many different means to interconnect different types of systems.

CCS CONCEPTS

• **Networks** → **Network architectures**; **Ad hoc networks**;

KEYWORDS

NDN, device-to-device, D2D

ACM Reference Format:
Wentao Shang, Alex Afanasyev, Yanbiao Li, Jeff Burke, and Lixia Zhang. 2017. Device-to-Device Communication with Named Data Networking. In *Proceedings of ICN '17, Berlin, Germany, September 26–28, 2017,* 2 pages. https://doi.org/10.1145/3125719.3132101

1 INTRODUCTION

With the proliferation of portable computing devices and the rapid development of the Internet of Things (IoT), there has been an increasing demand to support seamless direct device-to-device (D2D) communication. Example applications include short-range personal area networks (PANs), vehicular networks interconnecting nearby cars, and wide-area mesh networks supporting smart cities, smart grids, precision agriculture, etc. Creating such networks and applications over infrastructure-less D2D networks using the TCP/IP protocol stack is a non-trivial task with several technical hurdles [10].

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

ACM ISBN 978-1-4503-5122-5/17/09…$15.00
https://doi.org/10.1145/3125719.3132101

A promising way to meet D2D communication needs is provided by the Named Data Networking (NDN) [11] architecture. NDN names data packets using application-defined, hierarchical, and semantically meaningful names. NDN secures each named data packet using a cryptographic signature and encrypts the content whenever needed. Therefore, NDN-enabled nodes can communicate over any physical channel between them, without reliance on infrastructure support–in particular, without reliance on mapping application-layer names to network-layer addresses. A number of pilot NDN application trials have demonstrated successful operation in infrastructure-less D2D network environments [1, 8, 9].

In this poster, we report the results from our engineering efforts over the last two years in enabling D2D NDN connectivity. These efforts have evolved the NDN codebases to support NDN D2D over a variety of wireless communication technologies, which demonstrates NDN's ability to adapt to different link-layer technologies as a universal network-layer protocol. We also document the lessons we have learned and obstacles we faced while implementing support for the new networking stack on different platforms.

2 NDN SUPPORT FOR D2D NETWORKS

NDN codebases are currently supporting the following D2D communication mechanisms, summarized in Table 1.

Ad hoc Wi-Fi, as defined by the IEEE 802.11 specification, enables wireless stations to communicate directly with each other without using any infrastructure support, such as pre-deployed access points (APs). We have integrated ad hoc Wi-Fi support into the NDN Control Center [4] for macOS and Linux platforms, providing a one-click ability to setup the connectivity and maintaining proper entry in the Forwarding Information Base (FIB) of the *NFD* forwarding daemon ("/" towards the ad hoc Wi-Fi face). Unfortunately, Android platform has no support for ad hoc Wi-Fi mode on unrooted devices due to security and power concerns.

Wi-Fi Direct is a standard initially published around 2010 by the Wi-Fi Alliance to support peer-to-peer communication over Wi-Fi channel, and can run in parallel with any existing infrastructure-mode connectivity to AP. Wi-Fi Direct is natively available on Android and supported on Linux platforms for some Wi-Fi hardware. It allows adjacent Wi-Fi stations to discover each other and form peer-to-peer groups. Each group has an automatically selected *group*

Table 1: Available D2D networking technologies for NDN

	Ad Hoc Wi-Fi	Wi-Fi Direct	Bluetooth	IEEE 802.15.4	LoRa
Linux	✓	✓	✓		
macOS	✓		(✓)		
Android [5]		✓	(✓)		
RIOT-OS [7]				✓	
Electric Imp [6]					✓

owner that plays a role similar to an AP in infrastructure mode. Our Wi-Fi Direct implementation is integrated into NDN Android [5] and the NDN Control Center [4] (on Linux). It provides the ability for devices to advertise their presence and for users to select devices to connect. In addition, we have defined a protocol to automatically announce and discover NDN name prefixes available in the group [3], enabling NDN applications to run over single- or multi-hop ad hoc connectivity, and in hybrid ad hoc/infrastructure-mode environments. The macOS platform does not support Wi-Fi Direct. Instead it provides a similar but incompatible AirDrop technology with proprietary APIs. NDN codebases do not support AirDrop yet; we plan to add it in future versions of the NDN Control Center and NFD.

Bluetooth is a popular short-range wireless communication technology widely available on consumer electronic devices. We have integrated Bluetooth support into NFD/Linux codebase by leveraging the standard socket API provided the kernel modules of BlueZ, the official Linux Bluetooth protocol stack.Because Android and macOS provide their own, mutually incompatible Bluetooth APIs, we are currently working on providing platform-specific support for NDN.

IEEE 802.15.4 is a low-power wireless protocol for connecting constrained IoT devices in PANs. To communicate with each other, the devices tune into a pre-selected wireless channel (identified by the channel number) and PAN (identified by the PAN ID). Because of general unavailability of IEEE 802.15.4 network interfaces on common Android, Linux, and macOS devices, so far we have only implemented support for NDN connectivity over IEEE 802.15.4 as part of NDN-RIOT [7] which is based on RIOT-OS, a popular operating system for constrained IoT devices. The current implementation leverages the built-in APIs on RIOT-OS to send NDN packets to the broadcast address directly over the IEEE 802.15.4 interface.

LoRa or *LoRaWAN* is a Low Power Wide Area Network (LPWAN) specification introduced by the LoRa Alliance aimed to build a low-power, low-rate, and long-range communication network for Internet-of-Things [2]. A typical LoRa radio link can provide around 1 kbps of bandwidth over a half mile range with propagation delays up to several seconds. Similar to IEEE 802.15.4, LoRa network interfaces are available only on customized IoT platforms. We have added support for LoRa communication on Electric Imp platform as part of our NDN microforwarder [6].

3 LESSONS LEARNED

In developing NDN connectivity support for different platforms, we have learned a few lessons that may be helpful for future NDN development. We found out that developing a new D2D network protocol stack on common platforms is a daunting challenge, mainly due to three reasons. First, a standard cross-platform APIs to use D2D network interfaces is practically non-existent. Linux, macOS, and Android platforms all provide their own, mutually incompatible APIs to create and use ad hoc Wi-Fi, Bluetooth, and Wi-Fi Direct (AirDrop) interfaces.

Second, language restrictions imposed by the platforms (Objective-C and Swift on macOS, Java on Android, Squirrel on Electric Imp), together with widely varying documentation (e.g., macOS API documentation is not very newcomer-friendly), further complicate cross-platform implementations.

Third, access to low-level networking APIs is restricted on common platforms, perhaps due to (perceived/potential) security concerns. For example, macOS requires superuser privileges to access raw Ethernet frames; Linux requires at least appropriate capability configuration (`cap_net_raw=eip`); and Android does not provide such access unless one "roots" the device. On the other hand, the non-shared, single-app platforms, such as RIOT-OS and Electic Imp, usually give the application full access to the system, including direct low-level APIs.[1] This greatly simplifies the code development in support of new protocol stacks.

Our next step of exploring D2D communication with NDN includes investigating the interoperability of the D2D NDN connectivity on different platforms using the same technology (e.g., connecting NDN applications on Linux, macOS, and Android devices via Bluetooth) and the feasibility of bridging NDN communication over different ad hoc media (e.g., relaying sensor data collected by RIOT-OS devices to user's smartphone via a Linux box that support both Wifi-Direct and IEEE 802.15.4). Having those features available would help promote NDN applications at the edges, a fertile ground for emerging applications, such as augmented reality that we are pursuing under the newly launched NSF/Intel ICN-WEN program [12].

ACKNOWLEDGMENT

This work is partially supported by the National Science Foundation under awards CNS-1345318, CNS-1629922, and CNS-1719403. We would like to acknowledge Jeff Thompson, Allen Gong, Amar Chandole, and Da Teng for implementation of parts of NDN D2D support.

REFERENCES

[1] Emmanuel Baccelli, Christian Mehlis, Oliver Hahm, Thomas C. Schmidt, and Matthias WÃďhlisch. 2014. Information Centric Networking in the IoT: Experiments with NDN in the Wild. In *Proc. of ACM ICN Conference*.

[2] LoRa Alliance. 2017. Homepage. https://www.lora-alliance.org/. (2017).

[3] NDN Team. 2016. NDN Over WiFi Direct Protocol Specification. https://redmine.named-data.net/projects/nfd-android/wiki/NDN_Over_WiFi_Direct_Protocol_Specification. (2016).

[4] NDN Team. 2017. NDN Control Center. https://named-data.net/codebase/applications/ndn-control-center/. (2017).

[5] NDN Team. 2017. NDN on Android. https://github.com/named-data-mobile/NFD-android. (2017).

[6] NDN Team. 2017. NDN-Squirrel—An NDN Client Library for Squirrel. https://github.com/remap/ndn-squirrel. (2017).

[7] Wentao Shang, Alexander Afanasyev, and Lixia Zhang. 2016. The Design and Implementation of the NDN Protocol Stack for RIOT-OS. In *2016 IEEE Globecom Workshops (GC Wkshps)*. 1–6.

[8] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang. 2016. Named Data Networking of Things. In *Proc. of IoTDI*.

[9] Wentao Shang, Zhehao Wang, Alexander Afanasyev, Jeff Burke, and Lixia Zhang. 2017. Breaking out of the Cloud: Local Trust Management and Rendezvous in Named Data Networking of Things. In *Proc. of IoTDI*.

[10] Wentao Shang, Yingdi Yu, Ralph Droms, and Lixia Zhang. 2016. *Challenges in IoT networking via TCP/IP architecture*. Technical Report NDN-0038. NDN.

[11] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, et al. 2014. Named Data Networking. *ACM SIGCOMM Comp. Comm. Review* (July 2014).

[12] Lixia Zhang, Jeff Burke, Alex Afanasyev, Jason Cong, Mario Gerla, Songwu Lu, and Satyajayant (Jay) Misra. 2017. ICN-Enabled Secure Edge Networking with Augmented Reality (ICE-AR). http://ice-ar.named-data.net/. (2017).

[1] This is partly because of the overall simplicity of their system design.