The Design and Implementation of the NDN Protocol Stack for RIOT-OS.
Wentao Shang, Alexander Afanasyev, and Lixia Zhang.
NDN, Technical Report NDN-0043, Revision 1, July 16, 2016

Internet of (Named) Things: NDN Protocol Stack for RIOT-OS

Wentao Shang, Alex Afanasyev, Lixia Zhang

Presented by Alex Afanasyev

July 21, 2016
ICNRG Meeting, Berlin, Germany

ICN/NDN “Edge” for IoT

- Forget about hassle with managing IP addresses

- Bring IoT semantics to the network layer
  - Name the “things” and operations on “things”
    - “temperature in the room”, “humidity on the second floor”
    - “blood pressure”, “body temperature”
    - “max/min/avg pH of soil in specific point of US soil grid”
  - Focus on data associated with things, not devices
    - status information or actuation commands
  - Secure data directly

W. Shang et. al, "Named Data Networking of Things,” in proc. of IoTDI’2016
http://lasr.cs.ucla.edu/afanasyev/data/files/Shang/ndn-IOTDI-2016.pdf

IoT at the Edge

- Ultra low cost, longevity
  - constrained battery, low-power networking, limited memory, low CPU
  - SAMR21-PRO: 32-bit ARM, 48 MHz, 32KB RAM, 256KB flash
- RIOT-OS: multi-platform light-weight OS
  - https://www.riot-os.org/
  - C and C++ programming environment
  - micro-kernel for multi-threading, priority scheduling, interrupt handling, IPC
  - standard build tools (gcc, make)
  - simulator for testing on Linux PCs
  - gaining a lot of momentum

Other platforms
- Contiki
  - http://www.contiki-os.org/
- ARMmbed
  - https://www.mbed.com/
- tinyOS
  - http://tinyos.net/

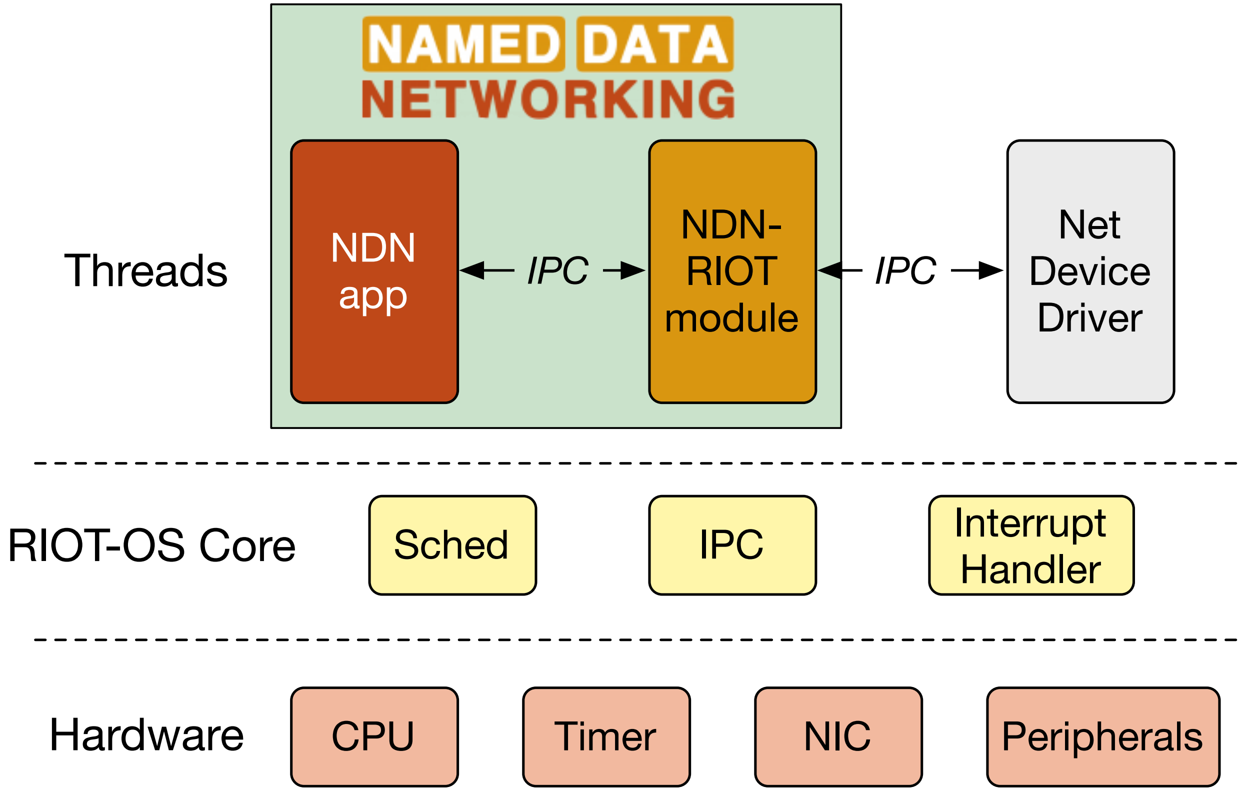
NDN-RIOT: NDN For RIOT-OS

- Optimized for IoT apps
- Memory efficient packet encoding & decoding
- Data-centric security support
- Basic stateful NDN packet forwarding
- Support for 802.15.4 and Ethernet
- Application API

Open source, contributions welcome
https://github.com/named-data-iot/ndn-riot

- A few basic examples
  - https://github.com/named-data-iot/ndn-riot-examples

NDN-RIOT Architecture



Memory-Optimized Packet Decoding

- Shared memory block structure to move packets
  - avoid memory copy in most cases
- On-demand packet field extraction
  - avoid memory for decoded meta data

Security Support

- ECDSA
  - micro-ecc library (https://github.com/kmackay/micro-ecc)
  - secp256r1 curve with 64-byte signatures
  - deterministic signing (RFC 6979) given lack of good entropy on many current devices
    - keys need to be generated outside the device
- no RSA
  - too much overhead and too expensive to produce signatures
- HMAC
  - RIOT-OS built-in APIs

Packet Forwarding

- PIT
  - exact match for interest
  - “any” prefix match for data (all interests that are prefix of the data)
- FIB
  - longest prefix match for interest names
  - static compile-time prefix registration
  - IPC-based run-time prefix registration (for local apps)
- CS
  - “any” match for interests (a data for which interest is a prefix)
  - compile-time adjustable size (~24KB default settings)
  - FIFO policy

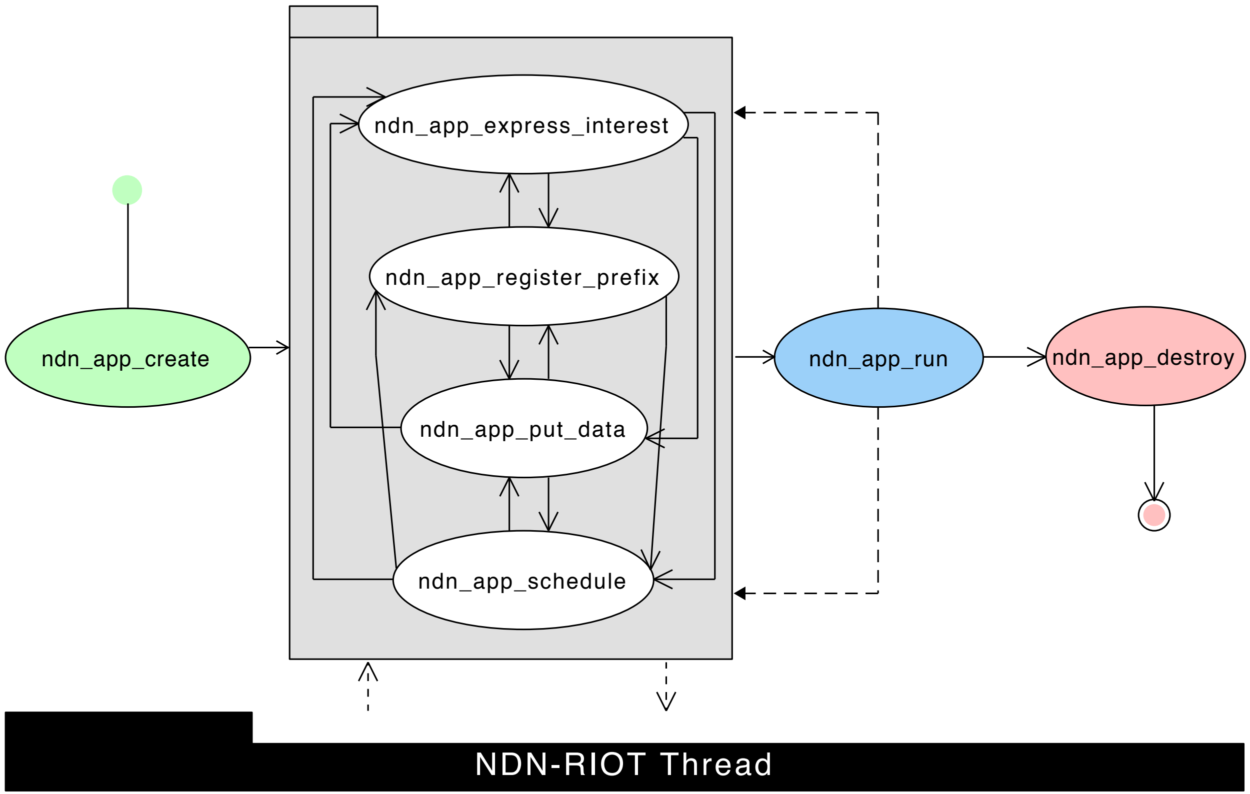
- Work in progress
  - Extendable / adaptive interest forwarding strategy
  - Support for basic Interest selectors
  - Extend dynamic prefix registration and maintenance

L2 Communication

- Run directly over layer 2 interfaces
  - IEEE 802.15.4
    - send packets to FF:FF (broadcast)
  - Ethernet (e.g., debugging on native plaform)
    - send packets to FF:FF:FF:FF:FF:FF:FF (broadcast)
- Simple hop-by-hop fragmentation if needed

```
 0                   1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1|0|M|   SEQ   |         Identification        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Application API



```c
static ndn_app_t* handle = NULL;

static int on_data(ndn_block_t* interest, ndn_block_t* data)
{
    ndn_block_t name;
    ndn_data_get_name(data, &name);
    ndn_name_print(&name);
    ndn_block_t content;
    ndn_data_get_content(data, &content);
    // do something with content...
    return NDN_APP_STOP;
}
```

```c
static int send_interest(void* context)
{
    const char* uri = (const char*)context;
    ndn_shared_block_t* sn = ndn_name_from_uri(uri, strlen(uri));
    ndn_shared_block_t* sin = ndn_name_append_uint16(&sn->block, 0);
    ndn_shared_block_release(sn);
    ndn_app_express_interest(handle, &sin->block, NULL, 1000,
                             on_data, on_timeout);
    ndn_shared_block_release(sin);
    return NDN_APP_CONTINUE;
}
```

```c
static void run_client(const char* uri)
{
    handle = ndn_app_create();
    ndn_app_schedule(handle, send_interest, (void*)uri, 1000000);
    ndn_app_run(handle);
    ndn_app_destroy(handle);
}
```

Memory Usage Numbers

Function Name	ARMv6-M	ARMv7-M
ndn_name_from_uri	420	408
ndn_name_append	232	232
ndn_name_get_size_from_block	124	124
ndn_name_get_component_from_block	152	164
ndn_interest_create	196	192
ndn_interest_get_name	92	94
ndn_data_create	668	692
ndn_data_get_name	98	100
ndn_data_get_content	160	168
ndn_data_verify_signature	450	502
ndn_app_run	612	596
ndn_app_schedule	96	88
ndn_app_express_interest	160	168
ndn_app_register_prefix	180	180
ndn_app_put_data	60	56

ISA	App	text	data	bss	Flash	RAM
ARMv6-M	Consumer	35,300	192	11,208	35,492	11,400
ARMv7-M	Consumer	33,900	192	11,208	34,092	11,400
ARMv6-M	Producer	35,212	192	11,208	35,404	11,400
ARMv7-M	Producer	33,800	192	11,208	33,992	11,400

Performance Numbers

Test Case	SAMR21-XPRO		IoTLab-M3	
	Time (μs)	Cycles	Time (μs)	Cycles
URI to Name	184	8,832	282	20,304
Get Name size	13	624	11	792
Get Name component	8	384	7	504
Append to Name	28	1,344	29	2,088
Create Interest	25	1,200	23	1,656
Get Interest Name	2	96	2	144
Create Data (HMAC)	1,806	86,688	1,333	95,976
Create Data (ECDSA)	451,215	21,658,320	269,314	19,390,608
Verify Data (ECDSA)	500,115	24,005,520	294,225	21,184,200
Get Data Name	3	144	2	144
Get Data Content	4	192	4	288

Data Size	Cached?	Fragmented?	RTT (ms)
100 bytes	No	No	280
	Remote	No	11
	Local	No	<1
196 bytes	No	Yes	286
	Remote	Yes	16
	Local	No	<1

Work in Progress

- Energy consumption evaluation / optimizations
- Advanced forwarding strategy supportData discovery
- Nearby data discovery
- Pub-sub API on top of Interest/Data exchange

Help welcome!

Use Cases and Other IoT-Related NDN Efforts

- **NDN-BMS**: encryption-based access control
  - Wentao Shang, Qiuhan Ding, Alessandro Marianantoni, Jeff Burke, Lixia Zhang. "Securing Building Management Systems Using Named Data Networking." In IEEE Network, Vol. 28, no. 3, May 2014.
- **NDN-ACE**: authorization framework for actuation apps
  - W. Shang, Y. Yu, T. Liang, B. Zhang, and L. Zhang, "NDN-ACE: Access Control for Constrained Environments over Named Data Networking," NDN Project, Tech. Rep. NDN-0036, Revision 1, December 2015.

- **NDN-IoT**: toolkit for NDN dev on Raspberry Pi
  - https://github.com/remap/ndn-pi
- **NDN on Arduino**: minimal app for Arduino
  - https://github.com/ndncomm/ndn-btle