iHEMS: An Information-Centric Approach to Secure Home Energy Management

Jianqing Zhang
Intel Labs
jianqing.zhang@intel.com

Qinghua Li
The Pennsylvania State University
qxl118@cse.psu.edu

Eve M. Schooler
Intel Labs
eve.m.schooler@intel.com

***Abstract*—Motivated by the "clean-slate" opportunity of the evolving Smart Grid, we propose an Information Centric Networking (ICN) approach for the home communications fabric and create an ICN-based secure publish-subscribe system to support home energy management. We design a secure group communication protocol with efficient key management specifically embedded in ICN for home data privacy. To validate the approach, we enhance an open-source ICN implementation and build a proof-of-concept Smart Home testbed.**

I. Introduction

With the proliferation of intelligent energy devices in our lives and in turn a tidal wave of energy-relevant data being generated by these devices, there is a heightened need for efficient data sharing, analytics, security and privacy. The Smart Grid presents a unique opportunity to experiment with "clean slate" Internet ideas because its cyber infrastructure is in the early stages of architectural development, particularly at the edges of the network, where it will enable personalized energy management solutions for smart homes, commercial buildings, neighborhoods and cities [12]. Because these networks are smaller-scoped and administratively-independent, there is even greater flexibility to experiment with next-generation Internet ideas and in particular with Information Centric Networking (ICN) [5], [15], [11], [8].

The principal idea behind ICN is to allow applications to ask for data in a content-centric manner regardless of the data's physical location. A data unit is requested, routed and delivered via its name rather than its IP address. It makes the data location transparent to applications and simplifies data access. Secondly, ICN supports distributed content caching in the network. The same named data may get stored at multiple different locations in the network. This improves data availability under conditions of device mobility, intermittent connectivity, and low-power operation. Thirdly, ICN provides self-contained data security. The security of data does not rely on the protection of end-to-end communication channels, as done by IPsec or SSL, but rather the data itself, which upon creation is signed (for integrity and authenticity) and encrypted (for confidentiality). These features strongly echo the requirements for secure, accessible and efficient data sharing in Home Energy Management Systems (HEMS).

In this paper, we propose *i*HEMS, a secure ICN-based publish-subscribe (*pub-sub*) system for home energy management. We adopt ICN as the basis for the home communications fabric and develop a pub-sub substrate to support HEMS applications. To protect data privacy, we propose a secure group communication scheme and an efficient group key management protocol specifically embedded in ICN. To validate the approach, we enhance an open-source ICN implementation and build a proof-of-concept Smart Home testbed. We evaluate the system from a functional perspective and demonstrate it is able to securely support a range of typical HEMS applications.

In the remainder of this paper, we motivate our choice of ICN for HEMS in Section II, present the secure and privacy-preserving pub-sub substrate and its integration with ICN in Section III, and discuss the proof-of-concept implementation, related work and conclusions in Sections IV, V and VI.

II. Home Energy Management and ICN

Home Energy Management Systems (HEMS) provide residents with real-time awareness of energy usage and in turn the potential for improved energy efficiency and a significant impact on energy savings and robustness of the Smart Grid. HEMS perform multiple functions: they collect measurement and status information of household elements, aggregate and analyze the data, and ultimately enable intelligent control decisions for actuation (*e.g.,* put appliances to sleep when unattended or reschedule their usage to a less expensive or less congested time of day). To achieve these functions, HEMS rely on the availability of a communications fabric among the many "devices" that populate a home (sensors, power monitors, computers, handhelds, appliances, *etc*).

A typical ICN system, like Content Centric Networking (CCN) [5], works in a pub-sub manner. A data consumer (subscriber) sends an "Interest" packet with the data name to the network. The Interest is routed based on the data name towards the data source (publisher). On receiving an Interest, the intermediate forwarding nodes will check for a cached copy. In case of a cache hit, a "Data" packet will be returned to the interface from which the Interest came. Each node maintains a Forwarding Information Base (FIB), which is comprised of destinations and forwarding interfaces for Interests, organized for retrieval by longest prefix lookup of the names. In case of a cache miss, the Interest will be passed to the next upstream node according to the FIB. At the same time, an entry is added to a Pending Interest Table (PIT), which indicates particular data has been requested and records the interface where the Interest came. If the node receives multiple Interests for the same missing data, the node does not forward Interests further; instead it only adds a new entry to the PIT. On the other hand, if an intermediate node receives the Data from an upstream node, it will check the PIT and pass the content to the recorded interface. If there are multiple PIT entries for the same data, it passes copies to the corresponding interfaces,

enabling multicast to behave in a copy-and-forward manner. When the content is served, the PIT entries are removed. At the same time, intermediate nodes will cache a copy of the data to serve future requests. To guarantee subscribers get the latest data, the cached data has a header field that indicates the expiration time of the copy. This field is set by the publisher and its value is determined by application requirements.

At creation, data is signed by the owner for integrity and for authenticity and data is encrypted for confidentiality, and so are the copies in the network. As long as an authorized subscriber has the appropriate credentials, it can verify the signature and decrypt the data content, no matter if the data is obtained directly from the owner or an intermediate node on the data forwarding path. Thus, data confidentiality, authenticity and integrity are provided intrinsically in ICN.

We argue that ICN is beneficial to HEMS because ICN capabilities match HEMS requirements very well. First, HEMS devices naturally communicate in a group-oriented pub-sub manner. For example, consider a typical HEMS shown in

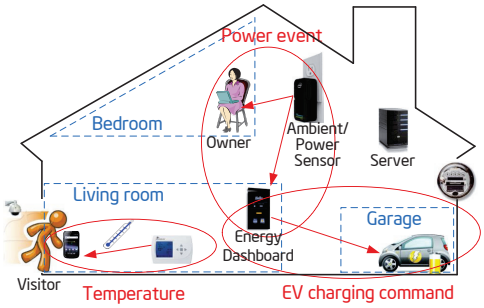

Fig. 1. HEMS Communications

Figure 1. A notable power event, for example that an electric vehicle (EV) is recharging, may be detected by a power monitor and then shared with (published to) an energy dashboard, a smart phone, and a computer. Similarly, the EV might accept (subscribe to) charging instructions from multiple places, *e.g.,* the owner's laptop, phone and car dashboard. Actuation commands also can be delivered in a one-to-many manner. For instance, if an ambient sensor detects no occupancy of a room, the HEMS may turn off some appliances and devices in the room, like lights, cooling fan, a TV monitor. For efficiency a controller could multicast the "turn off" command to all the devices rather than unicast it individually. Thus, devices often interact with energy data as groups. Actually, ICN's "interest-content" model maps naturally to data sharing pub-sub groups, and the entries in FIB and PIT tables map to a group of devices interested in the same piece of data.

Second, HEMS devices expect to access data regardless of where the data is stored and whether or not they and the data sources are continuously connected to the network. In HEMS, devices may be only intermittently connected due to mobility, sleep modes, interference, or being powered off. Temporarily disconnected devices should be able to retrieve data upon reconnection to the network. Also, the data previously generated by a device should be accessible to other devices even if the generating device is currently disconnected. These scenarios can be supported by name-based routing and distributed data caching among groups of devices.

Third, HEMS have a strong need to address security, especially confidentiality (keep data private among those authorized), integrity (data has not been tampered with) and authenticity (data is from who it says it is from). As energy data can be mined for sensitive information (someone is at home or work, away on vacation, has certain eating habits, uses a health device, *etc*.), leakage of such information may compromise privacy. Additionally, tampered or falsified energy data may cause unexpected or even dangerous side effects. For example, an EV's battery may not be recharged enough if the charging instruction is tampered with during transmission. ICN's self-contained data security meets the security requirement and data caching extends this capability if the data is not hosted by the original device. Because different devices should have different access privileges to data, yet most home wireless networks treat all devices equally, a mechanism - such as secure group communication (see Section III-B) - is needed to restrict data access [7] to only a select group of devices.

III. *i*HEMS

*i*HEMS is a secure, ICN-based pub-sub communication infrastructure for HEMS (Figure 2). Conceptually, it consists of

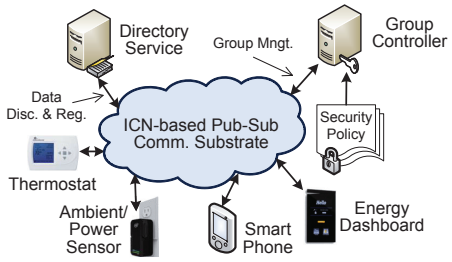

Fig. 2. The Architecture of *i*HEMS.

"devices" found in the home (*e.g.,* power sensors, thermostats, handhelds, display technology such as an energy dashboard), alongside of a *Directory Service* (DS) and a *Group Controller* (GC). Through the DS, a device can publicize its own data and can discover the data to which it wants to subscribe. The DS organizes available data by name. The GC controls data by only issuing group keys to authorized devices for data encryption/decryption according to the security policy. The DS and GC may run in a dedicated server or simply in one of the devices accessible on the network.

The design of *i*HEMS is based on CCN [5], [10]. We follow its URI format and human-readable name convention to create a schema for data in HEMS. An example is "$data/temperature/masterbedroom$", which means the temperature in the master bedroom.

CCN provides two primitives for network nodes to communicate and to exchange data. Both primitives take a data name, dn, as an input. A node uses $\mathbf{Advertise}(dn)$ to publicize its data named dn. At the same time, other nodes update routing information for the data such that they can reach it by dn. If a node wants to find content, it calls $\mathbf{Get}(dn)$, which sends an Interest message requesting the data. Based on the routing, CCN will forward the Interest to a node that has a copy of the data and send the Data to the requesting node.

To illustrate these basic functions, consider a non-intrusive power sensor [12] that detects whole-house power usage events and then disaggregates the signal into individual events such as "Heater ON". It advertises the event data to the HEMS network with the name "$/data/powerevent$" by calling $\mathbf{Advertise}(/data/powerevent)$. Then it listens for Interests for the data. If an energy dashboard wants to display the home power events, it calls $\mathbf{Get}(/data/powerevent)$. An Interest with the name "$/data/powerevent$" will be forwarded to the power sensor. Upon detecting a power event, the power sensor will publish the event in a Data message, which will be eventually delivered to the dashboard and other subscribers.

A. Enhanced Pub-Sub

Although CCN employs a pub-sub paradigm, it is unable to support the richness of the HEMS pub-sub scenarios and data discovery needs. Both motivate us to design a pub-sub layer above it for extended functionality (see Figure 3).

1) Pub-Sub APIs: HEMS applications have a range of pub-sub requirements. For example, a home owner may read the current power usage for a quick check, and the data is delivered only *once*. In a power event monitoring application, the home owner would like to be notified of interesting power events. Therefore, it is desirable to subscribe to the power event data only once and then receive notifications whenever they happen, *i.e.,* the data is delivered many times and the subscription is kept *persistent*. In an ambient sensing application, the energy dashboard may track temperature changes for data analysis. It is inefficient and undesirable to get all updates whenever the thermostat has a new reading. In this case the temperature data is usually sampled and delivered instead at a regular interval like every 10 seconds, *i.e.,* the subscription is *periodic*.

To handle these requirements, we design a set of enhanced APIs for pub-sub communications. The API $\mathbf{Pub}(dn)$ is designed for a publisher to publish data based on $\mathbf{Advertise}(dn)$. The API $\mathbf{Sub}(dn, mode, [interval])$ is used by subscribers, where the parameter $interval$ is optional. It provides three types of subscriptions depending on $mode$: 1) *One-time*. The caller gets the interested data once by calling the CCN primitive $\mathbf{Get}(dn)$; 2) *Periodic*. It enables the caller to periodically get the data by calling $\mathbf{Get}(dn)$ once every period of specified $interval$; 3) *Persistent*. It enables the caller to get the data whenever new data content is available.

One approach to implement persistent subscription is to iteratively call $\mathbf{Get}(dn)$ at the subscriber, which would introduce much communication overhead. It can be improved by extending the CCN protocol to support longer-lived forwarding information (for both Interest and Data packets) in the forwarding nodes and the publisher, so that the subscriber only needs to call $\mathbf{Get}(dn)$ once per time interval t. Upon the first $\mathbf{Get}(dn)$ call, an Interest is passed to the publisher, and the data content is forwarded back. Afterward, the forwarding rules used to forward data dn back to the subscriber are still maintained at the publisher and intermediate nodes for time t, instead of being deleted immediately as in the current CCN. Also, the Interest message is kept alive for time t at the publisher. During this time, if the data content is updated, the publisher directly pushes the content to the network, and eventually to the subscriber. If the time t elapses and the subscriber wants to continue receiving the data, it needs to re-call $\mathbf{Get}(dn)$.

Using a single Interest to enable the forwarding of multiple Data packets (as done for persistent subscription) may affect the traffic control and flow balance built into the CCN model because the PIT state is not consumed by the data packets and is meant to be short-lived. It may introduce a vulnerability to denial of service as well. However, these problems can be mitigated by careful selection of interval t. A shorter t may introduce more overhead caused by Interests but it also causes PIT states on forwarding nodes to be refreshed quickly. The interval t can even be dynamically updated based on traffic load or evidence of attack. More importantly, we argue that such effort is worthwhile because persistent subscription is important for many HEMS applications as well as group key management (see Section III-C).

2) Data Discovery and Registration: In *i*HEMS, the DS supports data discovery by maintaining a list of data names published in the network. The list itself is a type of control data with a uniform CCN name known by all nodes. A node can retrieve the list by name and discover the available data. It also helps requesting nodes find the right group they should join. If a node wants instead to publish data, it registers the data name with the DS, which makes the data discoverable to other nodes and guarantees the uniqueness of the name.

B. Secure Group Communication

To support secure group-oriented pub-sub communication, which is not addressed by CCN, we propose a secure group communication scheme above the pub-sub layer such that only a group of authorized devices are able to access private data. The scheme also enables subscribers to efficiently obtain the data from distributed caches with multicast rather than from the publisher individually. *i*HEMS uses a *group key* to encrypt confidential data. The key is shared among the publisher and subscribers with the appropriate privileges. Since all the authorized subscribers know the group key, data encrypted by the key can be cached in the network and used to serve the request of any subscriber. However, nodes that do not have the key are unable to decrypt and access the data.

Therefore the confidentiality of HEMS data falls to the GC, which manages the group keys for each pub-sub group. Before a node publishes or subscribes to data, it contacts the GC to join the relevant pub-sub group and obtains a group key from the GC. The GC determines if the node is authorized to join the group according to the *security policy*. It only issues the group key if the node has the appropriate publish or subscribe privileges to the data. Due to space, security policy is not addressed in this paper.

Figure 3 displays the secure group communication in the system protocol stack and the data format of the messages.

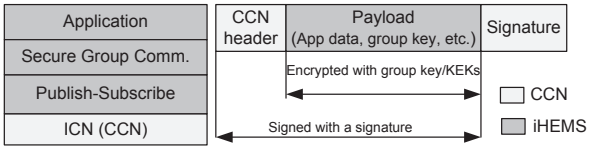

Fig. 3. Protocol Stack and Data Format

The original CCN layer provides integrity and authenticity using digital signatures, and the secure group communication layer proposed in this work provides data confidentiality by encrypting payloads. Usually, application data like a room temperature will be encrypted by a shared group key, whereas the group key management data like a group key distribution message will be encrypted by a key encryption key.

C. Group Key Management

The basic purpose of group key management is to securely distribute group keys to group members. It needs to handle a variety of group events which lead to group key update, including group member join, leave, key refresh and key revocation, *etc*. For consistency and to seamlessly integrate into ICN, group key management is implemented as an ICN-based pub-sub channel with certain well-defined control data.

1) One-way Function Tree Algorithm: One objective for the HEMS group key management algorithm is to strike a balance between security and cost. As for security, the algorithm should have three properties: 1) Forward secrecy. A departing node should not be able to access the group communications afterwards; 2) Backward secrecy. A joining node should not be able to learn the past group communications; 3) Collusion resistance. Even if multiple evicted group members collude, they cannot recover the current group key. As for cost, the algorithm should aim for low computation, storage and communication bandwidth needs, since many HEMS devices are resource-constrained. We choose the One-way Function Tree (OFT) algorithm [14] to satisfy the above requirements.

In OFT a group manager uses a binary tree structure to maintain the group key for each group. In this tree, every node is a key encryption key (KEK) derived from the one-way function of its children (if any), except that the root node is the group key. Each member stores a unique subset of KEKs to derive the group key. When a user joins the group, the group manager updates the key tree and derives a new group key. Then it sends the group key and a proper subset of KEKs to this new user via a secure unicast channel, and multicasts the new group key, which is encrypted with appropriate KEKs known to the other members. Other members can derive the new key based on the KEKs they know. When a user leaves the group, the group manager updates the group key in a similar manner. Since the binary tree is balanced, for a group with n members, each member stores $\lceil \log_2 n \rceil + 1$ KEKs, and key distribution messages contain $\lceil \log_2 n \rceil + 1$ KEKs.

2) ICN-based Group Key Management Protocol: We implement OFT by leveraging the pub-sub layer. The GC is the group manager of the OFT algorithm, and manages group keys for all groups. For each group, the GC maintains the whole key tree, and each publisher or subscriber stores the group key as well as a subset of KEKs. Without loss of generality, we provide an example showing one group that communicates application data D whose name is $./data/D$, and we assume all data names share the same prefix "./".

The group key management protocol consists of several procedures, involving the publication and subscription of certain control data. Table I shows the control data used and Figure 4 illustrates control data flows. Join Request/Response are used for a node to join a group and obtain the group key; Rekey is used by the GC to send a new group key to all group members; Leave Request is used for a group member to leave the group; Sync Request/Response are used for a temporarily disconnected group member to synchronize the group key (*i.e.,* get the group key currently used) after reconnecting to the system. For Join/Leave/Sync Request, group members are the publisher and the GC the subscriber; for Join/Sync Response and Rekey, the roles are reversed.

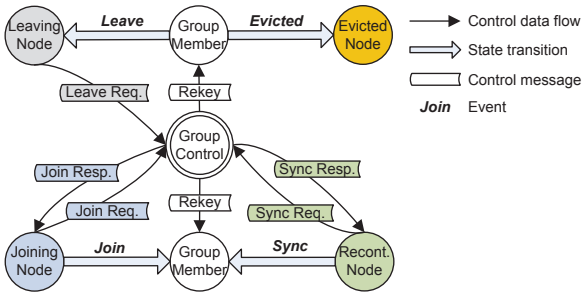

Fig. 4. Group Key Control Data Flow

TABLE I
THE CONTROL DATA OF THE GROUP KEY MANAGEMENT PROTOCOL

Ctrl. Data	Name	Message Content
Join Req.	$./req/group/join$	$\langle nid, op, ./data/D \rangle$
Join Resp.	$./resp/group/join/nid$	$\langle gkey, ver, auxinfo \rangle$
Rekey	$./resp/group/rekey$	$\langle ver,$ *rekey-info* $\rangle$
Leave Req.	$./req/group/leave$	$\langle nid, ./data/D \rangle$
Sync Req.	$./req/group/sync$	$\langle nid, ./data/D,$ $cur\text{-}version \rangle$
Sync Resp.	$./resp/group/sync/nid$	$\langle ver \rangle$, or $\langle gkey, ver, auxinfo \rangle$

Initialization. The GC listens to nodes' requests to join, leave or synchronize keys. Since these requests may come at any time, the GC grants a *persistent* subscription to Join/Leave/Sync Requests, receiving them in a timely manner.

Group Creation and Member Join. The creation of a pub-sub group is triggered by the first request to join the group. Suppose node nid wants to join the group for data D. It publishes control data *Join Request* (see Table I). The message content conveys that node nid wants to access data D with the operation of op, which is either pub or sub. The request will be delivered to the GC, which maintains a persistent subscription to it. If the security policy allows node *nid* to access data *D*, the GC processes the request as follows:

Case 1: Group Creation. Node *nid* is the first group member. The GC creates a new group for data *D*, generates an initial key tree, then adds a virtual node (a process running at the GC for periodic key refresh) and node *nid* to the group successively, and updates the key tree according to the OFT algorithm. The GC sends the group key to the joining node by publishing control data *Join Response*. In the data content, ver means the version number of the group key and $auxinfo$ is the subset of KEKs assigned to node *nid*. The content is encrypted with a pairwise secret key shared between the node and the GC.

The joining node waits to get the response from the GC and then issues a *one-time* subscription to control data *Join Response*. From the received join response, the node obtains the group key as well as the KEKs. Afterward, it issues a

persistent subscription to control data *Rekey* so as to receive future group key updates.

Case 2: Normal Join. Node *nid* is not the first group member. The GC updates the key tree and derives a new group key to ensure backward secrecy. Besides sending the new key to the joining node, it also sends the new key to the existing group members by publishing control data *Rekey*. In the message, *ver* denotes the version number of group key and *rekey-info* can be used by existing members to get the new group key. Since existing members have *persistent* subscriptions to *Rekey*, they receive the rekey data and obtain the new group key.

Member Leave. When a node leaves a group, the group key is updated to ensure forward secrecy. If a member *nid* wants to leave the group for data *D*, it publishes a control data *Leave Request*. When the GC receives the request, it updates the group key and constructs a *Rekey* message and publishes it, such that all other members can learn the new key from the *Rekey* message but not the leaving member.

Member Eviction. Sometimes the GC needs to evict a group member due to some reasons, for example, the node is compromised. This case is similar to a member leave, except the GC updates the group key and publishes a *Rekey* message without receiving a *Leave Request*.

Member Synchronization. A group member may become temporarily disconnected from the network. For example, a wireless device moves out of range of the WLAN or is powered off. When it reconnects, the group key may have been updated. The reconnecting node publishes a *Sync Request* that includes the version number of the latest group key it has. On receiving the request, the GC publishes a *Sync Response*, which includes the latest-version group key and a proper set of KEKs. As with a member join, the member *nid* gets the response by subscribing to *Sync Response* once.

Group Key Refresh. To resist cryptanalysis, a group key should be refreshed periodically. To do so, the GC evicts the virtual node from the group (or makes the virtual node join the group in the next refresh period) and updates the group key accordingly as done for a normal eviction (or join).

D. Discussion

The control data used in group key management is secure. The integrity and authenticity of the data is provided by ICN using digital signatures. As to confidentiality, Join/Sync Response are encrypted with a secret key only known to the joining/synchronizing node and the GC, and thus other nodes cannot learn the group key or KEKs included in the response. The group key included in Rekey is also encrypted with the proper KEKs, such that no parties other than the group members can learn the group key. Note that Join/Leave/Sync Requests do not need to be confidential since they do not leak sensitive information such as data content or a group key.

The overhead of group key management depends on the dynamics of nodes that cause group key management operations, such as group rekey and group key synchronization. There are two types of group dynamics. One case is a group membership change, where a new node is authorized to access the data of a group, or an existing group member is deprived of the privilege. In HEMS, group membership changes only when a device is installed or uninstalled, and when temporary data access privileges are granted to or reclaimed from a resident, a visitor or a device. It is expected that the frequency of group membership changes are low at home. The other type of group dynamics is the temporary disconnection and reconnection of group members, which can be mainly caused by device mobility. In most cases, the frequency of temporary disconnection is also low. Thus, the communication overhead of group key management should be acceptable in HEMS.

We also can observe that persistent subscriptions play a key role in group key management. Group key refreshment due to member leave or eviction are two asynchronous events. Only if the remaining members are notified quickly with the new keys can the security requirements be met. Therefore it is worthwhile to keep Interest and PIT state longer lived as discussed in Section III-A.

As an emerging technology, ICN has many open research problems, like data cache consistency/freshness and security of ICN networks, which are outside the scope of our work.

IV. A Proof-of-Concept System

To validate our approach, we design a proof-of-concept (PoC) home energy management system. The PoC includes an integrated ambient/power sensor [12] to measure temperature, brightness and noise. It is capable of detecting individual power events, such as "Air Conditioner ON" and "Washer OFF" through disaggregation of the whole-house energy signal. The PoC uses four tablets running Ubuntu10.04 with 1.82GHz Atom N470 CPU and 2GB RAM. One works as a proxy to connect the sensor to the home network. Ambient sensing and power events data are encapsulated into *i*HEMS messages and published to the network. Among the other three tablets: one emulates an Energy Dashboard, which receives and displays ambient sensing data and power events from the sensor; one acts as a GC; one emulates a Visitor's mobile device, which requests to join the group and subscribes to ambient data by a one-time or persistent subscription. Because the laptops support emulation, the PoC scales up, with more devices and more sophisticated pub-sub interactions.

All advanced *i*HEMS features, including the secure group communication scheme and the OFT-based group key management protocol, are built into the PoC by extending CCNx-0.4.0, an open source implementation of CCN [10]. Based on the CCNx library primitives, we developed the pub-sub APIs $\mathbf{Pub}(dn)$ and $\mathbf{Sub}(dn, mode, [interval])$ that support one-time, periodic and persistent subscriptions. Currently, persistent subscriptions are implemented by iteratively sending interest messages. All pub-sub messages are carried as SEP2.0 [18] application-level messages, to evaluate the emerging smart energy standard. Using OpenSSL, we employ *blowfish* for encryption and SHA-1 for the one-way function in OFT. We also implemented a simple security policy management module to handle access control and group key distribution. The back end system and the GUIs are implemented in 3400 lines of C code and 1300 lines of Java code respectively.

The PoC enables the evaluation of the system from a functional perspective with a focus on the secure group-oriented publish-subscribe in HEMS. We design experiments to validate native CCN as well as the enhanced secure HEMS

functions. The integrated ambient/energy sensor continuously publishes environmental data like temperature and brightness via the sensing proxy. It also publishes discrete power events once it detects them. The Energy Dashboard *periodically* subscribes to ambient sensing data, and *persistently* subscribes to power events. Most importantly, the PoC also allows scrutiny of security policies for access control. For example, when a Visitor makes a *one-time* content subscription request it actually is transformed into a group join request. The GC assesses if it matches a security policy rule, such as "Allow Visitor to subscribe to temperature". If so, the GC distributes the group key to the Visitor, who is now able to decrypt the data. Otherwise, the request is denied and the Visitor is unable to read the content. Our experiments show that the encryption scheme and group key management protocol can successfully protect confidential data from unauthorized access.

Although CCN has greater header overhead than TCP due to its security annotation, its bulk data transfer efficiency is comparable. According to [5], TCP throughput asymptotes to 90% of the link bandwidth and CCNx to 68%. Experiments also show the performance of secure content transfer of CCN matches unsecured HTTP and substantially outperforms secure HTTPS. As for *i*HEMS, the secure group communication scheme expands the CCN payload by less than 8 bytes for encryption padding. Assuming a smart home with approximately 100 participating nodes, the maximum size of group key management packets in Table I would be 220 bytes, a small burden to current Wi-Fi or Ethernet HANs.

V. Related Work

Gjermundrod *et al* present GridStat, a publish-subscribe middleware system for power grids [4] that aims to address QoS issues in the WAN versus HAN. Zhang and Gunter design a secure multicast scheme for power substations [16] and focus on group establishment and leverages off-the-shelf IPsec technology. Li and Cao [9] proposed an efficient one-time signature scheme for multicast authentication in the Smart Grid. As their scheme does not address data confidentiality it nicely complements our work. Schooler et al. present Trusted Personal Energy Cloud, a secure ICN architecture for personal energy data [13], into which *i*HEMS is intended to fit. Another CCN scheme enables group-based access control [1] leveraging the name hierarchy of content. Targeting distributed file systems, the scheme allows a new group member to read old content published prior to joining, which may not always reflect the typical HEMS usage. Zhang *et al* also study ICN group key management [17] but consider the scenario where the encrypted data cached in the network should be re-encrypted with a new group key when the key is updated. By attribute-based encryption, their approach allows the caching node to re-encrypt the cached data without the updated group key, which is only distributed to group members by unicast. This distribution method is not as efficient or scalable as our work that uses multicast. A recent effort addresses privacy protection in ICN [2]. As it aims to protect the anonymity of user interest rather than data content, it is complementary to our work. Jokar et al. propose a specification-based intrusion detection system for the HAN [6]. However, they focus on physical and medium access control layers, whereas our focus is on upper layers. Feamster proposes to outsource the security management of home networks to a third party [3], but that work does not design any specific communication scheme for HEMS.

VI. Conclusion and Future Work

We developed *i*HEMS, a secure and efficient ICN-based system for home energy management services, after identifying ICN as possessing features well-suited to HEMS and seizing the opportunity to incorporate next-generation Internet advances into the emerging Smart Grid edge architecture. To support the richness of HEMS scenarios and its data discovery needs, we design a pub-sub API on top of ICN. Based on that, we proposed a secure group communication scheme and an efficient group key management protocol to protect data privacy. To both validate and showcase our approach, we enhanced an open-source ICN implementation and built a proof-of-concept Smart Home testbed.

Going forward, we plan to enhance *i*HEMS by improving persistent subscriptions, exploring ICN's benefits to group key management and integrating security policy in ICN systems. In the longer term, we would extend the PoC to study how the architecture scales up for the Smart Neighborhood and Smart City, where there exist more demands on name-based routing, distributed data caching, security and privacy.

References

[1] CCNx Access Control Specifications, July 2010.

[2] S. Arianfar, T. Koponen, B. Raghavan, and S. Shenker. On preserving privacy in content-oriented networks. In *Proc. of the ACM SIGCOMM Workshop on ICN*, 2011.

[3] N. Feamster. Outsourcing home network security. In *Proc. of the ACM SIGCOMM Workshop on Home Networks*, 2010.

[4] H. Gjermundrød, D. Bakken, C. Hauser, and A. Bose. GridStat: A Flexible QoS-Managed Data Dissemination Framework for the Power Grid. *IEEE Trans. on Power Delivery*, 24(1), 2009.

[5] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. L. Braynard. Networking named content. In *Proc. of the ACM CoNEXT*, 2009.

[6] P. Jokar, H. Nicanfar, and V. Leung. Specification-based intrusion detection for HANs in Smart Grids. In *Proc. of the IEEE SmartGridComm*, 2011.

[7] T. H.-J. Kim, L. Bauer, N. James, A. Perrig, and J. Walker. Challenges in access right assignment for secure home networks. In *Proc. of the USENIX HotSec'10*, 2010.

[8] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. Kim, S. Shenker, and I. Stoica. A data-oriented (and beyond) network architecture. In *Proc. of the ACM SIGCOMM*, 2007.

[9] Q. Li and G. Cao. Multicast authentication in the smart grid with one-time signature. *IEEE Transactions on Smart Grid*, 2(4), 2011.

[10] Project CCNx. http://www.ccnx.org.

[11] Pursuing a Pub/Sub Internet. http://www.fp7-pursuit.eu.

[12] J. Rattner. Empowering Personal Energy Management. In *Intel Developer Forum*, April 2010.

[13] E. Schooler, J. Zhang, A. Omotosho, J. McCarthy, M. Zhao, and Q. Li. The trusted personal energy cloud for the smart home. *Intel Technology Journal*, 16(3), 2012.

[14] A. Sherman and D. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Trans. on SW Eng.*, 2003.

[15] D. Trossen, M. Sarela, and K. Sollins. Arguments for an information-centric internetworking architecture. *ACM SIGCOMM Computer Communication Review*, 40(2), 2010.

[16] J. Zhang and C. Gunter. Application-aware secure multicast for power grid communications. In *Proc. of the IEEE SmartGridComm*, 2010.

[17] X. Zhang. Secure mobile virtual group (smvg). In *CCNx Community Meeting*, 2011.

[18] ZigBee Alliance. Smart Energy Profile 2.0, March 2011.