Information Centric Networking in the IoT: Experiments with NDN in the Wild

Emmanuel Baccelli, Christian Mehlis, Oliver Hahm, Thomas Schmidt, Matthias Wählisch

Information Centric Networking in the IoT: Experiments with NDN in the Wild

E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, M. Wählisch

Information Centric Networking in the IoT:
Experiments with NDN
in the Wild

E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, M. Wählisch

Project-Team HiPERCOM

Abstract: This report explores the feasibility, advantages, and challenges of an ICN-based approach in the Internet of Things. We report on the first NDN experiments in a life-size IoT deployment, spread over tens of rooms on several floors of a building. Based on the insights gained with these experiments, the report analyses the shortcomings of CCN applied to IoT. Several inter-operable CCN enhancements are then proposed and evaluated. We significantly decreased control traffic (i.e., interest messages) and leverage data path and caching to match IoT requirements in terms of energy and bandwidth constraints. Our optimizations increase content availability in case of IoT nodes with intermittent activity. This report also provides the first experimental comparison of CCN with the common IoT standards 6LoWPAN/RPL/UDP.

Key-words: Network, Information-centric, ICN, CCN, NDN, internet, things, objects, IoT, routing, OS, energy, efficient, operating system, protocol, IPv6, wireless, radio, constrained, embedded

Réseaux Centrés sur les Contenus dans l'Internet des Objets: Experiences Grandeur Nature avec NDN

Résumé :   Ce rapport étudie la faisabilité, les avantages et les vérrous scientifiques afférrants à une approche réseau centré sur les contenus dans l'Internet des Objets. Cette étude est notamment basée sur des expériences avec NDN, sur un déploiement grandeur nature.

Mots-clés :   réseaux, internet, objets, routage, radio, sans-fil, capteurs, ICN, CCN, NDN, système d'exploitation, protocole, OS, efficace, énergie, contraint, embarqué

1   Introduction

The Internet is currently evolving in several directions. One path leads beyond end-to-end streams with Peer-to-Peer, CDNs and now ICN [1]. Endpoints in these information access models try to access named content, without direct mapping to a transport layer session at the (single) origin. The other evolves beyond traditional user terminal vs. router dichotomy: machine-to-machine (M2M) communications do not involve human source or destination, and interconnected machines include billions of cheap tiny communicating objects which play both the roles of host and router in spontaneous (wireless) networks, i.e., the Internet of Things [2]. In this dual context, this paper explores the feasibility, advantages and challenges of an ICN-based approach in the Internet of Things.

1.1   The Next Billion of Connected Machines

The next billions of interconnected machines are expected to consist in a variety of heterogeneous devices, ranging from wireless sensors to actuators, wearables, Radio-Frequency IDentification (RFID) tags, smart home appliances and many other types of machines that were typically not inter-networked so far. Connecting these devices to the global realm has been coined the Internet of Things (IoT). It is expected to profoundly transform our environment.

   Most IoT devices will be very limited in terms of memory, CPU, or power capacities (from small batteries). The term *constrained devices* [3] was recently introduced to define a category of connected devices that are resource-challenged compared to PCs, smartphones or laptops. Constraints include (i) orders of magnitude less power consumption measured in mWatt instead of Watt, (ii) orders of magnitude less computational power measured in MegaFLOPS instead of GigaFLOPS, and (iii) orders of magnitude less memory measured in Kilobytes instead of Giga-bytes. For cost reasons, and due to the specific nature of the envisioned (massive) deployments of IoT devices, such constraints are expected to remain the norm in this domain, in the foreseeable future [?].

   The sheer numbers and a lack of user interfaces make interconnecting IoT devices a challenge. Different approaches have been designed which leverage both traditional, infrastructure-based network paradigms, and spontaneous wireless network paradigms [4]. They allow for device autoconfiguration and dynamic self-organization to relay data towards destination – even without the help of infrastructure and pre-provisioned access points. Current approaches fall into two categories: silo approaches such as Zigbee [5], and approaches based on open standards, protocol stacks, such as IPv6 with 6LoWPAN [6] and RPL [7]. In the long run, one can expect that for the same reasons that led TCP/IP to prevail, an approach based on open standards and on a layered protocol stack will establish in the IoT. In the following, we will consider 6LoWPAN/IPv6/RPL as the reference networking solution for constrained devices in the IoT, with which ICN should measure up.

1.2   Why ICN for the Internet of Things?

Data in information-centric networking is delocalized and need not be retrieved via an end-to-end transport stream. Instead, hop-wise replication and in-network caching facilitate information dissemination in the IoT, and relax the demand for continued connectivity. Such perspectives, based on ICN, were recently mentioned as a potential alternative networking solution for the IoT [8].

   More specifically, common communication patterns of the IoT such as content retrieval 'upon request' and 'scheduled' content updates are easily accommodated by ICN and may noticeably

benefit from cache-assisted, hop-by-hop replication. The prevalent task of data fusion in the IoT may be implemented by augmented replication logic in a lightweight fashion. The combination of these mechanisms may save energy and radio resources, increase availability, and well reduce complexity. Most strikingly, ICN does reduce network layers and – in an optimized version – may subsume network, transport, and elementary application logic. Thus an ICN approach in the IoT might (i) offer opportunities to efficiently factorize functionalities e.g., caching and buffering for error control (ii) drastically reduce the complexity of autoconfiguration mechanisms compared to an approach based on a layered protocol stack, and (iii) achieve a smaller memory footprint compared to 6LoWPAN/IPv6/RPL.

However, a number of challenges should also be noted. Often, sensor data require freshness that conflicts with caching. Furthermore, there is also the demand for unscheduled traffic in the IoT e.g., the control of actuators, which is much easier to achieve in an end-to-end access model. Finally, in many ICN approaches, routing and forwarding significantly increases the burden over IP. In effect, state and cached content may blow up memory requirements of constrained nodes.

At the conceptual level, it remains fairly open whether benefits outweigh the shortcomings of ICN in the IoT, or not. It is the objective of the present paper to explore the basic feasibility and tradeoffs in an experimentally driven approach.

1.3   Related Work

While several ICN approaches have been developed, including NDN [9], PSIRP [10], Netinf [11], DONA [12], a number of key aspects remain challenges for ICN [?]. One example of such challenge is the design of routing schemes enabling automatic, efficient, and scalable forwarding information configuration on each ICN device. Related work proposed routing approaches based on proactive, link-state mechanisms [13], [14]. However, such approaches may not be directly applicable in the IoT, where constrained devices impose different requirements in terms of memory and power capacities. For instance, requirements for home, industrial and building automation [15] led to the design of the RPL [7] routing protocol, which can be more energy and memory efficient than standard link-state approaches. It does not require periodic flooding and allows partial topology knowledge.

Recent work has thus started to study ICN paradigms in IoT scenarios or similar contexts (e.g. mobile ad hoc networks). In [16], authors reports on early efforts to provide constrained devices with a CCN communication layer in practice. This implementation is however not interoperable with the full-blown, reference CCN implementation. This initial implementation was used in [17] to showcase a health monitoring application prototype in the context of a small home network. Several architecture design proposals emerged recently for ICN in the Internet of Things, such as [18] which proposes an overlay ICN architecture designed over the M2M ETSI standard, or [19] which identifies high-level requirements of ICN for IoT and proposes a network architecture for IoT based on ICN. Other efforts have proposed enhancements to tackle various issues with ICN in wireless scenarios. For instance, [20] focuses on MANETs scenarios and mobile nodes using ICN and proposes a mechanism reducing the overhead of NDN packet forwarding. On the other hand, [21] focuses on sensor networks and data collection from a data sink, and proposes in this context an NDN extension for directed diffusion with new packet types and neighbor distinction. This implementation is however not interoperable with the reference CCN implementation. In [22] authors propose a push mechanism for CCN, targeting sensor networks. In [23] a gossip mechanism for CCN is introduced, targeting wireless ad hoc networks. Another category of efforts have focused on tackling security and naming issues with ICN in the IoT, such as [24] which studies such issues with CCN in the context of lighting systems and building automation.

However, the above prior work only studied ICN approaches via theoretical analysis and simulations. In [17] and [16], preliminary tests are reported on small, toy networks. But to the best of our knowledge, there are no reports to date on larger scale deployments on IoT hardware, in environments matching requirements described by the industry, e.g., in [15]. Furthermore, prior work in this domain has either (i) focused on MANET, where machines are not constrained devices, or (ii) focused on wireless sensor networks and sink-centric data traffic (i.e., sensor-to-sink or sink-to-sensor) which is not representative of the whole IoT, where other types of devices participate, and other types of data traffic are significant, such as sensor-to-sensor traffic which is substantial in building automation scenarios (e.g., for lighting systems).

1.4   Contributions of this Paper

In this paper, we report on the first CCN experiments in a life-size IoT deployment, spread over tens of offices on several floors of a building, matching characteristics and requirements from building automation as specified in [15]. Based on the insights gained with these experiments, the paper analyses the shortcomings of NDN applied to IoT. Several interoperable CCN enhancements are then proposed and evaluated, which decrease interest traffic and focus data path and caching to match IoT requirements in terms of energy and bandwidth constraints, and increase content availability in case of IoT nodes with intermittent activity. This paper also provide the first experimental comparison of CCN with the alternative dominant approach in IoT based on 6LoWPAN/RPL/UDP. In addition to our real-world experiments, we discuss ICN in the context of IoT, based on an extensive literature survey.

The remainder of this paper is organized as follows. First, in § 2 we will compare IoT requirements with basic ICN characteristics to identify mismatches and challenges one faces with ICN in the Internet of Things. Then, in § 3 we will describe our ICN implementation for the IoT and our deployment setup in a building automation context. Based on insights gained from our experiments with the CCN implementation in this deployment, we will propose and evaluate in § 4 several interoperable enhancements for CCN operation in the Internet of Things. We will then present lessons learned in § 5. Finally, we will conclude and discuss future steps in § 6.

2   A Priori Challenge of ICN in IoT: Limited Memory

Limited memory resources are fundamental in IoT scenarios. Before an ICN solution can be deployed and experimented with, it needs to be aligned with these constraints. In this section, we discuss memory requirements introduced by ICN and how we overcome this basic challenge. We separately discuss aspects concerning caching, protocol stack architecture, and routing schemes. For challenges we derived based on our experiments, we refer to Section 5.

2.1   Implications on Caching Capabilities

One of the fundamental aspects of ICN is in-network caching, which requires memory dedicated to content cache on nodes in the network. On constrained devices, available RAM is very limited and usually in the order of 10 kBytes [3]. This memory is shared by all processes running on the device, including the operating system, the full network stack, the application(s). Considering typical sizes of these software components in the IoT, the remaining cache size for content on constrained devices is at most in the order of 1 kByte. This is extremely small compared to cache sizes expected on types of devices initially targeted by ICN [25, 26]. As readings of sensor values are ephemeral information by nature – sensor data are continuously replaced by new data

– one might argue to disable caching altogether. However, as we will show below, caching is not only doable, but also beneficial in the IoT (even with such limited resources).

First, a significant part of the data is expected to consist in small size content. The size of a common implementation of temperature values is 12 bytes, which allows to store $\approx$85 sensor values in a single cache. For medium-sized content (i.e., of size in the order of n kBytes, where n is the number of nodes in the network), distributed caching strategies could coordinate multiple devices to leverage in-network caching of all chunks. Typical medium-size content examples include accumulated, periodically-generated data, or software update binaries.

Second, beyond simple sensor scenarios with a single sink, the IoT envisions multiple consumers for the same content. For example, a temperature sensor asynchronously accessed by the air-conditioning system, the automated blinds, and windows of a room, each of which may react independently upon temperature evolution. For more powerful devices crowd computing [27] is an interesting application field. Similarily, caching ephemeral content within the network may significantly increase content availability because (i) nodes can then sleep as often as possible to save energy, and (ii) lossy multi-hop wireless paths towards content producers are shortened. We will study the effect of caching in Section 4.

2.2 Implications on Overlay Applicability

Deploying only the IP stack on constrained devices is already a challenge in terms of RAM and ROM. ICN approaches that work on top of IP might be impossible due to the additive memory requirements of both the ICN stack and the IP stack. Consequently, ICN implementations should work directly on top of the link layer. Note that for heterogeneous deployment border gateways can bridge between IP and ICN. For the experiments reported in this paper, we have thus used an ICN approach running directly above the MAC layer (see Sections 3 and 4).

2.3 Implications on Routing Approaches

Reduced memory of constrained devices also limits applicability of ICN routing approaches. Current proposals usually route either directly on names or indirectly via name resolution. Based on our previous observations, name resolution on top of IP is not viable. However, even some pure name-based routing schemes, such as [13] and [14] rely on an ICN overlay requiring an IP network, or use proactive link state algorithms. Link state routing results in both (i) a significant amount of control traffic, whether or not there is data traffic to carry in the network, and (ii) a significant amount of memory, typically in $O(n)$, where n is the number of nodes in the network. These characteristics do not match the memory and energy resources of constrained devices.

Routing protocols running on IoT devices should aim for $O(1)$ routing state and minimal control traffic – ideally none, especially when there is no data traffic to carry [28]. In Section 4, we introduce ICN routing with these properties.

3 Steps to Enable ICN in the IoT

In order to gain a full understanding of how ICN operates in the Internet of Things, it is inevitable to conduct experiments in real-world deployments or testbeds that reflect properties of such deployments. Testbeds help to avoid topologies and densities that are too artificial, too regular, or too isolated compared with the real word. They naturally include external interferences resulting from other radio networks, electrical devices, or simple human activity. The first step towards such experiments is implementing ICN code that runs on IoT hardware.
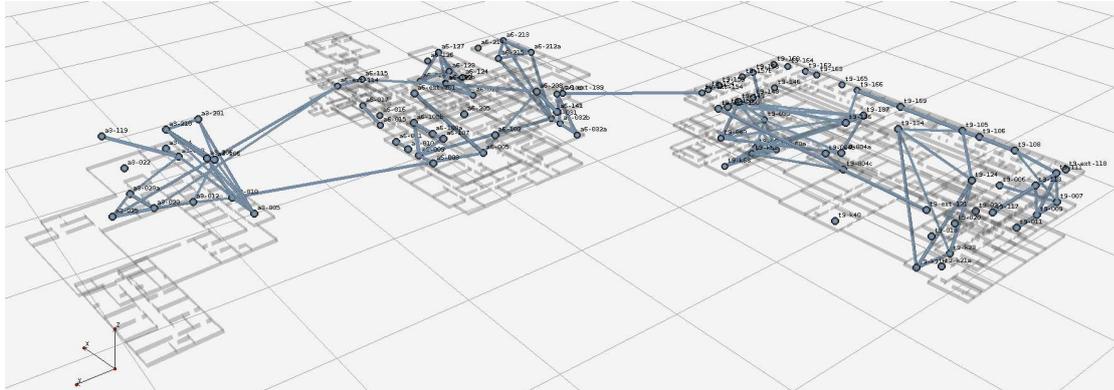
Figure 1: 3D visualization of the topology of the deployment, consisting in 60 nodes that interconnect via wireless communications (sub-GHz) and that are physically distributed in multiple rooms, multiple floors, and multiple buildings.

3.1 Porting CCN-Lite to RIOT

We have ported CCN-Lite [29], a bare-bone Linux open source implementation of NDN, to RIOT [30], an operating system for constrained devices. Among ICN approaches, we have chosen NDN because it can easily operate directly above the link layer – a requirement we identified in Section 2. We chose to base ourselves on CCN-Lite because this implementation is compliant with the reference NDN implementation (CCNx) while being very compact: less than 1,000 lines of C code and low memory footprint. And we chose RIOT as operating system to run on constrained devices because it is open source and fits IoT devices memory requirements, while allowing plain C code with all the standard headers, based on a (multi-)threading model comparable to POSIX. These characteristics guaranteed that porting Linux code to RIOT is straightforward, and a fair comparison with the non-IoT world. We also leveraged RIOT support for popular debugging tools such as Valgrind, Wireshark, gdb, and nativenet. Our implementation is open source and available online in GitHub [31].

Table 1 compares the ROM and RAM sizes of the binaries compiled for NDN network stacks and for 6LoWPAN/RPL network stacks, built upon state-of-the-art IoT operating systems (RIOT and Contiki), for state-of-the-art IoT hardware (Redbee Econotag board and MSB-A2 board). We observe that an ICN approach can significantly outperform common IoT protocols in terms of ROM size (down to 60% less) and RAM size (down to 80% less).

3.2 Configuring NDN Deployment

In order to obtain a fully functional NDN network stack for the IoT, a FIB autoconfiguration mechanism is needed: in IoT scenarios, even less than in other scenarios, one cannot expect humans in the loop, so manual configuration is not part of the deployment. In particular, predefined location-based naming and simple routing schemes based on the structure of such names may thus not be possible in general. Furthermore, as mentioned in Section 2, existing ICN routing approaches are not appropriate for constrained devices in the IoT: alternative routing mechanisms must be used in this context, which require drastically less state.

In the context of ICN, the naming scheme is crucial. NDN uses a hierarchical name space, which allows for aggregation in routing. The amount of content items that can be expressed depends on the character set and name length. MTUs of common IoT link layer technologies

(a) RIOT on MSBA2

Module	ROM	RAM
RPL + 6LoWPAN	53412 bytes	27739 bytes
CCN-Lite	16628 bytes	5112 bytes

(b) Contiki on Redbee-Econotag

Module	ROM	RAM
RPL + 6LoWPAN	52131 bytes	21057 bytes
CCNx	13005 bytes	5769 bytes

Table 1: Comparing memory resources for common IoT operating systems and hardware.

range between ≈30 bytes and ≈100 bytes. To the best of our knowledge fragmentation within ICN is not addressed, hence naming and chunk size need to be aligned with the packet size to prevent fragmentation (not supported by the link layer).

4 NDN Experiments and Optimizations for IoT Deployment

In the following, we will describe and evaluate several routing alternatives, as well as other aspects of NDN in the wild, such as the effect of caching in IoT.

4.1 Large-scale Deployment Setup

Typical IoT application scenarios, include building and home automation [15,32], smart metering (e.g., [33]), or environment monitoring (e.g., [34]). These scenarios usually require a multi-hop wireless network. For the NDN experiments, we deployed our ICN IoT implementation on the campus testbed of *Freie Universität Berlin*, consisting in 60 nodes distributed in various rooms, on several floors, and in several buildings, as shown in Figure 1. This deployment matches the typical device density (several meters between nodes), distribution (one node per room), and environment (e.g., co-located wireless networks) described in [15] for building automation, e.g., HVAC devices, lighting devices, or fire-detection devices. Each node is equipped with a CC1100 radio chip operating at 868MHz, and sensors that can measure various parameters including room temperature, humidity etc. For more details we refer to [35]. Most of the nodes are deployed inside rooms, while a few nodes are deployed outdoor to better interconnect nodes in different buildings. Nodes interconnect via their wireless interface, which offers a maximum link layer frame size of 64 Bytes.

In order to monitor closely energy consumption, verify individual node behavior, and manage experiments on this deployment (e.g., flash nodes, gather results) each node is furthermore connected to its own docking station. Docking stations are interconnected via an Ethernet backbone. However, these docking stations are used only to monitor and manage the nodes. Nodes operate autonomously, i.e., each node can only use its own CPU, its own memory, and its own wireless interface to communicate with other nodes.

Basic Configuration of Experiments The following experiments use 400 ms interest timeout (stop-and-go, giving up after 5 tries), and 900 ms nonce timeouts. The content is named in a

hierarchical fashion typical for NDN, without any encryption. Considering the maximum link layer frame size of 64 bytes in our deployment, we decide for a medium sized name length of 12 bytes including the chunk identifier (the exact names of the content chunks are */riot/text/a*, */riot/text/b* etc.). Note that with these names, the size of headers and names fit in a single link layer frame, both with CCN (16+12 = 28 bytes) and with 6LoWPAN/RPL/UDP (15+12 = 27 bytes), and still allow to carry realistic application data. Also note that the sizes of minimal CCN header (16 bytes, eliding optional fields) and of 6LoWPAN/RPL/UDP headers (15 bytes) are similar, and thus represent not a decisive factor in the differences observed in the following experiments. The length of content names is however a factor, as discussed in Section 5.1.

In the experiments, we consider a single content producer and one or multiple consumers. Due to the volatile nature of the wireless medium [36], the resulting link layer topologies based on our 60 node network might change on a per-transmission basis (cf., Figure 2). Note that IoT scenarios in home and building automation networks are typically multi-hop, but less than 5 hops in diameter [?]. Consequently, in our experiments, we placed content producer and consumers at least 2 hops apart.

To analyze the effects of NDN for typical radio packets payload in the IoT, we align the chunk size such that each chunk can be transmitted without fragmentation. In our case, MTU is 64 bytes, chunks are set to be 58 bytes long, of which 30 bytes of content. Since typical sensor content production is of the order of 200 bytes per minute [15], we set the basic configuration for consumers to periodically fetch 10 such chunks. However, other popular IoT radio technologies provide MTUs that are twice bigger (e.g. IEEE 802.15.4), or half smaller (e.g. Bluetooth LE). So we also check cases with 5 and 20 chunks per content item.

4.2 Vanilla Interest Flooding (VIF)

The simplest routing approach that requires minimal states is *interest flooding*, whereby each node in the network repeats an interest, upon first reception. In the following, we will call this simple mechanism Vanilla Interest Flooding (VIF). Using VIF, a consumer with an empty FIB can nevertheless disseminate its interest in content, and the flooded interest will reach the producer which can then send the content on the reverse path. VIF fits the constraints of IoT devices because (i) it does not rely on any additional control traffic to maintain the FIB, (ii) it requires minimal state, i.e., only temporary pending interests on the reverse path of content that is sought after.

Figure 3(a) shows the results of an experiment using NDN with VIF for a single consumer scenario. In this experiment, the consumer periodically accesses content of size 5, 10, or 20 chunks of data, all of which were produced by another constrained node in the network shown in Figure 2(a).

While the experiment is successful in that NDN was demonstrated to operate on IoT hardware (meeting memory requirements), and the consumer could fetch the content, Figure 3(a) shows that, compared to its size, many packets were transmitted to fetch the content. This is due to the fact that each chunk triggers an interest, which requires network-wide flooding. In general, in a network of n nodes, and for k chunks of content, the number of transmissions for a single content item is $k \cdot ((n-1) + \sqrt{n})$, assuming the average path length approximation $\sqrt{n}$. We observe that while VIF is simple and works in the scenario we tested, it does not scale well in terms of number radio transmissions when the network or the content grows in size. Radio transmission and reception are however very costly in terms of energy for battery-powered IoT devices. In the following, we have thus designed and tested enhancements reducing the number of radio transmissions and receptions in IoT environment.

4.3 Reactive Optimistic Name-based Routing (RONR)

In order to reduce the number of radio transmissions compared to basic interest flooding, we introduce Reactive Optimistic Name-based Routing (RONR), which automatically configures a temporary FIB entry on the reverse path taken by the first content chunk. That way, in case the FIB is empty (e.g., after booting) or if no FIB entry matches the name/prefix of the content in which the consumer is interested, only a single initial interest flooding is needed, while subsequent interests for chunks of that content can be unicast using the FIB entries thus auto-configured along the path. For example, in our experiments, after flooding an interest for chunk */riot/text/a*, nodes on the reverse path of that chunk store a temporary FIB entry for */riot/text/**, thus subsequent interests for chunks */riot/text/b*, */riot/text/c* can be unicast using the established path, instead of flooded.

RONR is optimistic because it first assumes that the whole content is stored on a single node (a cached replica or the original producer), which may not be the case in general. However, this assumption is reasonable in the IoT because typical content size is in the order of a few hundred bytes [15]. Furthermore, FIB entries timeout ensure that if the configured FIB entries do not lead to a node with the full content, the consumer will eventually revert to interest flooding, through which it can discover another node with the rest of the content, install new temporary FIB entries etc. This timeout strategy is common for reactive routing in multi-hop wireless scenarios [?].

In Figure 3(b), we show the results of an experiment using NDN with RONR, for the exact same topology and scenario as for Figure 3(a). We observe that the number of radio transmissions decrease about 50% compared to NDN with VIF. In particular the number of broadcast transmissions is drastically reduced because, with RONR, only the first interest packet of a content item is flooded, while subsequent interests are unicast, using temporary FIB entries established by RONR. A quick back-of-the-envelope analysis shows that in a network of n nodes, and for k chunks of content, the number of transmissions is $(n-1) + 2(k - \frac{1}{2})\sqrt{n}$, assuming again the average path length approximation $\sqrt{n}$. Therefore, RONR scales much better than VIF when network size or content size grows. RONR thus better fits IoT devices energy requirements compared to VIF, while still fitting other requirements of constrained devices by (i) not relying on any control traffic, and (ii) requiring minimal state, i.e., only temporary FIB entries on the reverse path of content that is sought after (not counting PIT state, of course).

An enhancement of RONR could be even more optimistic and tentatively aggregate prefixes in the following manner. If a FIB entry is pointing to an interface for content with prefix */riot/text/** and an interest for */riot/temp/c* is answered by a chunk of content through the same interface (after the initial interest flooding phase), the enhancement would optimistically aggregate the prefixes and create a FIB entry for */riot/** pointing to this interface. In the best case, this will indeed lead to all the requested content matching this prefix, via unicast only. In the worst case, after unicast transmission and time-out, the consumer will eventually revert to interest flooding, through which it can discover another node with the rest of the content, install new temporary FIB entries etc. For this paper, however, we have only tested RONR without this enhancement, and leave its analysis for future work.

4.4 Multiple Consumers & Impact of Caching

In this section, we evaluate experimentally the impact of ICN caching. The same content (20 chunks) is accessed alternatively by one, two, or three consumers that are topologically close to one another (pairwise, maximum hop distance is 1). In order to accommodate for more consumers while keeping them apart from the producer with at least 2 hops, a larger topology

shown in Figure 2(b) was used for the following experiments. To reduce signaling overhead, we use RONR as routing scheme for NDN interest packets.

In Figure 4(a) we show the results of our experiment with a disabled content cache. We observe that, as expected, the number of radio transmissions scales almost linearly with the number of consumers. In a network of n nodes, and for k chunks of content and m consumers within radio reach, the number of transmissions is $m \cdot ((n-1) + 2(k - \frac{1}{2})\sqrt{n})$, still assuming the average path length approximation $\sqrt{n}$.

Next, we enable cache capacity of 20 chunks on all nodes, which corresponds to RAM usage of 2 kBytes (2 % of 96 kbytes overall RAM). Figure 4(b) shows the results we obtained for the exact same topology and scenario as for Figure 4(a), except the caching. We observe that the number of radio transmissions needed to retrieve the content is drastically reduced, by up to 50% in this scenario. In detail, the number of broadcast transmissions is almost similar, while the number of unicast packets decreases substantially. This is consistent with the facts that the initial interest flooding (broadcasted) is not modified, while cached content chunk shorten unicast paths, thus reducing the number of unicast transmissions. In the best case, if the initial flood for subsequent consumers can be reduced to a local broadcast because only neighbors with cached content receive the interest, the number of transmissions becomes $2(k - \frac{1}{2})(\sqrt{n} + n - 1) + n + m - 2$.

4.5 Comparison with 6LoWPAN/RPL/UDP

In this section, we compare NDN with 6LoWPAN/RPL/UDP, a common protocol suite for the current IoT. For fair comparison, we use the following setup. On the ICN side, we deploy RONR with a cache size of 2 kBytes, as this leads to the best performance results in our previous analysis. On the RPL side, we first let the network converge until the RPL root and the routing entries are installed in nodes, before we start the experiment (i.e., we factor out the control traffic transmissions necessary to bootstrap the network).

In Figure 5, we show the results we obtained for the exact same topology and scenario as for Figure 4(b), except the network stack used was 6LoWPAN/RPL/UDP with default settings instead of NDN. We observe that the 6LoWPAN/RPL/UDP network stack yields much more transmissions compared to NDN (cf., Figure 4(b)), approximately three times more. This is due to two main factors. On one hand, the amount of control traffic generated by the *proactive* 6LoWPAN/RPL/UDP network stack is a big penalty compared to the *reactive* CCN approach we tested. On the other hand, compared to our CCN approach, the unicast paths created by the 6LoWPAN/RPL/UDP network stack do not benefit from *caching* and are thus always maximum length, which can in some cases be even longer than the shortest topological paths, as shown in [?]. Note that we have not used RPL extensions such as [?], which could reduce the length of unicast paths. Furthermore, as discussed in Section 4.1, we observed that the naming scheme and the header sizes were not a decisive factor explaining the performance gap between the CCN stack and the 6LoWPAN/RPL/UDP in the experiments we conducted. All in all, we can conclude that NDN may be a potential alternative to 6LoWPAN/RPL/UDP, which should be studied more in the context of IoT in future work.

5 A Posteriori Challenges: What are the Lessons Learned

In this section, we gathered further considerations and observations concerning ICN in the Internet of Things, based on our practical experience with NDN implementation and deployment. In the following, we distinguish energy consumption aspects, wireless connectivity aspects, and communication model aspects.

5.1 Energy Consumption

Energy consumption is mainly impacted by network transmissions, which are affected by content naming, content caching, network flooding, and local wireless broadcast.

5.1.1 Impact of Names

Routing information about names and prefixes should dynamically be auto-configured in IoT devices. The resulting overhead not only depends on the routing protocol but also on the size of names to be processed in ICN packets. In our experiments, we deployed VIF, a very basic approach based on *flooding*, whereby each node in the network repeats (on all interfaces) each flooded packet upon first reception (on any interface).

Flooding is used (i) to disseminate an interest message when no forwarding information is available, or (ii) to disseminate names and topology information, e.g., with link state routing approaches [13], [14]. However, flooding is costly in terms of energy since each flood requires $O(n)$ packet transmissions and $O(nm)$ packet receptions, where n is the number of nodes in the network and m is the average node degree. Each packet received will not only be costly in terms of pure packet reception but will also trigger its processing, which includes CPU-expensive string comparisons with variable lengths, trying to match received names with names stored locally. Furthermore, recent work [37] identifies ICN packet processing as a CPU bottleneck, serious enough to provide DOS attack opportunities. This processing is even more costly on constrained devices since their CPU typically does not benefit from advanced functionalities such as prefetching or super scalar instruction set, and thus needs one cycle per byte compared. Table 2 shows a benchmark for the number of required CPU cycles per CCNlite operation for our implementation in RIOT. The top 3 functions, which represent 85% of the CPU cycles, involve string comparison and name matching.

These observations thus call for (i) the least possible recourse to flooding and (ii) the shortest possible names. Note that short names also ensure that packet fragmentation is avoided at the link layer: long names that do not fit in the MTU of the link layer split interest packets in several transmissions which is inefficient in terms of energy. Shorter names should however not sacrifice prefix aggregability, so that scalability remains in terms of number of nodes in the network vs. routing state. In our experiments, we have demonstrated the use of small, hierarchical names of 12 bytes and the minimal CCN header all of which carried by link layer packets with very constrained MTU (64 bytes). In practice, it still allowed about 30 bytes of content payload, which is appropriate for IoT scenarios, where content generated by sensors is in the order of a few hundred bytes per minute [15]. Even with slightly longer names implementing a typically deeper hierarchy as described in [15], there is still enough space for payload. For instance, with names such as */zone1/room2/dev7/temp/a*, there is about 20 bytes for content payload with the link layer we used in our experiments.

Note however that human-readable names may not be required or useful in a context of machine-to-machine communication, whereby no humans are in the loop. More compact naming (e.g., a binary representation, or a more compact ASCII representation) may thus be applicable and would leave more space for content in packets constrained by small MTUs. Furthermore, the computations incurred by cryptographically-generated names (or parts of names) are expected to yield both substantial energy consumption penalty for constrained node in the IoT. In this paper, however, we do not consider security-related aspects for names – which are considerations that are orthogonal to the aspects studied here. Nevertheless, security mechanisms typically yield substantially longer headers. We can therefore also conclude that a standard CCN header compression scheme would be useful in the IoT.

# of instructions	Function
14,002,814	memcmp_ssse3
7,525,050	ccnl_nonce_find_or_append
4,062,659	ccnl_i_prefixof_c
1,462,304	dehead
956,238	ccnl_core_RX_i_or_c
895,590	ccnl_extract_prefix_nonce_ppkd
845,042	memcpy_ssse3

Table 2: CPU cycles per CCN function.

5.1.2 Impact of Caching

The impact of in-network caching on energy aspects with ICN approaches has been studied by recent work such as [38], which indicates that energy consumption incurred by caching reduces energy efficiency. But on the other hand, studies such as [39] show that CCN can be more energy efficient than other content delivery approaches such as CDN and P2P by leveraging the most energy efficient devices in the network. It remains to be seen at large scale on the Internet which ICN approaches introduce low overhead in terms of energy consumption. In the IoT, to the best of our knowledge, there are no studies yet that focused on energy aspects of ICN due to the use of caching.

In Section 4, we demonstrated experimentally that savings in terms of energy consumption are possible thanks to (even small) in-network caching since (i) on-path or near-path caching can decrease the number of intermediate energy-challenged devices on the path to reach content in some scenarios, and (ii) content producers such as sensors could sleep more while their content could still be available in other caches in the network.

5.1.3 Impact of Local Wireless Broadcast

In case of multiple PIT hits, the NDN stack could use a single multicast transmission if all matching neighbors are reachable through the same wireless interface – which is the case in most IoT scenarios where nodes only have a single interface (omnidirectional radio). We have thus enhanced our NDN implementation with such a link-local multicast awareness mechanism called Content Forwarding Aggregation (CFA). In scenarios where multiple geographically close consumers are interested in the same content at approximately the same time, CFA leads to substantial gains in terms of number of radio transmissions necessary to deliver the content. With CFA, a content chunk may be forwarded as a single multicast to multiple nodes that have expressed interest in this content. Using link-local multicast, CFA reaches nodes within the same radio range without implementing explicitly location-awareness mechanisms.

Another opportunity to leverage the multicast nature of IoT devices' wireless interface concerns caching. Very often, a node will overhear unsolicited chunks of content that are being transmitted in its radio vicinity. In such case, instead of discarding this content, the node could cache this unsolicited chunk in its content store, if there is space left, with a lower priority than solicited content. We have thus enhanced our NDN implementation with such a mechanism, called Opportunistic Near-Path Caching (ONPC), which increases availability of the content and further reduces the number of radio transmissions in case of several consumers of the same content. However, due to lack of space, we do not show experimental results with CFA or ONPC in this paper.

5.2 Wireless Connectivity

Although ICN is applicable in wireless networks, several issues arise when applied to the wireless regime at work in IoT. In the following, we distinguish aspects concerning frame size, fragmentation, and bidirectional links.

5.2.1 Frame Size and Packet Fragmentation

Several link layer technologies are currently used in the IoT, and it is likely that multiple technologies will be used in the future, too. Currently, the dominant IoT link layer in the field of building automation and industrial automation is IEEE 802.15.4. The maximum frame size is very small (127 bytes or less). Other popular wireless link layers provide an even smaller maximum frame size, such as Bluetooth Low Energy [40] which typically allows a payload of $\approx$30 bytes. These frame sizes are more than ten to a hundred times smaller compared to traditional Ethernet or WiFi frames. Consequently, fragmentation and reassembly mechanisms are necessary. While Bluetooth provides its own, IEEE 802.15.4 does not. To bridge this gap, 6LoWPAN introduced (i) a standard header compression scheme, and a (ii) standard fragmentation and reassembly mechanism for IPv6 operation in the IoT, both on top of IEEE 802.15.4 link layer. It is worth noting that ICN cannot benefit from the same mechanisms for fragmentation and compression. Overlay architectures conflict with the memory constraints in the IoT (cf., Section 2.2) as well as with packet sizes of common IoT link layers.

In our real-world deployment, we demonstrated that NDN can be implemented directly on top of an IoT link layer, without compression/fragmentation mechanisms (see Sections 3 and 4). Omitting these optimizations is suitable for basic scenarios in which small enough names and small enough chunks can be used in the first place. Our results give confidence that we can already start with ICN in the IoT. However, in the future, ICN approaches for the IoT need an equivalent of what 6LoWPAN is providing for IPv6. For illustration, NDN will typically use up to 40 bytes for the header and data encoding, which is negligible in the common Internet ($\approx$2% of the capacity of standard 1500 bytes MTU) but occupies $\approx$28% of the capacity of standard 802.15.4 frames. Neither can it be expected that all chunk sizes on all ICN networks will be defined by IEEE 802.15.4 frame size (which would be inefficient), nor can it be expected that names indicated in interest packets will always be short enough to fit in a single 802.15.4 frame of 127 bytes, for example. Note that fragmentation approaches need to take into account that altered chunks can break security and naming schemes.

5.2.2 Bidirectional links

Many ICN approaches assume bidirectional links. This is not true in general in spontaneous wireless networks [4], and thus this assumption does not hold in the IoT. In such context, a high proportion of links are asymmetric, e.g., 10% loss rate from A to B and 80% loss rate from B to A. In reality, a substantial fraction of the links are unidirectional, i.e., loss rate strictly below 100% in one direction, and 100% loss rate in the reverse direction. Last but not least, wireless link quality between two nodes A and B can vary significantly over time, even at small time scales [36] – a phenomenon we also experienced in our experiments.

The above wireless connectivity characteristics lead to the following observations. ICN routing protocols running on constrained devices need to satisfy conflicting requirements (i) negligible control traffic to reduce energy consumption and small state to fit memory constraints, while at the same time (ii) dynamic tracking of wireless link to avoid non-functional paths. The goal is to not forward an interest in the first place if reverse link is not "good enough". The overhead for failing is a reverse path taken by content which often fails and will lead to PIT time-outs,

interest flooding, etc. Subsequently, this might lead to the same failing reverse path – and thus be very inefficient both in terms of energy and delay.

5.3   Different Communication Models

The ICN communication model is based on a *pull* paradigm: in a first phase, a node expresses interest in some content, and in a second phase, the node should receive this content. However, this communication model alone is not sufficient to accommodate typical traffic patterns in the IoT. Aside of pull, these patterns include for instance *push* paradigms (e.g., for actuators), and *observe* paradigms [41] whereby a node can register for updates from a given content producer (e.g., a sensor measuring the real-time evolution of a given parameter). Note that explicit acknowledgements are also typically used in this context, for example patterns such as push+ACK, or request+reply+ACK are the norm in this domain. Recent work has started to integrate these patterns in ICN, such as [22] which proposes a push mechanism for CCN on sensor networks.

Furthermore, the simplified communication model at the base of ICN was initially designed with the assumption that the number of consumers is much larger than the number of producers, targeting use cases that are comparable to the scenarios CDNs aim for. Such an assumption does not hold in general in the IoT, where consumers (e.g., a data sink) are often outnumbered by producers (e.g., sensors). In consequence, content caching strategies designed for scenarios similar to CDN will not be efficient in the IoT, and thus, alternative strategies should be designed for content replication and content cache replacement in the IoT with ICN.

6   Conclusion and Perspectives

ICN has recently been identified as a potential alternative network paradigm for the Internet of Things. In this paper, we have carried out experiments with NDN on a real IoT deployment consisting in tens of constrained nodes in multiple rooms of multiple buildings. Based on this experience, we have shown that ICN is indeed applicable in the IoT, and that it can offer advantages over an approach based on 6LoWPAN/IPv6/RPL in terms of energy consumption, as well as in terms of RAM and ROM footprint. We have proposed several interoperable NDN enhancements to decrease energy consumption and routing state. Furthermore, we identified several areas where future work is needed. Topics include (i) an efficient header compression and fragmentation/reassembly adaptation layer below NDN to fit typically small frame sizes, (ii) IoT-specific content replication and cache replacement strategies, (iii) enhancements of the basic ICN communication model to accommodate IoT traffic patterns, (iv) further studies on the impact of caching on content availability in the context of sleeping nodes, and (v) short naming schemes optimized for constrained devices.

References

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] C. Bormann, M. Ersue, and A. Keranen, "Terminology for Constrained-Node Networks," RFC 7228 (Informational), Internet Engineering Task Force, May 2014. [Online]. Available: http://www.ietf.org/rfc/rfc7228.txt

[4] J. Cordero, J. Yi, T. Clausen, and E. Baccelli, "Enabling Multihop Communication in Spontaneous Wireless Networks," in *ACM SIGCOMM eBook on "Recent Advances in Networking", Volume 1, Chapter 9, pp. 413-457.* ACM, 2013.

[5] ZigBee Alliance, "ZigBee Specifications," 2012.

[6] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," RFC 4944 (Proposed Standard), Internet Engineering Task Force, Sep. 2007, updated by RFCs 6282, 6775. [Online]. Available: http://www.ietf.org/rfc/rfc4944.txt

[7] T. Winter and P. Thubert, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550 (Proposed Standard), Internet Engineering Task Force, Mar. 2012. [Online]. Available: http://www.ietf.org/rfc/rfc6550.txt

[8] A. Ghodsi, B. Ohlman, J. Ott, I. Solis, and M. Wählisch, "Information-centric networking: Ready for the real world?" *Dagstuhl Reports (Seminar 12361)*, vol. 2, no. 9, pp. 1–14, 2012.

[9] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," pp. 1–12, 2009.

[10] N. Fotiou, D. Trossen, and G. Polyzos, "Illustrating a publish-subscribe internet architecture," *Telecommunication Systems*, vol. 51, no. 4, pp. 233–245, 2012.

[11] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, "Network of information (netinf), an information-centric networking architecture," *Computer Communications*, vol. 36, no. 7, pp. 721 – 735, 2013.

[12] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 181–192, Aug. 2007.

[13] A. K. M. M. Hoque, S. O. Amin, A. Alyyan, B. Zhang, L. Zhang, and L. Wang, "NLSR: Named-data Link State Routing Protocol," in *Proc. of the 3rd ACM SIGCOMM WS on ICN.* New York, NY, USA: ACM, 2013, pp. 15–20.

[14] 2012, "Ospfn: An ospf based routing protocol for named data networking," Lan Wang, Mahmudul Hoque, Cheng Yi, Adam Alyyan and Beichuan Zhang. [Online]. Available: http://www.named-data.net/techreport/TR003-OSPFN.pdf

[15] J. Martocci, P. D. Mil, N. Riou, and W. Vermeylen, "Building Automation Routing Requirements in Low-Power and Lossy Networks," RFC 5867 (Informational), Internet Engineering Task Force, Jun. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5867.txt

[16] B. Saadallah, A. Lahmadi, O. Festor, "CCNx for Contiki: implementation details," in *INRIA, Tech. Report RT-0432*. INRIA, 2012.

[17] T. Biswas et al., "Contextualized information-centric home network," in *ACM SIGCOMM*, 2013.

[18] L. Grieco et al., "Architecting information centric etsi-m2m systems," in *Proc. of PERCOM*, 2014.

[19] Y. Zhang, D. Raychadhuri, R. Ravindran, G. Wang, "ICN based Architecture for IoT," in *IETF Internet Draft draft-zhang-iot-icn-architecture-00*. IRTF, 2013.

[20] Y.-T. Yu, R. B. Dilmaghani, S. Calo, M. Sanadidi, and M. Gerla, "Interest propagation in named data manets," in *Proc. of IEEE ICNC*, 2013, pp. 1118–1122.

[21] M. Amadeo, C. Campolo, A. Molinaro, and N. Mitton, "Named data networking: A natural design for data collection in wireless sensor networks," in *Wireless Days (WD), 2013 IFIP*. IEEE, 2013, pp. 1–6.

[22] J. Francois, T. Cholez, T. Engel, "CCN Traffic Optimization for IoT," in *International Conference on Network of the Future (NoF)*, 2013.

[23] F. Angius, M. Gerla, and G. Pau, "Bloogo: Bloom filter based gossip algorithm for wireless ndn," in *Proc. of the 1st ACM NoM Workshop*. New York, NY, USA: ACM, 2012, pp. 25–30.

[24] J. Burke, P. Gasti, N. Nathan, and G. Tsudik, "Securing instrumented environments over content-centric networking: the case of lighting control," *arXiv preprint arXiv:1208.1336*, 2012.

[25] S. Arianfar, P. Nikander, and J. Ott, "On Content-Centric Router Design and Implications," in *Proc. of ReARCH workshop*. USA: ACM, 2010.

[26] D. Perino and M. Varvello, "A Reality Check for Content Centric Networking," in *Proc. of the ACM SIGCOMM WS on ICN*. USA: ACM, 2011, pp. 44–49.

[27] D. G. Murray, E. Yoneki, J. Crowcroft, and S. Hand, "The Case for Crowd Computing," in *Proc. of the 2nd ACM SIGCOMM MobiHeld WS*. New York, NY, USA: ACM, 2010, pp. 39–44.

[28] P. Levis, A. Tavakoli, and S. Dawson-Haggerty, "Overview of existing routing protocols for low power and lossy networks," *IETF Internet Draft draft-ietf-roll-protocols-survey*, 2009.

[29] "CCN Lite: Lightweight implementation of the Content Centric Networking protocol," 2014. [Online]. Available: http://ccn-lite.net

[30] E. Baccelli, O. Hahm, M. Günes, M. Wählisch, and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *Proc. of the IEEE INFOCOM. Poster*, NJ, USA: IEEE Press, 2013.

[31] "RIOT open source code on GitHub," 2014. [Online]. Available: https://github.com/RIOT-OS/RIOT

[32] A. Brandt, J. Buron, and G. Porcu, "Home Automation Routing Requirements in Low-Power and Lossy Networks," IETF, RFC 5826, 2010.

[33] Z. Fan, G. Kalogridis, C. Efthymiou, M. Sooriyabandara, M. Serizawa, and J. McGeehan, "The new frontier of communications research: Smart grid and smart metering," in *Proc. of ACM e-Energy*. New York, NY, USA: ACM, 2010, pp. 115–118.

[34] G. Wittenburg *et al.*, "Fence Monitoring - Experimental Evaluation of a Use Case for Wireless Sensor Networks," in *Proc. of EWSN*, 2007.

[35] Michael Baar et al., "The ScatterWeb MSB-A2 Platform for Wireless Sensor Networks," Freie Universität Berlin Tech. Rep, Tech. Rep., 2008.

[36] E. Baccelli and C. Perkins, "Multi-hop Ad Hoc Wireless Communication," IETF, Internet-Draft – work in progress 03, March 2014.

[37] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the Data Plane – Threats to Stability and Security in Information-Centric Network Infrastructure," *Computer Networks*, vol. 57, no. 16, pp. 3192–3206, Nov. 2013.

[38] N. Choi, K. Guan, D. C. Kilper, and G. Atkinson, "In-network caching effect on optimal energy consumption in content-centric networking," in *Proc. of IEEE ICC*, 2012, pp. 2889–2894.

[39] U. Lee et al., "Greening the internet with content-centric networking," in *Proc. of the ACM e-Energy*, 2010.

[40] M. Isomaki et al., "Transmission of IPv6 Packets over BLUETOOTH Low Energy," *IETF Internet Draft draft-ietf-6lo-btle*, 2014.

[41] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained application protocol (coap)," *IETF Internet Draft draft-ietf-core-coap*, 2014.
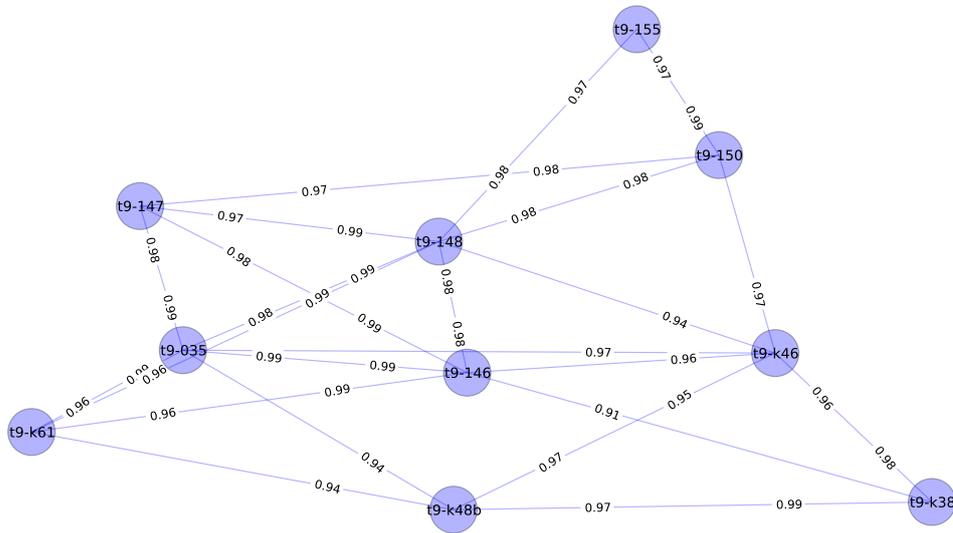
(a) 10 nodes are involved when a single consumer (t9-k38) requests content published by t9-155.
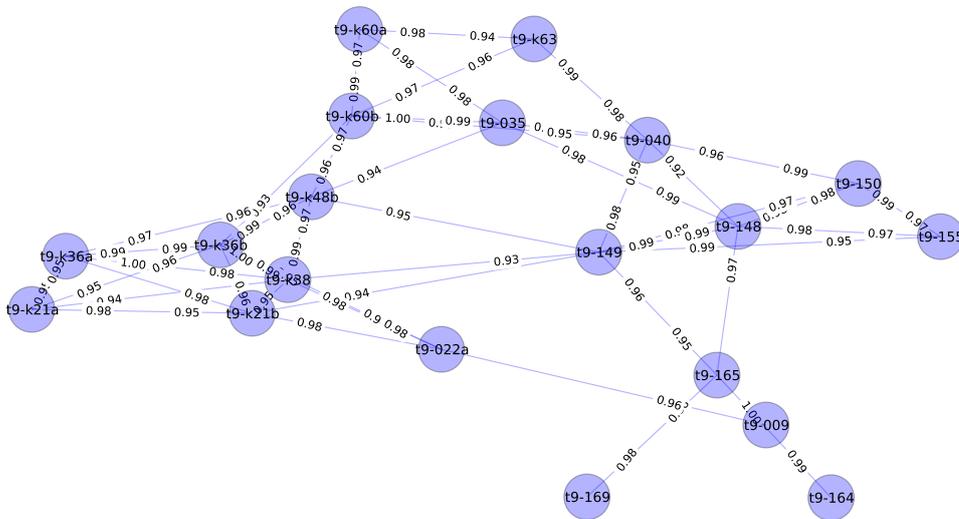
(b) 20 nodes are involved when multiple consumers (t9-149, t9-148, and t9-150) request content published by t9-k36a

Figure 2: Snapshot of the link-layer network topologies used in the experiments for single and multi consumer scenarios. Each topology spans over 3 floors in the right-most building shown in Figure 1. Link weights describe % of received packets, per link, per direction.
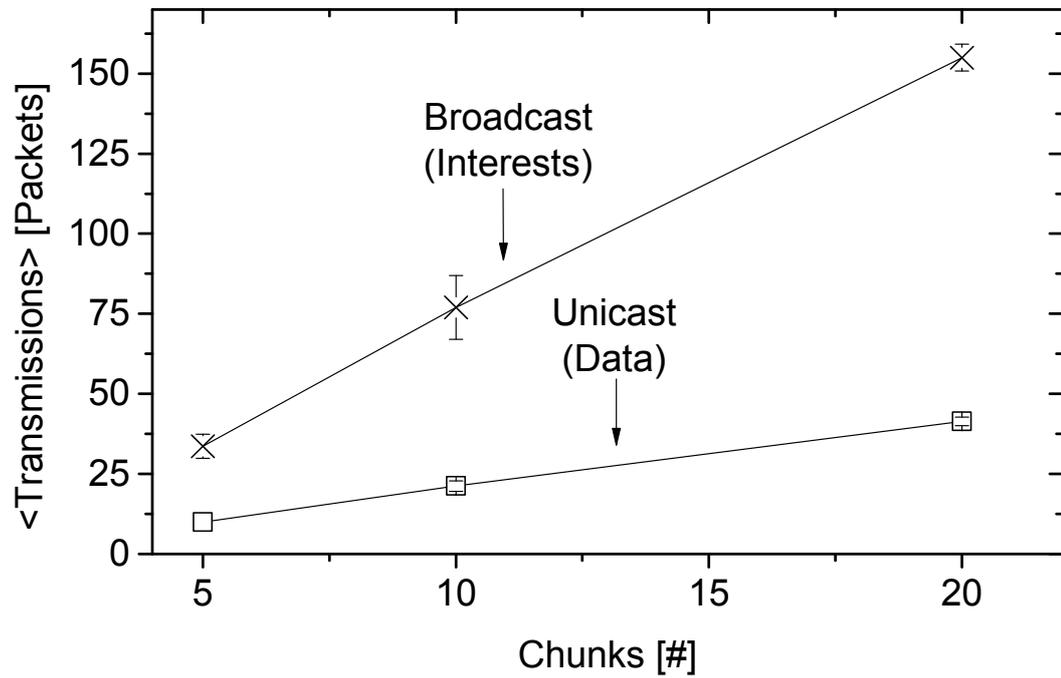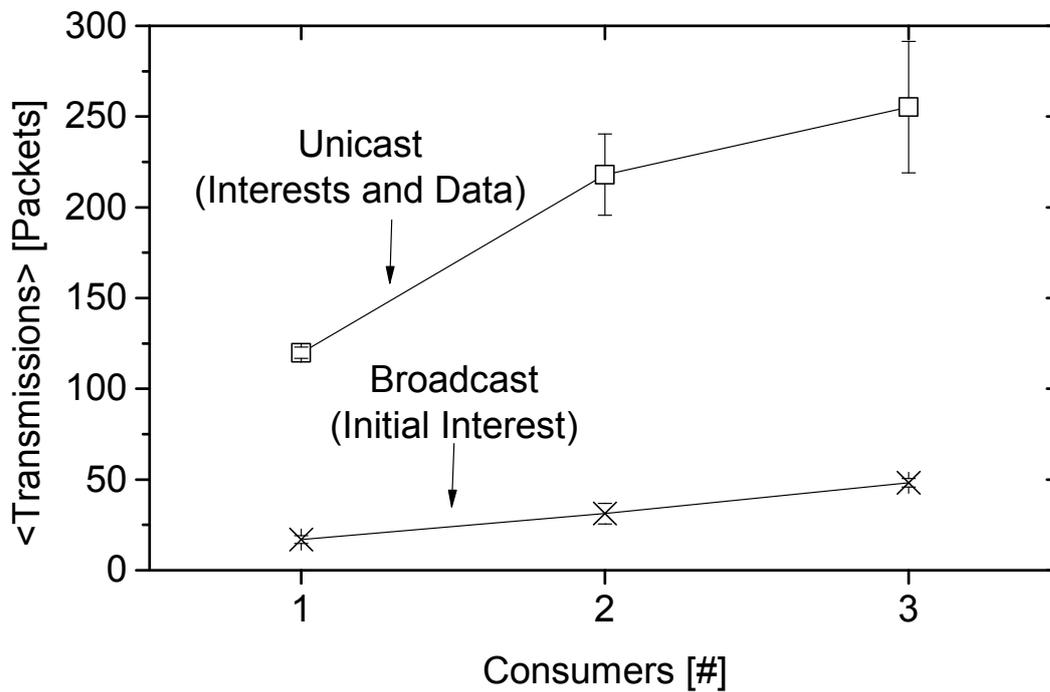
(a) Vanilla Interest Flooding
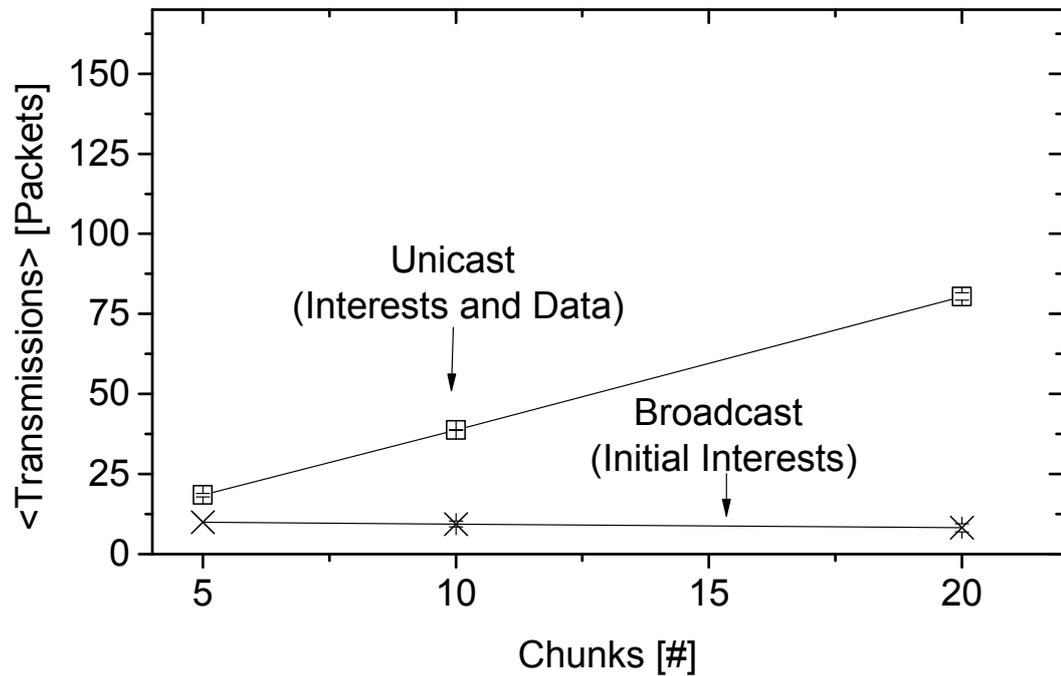
(b) Reactive Optimistic Name-based Routing

Figure 3: Single-consumer scenario. NDN performance for different routing schemes. Average number of packets transmitted in a network of 10 nodes to fetch content of various size.

(a) Without caching
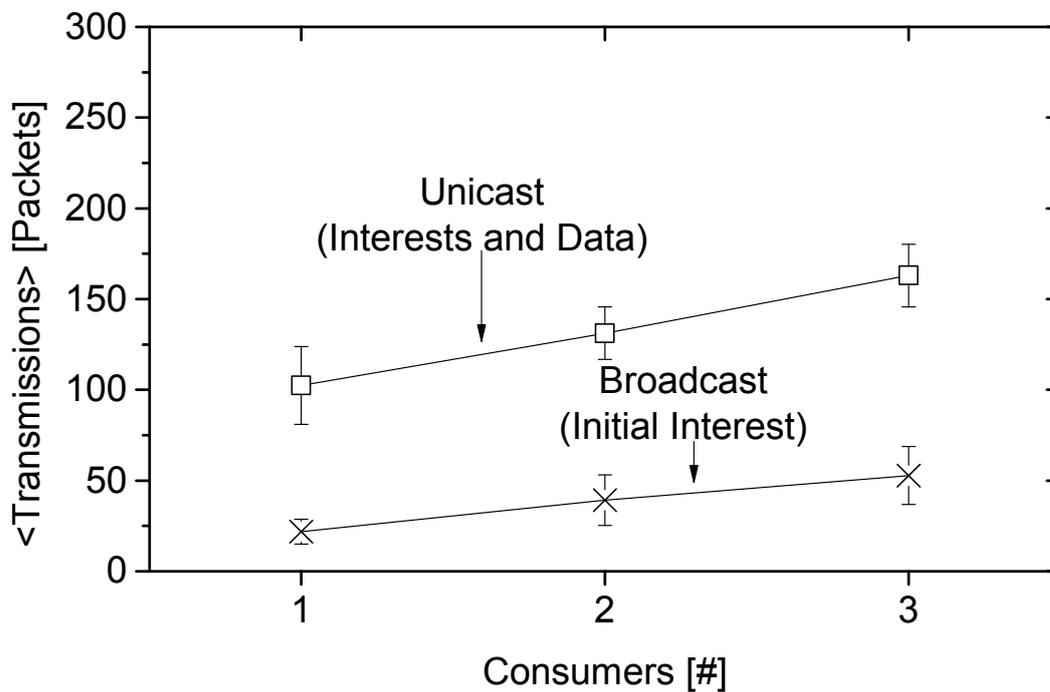
(b) With caching

Figure 4: Multi-consumer scenario. NDN performance for RONR and different content cache schemes. Average number of packets transmitted in a network of 20 nodes with a variable number of consumers.
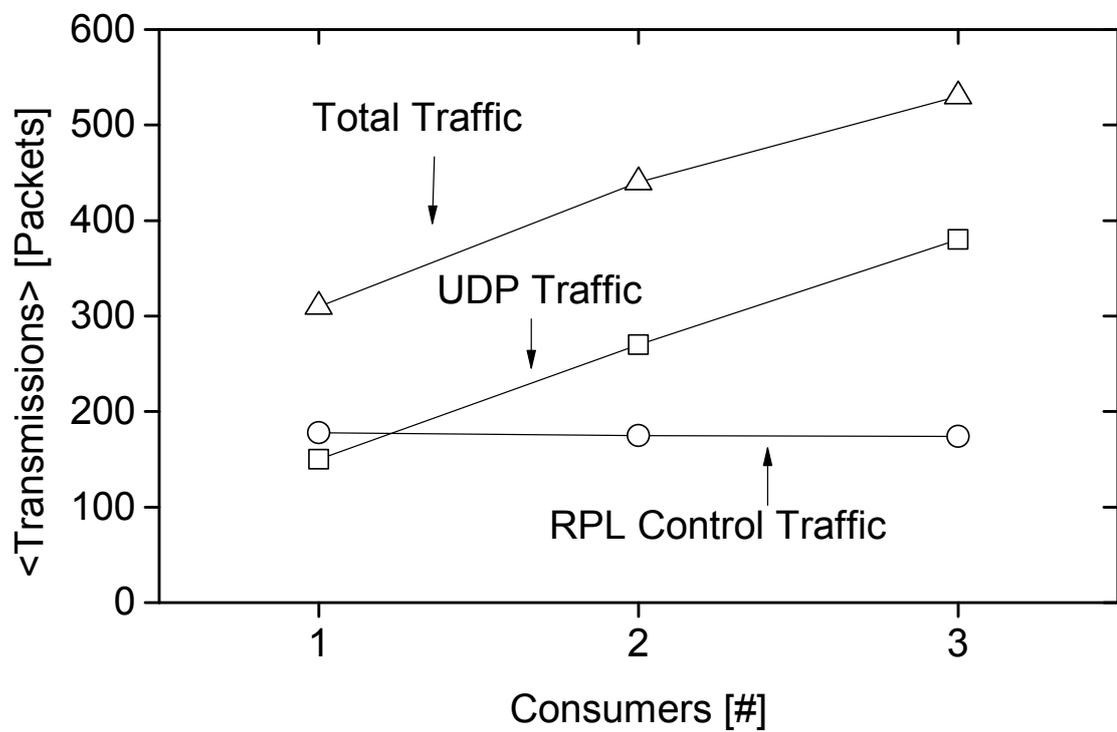
Figure 5: Multi-consumer scenario with 6LoWPAN/RPL/UDP. Average number of packets transmitted in a network of 20 nodes.