Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 19, 2014

J. Gilger

H. Tschofenig

October 16, 2013

Report from the 'Smart Object Security Workshop', March 23, 2012, Paris, France
draft-gilger-smart-object-security-workshop-02.txt

Abstract

This document provides a summary of a workshop on 'Smart Object Security', which took place in Paris on March 23, 2012. The main goal of the workshop was to allow participants to share their thoughts about the ability to utilize existing and widely deployed security mechanisms for smart objects.

This report summarizes the discussions and lists the conclusions and recommendations to the Internet Engineering Task Force (IETF) community.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 19, 2014.

Copyright Notice

Table of Contents

1. Introduction

In early 2011, the Internet Architecture Board (IAB) solicited position statements for a workshop on 'Interconnecting Smart Objects with the Internet', aiming to get feedback from the wider Internet community on their experience with deploying IETF protocols in constrained environments. The workshop took place in Prague on March 25, 2011. During the workshop, a range of topics were discussed, including architecture, routing, energy efficiency, and security. RFC 6574 [RFC6574] summarizes the discussion and suggested several next steps.

During the months following the workshop, new IETF initiatives were started, such as the Light-Weight Implementation Guidance (lwig) working group, and hackatons were organized at IETF#80 and IETF#81 to better facilitate the exchange of ideas.

With the contributions on security in the IETF CoRE working group as well as in the IETF TLS working group it became clear that further

discussions on security were necessary and that those would have to feed in implementation and deployment experience as well as a shared understanding how various building blocks fit into a larger architecture.

The workshop on Smart Object Security was organized to bring together various disconnected discussions about smart object security happing in different IETF working groups and industry fora. It was a one-day workshop, held prior to the IETF#83 in Paris on March 23, 2012.

The workshop organizers were particularly interested to get input on the following topics, as outlined in the call for position papers:

- What techniques for issuing credentials have been deployed?
- What extensions are useful to make existing security protocols more suitable for smart objects?
- What type of credentials are frequently used?
- What experience has been gained when implementing and deploying application layer, transport layer, network layer, and link layer security mechanisms (or a mixture of all of them)?
- How can "clever" implementations make security protocols a better fit for constrained devices?
- Are there lessons we can learn from existing deployments?

This document lists some of the recurring discussion topics of the workshop. It also offers recommendations from the workshop participants.

Note that this document is a report on the proceedings of the workshop. The views and positions documented in this report are those of the workshop participants and do not necessarily reflect the views of the authors or the Internet Architecture Board (IAB).

2. Terminology

This document uses security terminology from [RFC4949] and smart object related terms from [RFC6574].

3. Workshop Structure

With 36 accepted position papers there was a wealth of topics to talk about during the one-day workshop. The program committee decided to divide the discussion into four topic areas ("Requirements and Use Cases", "Implementation Experience", "Authorization" and "Providing Credentials"), with two or three invited talks per slot to get a discussion started. This section will summarize the points raised by the invited speakers as well as the essence of the ensuing discussions.

3.1. Requirements and Use Cases

To design a security solution, an initial starting point is to understand the communication relationships, the constraints, and the security threats. The typical IETF security consideration section describes security threats, security requirements, and security solutions at the level of a single protocol or a single document. To offer a meaningful solution for a smart object deployment it is, however, necessary to go beyond this limited view to the analysis of the larger eco-system. The security analysis, documented in [RFC3552] and in [RFC4101], still provides valuable guidance.

Typical questions that arise are:

1. Who are the involved actors?

   Some usage scenarios look very simple at first but then, after a longer investigation, turn out to be quite complex. The smart meter deployment, for example, certainly belongs to one of the more complex deployments due to the history of the energy sector, see [RFC6272].

2. Who provisions credentials?

   Credentials may, for example, be provisioned by the end user, by the hardware manufacturer, an application service provider, or other parties. With security provided at multiple layers, credentials from multiple parties may need to be provisioned.

3. What constraints are imposed on the design?

For example, a constraint can be the need to interworking with existing infrastructure. From an architectural point of view an important question is whether security is terminated at the border router (or proxy server) at the customer's premise or if end-to-end security to servers in the Internet is required. A more detailed discussion can be found at [I-D.iab-smart-object-architecture].

4. What type of authorization is required by the identified actors?

   This may, for example, be authorization to get access to the network, or authorization at the application layer. Authorization decisions may be binary, or may consist of complex role-based access control policies.

5. What tasks are expected by the customer who deploys the solution?

   An end customer may, for example, be expected to enter short PIN codes to pair devices, might need to update the firmware, or needs to connect to an appliance via a Web browser to make more sophisticated configuration settings. The familiarity of end-users with Internet-based devices certainly increases constantly but user interface challenges contribute to a large number of security weaknesses of the Internet and therefore have to be taken into account.

To illustrate the differences, consider a mass-market deployment for end customers in comparison to a deployment that is targeting enterprise customers. In the latter case, enterprise system administrators are likely to utilize different management systems to provision security and other system-relevant parameters.

Paul Chilton illustrated the security and usability requirements in a typical end-user scenario for small-scale smart lighting systems [PaulChilton] . These systems present a substantial challenge for providing usable and secure communication because they are supposed to be cheap and very easy to set up, ideally as easy as their "dumb" predecessors. The example of IP-enabled light bulbs shows that the more constrained devices are, the more difficult it is to get security right. For this reason, and the required usability, light bulbs might just be the perfect example for examining the viability of security solutions.

Rudolf van der Berg focused on large-scale deployments of smart objects, such as eBook readers, smart meters, and automobiles. The use of mobile cellular networks is attractive because they are networks with adequate coverage and capacity on a global scale. In order to make use of mobile networks you need to make use of

Subscriber Identity Modules (SIM)-based authentication. However, at this moment the SIM is controlled by the network operator. These companies could also use EAP-SIM [RFC4186] authentication in, for example, wireless LANs.

The end-user interaction may differ depending on the credentials being used: for a light bulb deployed in the user's home it is expected that the user somehow configures devices so that only, for example, family members can turn them on and off. Smart objects that are equipped with SIM-based credential infrastructure do not require credential management by the end-user since credential management by the operator can be assumed. Switching cellular operator may, however, pose challenges for these devices.

Furthermore, we have a technology that will be both deployed by end-users and large enterprise customers. While the protocol building blocks may be the same, there is certainly a big difference between deployments for large-scale industrial applications and deployments for regular end-users in terms of the architecture. Between these two, the security requirements differ significantly, as do the threats. It is difficult, if not impossible, to develop a single security architecture that fulfills the needs of all users while at the same time meeting the constraints of all kinds of smart objects.

In the consumer market, security should not incur any overhead during installation. If an end user has to invest more time or effort to secure a smart object network, he or she will likely not do it. Consumer products will often be retrofitted into the existing infrastructure, bought and installed by consumers themselves. This means that devices will have to come pre-installed to some extent and will most likely interoperate only with the infrastructure provided by the vendor, i.e., the devices will be able to connect to the Internet but will only interoperate with the servers provided by the vendor selling the device.

Closed systems (one bulb, one switch) typically work out of the box, as they have been extensively tested and often come with factory-configured security credentials. Problems do arise when additional devices are added or when these closed systems get connected to the Internet. It is still very common to ship devices with default passwords. It is, however, not acceptable that a device is in a vulnerable, but Internet-connected, state before it has been correctly configured by a consumer. It is easy to conceive that many consumers do not configure their devices properly and may therefore make it easy for an adversary to take control of the device by, for example, using the default password or an outdated firmware.

Once security threats for a specific deployment scenario have been identified, an assessment takes place to decide what security requirements can be identified and what security properties are desirable for the solution. As part of this process, a conscious decision needs to take place about which countermeasures will be used to mitigate certain threats. For some security threats, the assessment may also lead to the conclusion that the threat is considered out-of-scope and that therefore no technical protection is applied. Different businesses are likely to come to different conclusions about the priorities for protection and what security requirements will be derived.

Which security threats are worthwhile to protect against is certainly in the eye of the beholder and remains an entertaining discussion even among security specialists. For some of the workshop participants, the security threats against a smart lighting system were considered relatively minor compared to other smart home appliances. Clearly, the threats depend on the specific application domain but there is a certain danger that deployments of vulnerable smart objects will increase. As the systems evolve and become more pervasive, additional security features may be required and may be difficult to incorporate into the already installed base, particularly if smart objects have no software update mechanism incorporated in their initial design. Smart objects which require human interaction to perform software updates will likely be problematic in the future. This is particularly true for devices that are expected to have service schedules of five to twentyfive years. Experience shows that security breaches that are considered to be a prank usually evolve very rapidly to become destructive attacks.

Apart from the security requirements of individual households and users, it is also important to look at the implications of vulnerabilities in large-scale smart object deployments, for example in smart meters and the power grid.

Finally, there is the usual wealth of other requirements that need to be taken into account, such as ability for remote configuration and software updates, the ability to deal with transfer of ownership of a device, avoidance of operator or vendor lock-in, crypto agility, minimal production, license and IPR costs, etc.

3.2. Implementation Experience

The second slot of the workshop was dedicated to reports from first-hand implementation experience. Various participants had provided position papers exploring different security protocols and cryptographic primitives. There were three invited talks which

covered tiny implementations of the Constrained Application Protocol (CoAP) protected by Datagram Transport Layer Security (DTLS), a TLS implementation using raw public keys, as well as general experience with implementing public key cryptography on smart object devices.

All three presenters demonstrated that implementations of IETF security protocols on various constraint devices are feasible. This was confirmed by other workshop participants as well. The overall code size and performance of finished implementations will depend on the chosen feature set. It is fairly obvious that more features translate to a more complex outcome. Luckily, IETF security protocols in general, and DTLS/TLS is no exception, can be customized in a variety of ways to fit a specific deployment environment. As such, an engineer will have to decide which features are important for a given deployment scenario, and what trade-offs can be made. There was also the belief that IETF security protocols offer useful customization features (such as different ciphersuites in TLS/DTLS) to select the desired combination of algorithms and cryptographic primitives. The need to optimize available security protocols further or to even develop new cryptographic primitives for smart objects was questioned and not seen as worthwile by many participants.

The three common constraints for security implementations on smart objects are code size, energy consumption, and bandwidth. The importance to tailor a solution to one of these constraints depends on the specific deployment environment. It might be difficult to develop an architecture that minimizes for all constraints at the same time.

To wait for the next generation of hardware typically does not magically lift the constraints faced today. The workshop participants again reinforced the message that was made at the earlier smart object workshop [RFC6574] regarding future developments in the smart object space: "While there are constantly improvements being made, Moore's law tends to be less effective in the embedded system space than in personal computing devices: gains made available by increases in transistor count and density are more likely to be invested in reductions of cost and power requirements than into continual increases in computing power.".

The above statement is applicable to smart object designs in general; not only for security. Thus, it is expected that designers will continue having to deal with various constraints of smart objects in the future. A short description of the different classes of smart objects can be found in [I-D.ietf-lwig-guidance], which also provides security-related guidance. The workshop participants noted that making security protocols suitable for smart objects must not water

down their effectiveness. Security functionality will demand some portion of the overall code size. It will have an impact on the performance of communication interactions, will lead to higher energy consumption, and certainly make the entire product more complex. Still, omitting security functionality because of various constraints is not an option. The experience with implementing available security protocol was encouraging even though the need to make various architectural design decisions for selecting the right set of protocols and protocol extensions that everyone must agree on was pointed out. Sometimes, the leading constraint is energy consumption and in other cases it is main memory, CPU performance, or bandwidth. In any case, for an optimization it is important to look at the entire system rather than a single protocol or even a specific algorithm.

Equally important to the code size of the protocols being used in a deployed product are various other design decisions, such as the communication model, the number of communication partners, the interoperability requirements, and the threats that are being dealt with. Mohit Sethi noted that even the execution time for relatively expensive operations like asymmetric signature generation and verification are within acceptable limits for very constrained devices, like an Arduino UNO. In either case, public key cryptography will like only be used for the initial communication setup to establish symmetric session keys. Perhaps surprisingly, the energy cost of transmitting data wirelessly dwarfs even expensive computations like public key cryptography. Since wireless reception is actually the most power consuming task on a smart object, protocols have to be designed accordingly.

The workshop participants shared the view that the complexity of security protocols is a result of desired features. Redesigning a protocol with the same set of features will, quite likely, lead to a similar outcome in terms of code size, memory consumption, and performance. It was, however, also acknowledged that the security properties offered by DTLS/TLS/IKEv2-IPsec may not be needed for all deployment environments. DTLS, for example, offers an authentication and key exchange framework combined with channel security offering data-origin authentication, integrity protection, and (optionally) confidentiality protection.

The biggest optimization in terms of code size can be gained when looking at the complete protocol stack, rather than only cryptographic algorithms. This also includes software update mechanisms and configuration mechanisms, all of which have to work together. What may not have been investigated enough is the potential of performing cross-layer and cross-protocol optimization. We also need to think about how many protocols for security setup we

want to have. Due to the desire to standardize generic building blocks, the ability to optimize for specific deployment environments has been reduced.

Finally, it was noted that scalability of security protocols does not imply usability. This means that while smart object technology might currently be developed in large scale industrial environments, it should be equally usable for consumers who want to equip their home with just a few light bulbs.

For details about the investigated protocol implementations please consult the positions papers, such as the ones by Bergmann et al., Perelman et al., Tschofenig and Raza et al..

3.3. Authorization

The discussion slot on authorization was meant to provide an idea of what kind of authorization decisions are common in smart object networks. Authorization is defined as 'an approval that is granted to a system entity to access a system resource' [RFC4949].

Authorization requires a view on the entire smart object lifecycle to determine when and how a device was added to a specific environment, what permissions have been granted for this device and how users are allowed to interact with it. On a high level, there are two types of authorization schemes: First, there are those systems that utilize an authenticated identifier and match it against an access control lists. Secondly, there are trait-based authorization mechanisms that separate the authenticated identifier from the authorization rights and utilize roles and other attributes to determine whether to grant or deny access to a protected resource.

Richard Barnes looked at earlier communication security work and argued that the model that dominates the web today will not be enough for the smart object environment. Simply identifying users by their credentials and servers via certificates is not something that translates well to smart object networks because it binds all the capabilities to the credentials. The evolution in access control is moving in the direction of granting third parties certain capabilities, with OAuth [RFC6749] being an example of a currently deployed technology. Access to a resource using OAuth can be done purely based on the capabilities rather than on the authenticated identifier.

> At the time of the workshop OAuth was very much focused on HTTP-based protocols with early efforts to integrate OAuth into SASL and the GSS-API [I-D.ietf-kitten-sasl-oauth]. Further investigations need to be done to determine the suitability of OAuth as a protocol for the smart object environment.

Richard believed that it is important to separate authentication from authorization right from the beginning and to consider how users are supposed to interact with these devices to introduce them into their specific usage environment (and to provision them with credentials), and to manage access from different parties.

The relationship between the policy enforcement point and the policy decision point plays an important role regarding the standardization needs and the type of information that needs to be conveyed between these two entities.

> For example, in a AAA context the authorization decision happens at the AAA server (after the user requesting access to a network or some application level services had been authenticated). Then, the decision about granting access (or rejecting it) is communicated from the AAA server to the AAA client at the end of the network access authentication procedure. The AAA client then typically enforces the authorization decision over the lifetime of the granted user session. The dynamic authorization extension [RFC3576] to the RADIUS protocol, for example, also allows the RADIUS server to make dynamic changes to a previously granted user session. This includes support for disconnecting users and changing authorizations applicable to a user session.

The authorization decisions can range from 'only devices with password can use the network' to very detailed application specification authorization policies. The decisions are likely to be more sophisticated in those use cases where ownership of devices may be transferred from one person to another one, group membership concepts may be needed, access rights may be revocable, and fine grained access rights have to be used. The authorization decisions may also take environmental factors into account, such as proximity of devices to each other, physical location of the device asking access, or the level of authentication. With the configuration of authorization policies the question arises who will create them and where these policies are stored. This immediately raises the question about how devices are identified, and who is allowed to create these policies.

Since smart objects may be limited in terms of code size, persistent storage, and Internet connectivity, established authorization schemes may not be well suited for such devices. Obviously, delegating every

authorization decision to another node in the network incurs a certain network overhead, while storing sophisticated access control policies directly on the smart object might be prohibitive because of the size of such a ruleset. Jan Janak presented one approach to distribute access control policies to smart objects within a single administrative domain.

In those cases where access control decisions are bound to the identifiers of devices and humans need to either create or verify these access control policies, the choice of identifier matters for readability and accessibility purposes.

A single mechanism will likely not help with solving the wide range of authorization tasks. From the discussions it was not clear whether there is a need for new authorization mechanisms or whether existing mechanisms can be re-used. Examples of available protocols with built-in authorization mechanism are Kerberos, OAuth, EAP/AAA, attribute certificates, etc. In many cases, it is even conceivable that the authorization decisions are internal to the system, and that there is no need to standardize any additional authorization mechanisms or protocols at all. In fact many of the authentication and key exchange protocols have authorization mechanisms built-in.

3.4. Provisioning of credentials

When a smart object is to be introduced into an environment, like a home or an enterprise network, it usually has to be provisioned with some credentials first. The credentials that are configured at the smart object and at some entity in the network are often an implicit authorization to access the network or some other resource. The provisioned information at the smart object will include some identifier of the smart object, keying material, as well as other configuration information (e.g., specific servers it has to interact with).

Some devices will be pre-configured with default security codes or passwords, or will have per-device or per-user credentials pre-configured, when they are bought or when they arrive at the customer.

There is a limited set of solutions available (based on the available interface support). The solutions for imprinting vary between the enterprise and the consumer household scenarios. For large-scale deployments, the time needed to pair two objects further excludes other schemes which rely on manual steps.

Johannes Gilger dealt with the very basic ideas behind pairing schemes, including the kinds of out-of-band channels that could be employed and their limitations. Imprinting and pairing protocols

usually establish a security association between two equal devices, such as Bluetooth-equipped cell phones. To deal with man-in-the-middle attacks during this phase, various forms of additional verification checks exist. For example, devices with a display allow numeric values to be shown on each device and to let the user verify whether they match. For other devices that have a keypad, a PIN may need to be entered by the user. Where and how a smart object is to be paired with other devices in the network can differ substantially from the specific use cases and the hardware capabilities of devices. Note that pairing is not necessarily something that is only done once during the lifetime of a device. Is group pairing something to be looked at? Or can any group key establishment be reduced to pair wise pairing with a central master device?

Cullen Jennings presented a model for smart objects based on a deployment used for IP phones. The idea was that the smart object "phones home", i.e., contacts a server offered by the manufacturer, when it is first switched on. This initial interaction can then be used for managing the device and provisioning keying material for further use. Proof of ownership could be done by identifying the user who purchased the device. This is an approach that is increasingly being done today. Another option is some kind of secret information enclosed in the packaging.

For interface-constrained devices, the solution of using (semi)-public information in combination with an online manufacturer during imprinting seems like a possible solution. This solution approach created a lot of discussion among the participants, as it assumes an Internet connection and means that the manufacturer effectively knows about the trust relationships of all the devices it sells.

A few questions did arise with such a model: Will there be third parties which have a business interest in providing something like key distribution and key escrow over the lifetime of a smart object? For constrained devices, will it always be possible to fall back to the existing security associations between device and manufacturer to create new associations? Obviously, we do not want the lifetime of a smart object limited by the manufacturer product support lifespan. What happens if a manufacturer goes bankrupt, changes its business scope, or gets bought by another company? Will end customers not be able to use their smart objects anymore in such a case or will they loose the ability to re-sell their devices because the ownership can no longer be transferred?

One important design decision is that the compromise of the manufacturer must not have any impact on the smart objects, which have already been imprinted to their new owners. Furthermore, the

question of how to transfer ownership, e.g. when reselling a device arise. While this may not be a requirement for all devices, there will likely be classes of large or expensive devices where support for transferring the ownership is an absolute necessity.

Industrial users are comfortable when they have to rely on the manufacturer during the imprinting phase, but they want to be in exclusive control over their devices afterwards.

There are many classes of devices where we could assume online connectivity to be present, otherwise these devices would not make sense in the first place. But, there are also other devices which need to be imprinted completely offline.

Is it important to worry about security vulnerabilities, such as man-in-the-middle attacks, during the very short imprinting phase? Is it realistic that an adversary is in close proximity to mount an attack? Especially for devices with limited capabilities, such as lightbulbs, the concerns seemed rather small.

What happens if such a device is not enrolled by the customer but still connected in a "naked" state? How does this impact security and it is possible for an attacker to perform a 'drive-by' enrollment procedure of many devices? How should a device behave in this situation? The safest (for the user at least) would be to not allow the device to work with full functionality if it has not been enrolled. This concern is particularly applicable for cases where smart objects are sold with default passwords or passwords using semi-public information. Examples of those are Raspberry Pi's with Linux images that use a default password [RaspberryPi].

4. Summary

Designing for a smart object environment is about making an optimization decision that needs to take technical aspects, usage scenarios, security threats, and business models into account. Some design constraints may be considered fixed while others are flexible. Compromises will have need to be made but those should not only go at the expense of security functionality.

Designing a software update mechanism into the system is crucial to ensure that both functionality can be enhanced and that potential vulnerabilities can be fixed. Functionality as well as security will need to remain unchanged for several years. Also the importance of security threats changes over time.

New research and standardization on cryptographic algorithms (like encryption algorithms, hash functions, keyed message digests, public

key crypto systems) that are tailored to smart object environments was not seen as worthwhile by the participants. A huge range of algorithms already exists and standardized authentication and key exchange protocols can be customized to use almost any selection of algorithms already today. The integration of various building blocks into a complete system was considered important and this document highlights a number of those areas. Searching for the smart object security architecture was seen as a hopeless journey given the almost infinite design space.

5. Acknowledgements

We would like to thank the participants and the paper authors of the position papers for their input.

Special thanks go to Thomas Heide Clausen and Ecole Polytechnique (Paris) for providing the venue and organization.

Finally, we would like to thank Rudolf van der Berg for his review comments.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

The whole document is a report on the Smart Object Security Workshop. The focus of this workshop was on security only; privacy was not part of the workshop agenda.

8. References

8.1. Normative References

[RFC6574] Tschofenig, H. and J. Arkko, "Report from the Smart Object Workshop", RFC 6574, April 2012.

8.2. Informative References

[I-D.iab-smart-object-architecture]
Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", draft-iab-smart-object-architecture-02 (work in progress), March 2013.

[I-D.ietf-kitten-sasl-oauth]

Mills, W., Showalter, T., and H. Tschofenig, "A set of SASL and GSS-API Mechanisms for OAuth", draft-ietf-kitten-sasl-oauth-10 (work in progress), February 2013.

[I-D.ietf-lwig-guidance]
Bormann, C., "Guidance for Light-Weight Implementations of the Internet Protocol Suite", draft-ietf-lwig-guidance-03 (work in progress), February 2013.

[PaulChilton]
Chilton, P., "Experiences and Challenges in using constrained Smart Objects, Position Paper to the 'Workshop on Smart Object Security'", URL: http://www.tschofenig.priv.at/sos-papers/PaulChilton.pdf, March 2012.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

[RFC3576] Chiba, M., Dommety, G., Eklund, M., Mitton, D., and B. Aboba, "Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)", RFC 3576, July 2003.

[RFC4101] Rescorla, E. IAB, "Writing Protocol Models", RFC 4101, June 2005.

[RFC4186] Haverinen, H. and J. Salowey, "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", RFC 4186, January 2006.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", RFC 4949, August 2007.

[RFC6272] Baker, F. and D. Meyer, "Internet Protocols for the Smart Grid", RFC 6272, June 2011.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[RaspberryPi]
Raspberry Pi Foundation, "Raspberry Pi", URL: http://www.raspberrypi.org, February 2013.

Appendix A. Program Committee

The workshop was organized by the following individuals:

- Hannes Tschofenig
- Jari Arkko
- Carsten Bormann
- Peter Friess
- Cullen Jennings
- Antonio Skarmeta
- Zach Shelby
- Thomas Heide Clausen

Appendix B. Published Workshop Material

- Main Workshop Page: http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity
- Position Papers: http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/papers
- Slides: http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/slides

Appendix C. Accepted Position Papers

1. Michael Richardson, "Challenges in Smart Object Security: too many layers, not enough ram"
2. Mitsuru Kanda, Yoshihiro Ohba, Subir Das, Stephen Chasko, "PANA applicability in constrained environments"
3. Randy Bush, "An Operational View of Trust Needs of Moving Objects"
4. Andrei Gurtov, Ilya Nikolaevsky, Andrey Lukyanenko,"Using HIP DEX for Key Management and Access Control in Smart Objects"
5. Jens-Matthias Bohli, "Access Tokens for the IoT "
6. Sye Loong Keoh, Martina Brachmann, Oscar Garcia-Morchon, Sye-Loong Keoh, Sandeep S. Kumar, "Security Considerations around End-to-End Security in the IP-based Internet of Things"

7. Kazunori Miyazawa, "Convergence of Smart Objects in industrial wireless sensor network"

8. Thomas Bartzsch, Dirk Burggraf, Laura Cristina, Alexis Olivereau, Nouha Oualha, Emil Slusanschi, Dan Tudose, Markus Wehner, Sven Zeisberg, "AAA-based Infrastructure for Industrial Wireless Sensor Networks"

9. Philip Ginzboorg, Fida Khattak, Philip Ginzboorg, Valtteri Niemi, Jan-Erik Ekberg, "Role of Border Router in 6LoWPAN Security"

10. Thomas Fossati, Angelo Castellani, Salvatore Loreto, "(Un)trusted Intermediaries in CoAP"

11. Rene Hummen, Christian Roeller, Klaus Wehrle, "Modeling User-defined Trust Overlays for the IP-based Internet of Things"

12. Sam Hartman, Margaret Wasserman, "Federation, ABFAB and Smart Devices"

13. Cary Bran, Joseph Stachula "Device Pairing: Lessons Learned"

14. Jan Janak, Hyunwoo Nam, Henning Schulzrinne, "On Access Control in the Internet of Things"

15. Rene Struik, "Cryptography and Security for Highly Constrained Networks"

16. Zhen Cao, Hui Deng, "The Architecture of Open Security Capability"

17. Sujing Zhou, Zhenhua Xie, "On Cryptographic Approaches to Internet-Of-Things Security"

18. Monique Morrow, Nancy Cam Winget, "Security Implications to Smart Addressable Objects"

19. Jouni Korhonen, "Applying Generic Bootstrapping Architecture for use with Constrained Devices"

20. Olaf Bergmann, Stefanie Gerdes, Carsten Bormann, "Simple Keys for Simple Smart Objects"

21. Jari Arkko, Mohit Sethi, Ari Keranen, "Practical Considerations and Implementation Experiences in Securing Smart Object Networks"

22. Paul Chilton, "Experiences and Challenges in using constrained Smart Objects"

23. Vladislav Perelman, Mehmet Ersue, "TLS with PSK for Constrained Devices"

24. Richard Barnes, "Security for Smart Objects beyond COMSEC: Principals and Principles"

25. Rudolf van der Berg, "OECD Publication on Machine-to-Machine Communications: Connecting Billions of Devices", OECD Digital Economy Papers, No. 192, OECD Publishing

26. Cullen Jennings, "Transitive Trust Enrollment for Constrained Devices"

27. Barbara Fraser, Paul Duffy, Maik Seewald, "Smart Objects: Security Challenges from the Power Sector"

28. Hannes Tschofenig, "Smart Object Security: Considerations for Transport Layer Security Implementations"

29. Johannes Gilger, Ulrike Meyer, "Secure Pairing & Policy Frameworks"

30. Klaas Wierenga, "Scalable Authentication for Smart Objects"

31. Dirk Stegemann, Jamshid Shokrollahi, "Security in the Internet of Things - Experiences from Use Cases"

32. Alper Yegin, "Credentials for Smart Objects: A Challenge for the Industry"

33. Shahid Raza, Thiemo Voigt, Vilhelm Jutvik, "Lightweight IKEv2: A Key Management Solution for both the Compressed IPsec and the IEEE 802.15.4 Security"

34. Eric Rescorla, "A Brief Survey of Imprinting Options for Constrained Devices"

35. Fred Baker, "Security in distributed telemetry and control networks"

Appendix D. Workshop Participants

We would like to thank the following workshop participants for attending the workshop:

- Jari Arkko
- Carsten Bormann
- Cullen Jennings
- Antonio Skarmeta
- Sean Turner
- Thomas Heide Clausen
- Hannes Tschofenig
- Michael Richardson
- Yoshihiro Ohba
- Subir Das
- Randy Bush
- Andrei Gurtov
- Ilya Nikolaevsky
- Andrey Lukyanenko
- Jens-Matthias Bohli
- Kazunori Miyazawa
- Philip Ginzboorg
- Fida Khattak
- Angelo Castellani
- Salvatore Loreto
- Rene Hummen
- Klaus Wehrle

- Sam Hartman
- Margaret Wasserman
- Cary Bran
- Jan Janak
- Rene Struik
- Zhen Cao
- Hui Deng
- Zhou Sujing
- Xie Zhenhua
- Monique Morrow
- Nancy Cam Winget
- Jouni Korhonen
- Ari Keranen
- Paul Chilton
- Vladislav Perelman
- Mehmet Ersue
- Richard Barnes
- Rudolf van der Berg
- Barbara Fraser
- Johannes Gilger
- Sye Loong Keoh
- Olaf Bergmann
- Stefanie Gerdes
- Klaus Hartke

- Oualha Nouha
- Oliverau Alexis
- Alper Yegin
- Klaas Wierenga
- Jiazi Yi
- Juan Antonio Cordero Fuertes
- Antonin Bas
- David Schinazi
- Valerie Lecomte
- Ulrich Herberg
- Shahid Raza
- Stephen Farrell
- Eric Rescorla
- Thomas Fossati
- Mohit Sethi
- Alan Duric
- Guido Moritz
- Sebstian Unger
- Hans Loehr

Authors' Addresses

Johannes Gilger
Mies-van-der-Rohe-Str. 15
Aachen 52074
Germany

Phone: +49 (0)241 80 20 781
Email: Gilger@ITSec.RWTH-Aachen.de

Hannes Tschofenig
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at