# IoT Middleware over Information-Centric Network

Sugang Li*, Yanyong Zhang*, Dipankar Raychaudhuri*, Ravishankar Ravindran†, Qingji Zheng†, Guoqiang Wang†
*WINLAB, Rutgers University, North Brunswick, NJ, USA
†Huawei Research Center, Santa Clara, CA, USA

***Abstract*—To build a unified IoT platform in which physical and digital objects can be accessible by application across different organization and domain, many state of art approaches are based on IP-overlay architecture. These solutions inherit the constrain of the current internet, especially in terms of naming, heterogeneity, mobility and security. Thus, we propose a new Information-Centric Network (ICN) based IoT middleware to address these challenges. In this paper, we introduce detailed protocol design for the functions in the middleware and evaluate the efficiency of service discovery.**

## I. Introduction

For many years, many standalone Internet of Things (IoT) platforms have been developed and deployed in different domains. The recent trend, however, is to evolve towards a globally unified IoT platform, in which billions of objects connect to the Internet, available for interactions among themselves, as well as interactions with many different applications across boundaries of administration and domains.

Building a unified IoT platform, however, poses a set of unique challenges and requirements on the underlying network and systems.

- **Identity**: To realize a unified IoT platform, the first step is the capability to assign that are unique within the scope and lifetime of each device, data items generated by these devices, or a group of devices towards a common objective.
- **Heterogeneity**: IoT devices will have heterogeneous means of connecting to the Internet, and often have severe resource constraints, e.g., constrained resources in power, computing, storage, bandwidth.
- **Mobility**: In some scenario, the location of data producer changes by time, and it is not able to provide reliable connection with data consumer. Thus, mobility requirement for a IoT platform is to be capable to deliver IoT data below an application acceptable delay .
- **Security**: The heterogenity and openness of the IoT environment imposes great attack surface towards the IoT devices and service. Without carefully considering security concerns such as identity authentication, data integrity and privacy, any IoT platform is hard to be adopted by either end-users or enterprises.

### A. ICN-IoT Middleware

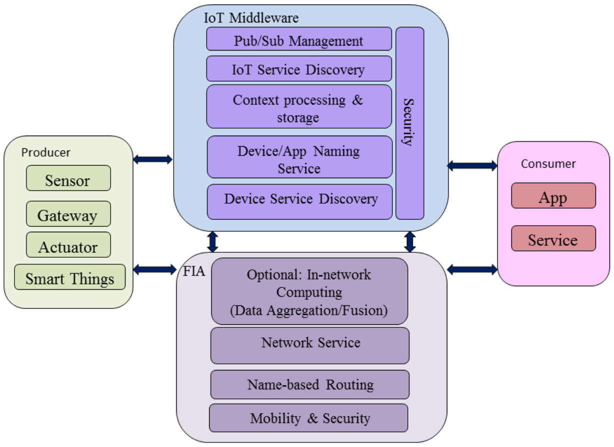


**Fig. 1:** ICN-IoT middleware functionality break down.

Current approaches towards a unified IoT platform are mostly based upon Internet overlays, whose inherent inefficiencies hinders the platform from satisfying the challenge outlined earlier. In recent years, in order to address the inefficiencies of today's Internet, Information-Centric Network (ICN) has been proposed. ICN identifies a network object (including a mobile device, content, or service) by application-centric name instead of its IP address, and adopts a name-based routing, lending itself to supporting the unified IoT platform. In this paper, we propose propose a unified IoT middleware which can work properly with various ICN architecture.Shown as Figure 1, the ICN-IoT middleware mainly consists of publish/subscribe management, device/service discovery, context context processing, naming service and discovery service, where the first one is a centralized service while others are implemented in a distributed manner (discussed in Section III-A). Within the platform, security component provides necessary security guarantee for all services. It is worth noting that functions in the middleware are required to comply with the IoT deployment requirements and protocol agnostic, and therefore the instantiation of the ICN-IoT middleware may differ depending on the underlying ICN protocol. To be specific, in this paper we will discuss there IoT functions with respect to the hierarchical name-based NDN and flat name-based MobilityFirst.

### B. Name Data Networking (NDN)

NDN [14] provides a receiver-driven architecture by transmitting two types of packets : Interest and Data. Consumer issues Interest of the hierarchical content name and forwards it to Data Producer. When the interest arrives at some node (either intermediate router or Data producer) that owns the requested content, the node will reply the requested content(in the form of Data Packet) along the reverse path to the Data Consumer. NDN provides three types of key components to support such functionality. Forwarding Information Base (FIB) collects the forwarding information at each NDN node. The content prefix and out-face mapping is recorded in the FIB. Once a certain face receives Data packet for a Pending Interest Table , it could be added into the FIB to indicate a healthy data retrieving path.Pending Interest Table (PIT) is a data structure that maintain the set of pending Interest routed by the node that expecting Data packet in return. PIT entry is created by receiving Interest, and filters the redundant Interest that carried the same content name. Due to the limitation of PIT size, every PIT entry comes with a timeout which is estimated based on a round trip time. Once a matching Data packet being received by the node, it is forwarded via the face recorded in the PIT before this entry being removed. Content Store(CS) is a data cache for storing Data packet for matching Interest along the reverse path. Also, due to the constraint of local CS size, Data Producer can define the freshness of the Data packet in order to timeout it in the cache. With such transmission mechanism, NDN eliminates the notion of source and destination as it is in IP network, and handle routing only for Interest Packet.

### C. *MobilityFirst Background (MF)*

MF [11] utilizes GUID to name every network object, while separating this GUID from its actual network address. This identifier(GUID)/locator(NA) split design allows MF supporting dynamic address binding, multiple addressing binding and late binding. Shown as Figure 2, MF core network architecture includes the following network component.

**Global Name Resolution Service(GNRS)**: GNRS is a centralized service that maintains mappings between GUID and network address. MF routers create the entries by performing an Insert for the GUIDs of attached network devices and the associated network address, and query GNRS for a translation from GUID to latest binding network address. Recent works shows that this translation performance is much better (50-100 ms delay) than DNS resolution [13].

**Hybrid GUID/NA address routing**: Each of the router in MF can make routing decision based on NA or GUID in the header of data packet, since routing decision are made on a hop-by-hop manner [8].

**Delay-Tolerant Network(DTN)**: The storage in each MF router provides the capability of caching the data packe. Hence, data can be stored or forwarded based on different routing policies, such as quality of the link, which is very useful over wireless interface.

**MF multicast**: MF multicast is based on the idea of group GUID that gathers multiple network objects into one entity. Group GUID to object GUID mapping is a one-to-many mapping that being maintained at GNRS server. Network object can claim to join the multicast group, and either edge router or a centralized management service will perform insertion of the object GUID to group GUID mapping to into GNRS server.

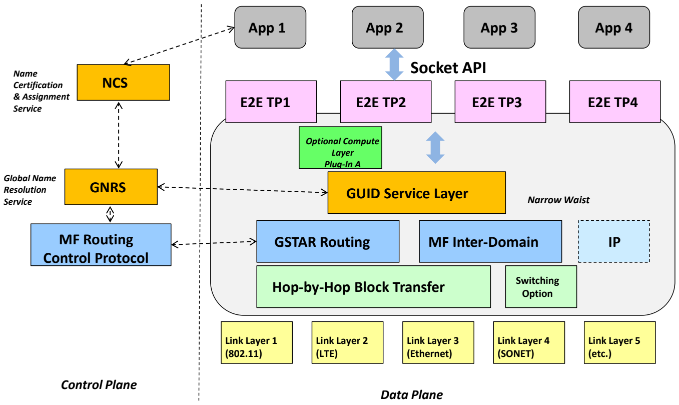
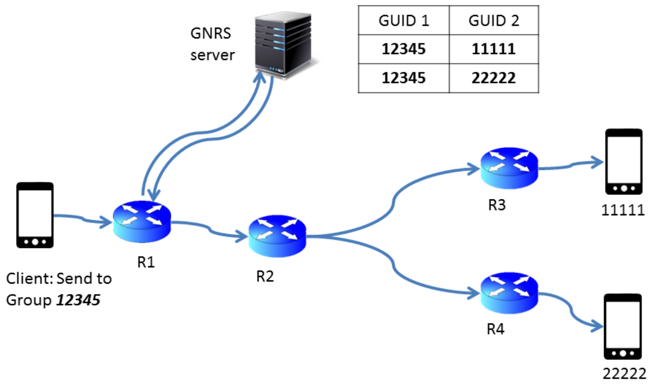


**Fig. 2:** Mobilityfirst architecture.



| GUID 1 | GUID 2 |
|---|---|
| 12345 | 11111 |
| 12345 | 22222 |

**Fig. 3:** MF multicast example: A client sends a message to Group 12345

When the first router queries GNRS server for a group GUID mapping, GNRS server returns a list of member GUIDs. The router assembles these GUIDs into the header and performs ”Longest Common Path(LCP)” algorithm to determine the next hop address.Shown as Figure 3, routers look up the routing table for these GUIDs, and check if there are multiple next hops. When the number of result is more than one, the router reassembles the header and forwards the copied packet to corresponding interfaces. If the number of multicast group member become significant, another approach can be adopted to resolve this issue–instead of assembling all GUIDs into the header after one query, each router on the path queries GNRS server for the group members and perform LCP to decide next hop.

## II. Related Work

ICN is a promising architecture for a general purpose future Internet, including Internet of Things. Previous works by Zhang Y. et al [15] and Li S. et al [7] states the benefits that ICN network is capable to bring to IoT, including efficient data retrieval, well support of mobility, naming, scalability and security.

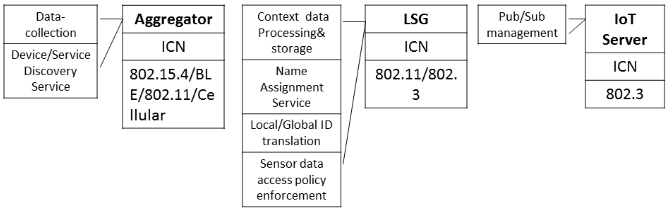

Data-collection
Device/Service Discovery Service
Aggregator
ICN
802.15.4/BLE/802.11/Cellular
Context data Processing& storage
Name Assignment Service
Local/Global ID translation
Sensor data access policy enforcement
LSG
ICN
802.11/802.3
Pub/Sub management
IoT Server
ICN
802.3

**Fig. 4:** Physical components of ICN-IoT middleware

Some early works have demonstrated the feasibility of ICN on particular IoT application requirements .

Study by Amadeo et al. [3] provides a solution on efficient multi-source data retrieval. It proposes a method to use one Interest to retrieve multiple data with different suffixes on different location by adjusting the traditional NDN protocol to support ”prefix-based” interest. For example, a subscriber is interested in the resource named ”/office/temperature”, which is the information from multiple sensor deployed in his office. At the same time, temperature sensors should advertise their resource with this prefix and their own suffix(e.g sensor in the conference room should be named ”/office/temperature/conferenceroom”). Hence, the subscriber can issue single interest to retrieve multiple resource.

George et al. [10] propose a Publish-Subscribe Internetworking ICN architecture where identifier can be used to represent a thing, an application, a group of similar, or any contextual specific entity in the network. A Rendezvous Node functions as a middle box where advertisement from publisher and subscription request from subscriber can meet up to form a membership.

Many studies take one step further to implement IoT system over ICN architecture. In the study by Shang et al. [12], a practical use case that integrates NDN with Building Management System (BMS) is represented. It shows that human-readable hierarchical naming scheme brings convenience in configuring and managing large number of BACnet devices. Baccelli et al. [4] report a CCN experiment over a life-size IoT system. They port light-weight CCN-lite code over RIOT operating system. Based on this platform, performance of various routing protocol and impact of caching have been measured.

## III. ICN IoT Middleware System Design

In this section, we present the overall IoT middleware system design which consists of three distributed physical components (shown in Figure 4) – Aggregator, Local Service Gateway, and IoT server, and is composed of four principle functionalities, including device discovery, device naming service, IoT service discovery and Pub/Sub management. In contrast to centralized or overlay functionality implementation in the legay IP-based IoT platform, our ICN-IoT middleware architecture pushes these functionalities down to the distributed compoents that not only can form a self-configurating subsystem to provide local service but also can act as a complete system to achieve global communcation.

### *A. Physical Components*

- *Aggregator*: It is the base-line component in our middlware architecture that interconnects various IoT service in the local network. An Aggregator usually plays two roles: one is to act as the gateway to bridge the communcation between other devices and resource-contraint wireless sensore and the other one is to integrate sensing/actuating service.
- *Local Service Gateway (LSG)*: It connects local IoT system to the outside world, handes local name assignment and enforces data access policy for local IoT devices. In addition, it can run context data processing service to publish only the information abstraction to IoT server.
- *IoT Server*: It is a centralized server that maintains subscription membership and provides lookup service for the subscribers. Unlike legacy IoT server that involves in the data path from publisher to subscriber, raising the concern of bandwidth of its interfaces being a bottleneck, IoT server in our architecture only involves in the control path where publishers and subscribers exchange their names and certificates.

In the following subsections, we will discuss the breakdown functionality for each physical component and demonstrate detail protocol design.

### *B. Device Discovery*

The objective of device discovery is to contextually establish relationship for nodes in proximity and remote. Device discovery is a key component of any IoT system, and can be considerably simplified by the ICN network. In today’s IoT systems, IP overlay device discovery module focuses on not only the reach-ability of a device, but also the device physical properties [2], [1]. Moreover, a translation service is required to maintain the mapping from network addresses to physical attributes and manual configuration is necessary. ICN-IoT device discovery, however, does not involve any manual configuration or name traslation because ICN uses names to discover new devices. In what follows, we explain the ICN-IoT device discovery in detail, including both devices that are able to run full-stack protocols (referred to as *resource-rich sensors*) and devices that are unable to do so (referred to as *resource-constrained sensors*).

**Resource-rich sensors**: Many resource-rich sensors come with manufacture secure ID and model name, which need to be exposed to both the aggregator and LSG to facilitate device discovery. This aim can be achieved by different ways with respect to NDN and MF. In NDN, this process is initiated by the configuration service running on LSG, which periodically broadcasts discovery Interests (using the name $/iot/model$). The new sensor replies to the discovery interest with its information, and the configuration service then registers the sensor and generates a local ICN name for the sensor. In MF, we can set the model number, $group\text{-}GUID$ as the destination address, and configuration service issues the request via multicasting. When receiving such request, the new device replies with the manufacture secure ID, and the configuration service registers the device and generates a local ICN name to it.

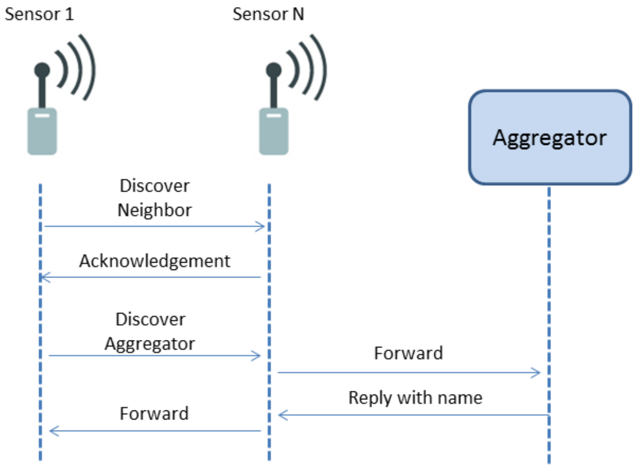


**Fig. 5:** Device discovery for resource-constrained sensor

**Resource-constrained sensors**: Many resource-constrained sensors connect to the Internet using the IEEE 802.15.4 technology via a border router [6]. These sensors voluntarily discover each other in proximity and establish forwarding paths to the border router through self-organizing into a mesh network. Similarly, for our ICN-IoT middleware, Aggregator acts as a border router, and neighbor discovery among sensors can be achieved differently with respect to NDN and MF as follows. In NDN, when a new sensor arrives, it broadcasts a neighbor discovery message to connect with neighbors. After establishing connectivity with neighboring sensors, it broadcasts an Interest named $/ndn/aggregatorservice$ to discover the Aggregator. In MF, neighbor discovery is naturally supported, and the new sensor simply needs to send a discovery request to a well-known broadcast GUID to discover the Aggregator. The above process is shown as Figure 5

### C. IoT Service Discovery

IoT services can be generally categorized into two classes: sensing and actuating. In today's IoT platforms, sensors are connected via a server, which requires considerable development effort such as maintaining a local name to IP mapping. In ICN-IoT, IoT service discovery and configuration becomes more efficient. Specifically, we consider two service discovery modes: $peer\text{-}to\text{-}peer$ and $master\text{-}slave$.

**Peer-to-peer Service Discovery**: In this mode, we consider the scenario in which an aggregator (referred to as the source aggregator) tries to discover an IoT service provided by another peer aggregator (referred to as destination aggregator). In NDN, the source aggregator broadcasts an interest using the well-known name $/area/servicename/certificate$, which will eventually reach the destination aggregator. In MF, this is handled by multicast (discussed in Section I-C) instead of flooding the entire local network. All sensors that provide relevant services can join the multicast group identified by a $Group\text{-}GUID$. After establishing the multicast group, the source aggregator sends a request containing the service name and certificate to the multicast group.The destination aggregator that hosts the service checks the certificate and registers the source Aggregator if there is a matched service. It replies with an acknowledgement containing its certificate to the source aggregator. The message flow is shown as Figure 6 In an example of NDN smart home, a thermostat expresses a request to discover a AC service using well-known name $/home/ac/certificate$ via broadcast channel. In MF case, a multicast group GUID $1234$ can be assigned to all home appliance IoT service. The thermostat sends request containing the service name and certificate to $1234$. In both cases, the AC hosting this services replies with acknowledgement if all conditions match.

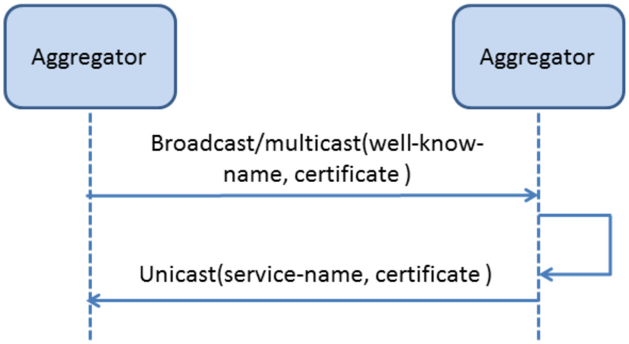

Aggregator
Aggregator
Broadcast/multicast(well-know-name, certificate )
Unicast(service-name, certificate )

**Fig. 6:** Peer-to-peer service discovery

**Master-slave Service Discovery**: In some IoT applications, there are more than one sensing service or actuating service connecting to one control service. A source aggregator hosting control service expresses a request with service name and its certificate to discover a service from all available aggregators. The destination aggregators verify the certificate. If valid, the destination aggregators register the source aggregator and answer with acknowledgement containing their certificates if there is a matched service.

In the AC control NDN example, the number of AC service provider increases to three, which can be named as $/office/ac/1$, $/office/ac/2$ and $/office/ac/3$ . The thermostat expresses a partial name $/office/ac/certificate$ via a broadcast channel periodically. In the MF case, a multicast group GUID $1234$ can be assigned to identify AC service. The thermostat sends request with service name and certificate to $1234$. The destination ACs reply with their certificate if the request meets all criteria.

### D. Secure Device Discovery, Naming Service and Service Discovery

Security guarantee is always treated as an essential function for the IoT middleware. Generally speaking, the security objective is to assure that the device that connects to the network should be authenticated, the provided services are authenticated and the data generated (through sensing or actuating) by both devices and services can be authenticated and kept privacy (if needed). To be specific, we consider the approach to secure device discovery, naming service and service discovery, because other services, such as pub/sub management and context processing & storage, can be properly secured according to application-specific demands.

Recall that our goal is to assure that either device or service itself is authenticated, attemptting to prevent sybil (or spoofing) attack [9] and the assigned name is closely binding to the device (or service). In what follows, let us consider how to secure device discovery and name assignment. Suppose that the device (i.e., sensor) can be either programmable so that before deployment the owner can preload some identity information (such as secure ID, a pair of public/private key and a certificate) , or has some manufacture ID and a pair of public/private key (which is certified by the manufacturer). That is, the device is associated with information including device identity, public/private keys ($PK_{device}, SK_{device}$) and a certificate either from the owner or the manufacturer which certifies the device identity and public/private keys. For the ICN-IoT middleware on top of NDN, when discovering a device, the aggregator will first verify the device identity (e.g., the device can generate a signature with the private key $SK_{device}$ and present the signature and the certificate to the aggregator so that the aggregator can verify it), and then assign a name to the device as follows: the aggregator will issues a request to LSG together with its device identity and $PK_{device}$, so that LSG can assign an ICN name and generate a certificate (certifying the binding of ICN name, $PK_{device}$). To this end, the ICN name and the certificate will be sent back to the aggregator and will be stored locally if the device is resource-restricted. Otherwise, the ICN name and the certificate will be passed to the device. For the MF-IoT, assigning a network name (GUID) for a device is rather straightforward: after verifying the device identity, the Aggregator inserts the public key $PK_{device}$ and device information to the upper layer component to verify if there is a conflict in the corresponding scope. Specifically, LSG is in charge of local scope and IoT server guarantees the global uniqueness. Finally, the unique public key is used a GUID for the new device. Analogously, service discovery can be secured in a similar way.

Note that the above discussion assumes that there exists public key infrastructure (a centralized and hierarchical system) , which greatly simplify the system trust model. In some ad hoc network this assumption may not hold and some other trust mechanisms may need to be employed, such as PGP. Moreover, in order to comply with the capability of resourced-restricted devices, light-weight cryptographic primitive (such as symmetric cryptography) may be used instead of public key cryptography.

### *E. Publish/Subscribe Management*

Data Publish/Subscribe (Pub/Sub) is an important function for ICN-IoT, and is responsible for resource sharing and management. In conventional IP network, most of the IoT platforms provide a centralized server to aggregate all IoT service and data. While this centralized architecture ensures the availability, but it poorly supports scalability and high bandwidth consumption due to high volume of control and data exchange. Thus, we consider a decentralized pub/sub model in ICN-IoT middleware. Specifically, we will discuss rendezvous mode and a data-control separated mode.

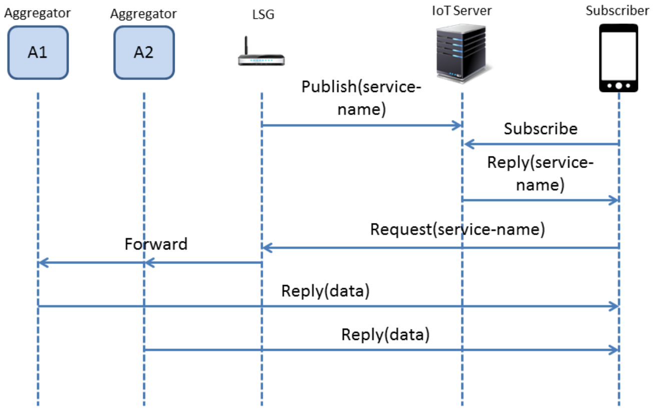


**Fig. 7:** Publish/subscriber management message flow

**Rendezvous Mode**: Rendezvous mode is a classical pub/sub scheme in which data and request meet at an intermediate node. While NDN is a Pull-based architecture without supporting the Pub/Sub mode naturally, COPSS [5] proposes a solution to fix this problem. It integrates a push based multicast feature with the pull based NDN architecture at the network layer by introducing Rendezvous Node(RN). RN is a logical entity that resides in NDN. The publisher first forwards a Content Descriptor (CD) as a snapshot to the RN. RN maintains a subscription table, and receives the Subscription message from subscriber. The data publisher just sends the content using Publish packet by looking up FIB instead of PIT. If the same content prefix is requested by multiple subscribers, RN will deliver one copy of content downstream, which reduces the bandwidth consumption substatially.

**Data-control separated Mode**:Compared with Rendezvous mode in which data plane and control plane both reside on the same ICN network layer, we consider an architecture where the control message is handled by the centralized server while data is handled by ICN network layer. Following the naming process mentioned above, the LSG has the ICN name for the local resource which is available for publishing on IoT server. IoT server maintains the subscription membership, and receives subscription requests from subscribers. Since the subscribers has non knowlege about the number of resource providers and their identities in a dynamic scenario, IoT server has to take responsibility of grouping and assigning group name for the resource. The generic message flow for publish/subscribe function is shown in Figure 7, and we will illustrate the detail in each individual architecture. In NDN, the grouping resource is intuitive : all resource have the same schematic prefix will be grouped together, and the prefix can be used to identify the service. The traditional NDN dose not support using a common partial name to retrieve multiple resource, but M. Amadeo et al. [3] provides a solution to resolve this issue by

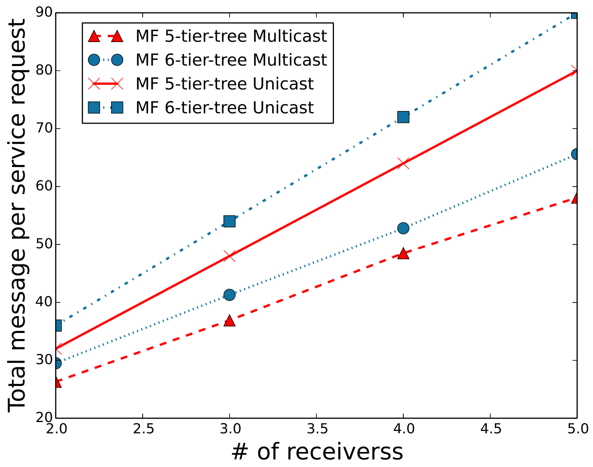


**Fig. 8:** Overhead of service discovery

introducing long-life multi-source Interest in PIT. MF takes advantage of *Group-GUID* to identify a service provided by multiple resources. This *Group-GUID* will be distributed to the subscriber as well as the publisher. In an example of NDN, it uses the common prefix$/home/monitoring/$ to identify a group of resource that provides multiple monitoring services such as $/home/monitoring/temperature$ and $/home/monitoring/light$. The subscriber retrieves the prefix from the IoT server, and sends Interest toward the resource. In a MF example, $GUID$-$x$ identifies the "home monitoring" service that combines with "light status" and "temperature". The resource producers, i.e. the host of "temperature" and the host of "light status" are notified that their services belong to $GUID$-$x$, then listen on $GUID$-$x$. The subscriber sends the request containing $GUID$-$x$ through multicasting which ultimately reach the producers at the last common node. Once receiving the request, the resource producer unicasts the data to the subscriber. In addition, if multiple resource consumers subscribe to the same resource, the idea of *Group-GUID* can be reused here to group the consumers to further save bandwidth using multicast.

## IV. Evaluation

In this section, we investigate the efficiency of service discovery based on MF multicast and MF unicast. We have conducted a detailed experiment in NS3-based MF simulator. MFnetwork protocol is implemented over standard NS3 P2P module and the experiments are running in a tree-based topology. In MF multicast case, $GroupGUID$ is used to identify a service at unknown location, i.e., service host (SH) can be at any leaves of the tree while service requester (SR) attaches to the root of the tree. SR issue a request and the first router then perform GNRS lookup for the $GroupGUID$. On receiving of the service request message, the matched receivers will reply with a acknowledgement to the service requester via unicast. While in MF unicast case, we configures the SR to issue request periodically and each matched SH will reply the request sequentially. Figure 8 shows the overhead in terms of total message per service request. We observe that MF multicast introduce less overhead than MF unicast, as it triggers only one GNRS lookup per request, and the routers only forwards request duplicates based on the next common hop(s). Note that response message is delivered via MF unicast in both cases, hence higher efficiency gain can be achieved if multiple SRs send request for the same service and the response are delivered via MF multicast.

## V. Conclusion

In this paper, we present the feasibility of a ICN-based middleware to support various IoT application, and define detailed protocol design for the funtionalities of the middleware. Through detailed simulation, we find that service discovery over MF multicast introduce less overhead than MF unicast if multiple service hosts belong to the same service group. To realized a full-fledged ICN-IoT middleware, our next step will involve prototyping the platform in a real enviroment and develop a naming service for both hierarchical and flat naming scheme.

## References

[1] "Alljoyn," https://allseenalliance.org/developers/learn, 2014.

[2] "Iotivity," https://www.iotivity.org/about, 2014.

[3] M. Amadeo, C. Campolo, and A. Molinaro, "Multi-source data retrieval in iot via named data networking," in *Proceedings of the 1st international conference on Information-centric networking*. ACM, 2014, pp. 67–76.

[4] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information centric networking in the iot: experiments with ndn in the wild," in *Proceedings of the 1st international conference on Information-centric networking*. ACM, 2014, pp. 77–86.

[5] J. Chen, M. Arumaithurai, L. Jiao, X. Fu, and K. Ramakrishnan, "Copss: An efficient content oriented publish/subscribe system," in *Seventh ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), 2011.*

[6] J. W. Hui and D. E. Culler, "Ip is dead, long live ip for wireless sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*. ACM, 2008, pp. 15–28.

[7] 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*. IEEE, 2014, pp. 158–163.

[8] S. C. Nelson, G. Bhanage, and D. Raychaudhuri, "Gstar: generalized storage-aware routing for mobilityfirst in the future mobile internet," in *Proceedings of the sixth international workshop on MobiArch*. ACM, 2011, pp. 19–24.

[9] J. Newsome, E. Shi, D. X. Song, and A. Perrig, "The sybil attack in sensor networks: analysis & defenses," in *Proceedings of the Third International Symposium on Information Processing in Sensor Networks, IPSN 2004, Berkeley, California, USA, April 26-27, 2004*, 2004, pp. 259–268.

[10] G. C. Polyzos and N. Fotiou, "Building a reliable internet of things using information-centric networking," *Journal of Reliable Intelligent Environments*, pp. 1–12, 2015.

[11] D. Raychaudhuri, K. Nagaraja, and A. Venkataramani, "Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet," *ACM SIGMOBILE Mobile Computing and Communications Review*, 2012.

[12] W. Shang, Q. Ding, A. Marianantoni, J. Burke, and L. Zhang, "Securing building management systems using named data networking," *Network, IEEE*, vol. 28, no. 3, pp. 50–56, 2014.

[13] T. Vu, A. Baid, Y. Zhang, T. D. Nguyen, J. Fukuyama, R. P. Martin, and D. Raychaudhuri, "Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet," in *IEEE 32nd International Conference on Distributed Computing Systems (ICDCS), 2012*.

[14] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[15] Y. Zhang, D. Raychadhuri, R. Ravindran, and G. Wang, "Icn based architecture for iot - requirements and challenges," IETF Internet Draft draft-zhang-iot-icn-architecture-00. IRTF, Tech. Rep., 2013.