Design Choices for the IoT in Information-Centric Networks

Anders Lindgren*†, Fehmi Ben Abdesslem*, Bengt Ahlgren*, Olov Schelén†, Adeel Mohammad Malik‡
*SICS Swedish ICT AB, Sweden {andersl,fehmi,bengta}@sics.se
†Luleå University of Technology olov.schelen@ltu.se
‡Ericsson adeel.mohammad.malik@ericsson.com

***Abstract*— This paper outlines the tradeoffs involved in utilizing Information-Centric Networking (ICN) for Internet of Things (IoT) scenarios. It describes contexts and applications where the IoT would benefit from ICN, and where a host-centric approach would be better. Requirements imposed by the heterogeneous nature of IoT networks are discussed in terms of connectivity, power availability, computational and storage capacity. Design choices are then proposed for an IoT architecture to handle these requirements, while providing efficiency and scalability. An objective is to not require any IoT specific changes of the ICN architecture per se, but we do indicate some potential modifications of ICN that would improve efficiency and scalability for IoT and other applications.**

I. Introduction

Information-Centric Networking (ICN) has been shown to efficiently meet current usage demands of computer networks, where users consume content from the network instead of communicating with specific hosts. The applications and usage of the Internet of Things (IoT) often imply information centric usage patterns, where users or devices consume IoT generated content from the network instead of communicating with specific hosts or devices.

However, while the IoT shares many characteristics with typical information centric applications, it differs because of the high heterogeneity of connected devices (including sensors and actuators), the very high rate of new information being generated, and the heterogeneity in requirements from applications regarding information retrieval and dynamic actuation. Because of these differences, using an Information-Centric Network to design an architecture of the IoT is often, but not always, beneficial. Depending on the context, the IoT architecture may benefit from using an ICN or a host-centric network (HCN). In practice, the right approach is a complex tradeoff that depends on the applications and usage of the IoT network.

This paper describes some advantages and inconveniences of using an ICN for the IoT architecture, and helps finding the right tradeoff between using an ICN or an HCN, depending on the context. We explore how to represent and model IoT on top of existing ICN solutions, without requiring IoT specific functionality in the ICN. We discuss this in terms of effectiveness, efficiency and scalability. However, in some cases we also open further discussion on possible additions to ICN functionalities in order to make the overall IoT solution more efficient and scalable. The paper serves as a problem statement and also proposes some design choices in order to efficiently and effectively provide IoT over ICN. This information is intended to be used as a basis for further discussion on architectural design for the IoT. This is in line with the recent efforts in the community [1], [2] to drive the discussion forward and reach standardization and best practice recommendations. This will also help in the design of ICN implementations [3] such as CCNx [4] and CCN-lite [5] to better take into account the particular case of the Internet of Things.

Previous work have considered using an ICN architecture for the IoT [6], [7] and have run some initial tests [8]. Quevedo et al [9] make a case for IoT usage in ICN environments, where they study the benefits and implications of caching in the network, especially in terms of bandwidth usage and energy consumption. Foutiou et al [10] present their vision for a global Internet of Things (IoT) architecture relying on information and its identifiers/names. They argue that the Information-Centric Networking (ICN) paradigm is the ideal candidate architecture for the realization of that IoT vision. Rayes et al [11] consider performance and security implications when developing IoT architectural framework, and present key ICN performance and security requirements of IoT networks, together with a case study. Amadeo et al [12] present a design of an ICN framework tailored to the smart home domain, considered as a major representative of IoT scenarios. Burke et al [13] address efficient and secure sensing over Named Data Networking NDN (a project in the ICN area), motivated by the convergence of the IoT vision with traditional Building Automation Systems (BAS). Sugang et al [14] present a unified IoT platform called ICN-IoT, and they explore and evaluate in terms of mobility support.

The remainder of this paper is organized as follows. Section II presents the practical advantages of using ICN for IoT. Section III outlines the design challenges in applying ICN to the IoT. Based on this, Section IV describes the fundamental choices to consider when using ICN, and a particular attention to security issues is given in Section V before the paper is concluded in Section VI.

II. Advantages of using ICN for IoT

A key concept of ICN is the ability to name data independently from the current location where it is stored, which simplifies caching and enables decoupling publishers and consumers. This section highlights these general benefits that ICN could provide to IoT networks.

A. Naming of Data and Services

In many common applications of IoT networks, data and services are the main goal, and communication with specific devices is secondary. The ICN network distributes content from IoT devices and provides a service, instead of establishing a communication link between two devices. In many IoT scenarios with redundant devices (e.g., a crowd), data content and services can be provided by several devices, or group of devices, hence naming data and services is often more important than naming the devices.

B. Distributed Caching

While caching mechanisms are already used by other types of overlay networks, IoT networks can potentially benefit even more from caching systems, because of their resource constraints. Wireless bandwidth and power supply can be limited for multiple devices sharing a communication channel, and for small mobile devices powered by batteries. In this case, avoiding unnecessary transmissions with IoT devices to retrieve and distribute IoT data to multiple places is important, and storing such content in the network can save wireless bandwidth and battery power. Moreover, applications for IoT networks requiring short delays can benefit from local caches to reduce delays between content request and delivery.

C. Decoupling between Publisher and Consumer

IoT devices may be mobile and face intermittent network connectivity. When specific data is requested, such data can often be delivered by ICN without any consistent direct connectivity between devices. Apart from using structured caching systems as described previously, information can also be spread by forwarding data opportunistically. Such decoupling between the publisher of data and the consumer also creates increased possibilities for horizontal sharing of IoT data between applications and services.

In particular, this provides an inherent support for mobility among devices acting as consumers of data. As devices move throughout the network, there is no need to maintain long-lived connections or update the topological location of the device as each request for an object is independent. Thus, in requesting a stream of objects, new requests will be sent from the new location as the device moves. If the device can anticipate handover between access technologies, it can further enhance performance by sending some requests over multiple paths to reduce the risk of some responses being lost.

III. Design Challenges of IoT over ICN

As outlined in Section II, there are potential benefits from using ICN to implement IoT communication architectures. However, in order to obtain a scalable and efficient architecture there are some aspects of ICN that must be specifically considered in making the right design choices for IoT. This section outlines some of the specific challenges that must be considered and describes some of the trade offs that will be involved. We will address these challenges in our proposed design choices later in Section IV.

A. Naming of Devices, Data and Services

The ICN approach of named data and services (i.e., device independent naming) is typically desirable when retrieving IoT data. However, data centric naming may also pose challenges.

- **Naming of devices:** Naming devices is often important in an IoT network. The presence of actuators requires clients to act specifically on a device, for example to switch it on or off. Also, managing and monitoring the devices for administration purposes requires devices to have a specific name allowing to identify them uniquely. This is addressed in more detail in Section IV-C2.
- **Size of data/service name:** In information centric applications, the size of data is typically larger than its name. For the IoT, sensors and actuators are common and may generate or use data as small as a short integer, or a one-byte instruction to switch on an actuator. The name of the content for each of these pieces of data has to uniquely identify the content. Many existing naming schemes have long names that are likely to be longer than the actual data content for many IoT applications. While this is an acceptable overhead for larger data objects, it is infeasible for use when the object size is on the order of a few bytes.
- **Hash-based content name:** Hash algorithms are commonly used to name content in order to verify that the content is the one requested. This is only possible in contexts where the requested object already exists, and where there is a directory service to look up names. This approach is suitable for systems with large data objects where it is important to verify the content, but it is a challenge for IoT systems where data is dynamically generated.
- **Metadata-based content name:** Relying on metadata allows to generate a name for an object before it is created. However this mechanism requires metadata matching semantics.
- **Naming of services:** Similarly to naming of devices or data, services in IoT networks can be referred to with a unique identifier. Contrary to HCN, this service can be provided in ICN by a group of devices, for example the ones matching certain metadata conditions. Example of services include content retrieval, taking a content name as input and returning that content, or actuation, taking an actuation command as input and possibly returning a status code afterwards.

B. Efficiency of Distributed Caching

Distributed caching is a key feature of ICN. However, an IoT framework must be carefully designed to reap the maximum benefits of ICN caching. When content popularity is heterogeneous, some content is often requested repeatedly. In that case, the network can benefit from caching.

However, using distributed caching mechanisms in the network is not useful when each object is only requested at most once, as a cache hit can only occur for the second request and later. It may also be less useful to have caches distributed

throughout ICN nodes in cases where there are overlays of distributed repositories, e.g., a cloud or a Content Distribution Network (CDN), from which all clients can retrieve the data. Using ICN to retrieve data from such services is beneficial, but in case of dense occurrence of overlay CDN servers the additional benefit of caching in ICN nodes would be lower.

C. Decoupling between Publisher and Consumer

Decoupling the publisher and consumer is a useful mechanism offered by the ICN approach, especially for content retrieval with duty cycling devices or devices with intermittent connectivity. However, in order to efficiently retrieve data it must be possible for consumers to easily deduce the name of the data to request, without any direct contact with the publisher.

The decoupling provides a solution to the problem of consumer mobility as discussed in Section II-C and also removes the need to maintain long-lived connections. However, publisher mobility, when a device producing and publishing a data object is mobile still exhibits many of the same challenges as device mobility in today's Internet as it is necessary to be able to route requests towards the network location of a named data object.

Furthermore, decoupling is a challenge when authentication is needed for management and actuation, or when real-time interaction between devices is necessary. Solutions for object security supporting decoupled authentication (e.g., similar to signing by proxy), and solutions for pushing data to decoupled entities must be explored.

IV. Proposed Design Choices for IoT over ICN

The previous two sections have outlined the advantages and tradeoffs of utilizing ICN for IoT systems, and Table I provides an overview of the key advantages and challenges. In this section, we use this knowledge to describe some fundamental design choices to allow for effective, efficient, and scalable handling of IoT applications in an ICN network. An objective with these choices is to facilitate the use of generic ICN principles without focusing on specific architectures and without requiring new functionality to be added to the ICN architecture to support IoT. However, in some cases we do invite discussion on tentative additions of functionality in order to make the overall IoT solution more efficient and scalable. As ICN networks are likely to coexist with traditional IP networks, we will also consider that there may be situations where a host centric addressing is more suitable for IoT networks.

A. Data naming considerations

As IoT data components are often small and simple, a general challenge in defining ICN applications is to decide how to compose/group the data so that it can be effectively named and requested. Requesting partial data inside a composition may become a challenge. Indeed, if data is composed and sub components are requested, which are not directly namable by the requestor, finding such a subset will resemble a database query which may require processing to resolve. The IoT framework should be defined so that no new functionality is required in the ICN for searching data or subcomponents of data. The ICN network supports just naming of atomic data objects, while any searching is provided by the IoT framework, which in itself may be constituted by a highly distributed set of nodes that provide processing, analysis and aggregation of IoT data.

A design choice regarding IoT data is therefore to not require the ICN network to support advanced queries and instead only support directly addressable data objects. Any advanced (de-)composition of IoT data would be handled at the application layer instead of inside the ICN network. This is to avoid making new requirements on the ICN and to make sure that the need for computation is kept low in the ICN network, essentially limiting it to deciding whether there is a cache hit or not. There are some considerations following from this design choice. First, the size of directly addressable objects should be kept fairly small to avoid that unwanted data is sent over resource constrained networks and cached in the ICN network. There is however a tradeoff in that smaller data objects results in a larger naming overhead. Second, this approach means that a flat ICN address space would be sufficient, but for practical reasons a hierarchical address space may add some benefits. In any case, there is flexibility in using different addressing schemes depending on what is supported by the existing ICN framework.

1) Data naming in streams of immutable data objects: The number of IoT devices as well as the amount of data produced by these devices can be very large, and data may be spread over large ICN networks. The potential problem of cache inconsistencies may therefore be large if we allow for data to be mutable objects. To support scalability and horizontal distribution it is essential to define data properties that facilitate independency and consistency, while minimizing the need for dynamic global synchronization.

A key design choice is therefore to mandate that IoT only uses immutable atomic data objects. This supports large scale distribution by ensuring that there is no stale data in the ICN domain. A cache hit is always a clean hit. A trade-off from this is that dynamic data must be modeled as a stream of immutable data objects, potentially consuming more resources. However, this challenge can be resolved by smart caching strategies where old data is dropped.

Many IoT devices produce new sensor readings at regular intervals or on demand. With the design choice of immutable atomic data objects, there is a need to model the resulting stream of sensor readings with a stream of immutable data objects in the ICN domain. The need in this situation is very similar, if not identical, to video streaming, where video frames or chunks are immutable data objects in a video stream. However, since new data objects (with new names) representing different versions of a sensor reading may be emitted frequently, there must be a way to differentiate the versions.

To support immutable streamed data efficiently, while adhering to the expected naming schemes of ICN, we recommend that names of data objects include a sequence number. When data can be named with sequence number, any request may or may not include such a sequence number. If no number is included in the request, the nearest cache hit will result in a response. If a sequence number is included in the request, only an exact cache match will result in a response. A client

TABLE I. ADVANTAGES AND CHALLENGES OF IOT OVER ICN

Aspect	Advantages	Challenges
Naming	• Naming of data and service inherent to ICN • Self-certifying names good for disconnected operation • Object-based security removes need for trust of individual nodes	• Naming of dynamic content and actuators non-trivial • Size of name can exceed size of data • Signature-based
Caching	• Reduce transmissions with heterogeneous request patterns • Improve performance for duty-cycled devices • Reduce latency	• Not useful when each object requested at most once • Cache placement/provisioning
Actuation	• Possibility to address actuator based on name connected to its service	• ICN not designed to address a specific node, often needed for actuation • Caching may be counter effective • May have strong latency requirements
Decoupling publisher/consumer	• Improves performance for intermittently disconnected/duty cycled networks • Data can be forwarded opportunistically • Consumer mobility inherently supported	• Security becomes a challenge • Need to deduce object name without contact with originator • Publisher mobility still unresolved
Security	• Object-based security removes need for trust of individual nodes • Secure retrieval of stream of objects from multiple sources possible	• Signature-based schemes for name-data integrity and autheticity requires PKI • Size of name can exceed size of data • Signature-based

that wants the ”latest” reading can according to our previously mentioned design choice, in Section IV-A, not ask the ICN network such a high level query, instead it must ask for the specific (version of) information. To avoid complicated searching in the ICN nodes, there is thus no way to explicitly ask the network for the ”latest” reading, or any other ”range” of sequence numbers.

Should a client want the latest reading from a sensor, one method for this is to make a subscription for the pushed stream of data, as described in Section IV-B2, provided that the particular ICN architecture supports this interaction model. The confirmation of that subscription can contain the latest reading, and then obviously the normal stream will be received. The reason for including the latest reading in the response is to immediately provide the ”state” of sensors that generate new data infrequently.

Another method to obtain the latest reading, or a particular reading in the past, from a sensor is to perform adaptive probing, for example by binary interval reduction. If a requested sequence number does not (yet) exist, there will be a negative answer from the ICN. This method is preferably combined with application knowledge, for example, in the form of capability advertisements as described in Section IV-C1that enable the consumer to better predict the sequence number to request. The consumer that always wants the latest value could also dynamically tune its requests for the next data value to the frequency of the publisher in order to minimise the latency and load on the network. If the ICN supports pending requests (i.e., long lived requests) the consumer may send requests for data that will soon be published (provided that the name of that data can be deduced beforehand). However, the fact that non-existing data is asked for would potentially pose an overload threat to the ICN since each request of non-existing data could result in cache misses that ripple through all the way to the source, which has to respond that the data doesn't exist. It may therefore be beneficial with negative caching so that some requests are immediately denied at the network edge. Serving requests for non-existing data is however a generic challenge to ICN (not specifically to IoT) to be resolved.

There is a third method ”in between” the above two. If requests for a not yet existing data object can be held for a short time until the data object is actually available, instead of immediately returning ”not found”, these pending requests act as one-time subscriptions. Provided that request aggregation is being used, this mechanism would be efficient and latency-minimising, and at the same time would not require persistent subscription state. However, such a solution may result in transient subscription (pending request) state all the way through the network.

The support for sequence numbers depends on the particular flavor of ICN. The naming scheme of CCN/NDN may here provide an advantage. It is for further study whether it is possible to use ICNs that do not support sequence numbers as part of naming (e.g., by clever use of metadata, namespace, and search functionality) and what the trade-offs would be.

Two issues for further study are the size of the sequence number space and gaps in the sequence numbers. Must sequence number wraparound be handled, or is it possible to require a large enough sequence number space? Wraparound means an exception to the assumption on immutable objects. Gaps in the sequence number space might result in inefficiencies in some of the above methods, or, if the gaps are large, making them unfeasible. Yet, it might not always be possible to guarantee that there are no gaps.

B. Network functionality and roles

1) Decoupling and roles of publishers and consumers: Since ICN networks essentially support a request/response model of interaction, we denote the consumers of information as requestors, and the publishers of information as responders. The ICN network in itself provides decoupling of requestors and responders. It is an important feature of the ICN that it will allow responders (e.g., IoT devices) to be occasionally unreachable (e.g., due to intermittent connectivity, low battery level, duty cycling). Another advantage is that caching in the ICN will ensure that data objects are normally delivered only once from the IoT devices, independently of the number of immediate requestors. While this solves the issue of mobility among requestors/consumers, mobility of responders/publishers is still an issue. The network needs to support methods for locating a copy of a data object (through routing or name resolution), and this can be handled differently in different ICN architectures. By allowing IoT devices to not necessarily be full fledged ICN nodes, it is possible for IoT devices that are highly mobile or operate on a low duty cycle

to delegate the responsibility of responding to requests for its data (and thus acting as the data publisher in the ICN domain) to an ICN node with a more stable connection.

Note however, that the ICN does not (and should not) provide any transformation or aggregation of data. The IoT dissemination architecture should therefore allow for any number of intermediate processing nodes. An intermediate node will be an endpoint in the ICN network that can act as both requestor and responder. Such a node may perform aggregation, filtering, selection, etc. The instantiation of such nodes may for example form a directed (acyclic) graph between ultimate responders (IoT devices) and ultimate requestors (the final applications). It is for further study how to define such an architecture.

It is a design choice to keep the IoT dissemination and aggregation functionality outside of the ICN domain. That architecture would be an overlay that may have intricate structure, and put the ICN usage in a new context, where content from ultimate requestors to ultimate responders may go through many IoT processing nodes that collect, process and re-publish data through an ICN for various purposes.

2) Combination of PULL/PUSH model: A critical decision regarding IoT data is whether to use a PULL model, a PUSH model, or both. In this paper, we define a PULL model as a system where data is only sent when explicitly requested, while a PUSH model indicate that data transmission is initiated by the source based on some trigger (either periodic, for each new object, or based on some condition on the generated data). There are some intrinsic trade-offs between these models. The PULL model is for example resource efficient when there is an abundant amount of IoT information, potentially redundant from many devices, and the clients only occasionally or partially are interested in the information. The PUSH model is for example efficient when there is real-time information and the clients are interested in all information from specific devices all the time.

A design decision in the IoT domain is to support PULL, while having some options for PUSH. The base model should be PULL, since this is the native mode of ICN, meaning that requesters must always start by sending a request. If the request is for some specific data, it can be resolved by returning the data (if it exists). The pure pull model can be supported efficiently and scalably by an ICN network.

A pull model can be used also for retrieving periodic real-time information if the name of the most recent and next upcoming immutable data item can be deduced directly or indirectly by the requester. A challenge with the pull model is however that it may be inefficient for retrieving new data that occur sporadically or based on specific conditions. The network would then have to support pending request state (or subscriptions) for indefinite time, which would require state in the network. Some ICN may for that reason not support such a feature. Our proposal for an IoT framework is therefore that there must be support for efficiently retrieving such triggered information, without having to poll for it through the ICN. Our proposal is that a request can also include triggers, which means that data will be returned (pushed) when triggers are fulfilled, which may be immediately, or in the future at one or several occasions. This can be used to select alarm conditions, to request continuous or periodic push, etc. The trigger conditions could in principle be set by the requester, or be pre-defined by the responder. The former would be more flexible but also have performance/scalability issues since the number of trigger conditions and consequent data generation would depend on a potential large number of requesters. The latter is more scalable since there will be a predefined and finite number of trigger conditions (as defined in capability advertisements). Our recommended choice, at least for the initial phase, is to go for a simple and scalable solution and therefore adopt the model where available trigger conditions are defined and advertised by the responder.

With this, there is no requirement raised on ICNs supporting data push, but we recommend to have a discussion on whether an ICN network can or should provide an option to effectively support a push model of data. Such support could make real-time IoT data dissemination more efficient and scalable as previously mentioned in Section IV-A1. However, since we assume that the ICN works with existing IP protocols, such functionality can be provided without ICN, by using traditional unicast or multicast communication. We finally note that an ICN supported push service model would make the ICN network more like a publish/subscribe system.

3) Name-based routing vs name resolution: As described in Section IV-A, the IoT framework should be defined so that new ICN functionality is not needed. For data that is frequently generated and regenerated, it makes sense to keep simple structures and provide directly inferable naming/addressing of data objects, so requesters can directly address the data. For more complex data, such as pre-processed, aggregated and structured data a two-step resolution model is recommended. IoT devices can provide a higher level resolution based on, for example, queries and searching, resulting in a number of concrete directly addressable ICN objects. Consequently, the IoT framework should have no requirement that the ICN network itself should support 2-step addressing (although such 2-step methods may exist in some ICNs).

C. Node capabilities and roles

1) Capability advertisements: Capability advertisements and discovery can be used by requesters to discover which data is available and/or to which responders to connect to. In a deployment with large numbers of responders, the functionality of automatic advertisement and discovery becomes a critical factor to support scaling. Responders should advertise their methods (inputs, outputs, parameters, triggers, etc) and provide relevant metadata in the responses as advertised. Such capability advertisements should be conservative with resources, which suggests that new advertisements should be posted with reasonably low frequency. This implies that an ICN network can be used for providing capability advertisements. Should there be a need for real-time awareness of dynamic changes, a subscription/push model of data dissemination could be used as earlier described in Section IV-B2.

Capability might also include resource constraints, as it is questionable whether IoT devices also should provide caching for data produced by other IoT devices. In ad-hoc networks this may be desirable, but often there is a desire for wireless nodes to minimize communication by handling only data of their own concern. Our design decision in this regard is

that we logically separate IoT server functionality (such as sensing and transmitting IoT data) and ICN functionality (such as routing and caching data generated by other devices). A resource constrained device may choose to only implement IoT functionality and act as a server to the ICN, i.e., not act as intermediate ICN node. However, since storage is getting cheaper, IoT devices should be able to cache their own content and, in essence, act as sources to ICN.

2) Handling actuators in the ICN model: If actuators should be controlled using the ICN communication model, we need to map the functionality of the actuator to named data and/or the requesting of named data. We see two main models with some variants as described in the following paragraphs.

In the first model, the state of the actuator is represented by a stream of immutable named data objects. The actuator periodically requests a new state using the name of its designated state object. There then has to be a publisher of that state data responding with the current state. When the actuator receives the response, it invokes its actuation function to set the new state. Authentication of the publisher of the state is important, but as this corresponds directly to publisher and data object authenticity that are fundamental in the ICN model, there are no additional requirements for the IoT domain.

A variant of this first model is that a requester first requests the state of the actuator. The requester supplies additional information with the request including the name of the new state data it will produce. The actuator responds with its state, and then requests its new state using the name that was supplied with the additional information in the first request. This variant enables low latency without high frequency polling.

In the second model, the actuation function is invoked as a side-effect of receiving a particular request. There are several plausible variants. The new state could be encoded in the name of the requested data in the request, or could be supplied as additional information with the request. Regardless, the actuator acts on the new state information as a side effect, and responds with data, possibly its state, to the requester. The security issues are potentially larger with this model since, in the ICN model, anyone could make the request. Access control and/or requester authentication are therefore required.

We think that ICN caching is not as relevant for actuation as it is for data retrieval, and can even be problematic. For the first model less so, since the actuator can make sure that its state is arbitrarily up-to-date by sending unique requests. The variant of the first model and the second model have larger issues. With caching, it is hard for a requester to make sure that its request actually reaches the actuator, and thus, it is hard to bound actuation latency. Some caching directive might be needed in this case for reliable functionality.

D. The importance of time

Time is almost always a very important property of IoT data, and especially so for data that change over time. When modeling dynamic IoT data with a stream of immutable data values, it is often the case that a certain IoT data value is a sensor reading at a particular point in time, and the next value in the stream is the next reading in time. Thus, dynamic data is in this case dynamic over time, with well defined (immutable) values for particular points in time.

We argue that it is important to find a way to represent such time-related streams of immutable data values in ICN. It should be possible to request a data value from a certain time, and to infer/find the name (sequence number) of the most current data value. The question is whether or not stream sequence numbers are sufficient to support time. If not, the ICN system needs to be extended with explicit support for time, something we want to avoid. In general, the methods outlined in the previous section are applicable for finding an IoT data value from a particular point in time, including the latest. What is missing is the mapping between sequence number and time.

One possibility could be to use sequence numbers that directly correspond to time, for instance, the Unix (POSIX) time in form of seconds since January 1st, 1970. This would however both limit the time resolution to seconds, and also result in large gaps in the sequence numbers, something that can be problematic, as discussed in the previous section.

There are several other methods for finding readings from a certain time, or the latest reading, for example through a high level request from a server/endpoint, or by using a naming scheme where the name can be directly inferred, e.g., if an IoT device has advertised under which conditions it produces data and how it is named.

To represent absolute time so that it can be directly inferred, one method is that the publisher of data in its capability advertisements provide a mapping function between sequence number and time. Thereby also readings on the time axis are immutable while it is still possible to efficiently find the latest reading, as described in Section IV-A1. Note that sequence numbers then may have gaps in order to cater for triggered non periodic data, etc. Another method is to include meta data with information on absolute time. We note that by using any of the proposed mapping schemes, data from current time can be efficiently requested, provided that clock synchronisation is accurate enough (which is out of scope of this paper).

V. Security Considerations

ICN advocates the model of trust in content rather than trust in hosts. This brings in the concept of Object Security which is contrary to session-based security mechanisms such as TLS/DTLS prevalent in the current host-centric internet. Object Security is based on the idea of securing information objects unlike session-based security which secure the communication channel between pairs of nodes. In the context of IoT, the Object Security model has several concrete advantages; for many IoT applications, data and services are the main goal and specific communication between two devices is secondary. Thus it makes more sense to secure IoT objects instead of the session between communicating endpoints.

It is important that while security mechanisms complement the ICN architecture in a coherent fashion, they do so without laying down any strict requirements or constraints. Therefore, the decision of what security mechanisms are employed should be handled at a layer above ICN, in this case within the IoT framework. However, the ICN layer should not be completely oblivious of Object Security. At this point it is important to distinguish between the different aspects of Object Security: integrity, authenticity and confidentiality. ICN provides data integrity through Name-Data Integrity, the guarantee that the

given data corresponds to the name with which it was addressed. Typical ICN protocols provide Name-Data integrity using various schemes such as hash-based names and signatures. Signature-based schemes additionally provide data authenticity. Otherwise data authenticity should be provided in layers above the ICN layer. Data confidentiality should also be handled above the ICN layer. This facilitates flexibility and allows IoT applications more freedom to decide which encryption scheme suits them best.

In an ICN network, an IoT client relies on the network to deliver requested content without concerning itself with content location, potentially meaning that individual objects within a stream are retrieved from different sources. Having a trust relationship with each sources is not realistic and gives rise to the need of retrieving trusted content from untrusted nodes/caches in an ICN network. Through Name-Data Integrity, ICN automatically guarantees data integrity to the requester regardless of the source from where it is delivered. Additionally, Object-based signatures and encryption are ideal because it relieves IoT clients from having to establish trust with each node. This means that clients can use more caches in the network, resulting in better throughput and latency.

A. Energy efficiency of cryptographic mechanisms

Session-based security protocols rely on the exchange of several messages before a secure session is established between a pair of nodes. Use of such protocols in constrained IoT devices can have serious consequences in terms of energy efficiency because transmission and reception of messages is often more costly than the cryptographic operations, especially for wireless devices.

The problem is amplified proportionally with the number of nodes the constrained device has to interact with because a secure session must be established with every node. If the constrained device acts as a consumer of data this means setting up secure sessions with every caching node that the device retrieves data from. When acting as a publisher of data, the constrained device would have to setup secure sessions with all the consumers. The Object Security model eliminates this problem because the content is readily available in a secure state in the network.

VI. Conclusions and Future Work

In this paper, we have looked at some of the benefits and tradeoffs associated with using ICN technology for an IoT scenario. The key advantages identified are: 1) naming of data and services in a manner that is not dependent on the device providing that data or service, 2) possible gains from distributed caching in terms of reduced energy consumption due to fewer wireless transmissions and increased duty cycling possibilities, as well as reductions in information access latency, 3) decoupling between publisher and consumer of data in the network, leading to improved performance in networks with intentional or spurious communication disruptions and the possibilty for increased sharing of data between applications. Each of the advantages outlined above have several challenges and tradeoffs that need to be addressed in order to realise their full potential. These challenges include: 1) how to create and format efficient names suitable for huge numbers of, often very small, data objects, frequently created continuously and accessed in near real-time, such that they can still be handled by a large number of IoT devices, 2) how to maximize the benefits of in-network caching depending on data creation and consumption patterns and device requirements and capabilities, 3) security and application design issues arising from the new disconnected paradigm created by the decoupling of publishers and consumers.

In this paper, we have given some architecture-agnostic design choices and guidelines on how to address these issues in a generic ICN. To be able to evaluate the different design tradeoffs, we have started work on mapping the ideas outlined in this paper to the CCN1.x ICN architecture. Solutions for the challenges in this paper will be implemented in the CCN-lite [5] code base and large-scale evaluations will be conducted to quantify the advantages and tradeoffs of using an ICN architecture for the Internet of Things.

References

[1] A. Lindgren, F. Ben Abdesslem, B. Ahlgren, O. Schelén, and A. M. Malik, "Applicability and Tradeoffs of Information-Centric Networking for Efficient IoT," individual, IRTF Internet Draft – work in progress 03, July 2015. [Online]. Available: https://datatracker.ietf.org/doc/draft-lindgren-icnrg-efficientiot/

[2] Y. Zhang, D. Raychadhuri, L. A. Grieco, E. Baccelli, J. Burke, R. Ravindran, and G. Wang, "ICN based Architecture for IoT - Requirements and Challenges," individual, IRTF Internet Draft – work in progress 01, December 2014. [Online]. Available: https://datatracker.ietf.org/doc/draft-zhang-iot-icn-challenges/

[3] G. Carofiglio, G. Morabito, L. Muscariello, I. Solis, and M. Varvello, "From Content Delivery Today to Information Centric Networking," *Computer Networks*, vol. 57, no. 16, 2013.

[4] PARC. Project CCNx. [Online]. Available: http://www.ccnx.org/

[5] U. of Basel. CCN-lite. [Online]. Available: http://www.ccn-lite.net/

[6] G. C. Polyzos and N. Fotiou, "Building a reliable internet of things using information-centric networking," *Journal of Reliable Intelligent Environments*, 2015.

[7] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in named data networking for the wireless internet of things," in *Recent Advances in Internet of Things (RIoT), 2015 International Conference on*, 2015.

[8] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information centric networking in the iot: experiments with ndn in the wild," *arXiv preprint arXiv:1406.6608*, 2014.

[9] J. Quevedo, D. Corujo, and R. Aguiar, "A case for icn usage in iot environments," in *Global Communications Conference (GLOBECOM), 2014 IEEE*, Dec 2014, pp. 2770–2775.

[10] B. Xu, L. D. Xu, H. Cai, C. Xie, J. Hu, and F. Bu, "Ubiquitous data accessing method in iot-based information system for emergency medical services," *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 2, pp. 1578–1586, May 2014.

[11] A. Rayes, M. Morrow, and D. Lake, "Internet of things implications on icn," in *Collaboration Technologies and Systems (CTS), 2012 International Conference on*, May 2012, pp. 27–33.

[12] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Information centric networking in iot scenarios: The case of a smart home," in *Communications (ICC), 2015 IEEE International Conference on*, June 2015, pp. 648–653.

[13] J. Burke, P. Gasti, N. Nathan, and G. Tsudik, "Secure sensing over named data networking," in *Network Computing and Applications (NCA), 2014 IEEE 13th International Symposium on*, Aug 2014, pp. 175–180.

[14] S. Li, Y. Zhang, D. Raychaudhuri, and R. Ravindran, "A comparative study of mobilityfirst and ndn based icn-iot architectures," in *Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine), 2014 10th International Conference on*, Aug 2014, pp. 158–163.