WSACd - A Usable Access Control Framework for Smart Home Devices

Konstantinos Fysarakis[1(✉)], Charalampos Konstantourakis[2], Konstantinos Rantos[3], Charalampos Manifavas[4], and Ioannis Papaefstathiou[1]

[1] Department of Electronic and Computer Engineering, Technical University of Crete, Chania, Greece
kfysarakis@isc.tuc.gr, ypg@mhl.tuc.gr
[2] Department of Computer Science, University of Crete, Heraklion, Greece
harconst@csd.uoc.gr
[3] Department of Computer and Informatics Engineering, Eastern Macedonia and Thrace Institute of Technology, Kavala, Greece
krantos@teiemt.gr
[4] Department of Informatics Engineering, Technological Educational Institute of Crete, Heraklion, Greece
harryman@ie.teicrete.gr

Abstract. Computing devices already permeate working and living environments; a trend which is expected to intensify in the coming years. However, the direct interaction smart devices often have with the physical world, along with the processing, storage and communication of private sensitive data pertaining to users' lives, bring security concerns into the limelight. This paper presents Web Service Access Control for devices (WSACd), a framework that combines access control provided by the eXtensible Access Control Markup Language (XACML) with the benefits of Service Oriented Architectures through the use of the Devices Profile for Web Services (DPWS). Based on standardized technologies, it enables fine-grained policy-based management of the heterogeneous embedded devices that may be found in a smart residential setting. The proposed framework is implemented in full and its performance is evaluated on a test-bed featuring devices expected to be found in a typical residential environment.

Keywords: Policy-based access control · XACML · Service architectures · DPWS · Smart home · Ubiquitous computing

1 Introduction

In recent years, massive advancements in computing and communication technologies have led to significant changes in terms of how people perform the various tasks comprising their everyday lives; a development enabled by the ubiquitous presence of computing devices in all aspects of modern life. These major changes could not leave the residential environment unaffected, with smart homes gradually becoming a reality, i.e. homes featuring sophisticated lighting (e.g. smart light bulbs), ambient environment controls (e.g. heating, ventilation and air conditioning via smart thermostats),

R.N. Akram and S. Jajodia (Eds.): WISTP 2015, LNCS 9311, pp. 120–133, 2015.
DOI: 10.1007/978-3-319-24018-3_8

appliances (smart -fridge, -oven, -washing machine, -coffee makers etc.), communication systems (including smart phones), entertainment (e.g. smart TVs), and home security (smart cameras, door and window controls etc.) devices. Moreover, the residential environment integrates with other ubiquitous computing applications, like smart metering and e-health, found in the smart home ecosystem. Nevertheless, as said devices typically handle personal sensitive data and often feature direct interaction with the physical world, a key factor in the wider adoption and success of these new technologies will be the effectiveness with which the various security [1][2] and privacy [3] concerns are tackled. A necessary instrument in successfully addressing these issues is the presence of robust access control mechanisms.

This paper presents Web Service Access Control for devices (WSACd), a scheme which aims to address the above requirements by defining a policy-based Access Control (AC) mechanism based on the eXtensible Access Control Markup Language (XACML [4], an OASIS standard), thus providing the means to control access to the resources of smart home nodes based on policy constraints centrally managed by the system owner. Typical XACML deployments require the setup of complex infrastructures to enable entities' interaction and policy retrieval (e.g. via LDAP); such an approach may be acceptable for corporate environments but is not suitable in the context of consumer applications and the average home user. To this end, the proposed framework leverages the benefits of Service Oriented Architectures (SOAs) by implementing key entities using the Devices Profile for Web Services (DPWS [5]), also an OASIS standard, which allows the deployment of devices aligned with the Web Services technologies, thus facilitating interoperability among services provided by resource-constrained devices. The adoption of DPWS facilitates seamless Machine-to-Machine (M2M) discovery and interactions, allowing the deployment of the framework's entities to any platform, anywhere on the home network, with minimal involvement on behalf of the user. By combining the above technologies, new devices can easily join existing networks and offer services protected by a predefined or dynamic policy set. Based on the policy rules set by the system owner, the proposed architecture provides fine-grained AC over the plethora of devices and services that may be found in smart home environments. Thus, WSACd assists in the use of the various smart devices aiming to enhance consumers' lives, while addressing their security concerns.

This work is organized as follows: Section 2 provides the rationale behind this work, and relevant research efforts identified in the literature. Section 3 presents the proposed framework and its key entities, along with our approach to implementing the framework. Section 4 includes the performance evaluation of the developed solution on typical devices that may be found in a smart home environment. Finally, Section 5 concludes the article, containing pointers to areas that future work could explore to further enhance the presented scheme.

2 Rationale and Related Work

In a typical ubiquitous-computing-enhanced residential setting, various smart devices are expected to be present on appliances (e.g. smart fridge) and automation-enabled

structures (e.g. smart doors), also including environmental sensors and actuators. Moreover, these are typically complemented by purpose-built devices intended to organize, manage and enhance the functionality of the rest of the smart infrastructure, like energy monitors and control nodes (e.g. a computing system with touch-based input to allow seamless monitoring and interaction with the devices).

This heterogeneous assortment of devices will feature a variety of services, each with its own intrinsic characteristics (some being critical in terms of the residents' safety, others dealing with private sensitive data etc.), thus requiring a different protection profile. For example, all residents should be able to control the smart doors and windows of a house, but, perhaps, children should not be able to tamper with a subset of those (e.g. front door) at certain timeframes (e.g. during the night). In another scenario, visitors may have the rights to monitor the environmental sensors of the residence, but not to set the climate control at their will. Moreover, the residence owners may decide they feel alright with visitors checking the contents of their smart fridge, but they, expectedly, should not be able to add goods to the shopping lists. Assuming the presence of e-health devices in the smart home ecosystem, it is anticipated that the patient, her spouse and medical staff should be able to monitor the various readings and control the actuators that deliver the prescribed medicine, but only the latter group should have access to the service that controls the drug dosage. In cases where the residence is equipped with smart-metering devices, authorized power company staff should be the only ones able to adjust and/or reset the meters (for billing purposes), but, nevertheless, the owners should be able to access the consumption readings. A more thorough analysis on the security risks associated with smart homes can be found in [6]; a report which identified threats, with high exposure, to all of the smart home assets, including the human inhabitants.

Furthermore, a survey [7] on smart home users revealed that inflexibility (often forcing users to adopt solutions offered by a single manufacturer) and difficulties in achieving security constitute significant barriers to the broader adoption of pertinent technologies and devices.

From the above, and considering that, typically, the only pervasive protection mechanism present in home environments is the access to the wireless network, it is evident that strong and interoperable access control mechanisms are required to safeguard a variety of aspects pertaining to the operation of a smart home environment. Additionally, this should be achieved in a flexible, platform-agnostic manner, acting as an enabler instead of introducing new (or further exacerbate existing) obstacles to the adoption of "smart" devices and services.

To this end, the presented framework is based on standardized mechanisms, which also allows leveraging work already carried out both in terms of Web Services as well as XACML policy definitions. DPWS can enable user-to-machine and M2M interactions in a unified manner, moving on from the current state of the field, where consumer electronics manufacturers offer a variety of proprietary protocols which are not interoperable and essentially lock-in consumers, forcing them to use a specific vendor/ecosystem. With regard to XACML, the scheme can trivially be expanded to cater for additional specific concerns, such as privacy issues and/or the handling of sensitive data (e.g. healthcare, as covered by the relevant XACML profile [8]).

2.1 Service-Oriented Architectures

SOAs evolved from the need to have interoperable, cross-platform, cross-domain and network-agnostic access to devices and their services. This approach has already been successful in business environments, as web services allow stakeholders to focus on the services themselves, rather than the underlying hardware and network technologies.

The Devices Profile for Web Services (DPWS) specification defines a minimal set of implementation constraints to enable secure Web Service messaging, including discovery, description, interactions and event-driven changes on resource-constrained devices. DPWS was introduced in 2004 and is now an OASIS open standard (at version 1.1 since July 2009). It employs similar messaging mechanisms as the Web Services Architecture (WSA), with restrictions on complexity and message size, allowing the provision of totally platform- and language-neutral services, similar to those offered by traditional web services, allowing system owners to leverage the SOA benefits across heterogeneous systems that may be found in the various smart environments (residential, enterprise etc.).

In this context, the work of Leong et al [9] presents a rule-based framework for heterogeneous smart-home systems management. Their work focuses on the use of SOAP for interoperability and uses an Event-Condition-Action (ECA) mechanism for machine-2-machine interactions and orchestration of the devices. The SOAP-based interoperability framework has been further extended by Perumal et al [10] with the addition of a service stub to facilitate the addition of new devices and a database module to handle the queries of the SOAP messages (including home service functions, operation logic and access to other local or remote databases).

SOA-DOS [11] is a SOA-based distributed operating system proposed in the relevant literature, aiming to manage all embedded devices in a home network and facilitating interoperability between the various systems. The work manages to provide a SOA-based solution that is also applicable to very resource-constrained platforms (like sensor nodes), but deviates from standardized mechanisms, e.g. resorting to the use of the JSON [12] format instead of XML for data exchange.

The use of SOA concepts to tackle the dynamic and heterogeneous nature of home-control applications has also been proposed by Bourcier et al [13]. The authors introduce an implementation of their approach based on open source, standardized platforms, providing bridges to seamlessly integrate disparate devices (including DPWS devices) and their services into their home control infrastructure.

The DPWS stack also forms the basis of iVision [14], a purpose-built hardware platform used to add context-awareness to a service architecture for controlling home appliances, and its accompanying architecture. In the above work, the context information extracted by the iVision camera and all the necessary smart home appliance communications are exposed as web services using DPWS.

The use and benefits of DPWS have also been studied extensively in the context of various other applications areas, including automotive and railway systems [15], industrial automation [16], smart grid [17], eHealth [18] and wireless sensor networks [19]. All of the above are positive indicators for the future of the technology chosen as the underlying implementation and communication mechanism for the presented framework, and its potential for ubiquitous adoption.

2.2 Access Control

Among the studied access control schemes proposed for systems with different requirements and properties, a cross-platform solution that meets the requirements of all types of embedded systems and provides interoperability (which is crucial for next-generation pervasive computing devices), is based on eXtensible Access control Markup Language (XACML) policies. XACML defines the structure and content of access requests and responses exchanged among access control entities. In this work, the typical policy based access control architecture, combined with XACML, is mapped to a SOA network of nodes to provide protected access to their distributed resources.

A survey of the literature reveals a wealth of related work, including various diverse approaches and attesting to the applicability of XACML in the context of smart homes.

Kim et al [20] have proposed the use of an OSGi (Open Services Gateway initiative) -based framework to integrate heterogeneous smart-home devices and services, including an access control model, combining XACML mechanisms with OSGi services to appropriately create the queries that will be forwarded to the entity responsible for access control decisions (i.e. the Policy Decision Point, PDP). While the proposed approach theoretically supports a variety of protocols (including DPWS devices), the presented analysis and proof of concept implementation are mainly based on Universal Plug and Play (UPnP), a protocol lacking in many respects (e.g. security & scalability). Furthermore, the performance of the proposed mechanisms – an important aspect considering the resource-constraints of many smart devices – is not evaluated.

Busnel et al [21] present a case study for remote healthcare assistance in smart homes. Most of the smart home security & dependability requirements are discussed extensively, identifying the use of SOAs along with XACML as the most applicable technologies to fulfill these requirements. An XACML-based authorization solution is applied using the security pattern approach to satisfy security requirements typically existing in such environments. This work presents the outline of such a framework, but not an actual implementation of the SOA and XACML mechanisms, nor a performance evaluation. The resource-constrained nature of the target devices and the use of appropriate security mechanisms do not appear to have been considered during the design phase.

Researchers have also studied the use of access control mechanisms to safeguard the users' privacy, a key concern in the context of smart environments. Faravelon et al [22] outline such an architecture in the context of SOA-enabled pervasive environments, using a medical scenario as a test case. The interoperability with DPWS is considered, among other SOA technologies, but a non-standardized approach is adopted for the access control functionality.

Privacy issues have also been considered by Jung et al [23], who have presented a generic concept of access control for home automation gateways, aiming to safeguard the privacy and security of users and their data. The scheme is based on a customized SOAP message structure that integrates XACML attributes within SAML-based

access token. However, the initial, theoretical evaluation of the proposed scheme indicates that this approach is quite costly (especially in terms of packet size), which questions its applicability in the context of embedded smart home devices. The authors acknowledge this drawback and indicate it will be investigated, as future work, on actual platforms.

Müller et al [24] have also proposed the combined use of DPWS with XACML, but focusing on end-user content (e.g. the distribution of multimedia files) and the use of proxies to establish trust relationships across smart home domains (an approach which could be exploited in WSACd as well). Moreover, the authors did not exploit DPWS in the implementation and deployment of the XACML architecture.

3 Proposed Model and Implementation Approach

This section aims to detail the basic components of the proposed solution, as well as the toolkits chosen to develop the proof of concept implementation.

3.1 XACML Implementation

In the proposed framework, the following key entities are present and can be deployed on different smart home nodes, depending on their role and resources:

- *Policy Enforcement Point (PEP):* Makes decision requests and enforces authorization decisions. This is expected to be present in every smart device (appliances, sensors, e-health devices, energy monitoring or smart metering devices etc.) which provides its resources to the end users, and which need to be protected by the active policy set.
- *Policy Decision Point (PDP):* Evaluates requests against applicable policies and renders an authorization decision. It is expected to be deployed on more feature-rich nodes, typically a personal computer or an embedded system that acts as a controlling node for the whole smart home infrastructure
- *Policy Administration Point (PAP) & Policy Information Point (PIP):* The former creates and manages policies or policy sets, while the latter acts as a source of attribute values. These two entities will typically be deployed on the same feature-rich node, facilitating direct interaction with end-users (e.g. home owners). A desktop computer or a laptop are good candidates for this role.

As is evident from the above, and considering that nodes embedded in a smart home may not have the computing resources to accommodate expensive mechanisms, the core decision process is undertaken by more powerful nodes expected to operate within the node's trusted environment. Such an approach allows requests to be directly addressed to the node in question, while maintaining the capability to centrally manage and control access to these nodes. An overview of the architecture can be seen in Fig. 1.

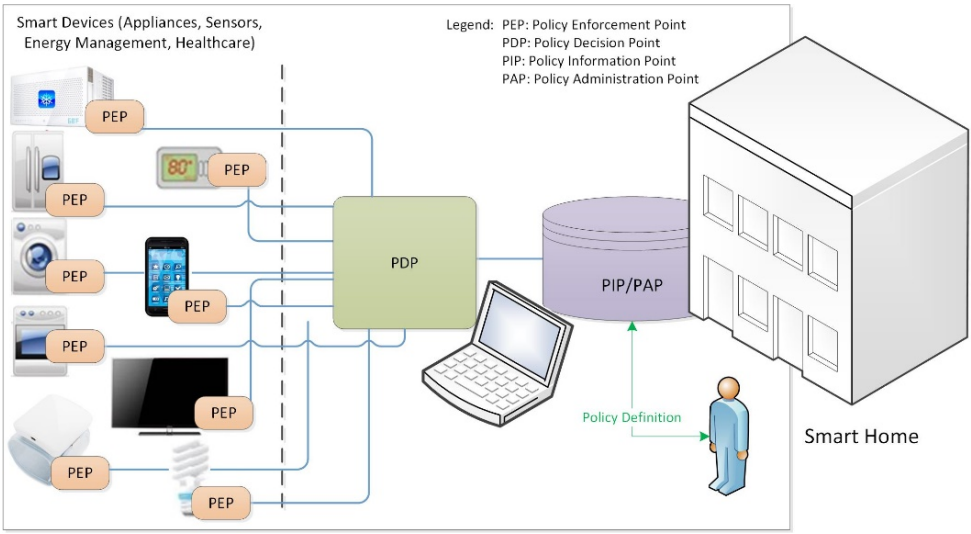

Fig. 1. Smart Home Access Control architecture

There are various open-source and commercial implementations of XACML that could be used as a basis for the AC entities needed to implement the proposed framework. We chose to use Sun's Java-based XACML implementation for all the infrastructure components, as it remains popular among developers and is actually the basis of various current open-source and commercial offerings.

3.2 DPWS Implementation of the XACML Entities

All of the framework's entities are exposed to the network using DPWS, thus leveraging the benefits provided by SOAs. There are a variety of APIs available for DPWS development, including the tools provided by the Web Services for Devices (WS4D) initiative and the SOAs for Devices (SOA4D) toolkits, based on various programming languages (C, C++, Java etc.) and each featuring its own intrinsic characteristics. Nevertheless, when focusing on key features such as code portability, deployment on heterogeneous platforms, support for IPv6 (necessary for ubiquitous computing applications) and, most importantly, active development and support of the tools, WS4D-JMEDS currently stands out as the most attractive choice. It is the most advanced and active work of the WS4D initiative, supporting most of the existing DPWS features and providing portability to a wide range of platforms; it is also our toolkit of choice to develop the DPWS entities presented in this work.

The XACML features are exposed as follows:

— *PEP to PDP implementation:* The Policy Enforcement Point must reside on every device with resources that must be protected from unauthorized access. Other than the functional elements of the devices which the framework intends to protect (e.g. access to its sensors), two extra operations must be present on each DPWS device.

These operations, in essence, constitute the PEP functionality and its communication with the PDP. The latter acts as a DPWS client which accesses these PBAC-specific operations. More specifically, the first operation is the "SAREvent" (Service Access Request Event), referring to an operation following the WS-Eventing specification to which devices can subscribe. When fired, the operation outputs “SAROut”, a message which includes all the information the PDP needs to have in order to evaluate a request (i.e. Subject, Action and Resource). The second operation is "PDPResponse" (Policy Decision Point Response), which is invoked by the PDP to relay an answer to a pending access request.

— *PDP to PIP/PAP implementation:* In terms of the discovery and information exchange that must take place between infrastructure entities (PDP, PIP, PAP), an extra operation must reside with the entity that stores the active policy set (namely the PIP/PAP). This extra operation is named "PIPOperation" (Policy Information Point Operation). It features an input for the request issued by the PDP (requesting all applicable policy rules), and an output containing all the pertinent information (i.e. policies and rules) that the PIP has identified.

The above DPWS operations and the sequence of events that take place when an access request is received for a protected resource are depicted in Fig. 2.

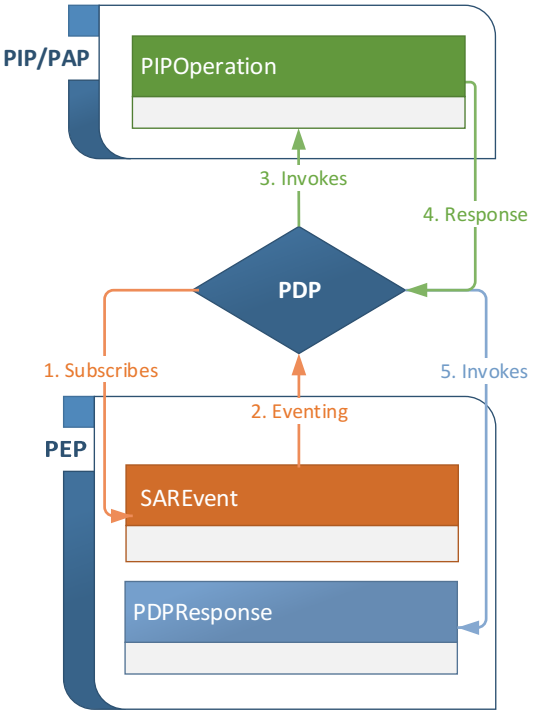

Fig. 2. DPWS implementation of the XACML mechanisms.

3.3 Security Considerations

The effectiveness of any access control mechanism can easily be compromised unless appropriate security mechanisms are deployed to protect policy messaging. A malicious entity would otherwise be able to eavesdrop, replay or tamper with the access control messaging, overriding the offered protection to provide access to unauthorized entities or denying access to authorized ones. When feasible, deployments over trusted and/or secure networks (e.g. over a Virtual Private Network, VPN) can address most of these concerns, but an alternative mechanism has to be considered for deployments where these provisions are not realistic.

The Web Services Security Specification (WS-Security or WSS, [25]) is part of the WS-* family of specifications published by OASIS. The protocol specifies integrating security features in the header of SOAP messages. Working in the application layer, it ensures the end-to-end integrity and confidentiality of SOAP messages.

Therefore, a variation of the proof-of-concept implementation was developed as well, adopting the security mechanisms specified in WSS. These mechanisms authenticate entities and safeguard the integrity and confidentiality of the policy messaging exchanged by the framework's entities.

4 Performance Evaluation

The use of platform-agnostic technologies (i.e. DPWS and Java) enables the proposed framework to be deployed, by design, on a variety of platforms and operating systems. However, in order to realistically assess the performance of the proposed framework and its impact on the target devices, the developed entities have to be deployed on devices expected to be present in smart home environments. Therefore, the infrastructure entities, namely the PDP and PIP/PAP, were deployed on a laptop (quad core CPU at 2.6GHz, 4GB RAM), as a personal computer is typically available in home environments and is expected to act as a management hub through which the residents monitor and control their smart residence. A total of 50 policies were stored in the policy repository, which we consider a realistic approximation of the number of policies needed, considering the relatively limited number of devices expected to reside in a smart home. Tests were also carried out with 500 policies, to assess the impact more policies would have on the framework's performance.

Regarding the target platforms – i.e. the platforms featuring the services that need to be protected – we chose to use relatively resource-constrained smart embedded devices (600MHz low power single core CPU, 512MB RAM) running a popular open source operating system for mobile devices. Such operating systems are already found in many smart commercial appliances (e.g. smart fridges) offered by the various consumer vendors. Moreover, their adoption is expected to spread as more sophisticated home devices become available to end users; thus, the above platform can be considered a realistic choice for evaluating the performance of the proposed mechanisms.

The DPWS device deployed on the smart platform not only featured the access control related operations (as depicted in Fig. 2) but also featured three simple operations, emulating part of the functionality of a smart appliance. Via the above operations, the

user can get the current temperature, subscribe to a service that periodically informs of said temperature and also set the desired temperature when needed. A basic touch GUI was developed for this device, which can be seen in Fig. 3.

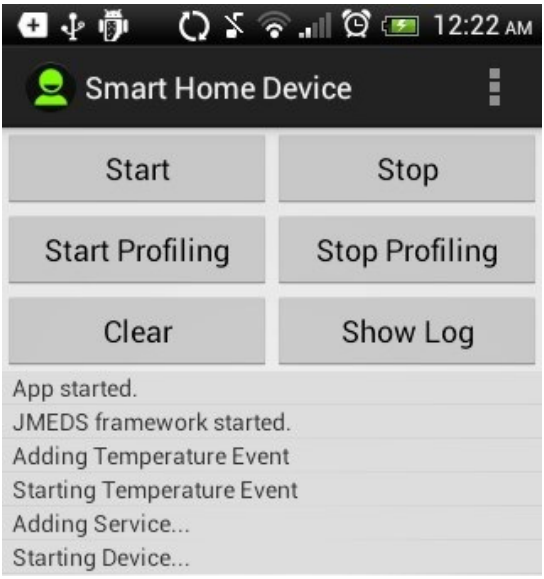

Fig. 3. Screen capture of the (access control-protected) DPWS test device deployed on the touch-enabled smart platform (e.g. a smart fridge).

A client application was also developed for testing purposes; the "Smart Home Browser". This application is deployable on various end devices (personal computers, smart phones or tablets) and allows users to discover and control the various DPWS-enabled smart appliances (to get the current contents of the smart fridge, to subscribe to the power consumption readings provided by the smart metering device etc.). A screenshot of the Smart Home Browser prototype implementation appears in Fig. 4.

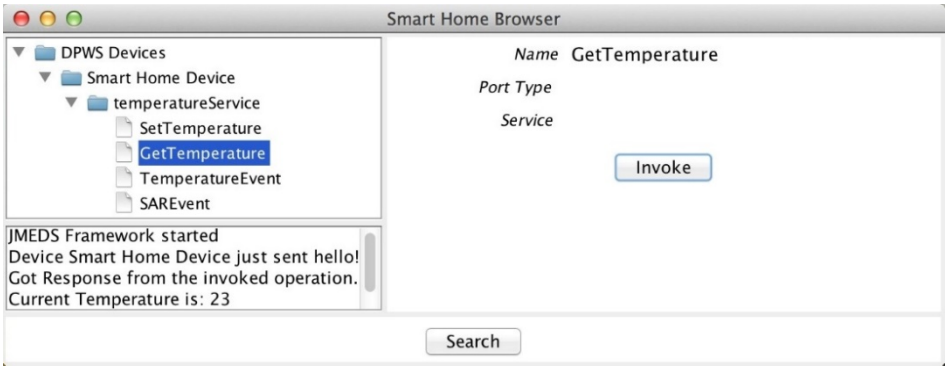

Fig. 4. The "Smart Home Browser"; an application developed to facilitate the discovery of DPWS devices and provide access to their hosted services.

A command line-only variation of this client, programmed to automatically invoke operations and record response times, was developed for benchmarking purposes. This benchmark client was used to evaluate the performance of three setups: a simple DPWS device with no PEP implemented (i.e. with direct access to its services), a DPWS device protected by the presented access control entities communicating in plaintext, and a third setup with the entities' communications being protected via WS-Security. This allowed us to separately assess the impact of the access control functionality and the impact of the security mechanisms that may be needed to protect the policy messaging in some deployments.

In addition to the client-side measurements, the CPU and memory utilization was also monitored on both the personal computer that hosted the PDP and PIP/PAP as well as on the PEP-equipped smart device. Furthermore, two different usage scenarios were investigated: In the first scenario, the client issued 100 concurrent requests to invoke the services, allowing the investigation of the performance under heavy load conditions. In the second scenario, the client issued 20 requests, one every 3 minutes, emulating more realistic usage conditions in the context of a smart home environment.

The results of the above assessments in terms of the average response time (i.e. the time the user has to wait before she receives the data she intended to access) are depicted in Fig. 5. In both usage scenarios, the overhead of the access control mechanism are considered acceptable. The impact of the WSS protection is significant in cases of infrequent requests (as in the second scenario, where the connections close between the request timeouts and, thus, have to be reinitiated).

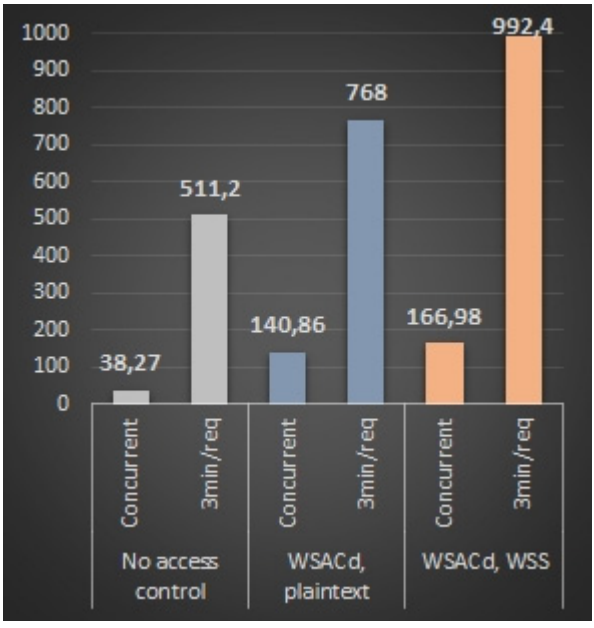

Fig. 5. Average client-side response time (in ms) for the investigated deployments and usage scenario

In terms of the resources consumed on the target, PEP-protected device, and focusing on the most demanding scenario (i.e. concurrent requests), profiling indicated a mild footprint during tests, even in the case of the relatively resource-constrained smart platform used in this setup. Average memory consumption is presented in Fig. 6, where the overhead of the access control mechanisms appears trivial compared to the simpler device.

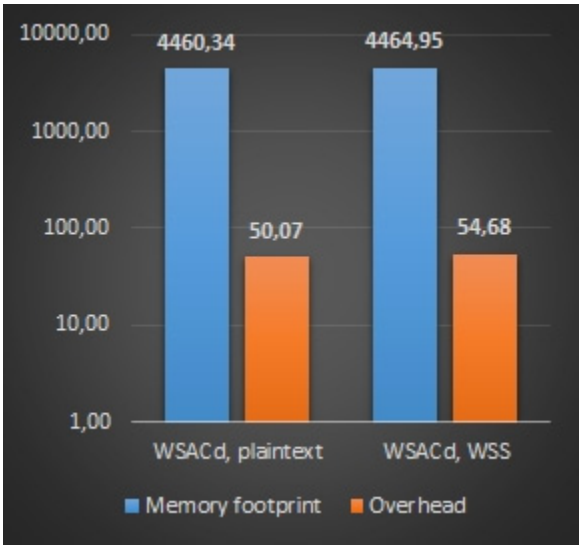

Fig. 6. Memory footprint (in kB, logarithmic scale) on the PEP-protected device, including the overhead compared to the simple DPWS device with no access control protection

The average CPU load was inversely proportional to the client response times (appearing in Fig. 5); when the device has to wait for a reply from the framework (i.e. the PDP) before serving the client, its CPU load is expectedly lower. The recorded values were 11.6%, 9.3% and 8.4% for no access control, WSACd and WSACd with WSS respectively. The same ranking was also documented when monitoring average transmission (TX) and reception (RX) rates on the target device – not depicted here to conserve space. The most taxing scenario network-wise was that of the device with no access control, but in all cases the data rates were relatively low, with the lowest recorded value being 16.13kB (average TX of WSACd, WSS device) and the highest being 26.5kB/sec (average RX of DPWS device without access control).

5 Conclusions and Future Work

This paper presented WSACd, a framework that leverages the benefits of SOAs, and DPWS specifically, to enable the integration of well-studied fine-grained and adaptable access control provided by XACML into smart homes. The intrinsic requirements of the smart home environment, its users and the often resource-constrained nature of

its devices fundamentally affected the choice and implementation of the standardized mechanism that form the basis of this work. Thus, WSACd's entities are platform-agnostic, lightweight and interact seamlessly, minimizing the home users' involvement in deploying, setting up and maintaining the system.

Nevertheless, home owners will be responsible for defining some parameters of the active policy set, depending on their requirements and preferences. Thus, an important aspect to be investigated is the provision of user-friendly interfaces for specifying access control policies, e.g. using a GUI with easy to use drop-down menus and tick boxes or having the user answer simple questions, automatically translating the user input to policies. Lastly, while this paper focused on authorization aspects of ubiquitous smart devices, another important building block is the user authentication, which the authors currently investigate and aim to present in future work.

References

1. Fysarakis, K., Hatzivasilis, G., Rantos, K., Papanikolaou, A.: Embedded systems security challenges. In: Measurable security for Embedded Computing and Communication Systems (MeSeCCS 2014), within the International Conference on Pervasive and Embedded Computing and Communication Systems (PECCS 2014), Lisbon, Portugal (2011)
2. Petroulakis, N.E., Askoxylakis, I.G., Tryfonas, T.: Life-logging in smart environments: challenges and security threats. In: IEEE International Conference on Communications, pp. 5680–5684 (2012)
3. Petroulakis, N.E., Askoxylakis, I.G., Traganitis, A., Spanoudakis, G.: A privacy-level model of user-centric cyber-physical systems. In: Marinos, L., Askoxylakis, I. (eds.) HAS 2013. LNCS, vol. 8030, pp. 338–347. Springer, Heidelberg (2013)
4. Parducci, B., Lockhart, H.: eXtensible Access Control Markup Language (XACML) Version 3.0. OASIS Standard, 1–154 (2013)
5. Devices profile for web services, version 1.1, OASIS Standard (2009). http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.pdf
6. European Union Agency for Network and Information Security (ENISA): Threat Landscape and Good Practice Guide for Smart Home and Converged Media (2014)
7. Brush, A., Lee, B., Mahajan, R.: Home automation in the wild: challenges and opportunities. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 2115–2124 (2011)
8. DeCouteau, D., Davis, M., Staggs, D.: OASIS Cross-Enterprise Security and Privacy Authorization (XSPA) Profile of XACML v2.0 for Healthcare. OASIS Standard Specification, pp. 1–21 (2009)
9. Leong, C., Ramli, A., Perumal, T.: A rule-based framework for heterogeneous subsystems management in smart home environment. IEEE Trans. Consum. Electron. **55**, 1208–1213 (2009)
10. Perumal, T., Ramli, A., Leong, C.: Interoperability framework for smart home systems. IEEE Trans. Consum. Electron. **57**, 1607–1611 (2011)
11. Sleman, A.: Moeller R: SOA distributed operating system for managing embedded devices in home and building automation. IEEE Transactions on Consumer Electronics **57**(2), 945–952 (2011)
12. Crockford, D.: The JavaScript Object Notation (JSON) Data Interchange Format. The Internet Engineering Task Force RFC 7159, pp. 1–15 (2006)

13. Bourcier, J., Escoffier, C., Lalanda, P.: Implementing home-control applications on service platform. In: IEEE Consumer Communications and Networking Conference, Las Vegas, USA, pp. 925–929 (2007)
14. Igorevich, R.R., Park, P., Choi, J., Min, D.: iVision based context-aware smart home system. In: The 1st IEEE Global Conference on Consumer Electronics 2012, pp. 542–546. IEEE (2012)
15. Venkatesh, V., Vaithayana, V., Raj, P., Gopalan, K., Amirtharaj, R.: A Smart Train Using the DPWS-based Sensor Integration. Res. J. Inf. Technol. **5**, 352–362 (2013)
16. Garcia Valls, M., Lopez, I.R., Villar, L.F.: ILAND: An enhanced middleware for real-time reconfiguration of service oriented distributed real-time systems. IEEE Trans. Ind. Informatics **9**, 228–236 (2013)
17. Zhou, L., Rodrigues, J.: Service-oriented middleware for smart grid: Principle, infrastructure, and application. IEEE Commun. Mag. **51**, 84–89 (2013)
18. Rantos, K., Fysarakis, K., Manifavas, C., Askoxylakis, I.G.: Policy-Controlled Authenticated Access to LLN-Connected Healthcare Resources. IEEE Systems Journal **PP**(99), 1–11 (2015). doi:10.1109/JSYST.2015.2450313. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7160675&isnumber=4357939
19. Dohndorf, O., Krüger, J., Krumm, H., Fiehe, C., Litvina, A., Luck, I., Stewing, F.J.: Towards the web of things: Using DPWS to bridge isolated OSGi platforms. In: 2010 8th IEEE International Conference on Pervasive Computing and Communications Workshops, PERCOM Workshops 2010, pp. 720–725 (2010)
20. Kim, J.E., Boulos, G., Yackovich, J., Barth, T., Beckel, C., Mosse, D.: Seamless integration of heterogeneous devices and access control in smart homes. In: 2012 Eighth Int. Conf. Intell. Environ., pp. 206–213 (2012)
21. Busnel, P., El-Khoury, P., Giroux, S., Li, K.: An XACML-based Security Pattern to achieve Socio-Technical Confidentiality in Smart Homes. J. Smart Home **3**, 17–26 (2009)
22. Faravelon, A., Chollet, S., Verdier, C., Front, A.: Enforcing privacy as access control in a pervasive context. In: IEEE Consumer Communications and Networking Conference, Las Vegas, USA, pp. 380–384 (2012)
23. Jung, M., Kienesberger, G., Granzer, W., Unger, M., Kastner, W.: Privacy enabled web service access control using SAML and XACML for home automation gateways. In: International Conference for Internet Technology and Secured Transactions, Abu Dhabi, UAE, pp. 584–591 (2011)
24. Müller, A., Kinkelin, H., Ghai, S.K., Carle, G.: A secure service infrastructure for interconnecting future home networks based on DPWS and XACML. In: Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks - HomeNets 2010, p. 31. ACM Press, New York (2010)
25. Lawrence, K., Kaler, C., Nadalin, A., Monzilo, R., Hallam-Baker, P.: Web Services Security: SOAP Message Security 1.1. https://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf