# Access and Authentication Control for Internet of Things (IoT Network)

*Jay Prakash Kumar[1], Arun Rai[2]*
*[1,2]Department of CSE, Vedica Institute of Technology, RKDF University, Bhopal, M.P*

***Abstract:** This paper introduces the importance of resource and information protection in distributed systems, highlighting the limitations of current authorization frameworks in terms of scalability, manageability, effectiveness, and efficiency. With the rise of the Internet of Things (IoT), there is an increasing demand for solutions that can handle the potentially limitless number of sensors, actuators, resources, services, and subjects, along with the enhanced interaction dynamics these environments entail. The paper proposes a capability-based access control system designed for both enterprises and individuals to manage their access control processes for services and information. This mechanism supports rights delegation and advanced access control customization, and it is being developed as part of the European FP7 IoT@Work project to manage access control for the project's services deployed on the shop floor.*

***Keywords:** Capability Based Access Control, Delegation, Internet of Things, Authorization, Rights Revocation*

## I. INTRODUCTION

In the following sections, we describe the Capability-Based Access Control (CapBAC) system that we are developing within the EU FP7 IoT@Work project for managing access control to some of the project's services.

This authorization approach is based on the capability-based authorization model (sometimes referred to as capability-based security). This model is one of the existing security models, where a capability (known in some systems as a key) is a communicable, unforgeable token of authority. It represents a value that references an object along with an associated set of access rights. A user program must use a capability to access an object. A capability is defined as a protected object which, by virtue of its possession by a user process, grants that process the capability (hence the name) to interact with an object in certain ways. These ways might include reading data associated with an object, modifying the object, executing the object, or other specific actions that the capability allows.

*data in the object as a process, and other conceivable access rights. The capability logically consists of a reference that uniquely identifies a particular object and a set of one or more of these rights*".

In distributed contexts, like SOA ([3], [4], [5], [6], [7]) and Grid computing [8], this model provides many advantages over more consolidated approaches and is gaining attention thanks to its flexibility and greater support for least-privilege operations and for avoiding security issues like the Confused Deputy problem [9].

As depicted in Figure 1., in a CapBAC system it is the user that have to present his/her/its authorization capability (and demonstrate he/she/it is the owner of it) to the service provider, while in a traditional ACL system it is the service provider that has to check if the user is, directly or indirectly (for example via a role owned by the user), authorized to perform the requested operation on the requested resource:

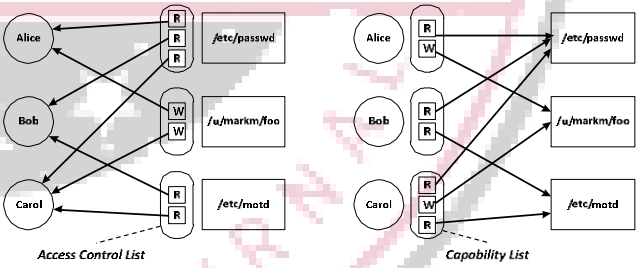


Figure 1. ACL vs Capability-based authorization models.

The CapBAC described in the following borrows ideas and approaches in previous works extending and adapting them to address IoT requirements and specifically the IoT@Work ones.

As compared to the previous approaches, the capability based authorization we are designing provides the following additional features:

- **delegation support**: a subject can grant access rights to another subject, as well as grant the right to further delegate all or part of the granted rights. The delegation depth can be controlled at each stage;
- **capability revocation**: capabilities can be revoked by properly authorized subjects, therefore solving

One of the issues of capability-based approaches in distributed environments is information granularity.

- A capability can specify dynamic adaptation of the granted rights, such as defining a "level of detail" for a read access right on a specific piece of information. This means that the service provider can adjust its behavior and the data it provides based on the specifications within the capability.

This document is organized as follows:

- The structure of the paper is organized as follows:
- **Section II:** Surveys the research activities on access control models, with a particular focus on
- capability-based models.
- **Section III:** Provides a quick overview of the specific issues that IoT contexts present and highlights how a capability-based approach can
- address these IoT access control issues.
- **Section IV:** Describes an application scenario and illustrates how the proposed capability-based
- access control system operates within this scenario.
- **Section V:** Details the functional models
- envisaged by the proposed access control model.
- **Section VI:** Analyzes how the proposed model
- can contribute to enhancing privacy.
- **Section VII:** Reports the current status of the implementation of the CapBAC system and outlines

Even if some of the issues analyzed in the following pages are focused on the FP7 IoT@Work context, they are actually more general and suitable for almost any IoT context.

## II. Related Work

Resources' protection requires the resource provider to know which client is accessing what resources and for what purpose. Information about clients and their purposes when accessing a specific resource is critical for a resource provider to grant or deny the requested operation.

The most common form of access control is based on access control lists (ACLs), which assign access rights to specific subjects. However, ACLs become very complex to manage when the number of subjects and resources increases. To reduce the burden of simple ACL systems, the Role-Based Access Control (RBAC) approach was designed. RBAC assigns access rights to roles, and subjects are assigned to these roles. This approach, however, can lead to roles explosion when the number of resources and/or administrative domains grows.

The Attribute-Based Access Control (ABAC) approach, well exemplified by the XACML standard, attempts to solve the problem of roles explosion by allowing the use of subject's properties (e.g., age, location, position in an organization) as well as resources and environmental properties to specify access policies. Despite this, ABAC still requires a consistent definition of the attributes within a domain or across different domains.

In widely open contexts like Service-Oriented Architecture (SOA) and Grid Computing, the traditional access control approaches (ACLs, RBAC, ABAC) face scalability issues. For instance, in cross-domain environments, they require managing trust among the involved Identity Providers, Attribute Providers, and Service Providers, which leads to increased management effort. Additionally, these approaches do not provide flexible and easy-to-use rights delegation features. In an IoT context, with its vast number of resources and subjects, these issues become even more critical.

The capability-based security model is not a new concept and has been used in devising standards like RFC 2693. Simple Public Key Infrastructure (SPKI) focuses on authorization rather than authentication by defining and exchanging authorization certificates. Since 1997, X.509 has included Attribute Certificates to specify subject information useful for authorization management, such as group membership, role, security clearance, etc.

In recent years, capability-based security models have been used to address usability issues and rights delegation in grid or service-oriented systems. For example, the Xerox Casca application and the XPOLA access control system have utilized this model. In the SUN-promoted Digital Ecosystem environment, a capability-based authorization approach was proposed by Skinner to address the dynamicity and scalability issues of such an environment. Similar approaches were proposed by Jun L and Karp to tackle similar issues and rights delegation.

## III. IoT Authorization Issues and CapBAC

The Internet of Things (IoT) presents a more demanding environment in terms of scalability and manageability compared to previous ones, including those based on an extended use of dynamically orchestrated SOA services. This is due to the potentially unbounded number of things (resources and subjects) and the significant need to support the orchestration and integration of different services, as envisaged by the Do It Yourself (DiY) socio-cultural practice. These IoT-specific aspects imply that access control management can become a nightmare in IoT if not addressed with new approaches. More complex and efficient access control mechanisms and delegation chains are required.

Both Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC) systems have been found to be inflexible, do not scale well, and are difficult to use and upgrade. Additionally, these systems have substantial management overhead, security issues (e.g., the confused deputy problem, rights revocation), and complex arrangements to support delegation and transitivity, as well as for managing access policies and ensuring policy compliance. Therefore, addressing these challenges in IoT environments requires rethinking access control mechanisms to accommodate the dynamic and large-scale nature of IoT systems.

A capability based access control and rights delegation approach has, instead, the following advantages:

- the *Principle of Least Authority* (*PoLA*) (Least Privilege) is the default;
- supports a more fine-grained access control;
- has less security issues (e.g. no Confused Deputy problem);
- externalizes and distributes the management of the authorization process;
- does not need to manage issues related to complexity and dynamics of subject's identities.

Additionally, identity management does not play a critical role in CapBAC, which provides huge advantages especially when managing access control in cross-domain In capability-based security models, each capability directly identifies the resource(s), the subject (grantee) to whom the rights have been granted, the granted rights, and the authorization chain. The grantee must prove the ownership of the identity specified in the capability for their access request to be accepted.

- In capability-based security models, each capability directly identifies the resource(s), the subject (grantee) to whom the rights have been granted, the granted rights, and the authorization chain. The grantee must prove the
- ownership of the identity specified in the capability to have their access request accepted. In an IoT context,
- such as in the IoT@Work project, it is not unusual to have the following requirements:

  1. **Dynamic and Scalable Management:** The ability to dynamically manage and scale the access control system to accommodate a potentially unbounded number of resources and subjects.
- 2. **Orchestration and Integration:** Support for the orchestration and integration of various services, which is crucial in complex IoT environments.

  3. **Fine-Grained Access Control:** The need for fine-
- grained access control mechanisms that can specify detailed and context-aware access rights.

## IV. A QUICK SURVEY OF CAPABILITY BASED SECURITY

Figure 2. provides examples of potential usage of capability based authorization to control access to Bob's car information and services (e.g.: car's location in section (a) of the figure, car's engine status services in section (b)). The subjects involved in the examples are: Bob Smith, the car's owner (and car's services access control policies manager); Alice Cooper, Bob's wife (interested in having information on Bob's car location); the Bob's City Traffic Management Service (interested in monitoring cars location); the Car's Manufacturer Maintenance Service (the application service in charge of monitoring engines status); Dave Jones (manager of the car's manufacturer Maintenance Service).

As depicted in the figure, Bob provides access capabilities to some of the indicated subjects. In particular he provides an access capability to:

- Consider an example where Bob grants different access capabilities to various entities regarding his car's data:
- 1. **Alice**: Bob grants his wife, Alice, the **Query** right on his car's location with **High Granularity** (Access Capability α2). This means Alice can access detailed location information about
- Bob's car.

  2. **City Traffic Management Service**: Bob grants this service the **Query** right on his car's location with a detail at the **Block Level** (Access Capability α1).

As indicated in section (b) of the figure, Dave Jones, on the basis of the received capability, has created an additional capability (Access Capability β2) for the car's manufacturer maintenance service in charge of periodically monitor Bob's car engine status. This capability contains a subset of the Dave Jones' rights, as well as Dave's capability (see *Auth. Capability* element in the figure).

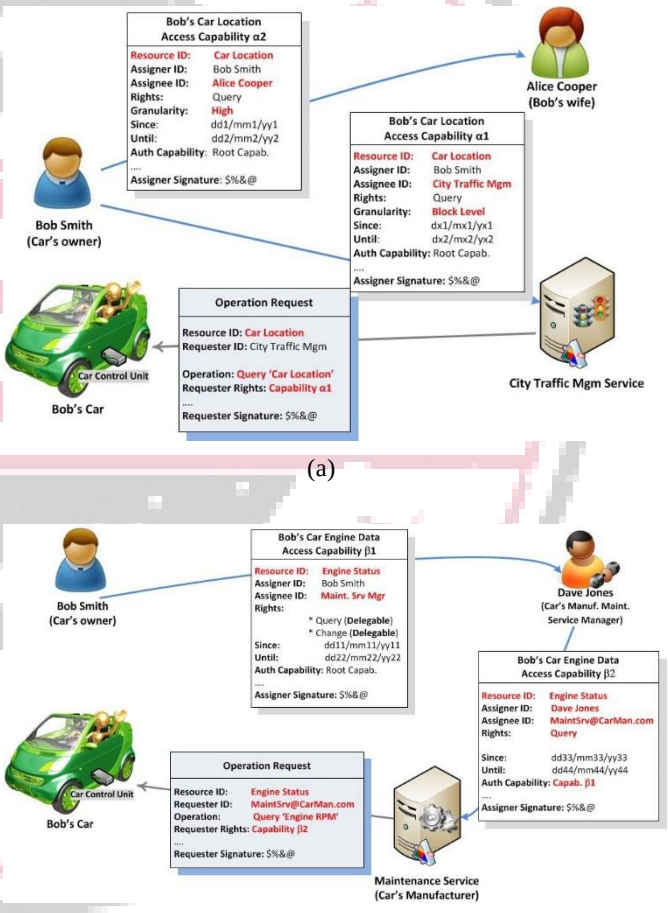


Figure 2. Capability-based authorization – Examples of potential scenario.

Two service requests submitted to Bob's car control unit are also shown in the figure sections. Each request states

When an access request is made to Bob's car control unit, it includes several critical elements: the access capability that grants the right to act on the resource in question, the equestor's identity, and proof of identity ownership. This information allows the control unit to evaluate whether the access request meets the necessary criteria. The control unit checks the access capability to ensure it covers the requested permissions and verifies the requestor's identity and proof of ownership. Additionally, it can assess the request against any local policies Bob has defined, which might impose specific rules or restrictions based on This transparency ensures that every subject involved in the authorization process is fully accountable, enhancing both security and oversight.V. CAPBAC FUNCTIONAL ELEMENTS

by the IoT@Work capability based authorization. These elements can be shortly characterized as follows:

- the **resource** object of the capability (*Service A* in the figure); it can be a specific information service (e.g. the measures of sensor XYZ), an application service (e.g. Alice's mailbox IMAP service) or a mix of services. The only real constraint is that the resource must be a univocally identifiable and actable upon object (much like a RESTful resource);
- the **authorization capability** that details the granted rights (and which ones can be delegated and, in case, their delegation depths), the resource on which those rights can be exercised, the grantee's identity, as well as additional information (e.g.: capability validity period, XACML conditions, etc.). As stated, a capability is a communicable object hence it can be provided to the subject using any communication mean. An authorization capability is valid as specified within the capability itself or until it is explicitly revoked;
- the **capability revocation** is used to revoke one or more capabilities. Like a capability, a capability revocation is a communicable object a subject, having specific rights (e.g. the revoker must be an ancestor in the delegation path of the revoked capability), creates to inform the service in charge of managing the resource that specific capabilities have to be considered no more valid. A capability revocation can revoke a single capability, a specific capability and all its descendants, or all descendants of a specific capability;
- the **service/operation request** is the service request as envisaged by the provided service with the only additional characteristics to refer or include, in an unforgeable way, a capability. For example, for a RESTful service, an HTTP GET request on one of the exposed REST resource has to simply include the capability and its proof of ownership to use our access control mechanism;
- the **resource PDP** (*Policy Decision Point*) is the service in charge of managing resource access request validation and decision. In a CapBAC environment, it is in charge of validating the In the evaluation of an access request, the system verifies the capability included in the service request against the provided access capability and any additional, locally available access policies. Similar to an XACML Policy Decision Point (PDP), the outcome
- of this evaluation is either an Allow or Deny decision. The resource manager, responsible for handling service requests for the identified resource (such as a CapBAC-aware RESTful service), also functions as an XACML Policy Enforcement Point (PEP). It must enforce the validation outcomes of the PDP.
- Additionally, a revocation service manages capability revocations. This involves validating received capability revocations and updating the PDP's capabilities database and access policies accordingly. The PDP needs to ensure that the capability presented within a service request is valid—i.e., not forged, with all data correct, and that it corresponds to the resource and assignee identified in the request. It also verifies that the capability has not been revoked. This comprehensive process ensures that the capability-based access control system operates securely and effectively.

As based authorization differs from traditional or more usual

The presence of additional elements such as access capability, capability revocation, and a revocation service introduces significant flexibility to the authorization framework. These elements enhance the system by providing greater granularity, scalability, and ease of access rights delegation. They also contribute to reducing security issues and ensuring full accountability throughout the authorization chain behind a service request.

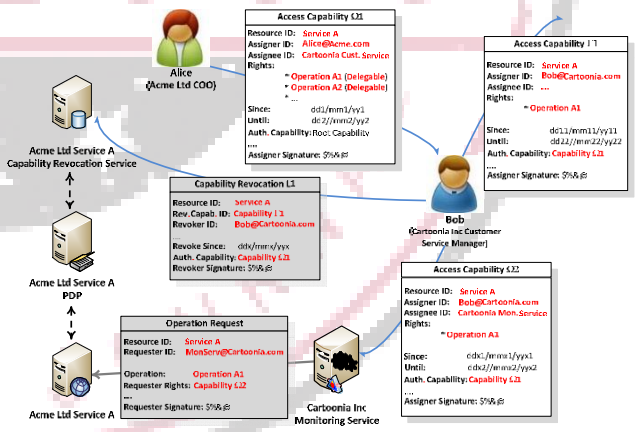


Figure 3. Capability-based authorization functional elements.

The main drawback of capability-based authorization is that it requires issuing capabilities to all subjects. This can be a management challenge, though the delegation mechanism simplifies this by allowing the distribution of management tasks among multiple subjects. Additionally, the requesting subject must select the appropriate capability when submitting a request.

Based on capability granting policies, it is possible to generate access capabilities on the fly for suitably identified and authorized users. This dynamic generation of capabilities has been explored in some projects ([7], [8]), demonstrating its potential for enhancing flexibility and responsiveness in access control systems.Moreover, in open, cross-domain, or cross-enterprise contexts, it is crucial to standardize the structure of capability tokens, supporting services, and their access protocols.

### A. Introduction

*In the preceding sections, we discussed the access control and delegation features of our approach. Section IV emphasized how capabilities can significantly simplify the process of granting access with varying levels of granularity to resources, without adding complexity to the access control system. This section shifts focus to describe the Privacy Enhancing features of CapBAC. Although privacy issues are not a primary concern in our current project, these features are included for completeness and potential future extensions, though they are not currently implemented.*

### B. Encrypted Capability Chain

*Figure 4 illustrates a scenario where Bob uses an Internet service (Share Your Pictures service, www.SYP.com) to store and share his photos. Bob wishes to use a printing service (www.HQP.com) to print one of his pictures but does not want to grant the printing service access to all his photos or disclose his SYP.com credentials. To address these concerns, Bob can utilize an access capability as defined previously.*

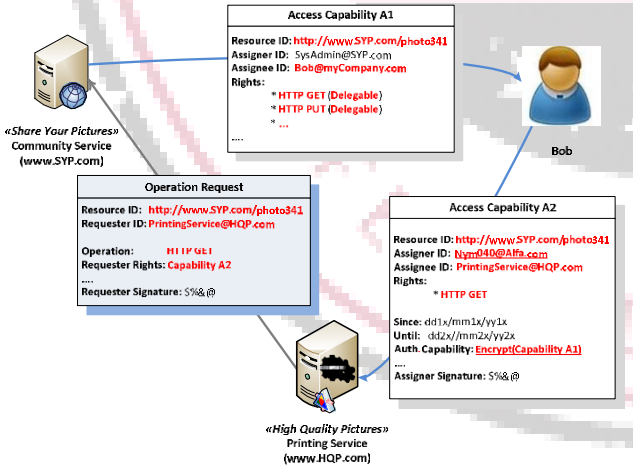


Figure 4. Encrypted capability chain.

Suppose Bob does not want to reveal to the printing service any personal information. A capability as defined in the previous sections would provide to the printing service

### B. Encrypted Capability Chain

To protect his privacy, Bob can use an authorization capability (capability A1 in Figure 4) that he owns to generate a new capability, which is not depicted in the figure. This new capability grants access rights to a Bob anonymous ID (Nym040@Alfa.com, as shown in the figure). Bob can then issue an additional capability (capability A2 in the figure) using his anonymous ID, which grants access to the printing service for the specific picture.

Capability A2 contains an encrypted version of capability A1, which fully masks Bob's authorization chain. The encryption uses the public key of the Share Your Pictures service (www.SYP.com), ensuring that while the SYP.com service can verify and decrypt the authorization chain, Bob's personal information remains concealed from the printing service. Thus, Bob's identity is obscured in this scenario, but his authorization chain remains traceable upward.

### C. Anonymous Capabilities

Figure 5 presents a scenario where Bob wishes to maintain complete anonymity, even while utilizing all the CapBAC features, including the ability to delegate.

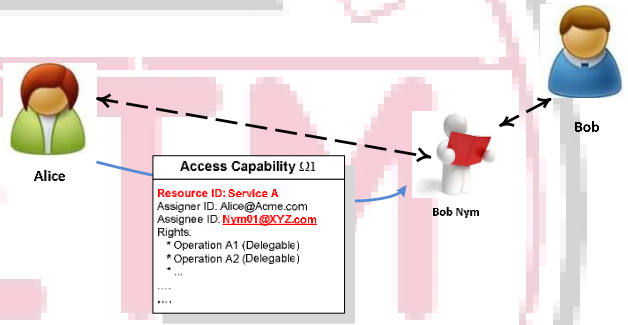


Figure 5. Anonymous capability.

### C. Anonymous Capabilities

To maintain complete anonymity while utilizing all CapBAC features, Bob can employ techniques such as Zero Knowledge Proofs. This allows him to prove, without revealing any personal information, that he is entitled to receive an access capability for a specific resource. As illustrated in Figure 5, Bob obtains a capability that contains no personal information about him, even though he retains full control and availability of it. This ensures that Bob can fully leverage the capabilities of the CapBAC system while preserving his anonymity.

### VII. Conclusion and Next Steps

In the preceding sections, we have detailed our capability-based authorization access control system, which builds on and extends recent research in this field. Our system, currently under development, is being implemented in Java as a collection of libraries, tools, and services. We have developed both an OSG and a Java library to validate capabilities and capability revocations, and we have created a Java application for generating capabilities.

In the preceding sections, we have outlined our capability-based authorization access control system, which extends recent research in this area. The system is being developed in Java and includes a suite of libraries, tools, and services. We have already implemented an OSG and Java library for validating capabilities and capability revocations, along with a Java application for generating capabilities. We are nearing completion of the CapBAC Policy Decision Point (PDP) and the Revocation Service, both of which are being developed as Java web applications with an AJAX U for management.

In a production context like the ones addressed by IoT@Work there are many subjects, internal (e.g.: workers, production supervisors) and external (e.g.: suppliers, maintainers), that need access both directly (e.g. via mobile or desktop computing sets) or indirectly (e.g. via application services) to production data. Most, if not all, of this data is sensitive and necessitates strict enforcement of access control policies. This includes finer-grained access control, which requires a management effort that is decoupled from the number of managed resources or subjects, especially when the many subjects are external ones (e.g.: suppliers, maintainers).

The capability based access control system described above is being made available openly in order to both speed up its adoption and to refine and enhance it.

## REFERENCES

[1] See project home page: http://iot-at-work.eu

[2] Wikipedia. *Capability-based security*. Available: http://en.wikipedia.org/wiki/Capability-based_security

[3] J. Li, A. H. Karp, "Access Control for the Services Oriented Architecture," in *ACM Workshop on Secure Web Services* (SWS'07), November 2007

[4] A. H. Karp, "Authorization-Based Access Control for the Services Oriented Architecture," in *Proc. 4th Int. Conf. on Creating, Connecting, and Collaborating through Computing* (C5), January 2006

[5] A. H. Karp, H. Haury, M. H. Davis, "From ABAC to ZBAC: The Evolution of Access Control Models," HP Laboratories, Tech. Report HPL-2009-30, February 2009

[6] A. H. Karp, J. Li, "Solving the Transitive Access Problem for the Services Oriented Architecture," in *Proc. 2010 Int. Conf. on Availability, Reliability, and Security* (ARES '10), February 2010

[7] G. D. Skinner, "Cyber Security Management of Access Controls in Digital Ecosystems and Distributed Environments," in *Proc. 6th Int. Conf. on Information Technology and Applications* (ICITA 2009), November 2009

[8] L. Fang, D. Gannon, F. Siebenlist, "XPOLA – An Extensible Capability-based Authorization Infrastructure for Grids," in *4th Annual PK R&D Workshop*, April 2005

[9] N. Hardy, "The Confused Deputy: (or why capabilities might have been invented)," *ACM SIGOPS Operating Systems Review*, vol. 22, no. 4, October 1988

[10] D. F. Ferraiolo, D. R. Kuhn, "Role Based Access Control," in *15th National Computer Security Conf.*, 1992

[11] *Information Technology: Requirements for the Implementation and Interoperability of Role Based Access Control*, ANSI/INCITS Standard 459-2011, January 2011

[12] *eXtensible Access Control Markup Language Version 3.0*, OASIS XACML v. 3.0, August 2010

[13] J. B. Dennis, E. C. Van Horn, "Programming Semantics For Multiprogrammed Computations," MIT, Tech. Report MIT/LCS/TR-23, 1965

[14] H. Levy, "Capability-Based Computer Systems," Digital Press, Bedford, Massachusetts, 1984. Available: http://www.cs.washington.edu/homes/levy/capabook/

[15] A. S. Tanenbaum, S. J. Mullender, R. van Renesse, "Using Sparse Capabilities in a Distributed Operating System," in *Proc. 6th Int. Conf. on Distributed Computing Systems*, 1986, pp. 558–563. Available: ftp://ftp.cs.vu.nl/pub/papers/amoeba/dcs86.ps.Z

[16] *SPK Certificate Theory*, IETF RFC 2693, September 1999. Available: http://www.ietf.org/rfc/rfc2693.txt

[17] *ITU-T Recommendation X.509: Information technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks* (also know as ISO/IEC 9594-8), ITU-T Recommendation X.509, November 2008

[18] *An Internet Attribute Certificate Profile for Authorization*, IETF RFC 3281, April 2002. Available: http://www.ietf.org/rfc/rfc3281.txt

[19] D. Balfanz, G. E. Durfee, D. K. Smetters, "Making the impossible easy: usable PKI," in *Security and Usability*, Chap. 16 edited by L. Cranor and S. Garfinkel, O'Reilly, 2005

[20] A. Lackorzynski, A. Warg, "Taming subsystems: capabilities as universal resource access control in L4," in *Proc. 2nd Workshop on Isolation and Integration in Embedded Systems* (IIES '09), April 2009

[21] M. Miller, Ka-Ping Yee, J. Shapiro, "Capability Myths Demolished," Systems Research Laboratory, Johns Hopkins University, Tech. Report SRL2003-02, 2003

[22] D. Uckelman, M. Harrison, F. Michahelles (eds.), *Architecting the Internet of Things*, Springer-Verlag Berlin Heidelberg, 2011

[23] L. Trappeniers, M. Roelands, M. Godon, J. Criel, P. Dobbelaere, "Towards Abundant DiY Service Creativity Successfully Leveraging the Internet-of-Things in the City and at Home," presented at *13th Int. Conf. on Intelligence in Next Generation Networks* (ICIN 2009), September 2009

[24] E. Yuan, J. Tong, "Attributed Based Access Control (ABAC) for Web Services," in *Proc. of the IEEE Int. Conf. on Web Services* (ICWS'05), July 2005

[25] D. R. Kuhn, E. J. Coyne, T. R. Weil, "Adding Attributes to Role-Based Access Control," *IEEE Computer*, vol. 43, no. 6, June 2010