Securing Tags to Control Information Flows within the Internet of Things

Jatinder Singh, Thomas F. J.-M. Pasquier, Jean Bacon
Computer Laboratory, University of Cambridge
Email: firstname.lastname@cl.cam.ac.uk

***Abstract*—To realise the full potential of the Internet of Things (IoT), IoT architectures are moving towards open and dynamic interoperability, as opposed to closed application silos. This is because functionality is realised through the interactions, i.e. *the exchange of data*, between a wide-range of 'things'.**

Data sharing requires management. Towards this, we are exploring distributed, decentralised Information Flow Control (IFC) to enable controlled data flows, end-to-end, according to policy. In this paper we make the case for IFC, as a *data-centric* control mechanism, for securing IoT architectures.

Previous research on IFC focuses on a particular system or application, e.g. within an operating system, with little concern for wide-scale, dynamic systems. To render IFC applicable to IoT, we present a certificate-based model for secure, trustworthy policy specification, that also reflects real-world IoT concerns such as 'thing' ownership. This approach enables decentralised, distributed, verifiable policy specification, crucial for securing the wide-ranging, dynamic interactions of future IoT applications.

Keywords*—*Internet of Things, Information Flow Control, Security, Distributed Systems, PKI, Privacy, Certificates

I. Introduction

Information sharing underpins the broad vision of the "Internet of Things" (IoT). IoT architectures are evolving from application-specific 'silos', where 'things' (see §II) operate for limited, predefined purposes (e.g. within a managed sensor network, monitoring of the elderly, warehouse logistics, etc.), towards those facilitating more open, dynamic interoperability. This is because IoT is driven by 'things' interacting, exchanging data, when and where necessary, in order to realise some required functionality. Thus flexible, but strictly controlled sharing is key to realising the full potential of IoT. But if data is to be shared, great attention must be paid to their protection, i.e. that data is shared as specified by their owners.

It follows that mechanisms are required for regulating the *flow of information across the range of 'things'*. Towards this, we are exploring the potential of Decentralised Information Flow Control (IFC), where management policy is encapsulated within tags that are attached to data (and processes). Policy is enforced at each point of flow, in accordance with these tags.

IFC offers much potential for IoT, as it represents a *data-centric approach* to information flow management. However, work on IFC has focused on closed, pre-defined systems environments, for example within the scope of an operating system (OS) [1] or database [2]. To support IoT, flow policy operate across the range of 'things'. By protecting information flows wherever they occur, IFC embodies the nature of IoT interactions, addressing the flow-based security concerns.

In this paper we present initial steps in using IFC to protect IoT information flows. The contributions of this paper are: 1) Making the case for flow-based controls in IoT; 2) Demonstrating a certificate-based model for enabling robust, verifiable data flow policy (i.e. secure tags); 3) Showing how this mechanism aligns with practical IoT considerations, such as ownership, by binding privileges and policies with 'things' and identities (be they individuals or organisations).

II. Background and Architecture

In this section we give some background on the properties of 'things' within the context of a general IoT architecture.

IoT is a broad term, often used in a variety of contexts from low-level radio concerns, sensor networks, pervasive computing, HCI, etc. Our focus is on data-sharing, to support the broader vision of IoT, in which 'things' **interact**, i.e. *exchange data*, to realise particular functionality. As such, we use the term '**thing**' to encompass the whole range of sensors, devices, applications, systems, servers, etc.; anything capable of exchanging data across the Internet.

The IoT landscape also contains *subsystems:* closed and/or self-contained networks of 'things' operating in a particular scope. Subsystems may be application-centric networks, fixed or ad-hoc, such as a smart home, emergency services operating in a catastrophe; or those resource constrained or using a particular technology-stack, e.g. a proprietary sensor network, or industrial control system [3]. We treat a subsystem as a single 'thing'. This is because our focus is on data sharing, where interactions between the subsystem (data in/out) and other 'things' are managed through *gateway* (or hub) components. Conversely, a single device could be considered several 'things'; e.g. a smartphone can host a range of applications capable of direct communication, each of these applications acting as a separate gateway over the phone's sensors.

A. *The nature of 'things'*

As we are concerned with the flow of data, we focus on the points of data exchange between 'things'. This is illustrated in Fig. 1. It is worth noting that cloud services are increasingly playing a role in IoT provisioning; a recent survey showed that 33 out of 38 IoT architectures leveraged the cloud [4]. There are ongoing issues in cloud security [5] and we have recently outlined a number of security issues concerning cloud-supported IoT [3]. We mention this for two reasons. First, IFC offers potential in addressing many of these concerns, be they for IoT, cloud-enabled IoT, or the cloud in general. Secondly, our initial work has focused on IFC for

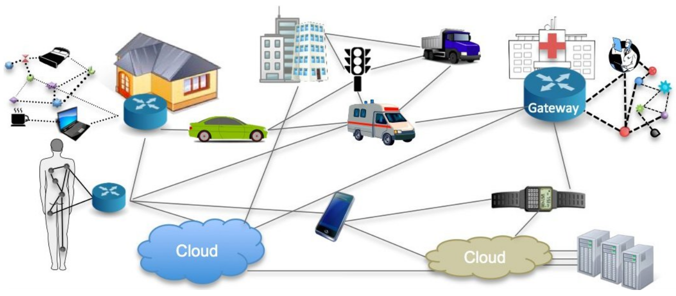

Fig. 1. IoT architecture including cloud services and gateways

cloud computing, which benefits tenants, users and the cloud providers by improving transparency, minimising the potential for data leakage and constraining untrusted applications to run within a secure, cloud-provided IFC context [6]. We use this work as a starting point for exploring IFC for IoT.

B. Data flow concerns

Though data security is a general concern in information systems [7], IoT introduces a number of additional challenges: (1) Much data will be personal, and thus highly sensitive; (2) IoT environments include actuators; an actuation data flow (command) can impact the physical world; (3) There is potential for dynamic, ad-hoc interactions, including between 'things' that have never encountered each other before; (4) IoT functionality typically entails coordination, composition and aggregation, across a range of 'things' (systems, services, etc.), so controls must operate end-to-end, beyond the point of access/production/consumption.

These considerations all concern data sharing, which is unsurprising given that realising the full potential of IoT involves using/reusing 'things', when and where appropriate, to realise new, possibly previously unforeseen functionality, data inference, and services. To date, the bulk of IoT research has focused on particular applications and/or technical constraints, e.g. in terms of protocols, radio and power management, etc. There is comparatively little focus on the security concerns across the range of 'things'. It follows that there is a real requirement for mechanisms enabling *secure, reliable data sharing, enforced end-to-end.*

C. Ownership and control

Ownership is a key factor in considering data management for IoT. This is because it is the owner of a 'thing' (or an authorised delegate) that controls how it may be used, and how/when it may interact with others.

Ownership considerations are naturally reflected in common IoT scenarios. Many IoT applications are described as *user-centric*, where an individual may own a number of devices that provide various services. Consider an individual with a smartphone, which is used to interact with some wearables (for health and fitness), and also their home, perhaps to manage energy usage. The user may also employ some cloud-based applications, perhaps for data aggregation, e.g. collecting a range of physiological data relevant to their health and wellbeing, or for automatic context-driven response, e.g. turning off the heat when they leave their house. The user, as the owner and controller, uses and manages these 'things', including coordinating them when and where necessary, to provide tailored functionality. Of course, such ownership (and thus management) is not restricted to people; organisations might employ 'things' to help with a variety of concerns, e.g. tracking inventory within a warehouse, or delivery logistics.

Ownership (and thus identity) is an important consideration for IoT security, that directly relates to information flow management. This is because those in control of each 'thing' must have policy that authorises any interaction with another. For instance, 'retail theatre' and gallery applications, which might offer location-specific advertisements when in a store, or explain details on nearby exhibits, involve 'things' owned by the store or gallery space, interacting with a device owned by an individual. The information flow policy must accord on both sides of an interaction. There are also interesting cases of shared ownership, e.g. a health authority that loans devices to patients. And of course, rights of ownership, usage and control (including policy specification) may be delegated, e.g. where an organisation gives devices to employees. These issues must be accounted for in the secure information flow model.

D. Certificate infrastructure

A clear challenge is in ensuring that security policy applies reliably and uniformly across the range of 'things', used and re-used in numerous ways. Towards this, a certificate-based model is highly amenable to the constraints we have outlined. A key contribution of this paper is to show how IFC policy (represented in tags) can be securely defined and managed.

For this, we assume the availability of a *Public Key Infrastructure* (PKI) where the owners of 'things' have private keys and *Public Key Certificates* (PKCs), signed by a *Certificate Authority* (CA). In this paper we explore the use of X.509 *Identity Certificates* and *Attribute Certificates* [8], to allow verifiable and trustworthy IFC labelling to support secure flows throughout an IoT architecture.

This approach relies on widely available and well-understood libraries/techniques and requires very little engineering overhead in order to be integrated. Authorisation based on Attribute Certificates has already been shown to work with open source technology such as OpenSSL [9]. We assume the availability of PKI technology as a building block leveraged by 'things'. Gateways or proxies can manage such aspects for resource constrained 'things' (see §II).

III. Why Information Flow Control?

We outline our IFC model, sufficient to understand the remainder of this paper. See [6] for further details of the model and implementation of IFC as a Linux kernel module.

The purpose of IFC is to prevent data leakage by controlling the exchange of information. We follow the classic pattern for IFC-system guaranteed secrecy (*no read up, no write down* [10]) and integrity (*no read down, no write up* [11]).

In IFC, entities, such as processes and data, are labelled with *secrecy* (S) and *integrity* (I) labels. A label comprises a set of tags, each of which represents some security concern such as secrecy:medical or integrity:sanitised. The *security* *context* of an entity is the state of its S and I labels. In this way, tags are used to encapsulate security policy.

A flow of information $A \rightarrow B$ is safe if and only if:

$$A \rightarrow B, \text{ iff } \{S(A) \subseteq S(B) \wedge I(B) \subseteq I(A)\}$$

IFC therefore ensures protected information flow throughout a system, which is crucial for IoT where 'things' are coordinated, (re)used for a variety of purposes. IFC can help ensure IoT system security as follows:

Confidentiality: A person may have sensor devices for health and lifestyle monitoring. The data streams from these sensors can only flow into remote data storage labelled to receive data from that person. This data may augment a conventional health record application, where the records may be stored on a private cloud, accessed via a web portal. Alternatively, a medical professional may be able to directly access a person's physiological sensors (through the respective gateways, if applicable) when in physical proximity, e.g. when a nurse visits the patient's home. A person's medical data may only be accessed for medical research after being anonymised. We assume an anonymisation process is able to tag an output research data set with e.g., secrecy:medical, integrity:anonymised.

Integrity: Sensor data may be tagged as from an approved, standard device, i.e. tagged with an attribute representing the source of data (e.g. belonging to a certain individual). For a smart home or driverless car, only actuation commands based on data approved by the owner are obeyed. This is achieved by tagging sensors, the actuation decision process and the device to be actuated with, e.g., integrity:bob-home.

Protection & Sharing: In a cloud context, data can be protected from leakage and interference by the strong isolation provided by VMs or containers. For IoT in general, including within cloud components, protection with sharing is required that is more subtle and flexible. IFC achieves data sharing by appropriately labelling data and endpoints to allow data flows. Data is protected, as flows are *only* authorised where these explicitly-defined labels agree. The underlying assumption is that IFC enforcement can be trusted; there are hardware-based approaches that assist [12]–[14].

Highly decentralised security policy: In decentralised IFC, new, application-meaningful tags can be created (by potentially anyone) to enforce any required policy, on a totally decentralised basis. There is no need for a central authority. This assists both in sharing data and in breaking application 'silos', both of which are crucial for in moving IoT forward. The assumption is that IoT device owners and management components cooperate in achieving the appropriate data tagging to build new applications and meet their policy requirements.

Increasing trust in cloud-hosted applications, and cloud providers: When cloud services are included in an IoT architecture, the cloud provider is more likely to be trusted to enforce policy than lesser-known applications. In turn, this motivates sharing, as a high-level of trust in applications is no longer needed when behaviour is constrained by IFC policy. Further, much IoT functionality is provided through the coordination of 'things'. Applications (policy engines) facilitating this, causing 'things' to interact when and where appropriate, are highly amenable to a cloud deployment. Given the power of such applications, the trustworthiness of the host is a prime consideration.

Assurance through audit: IFC is a data-centric security policy which can cause data-centric logs to be generated during IFC enforcement. These are more relevant in IoT than legacy system logs [15] (i.e. OS, hypervisors, databases, etc.), as they will offer traces indicating how, when and why 'things' were brought together, and the data exchanges that took place.

Support for compliance: IFC policy is able to capture data-centric legal and regulatory concerns [16]. The logs generated at IFC enforcement points can be used to demonstrate compliance, and invalidate false claims of leakage.

IV. Certificates: Managing ownership and tags

In previous work on IFC for distributed systems [6], [17] tags in inter-machine data exchange were represented by strings. Such tags were typically defined within an application framework and allocated by an application manager on creation of application instances, after authentication of users. Tags are converted between this external representation and bit strings, suitable for kernel implementation, by trusted system processes. In short, tags are application-specific, centrally managed and controlled.

In contrast, an IoT system is by nature decentralised; 'things' are owned and can have owner-initiated as well as more transparent interaction with other 'things', such as the offloading of data for processing and storage. The model of a user initiating authentication with a known service is no longer appropriate. However, it is important that data is protected, and only shared with other 'things' as authorised by its owner.

We believe that a *Public Key Infrastructure* (PKI) [18] and a *Privilege Management Infrastructure* (PMI) [19] can be used to capture the *ownership* of 'things' through *Identity Certificates* (IDCs) and to represent IFC tags through *Tag Certificates* (TCs) respectively. These are shown in Fig. 2.

We use IDCs (Public Key Certificates that tie a *subject*'s identity to a private/public key pair) to associate an IoT 'thing' with the identity of an individual. We discuss this further in §IV-A. An Attribute Certificate (AC) certifies some attribute of the certificate's *holder* (the identity to which the AC is tied) [8], [19], [20]. ACs are bound to IDCs through various techniques described in §IV-B. TCs are ACs that associate an IFC tag with the holder.

A. Identity Certificates (IDCs)

A certificate may be issued and signed by a *Certificate Authority* (CA), or the subject can create a key-pair and self-issue and self-sign a certificate.

If an IDC is self-signed, there is no process for truly validating the identity of the subject, i.e. anyone can claim to be Bob, and one could only trust they are telling the truth. As such, for the wide-spread use of 'things' it is important that IDCs are signed by some sort of authority,[1] one could imagine a commercial CA for this purpose (just as they exist for domain names). For IoT, each owned 'thing' should be

[1]This, of course, depends on the circumstances. One could imagine self-signed ICs functioning appropriately in closed/local environments.

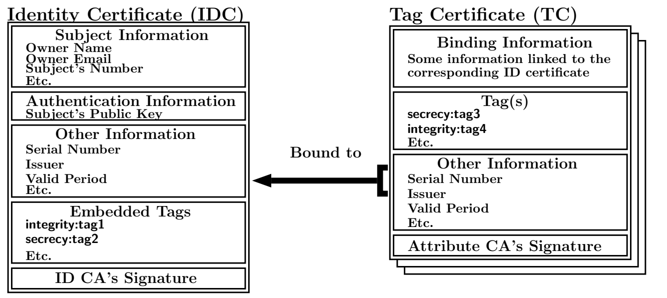

Fig. 2. An Identity Certificate with a Tag Certificate as Attribute Certificate

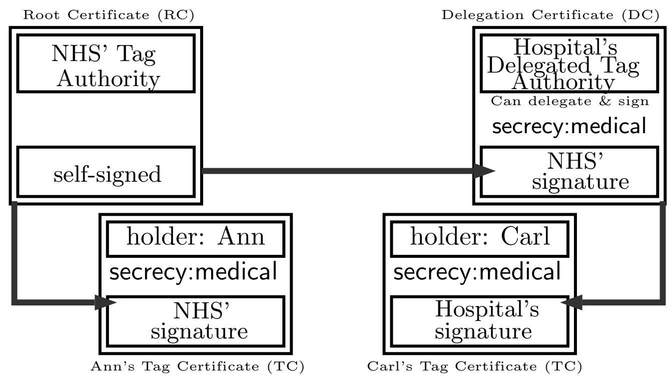

Fig. 3. Two (simplified) Tag Certificates representing the same tag (same name, valid signature chain with the same root)

issued with an IDC indicating the owner (i.e. they are Bob's devices/applications), signed by the same authority. This certificate is used to authenticate and identify the corresponding 'thing' and its relation to the owner.

B. Tag Certificates (TCs)

We define *Tag Certificates* (TCs) as ACs that encode tags, i.e. secrecy and integrity attributes, and are bound to a particular identity. Thus, a TC specifies a set of tags and who may use them. A binding (which defines the holder) may be to the owner's identity, e.g. so the tags apply to that person's 'things'; to another's identity, e.g. another user to allow a third party data access; or to a particular device.

There are several approaches for binding ACs (in our case, TCs) to IDCs (see [20] for a fuller description of each):

Monolithic signature: where a single authority manages both identity and tags. The tags and the IDC are tightly coupled, comprising a single *monolithic block*. Adding or revoking tags requires a new certificate to be issued. This approach is suitable where tags represent an intrinsic characteristic of the principal that is unlikely to change, e.g. a medical sensor generating data tagged as secrecy:medical.

Autonomic signature: where the binding information in the TC directly refers to the IDC certificate's *subject* (the device). The fact that IDCs and TCs are independent adds flexibility to certificate management (revocation, and new assignments of tags to subjects are facilitated). In IDCs, subject information is generally composed of several fields, which together, are guaranteed to uniquely identify the subject, according to PKI policy, as discussed in the IETF specification [8]. A TC is bound to its related IDC by including in it guaranteed unique information fields about the 'thing's' owner. We also assume that *some owner-unique* field(s) may be shared by all the IDC owner's devices, so the same TC can be shared by all devices belonging to that owner. This approach greatly simplifies TC management when a great number of devices are associated with a single owner, for example if Bob wants to tag all his home devices with the same tag secrecy:bob-home.

Chained signature: the TCs are bound to IDCs, using the signature of the IDCs. This creates a 'hard' link between the TC and the IDC, in that a TC only has one IDC and thus revoking the IDC implicitly revokes any TCs. However, it differs from the monolithic approach in that TCs can be independently created/revoked without affecting the IDC. This can be useful in an IoT context in terms of 'thing' management, by issuing/revoking TCs to assign various properties that may change over time, e.g. to specify that Bob's phone has different TCs associated with it when he's at work secrecy:work, so that he can access corporate data only while in the office.

The tags embedded in IDCs and TCs form the S and I labels of their holder: the 'thing', e.g. a device, a mobile app, a cloud service, etc. These labels are used to realise the safe flow constraint described in §III. We describe below (§IV-C) how TCs are distributed to make authorised flows possible.

On establishing a new connection, both parties authenticate each other and agree on a key for a secure channel, as in standard Mutual TLS (MTLS). Then, each side independently makes a decision (see §V) on whether data can flow, i.e. whether the TCs held by the parties obey the safe flow rule.

Importantly, separate TCs can be used to represent a short-lived security property. For example, in case of emergency, Bob (or rather a *break-glass mechanism* operating on his behalf, that automatically issues TCs when certain conditions are met) could issue a short-lived TC for the subject "Emergency Service" for a certain period of time (defined by certificate validity dates), in order for the response team to access Bob's live flow of medical data and cloud-stored historical data. Such a certificate will expire relatively quickly, so that information will not be allowed to flow to the emergency service outside of that time window, thus protecting Bob's privacy under normal circumstances. Note that TCs can be revoked at any time, e.g. in this case to prevent further data flows on a false-alarm.

C. Tag Ownership and Delegation

Conceptually, we define a tag t as a $\{\text{name}, \text{tag-owner}\}$ pair. If a tag $t = t'$, then $\text{name} = \text{name}'$ and $\text{tag-owner} = \text{tag-owner}'$. The tag-owner may or may not be the *holder* of the TC. For example, Bob can issue a TC for Ann's smart-watch allowing the device to receive some of Bob's information flows; though Ann is the holder, Bob remains the tag-owner.

In our model, two tags are said to have the same tag-owner if the signing authority's certificate chains have the same root. This means that it is possible to *delegate* the creation of new TCs by building a valid authority certificate chain. This is illustrated in Fig. 3 where the TCs held by Ann and Carl have the same root, with tag-owner *National Health Service* (NHS). A hospital is trusted and authorised to issue secrecy:medical

TCs for its employees on behalf of the NHS. The Root Certificate is an NHS *self-issued* certificate. The Delegation Certificate (DC), issued to the hospital, is a certificate signed by the Root Authority (the TC's original issuer), that specifies for which tags management is delegated, see Fig. 3.

Verifying the TCs' certification chains, in addition to the standard verification of signatures, consists of verifying that every certificate in the chain has been approved to generate the tag or delegate further as appropriate.

Delegation allows TCs to be created for, and transmitted to, authorised parties, e.g. to allow them to receive data. For example, sensor data may be routinely streamed to cloud services for processing and storage, or temporarily to emergency services, as described above. Further, delegation enables *break-glass* policies, e.g. override policies in emergency situations (by issuing TCs with a well defined lifetime to emergency services when some criteria are met); and also facilitates the management of tags by trusted third parties where appropriate.

Importantly, because TC creation/revocation does not require trusted CAs, it paves the way for decentralised policy specification and management. The approach is in line with the IoT vision, as users are put in control, having the authority to create and define tags to meet their particular security requirements; passing them to others (including 'things') without the involvement of a central trusted third party, except to establish identity when required.

D. Example

Extending the NHS scenario, Bob may have been given IoT devices by the NHS, each of which has secrecy:medical as an embedded tag and the corresponding IDCs. After authenticating with these devices, Bob issues new TCs for them, to ensure that they are now tagged as $S = \{\text{secrecy:medical}, \text{secrecy:bob}\}$. The NHS could also issue a cloud-based application a secrecy:medical TC, to allow the application to receive data flows from patient devices. To allow the application to process Bob's information, he must issue a secrecy:bob TC to the cloud application. This example illustrates flexible, decentralised policy management, where the NHS essentially 'approves' the application by imposing some general data flow constraints, while allowing Bob to define specific policy relating to his personal data.

V. Relation to IFC Enforcement

IFC enforcement mechanisms operate *locally*, to protect the flows within the system. For example, enforcement may occur by an OS running in a server to which 'things' upload data, protecting flows between labelled OS entities (processes, files, sockets, pipes) [6]; or by a gateway that manages IFC on behalf of a subsystem of low-end devices (see §VII).

The focus of this paper is on secure *policy specification* (tagging). The certificate-based approach we describe integrates with these local enforcement mechanisms to enable *cross-system enforcement*. Managing this interplay requires: a) translation between the external, certificate-based representation of IFC labels (the tag sets of S and I labels) and the relevant local representation, during interactions with other 'things'; and b) the proper enforcement of IFC rules within the local system—hardware-based approaches can assist (see §III).

For example, suppose a 'thing' streams out data for processing. The receiver, a cloud-based application running over an IFC enforcement mechanism, must: a) hold the TCs corresponding to the data stream. These may have been previously transmitted or be passed as part of any handshaking/interaction protocol; and b) be assigned the local representation of the S and I labels corresponding to the certificate. Similarly, if the data stream (perhaps in a processed form) is appended to a file, that file must also be labelled with the 'thing's' S and I labels. This is because all data flows must be protected.

VI. Related Work

Much work considers establishing secure communication channels between 'things' [21], [22], rather than on protecting data beyond the channel's endpoints. Some research focuses on lightweight authentication mechanisms [23]–[25]. Again, these do not provide end-to-end security guarantees. Such concerns are orthogonal to our work, and could be leveraged to support lower-end device implementations. Others have designed access control schemes specifically for IoT [26], [27]. These focus on adapting existing techniques to resource constrained devices, rather than on developing new approaches towards the wider IoT requirements.

Henze et al. [28] propose a policy negotiation mechanism enforced between a 'thing' and cloud storage provider. Much like an enforceable SLA, it considers a particular 'thing'-cloud interaction. It does not provide for 'thing'-'thing' interactions, nor does it aim to operate end-to-end.

De Leuss et al. [29] propose a *self-managed cell* that builds on identity federation systems and standard access control mechanisms. The model encourages silo interoperability (i.e. boundary-controls) rather than breaking those silos to enable wide-ranging interactions.

The approaches are typically boundary or one-to-one interaction focused (i.e. authentication, establishing a secure channel, negotiation, access control). To our knowledge, IFC has not been considered for IoT. We believe that by leveraging the mechanism described here we are able to build data- and user-centric policy specification and enforcement mechanisms that address the requirements for building secure, collaborating, decentralised IoT systems.

VII. Conclusions & Future Work

The wider vision of IoT requires support for controlled data sharing via inter-'thing', managed connections. We believe that IFC captures these requirements ideally and we have proposed the use of IFC to manage the flows between 'things'.

Most work in IFC considers policy specification (tagging) in the context of a particular system or application. IoT, however, represents a highly-distributed, dynamic interaction environment. We therefore present a certificate-based approach for secure tagging, to enable robust, verifiable and distributed data flow policy. The approach takes account of practical, real-world IoT considerations, including ownership, delegation, and dynamic (temporal) privilege. The aim is to build on well-understood, widely available security technology.

These contributions are an initial step towards a wider, IFC-enabled IoT. Areas of work we are continuing to address include: **(1)** The potential privacy implications of linking ownership with tags. **(2)** A wider range of enforcement mechanisms: Assuming a secure tag model (presented here), IFC can be enforced across all data flows within OSs [1], web applications [30], and cloud services [6] (see §V). Indeed, this will cover enforcement for many 'things' (e.g. the 'things' with an OS, or running in a cloud). However, work remains in exploring IFC enforcement in gateways, on behalf of low-powered devices. **(3)** Event-driven policy management: Changes in context, e.g. detected by policy engine components monitoring data streams, can effect changes in privileges and connections. The break-glass example presented in §IV-C represents an extreme case. This involves exploring the mechanism by which certificates can be *dynamically created, revoked and distributed*, regulating flows to meet user-goals. **(4)** Issues of shared 'things': We have considered policy management by 'thing' owners and their delegates. In some cases, issues are straightforward (e.g. §IV-D), in others, more care is needed. Consider a 'thing' in a family home; family members may have different policies, that could conflict. Some concerns could be managed through event-driven policy engines (see (2)), e.g. that dynamically change IFC tags based on a person's proximity to a 'thing'. We need to take account of potential policy conflict in certificate management. **(5)** Finer-grained (field-level) flow controls: So far, we have considered protection at the level of a communication stream, via the labels of data and communication end-points. However, more fine-grained policy is also desirable, e.g. to prevent the flow of an address or DoB in a personnel record, or restricting the flow of location data in certain circumstances. Naturally, this entails some form of structured data (e.g. messages), within which individual attributes can be separately protected.

Acknowledgement

This work was supported by the UK Engineering and Physical Sciences Research Council under grant EP/K011510 "CloudSafetyNet: End-to-End Application Security in the Cloud". We acknowledge the support of Microsoft for work on Cloud Law with QMUL colleagues.

References

[1] M. Krohn, A. Yip, M. Brodsky, N. Cliffer, M. F. Kaashoek, E. Kohler, and R. Morris, "Information Flow Control for Standard OS Abstractions," in *Symposium on Operating Systems Principles*. ACM, 2007, pp. 321–334.

[2] D. Schultz and B. Liskov, "IFDB: Decentralized Information Flow Control for Databases," in *European Conference on Computer Systems (Eurosys'13)*. ACM, 2013, pp. 43–56.

[3] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "20 Cloud Security Considerations for Supporting the Internet of Things," *under review*.

[4] J. Mineraud, O. Mazhelis, X. Su, and S. Tarkoma, "A Gap Analysis of Internet-of-Things Platforms," 2015, Arxiv, arXiv:1502.01181 [cs.CY].

[5] S. Subashini and V. Kavitha, "A Survey on Security Issues in Service Delivery Models of Cloud Computing," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1 – 11, 2011.

[6] T. F. J.-M. Pasquier, J. Bacon, and D. Eyers, "FlowK: Information Flow Control for the Cloud," in *6th International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2014.

[7] R. J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd ed. Wiley Publishing, 2008.

[8] S. Farrell and R. Housley, "An Internet Attribute Certificate Profile for Authorization," IETF, Tech. Rep., 2002.

[9] J. A. Montenegro and F. Moya, "A Practical Approach of X. 509 Attribute Certificate Framework as Support to Obtain Privilege Delegation," in *Public Key Infrastructure*. Springer, 2004, pp. 160–172.

[10] D. E. Bell and L. J. LaPadula, "Secure Computer Systems: Mathematical Foundations and Model," The MITRE Corp., Bedford MA, Tech. Rep. M74-244, 1973.

[11] K. J. Biba, "Integrity Considerations for Secure Computer Systems," MITRE Corp., Tech. Rep. ESD-TR 76-372, 1977.

[12] N. Vachharajani, M. Bridges, J. Chang, R. Rangan, G. Ottoni, J. Blome, G. Reis, M. Vachharajani, and D. August, "RIFLE: An Architectural Framework for User-Centric Information-Flow Security," in *37th International Symposium on Microarchitecture*. IEEE, 2004, pp. 243–254.

[13] K. R. Jayaram, D. Safford, U. Sharma, V. Naik, D. Pendarakis, and S. Tao, "Trustworthy Geographically Fenced Hybrid Clouds," in *ACM/IFIP/Usenix Middleware*. ACM, 2014.

[14] "Software Guard Extensions Programming Reference," Intel, Tech. Rep. 329298-001US, 2013. [Online]. Available: https://software.intel.com/sites/default/files/329298-001.pdf

[15] R. K. Ko, M. Kirchberg, and B. S. Lee, "From System-centric to Data-centric Logging-accountability, Trust & Security in Cloud Computing," in *Defense Science Research Conference and Expo (DSR), 2011*. IEEE, 2011, pp. 1–4.

[16] J. Singh, J. Bacon, J. Crowcroft, A. Madhavapeddy, T. Pasquier, W. K. Hon, and C. Millard, "Regional Clouds: Technical Considerations," University of Cambridge, Tech. Rep. UCAM-CL-TR-863, 2014. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-863.pdf

[17] N. Zeldovich, S. Boyd-Wickizer, and D. Mazières, "Securing Distributed Systems with Information Flow Control," in *5th USENIX Symposium on Networked System Design and Implementation*, 2008, pp. 293–308.

[18] D. Solo, R. Housley, and W. Ford, "Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile," 2002.

[19] D. W. Chadwick and A. Otenko, "The PERMIS X. 509 role based privilege management infrastructure," *Future Generation Computer Systems*, vol. 19, no. 2, pp. 277–289, 2003.

[20] J. S. Park and R. Sandhu, "Binding Identities and Attributes Using Digitally Signed Certificates," in *16th Annual Conference on Computer Security Applications*. IEEE, 2000, pp. 120–127.

[21] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security Challenges in the IP-based Internet of Things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, 2011.

[22] O. Garcia-Morchon, S. Kumar, R. Struik, S. Keoh, and R. Hummen, "Security Considerations in the IP-based Internet of Things." IETF, 2013.

[23] J.-Y. Lee, W.-C. Lin, and Y.-H. Huang, "A Lightweight Authentication Protocol for Internet of Things," in *International Symposium on Next-Generation Electronics (ISNE)*. IEEE, 2014, pp. 1–2.

[24] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, "Two-phase Authentication Protocol for Wireless Sensor Networks in Distributed IoT Applications," in *14th Int. Conf. on Wireless Communications and Networking (WCNC)*. IEEE, 2014, pp. 2770–2775.

[25] N. Park, M. Kim, and H.-C. Bang, "Symmetric Key-Based Authentication and the Session Key Agreement Scheme in IoT Environment," in *Computer Science and its Applications*. Springer, 2015, pp. 379–384.

[26] S. W. Oh and H. S. Kim, "Decentralized Access Permission Control Using Resource-oriented Architecture for the Web of Things," in *6th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2014, pp. 749–753.

[27] A. Cherkaoui, L. Bossuet, L. Seitz, G. Selander, and R. Borgaonkar, "New Paradigms for Access Control in Constrained Environments," in *9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC)*. IEEE, 2014, pp. 1–4.

[28] M. Henze, L. Hermerschmidt, D. Kerpen, R. Häußling, B. Rumpe, and K. Wehrle, "User-driven Privacy Enforcement for Cloud-based Services in the Internet of Things," in *2014 International Conference on Future Internet of Things and Cloud (FiCloud)*, 2014.

[29] P. De Leusse, P. Periorellis, T. Dimitrakos, and S. K. Nair, "Self Managed Security Cell, a Security Model for the Internet of Things and Services," in *1st International Conference on Advances in Future Internet*. IEEE, 2009, pp. 47–52.

[30] T. F. J.-M. Pasquier, J. Bacon, and B. Shand, "FlowR: Aspect Oriented Programming for Information Flow Control in Ruby," in *13th International Conference on Modularity*. ACM, 2014.