Implementación del control de uso en Internet de las cosas: Un caso de uso doméstico inteligente

Antonio La Marra, Fabio Martinelli, Paolo Mori, Andrea Saracino Istituto di Informatica e Telematica Consiglio
Nazionale delle Ricerche Pisa, Italia
Correo electrónico: nombre.apellido@iit.cnr.it

Resumen—Internet de las cosas (IoT) es un paradigma que se ha vuelto extremadamente popular, con aplicaciones que van desde la salud electrónica hasta los controles industriales. Las arquitecturas de IoT están distribuidas y a menudo se basan en dispositivos restringidos, lo que dificulta la tarea de introducir mecanismos de seguridad, en particular aquellos que requieren una evaluación dinámica de políticas. En este artículo presentamos UCIoT (Control de uso en IoT), un marco adaptable y tolerante a fallas para la aplicación de políticas de control de uso en entornos de IoT. UCIoT ofrece las funcionalidades de un marco de control de uso basado en U-XACML en una arquitectura descentralizada, distribuida y de igual a igual (P2P). En el presente trabajo, describimos una aplicación de UCIoT en un entorno de hogar inteligente, presentando también dos posibles casos de uso donde se explota el control de uso para implementar una política de ahorro de energía y una política de seguridad. Finalmente se presenta una serie de experimentos en dispositivos reales para informar el rendimiento del sistema, midiendo la sobrecarga introducida por el marco UCIoT.

I. INTRODUCCIÓN

En los últimos años, los objetos conectados están cada vez más presentes en nuestra vida diaria. Después de que los teléfonos inteligentes y las tabletas trajeron la conectividad a nuestros bolsillos, permitiendo a los usuarios estar perfectamente conectados también en movilidad, los objetos conectados ahora están entrando rápidamente en nuestras casas. "Gartner, Inc. pronostica que en 2017 se utilizarán 8.400 millones de cosas conectadas en todo el mundo, un 31 por ciento más que en 2016, y alcanzará los 20.400 millones en 2020. El gasto total en terminales y servicios alcanzará casi 2 billones de dólares en 2017". 1 Estos dispositivos son los componentes centrales de un nuevo paradigma conocido como Internet de las cosas (IoT), que se convierte en hogar inteligente cuando los dispositivos, incluidos sensores y actuadores, se instalan en una casa. Casa inteligente Los electrodomésticos y dispositivos domésticos inteligentes están entrando poco a poco en nuestras casas, sustituyendo a los tradicionales. Los primeros dispositivos que llegaron a los estantes fueron los televisores inteligentes, seguidos por refrigeradores inteligentes, termostatos inteligentes, hornos inteligentes, cámaras inteligentes, luces inteligentes, duchas inteligentes y muchos otros. Estos dispositivos, a diferencia de los tradicionales, tienen la posibilidad de ejecutar un sistema operativo (como Google Android), que en varios casos permite la instalación de aplicaciones adicionales, para mayor funcionalidad y personalización.

Sin embargo, las capacidades de conexión de los dispositivos inteligentes, así como la capacidad de instalar software de terceros, exponen los dispositivos, los usuarios y todo el entorno (la casa en el caso de una configuración de hogar inteligente) a posibles problemas de seguridad. Los riesgos de seguridad y protección se pueden mitigar agregando sistemas configurables para hacer cumplir las políticas de seguridad. La necesidad de introducir mecanismos de control de acceso y control de uso en IoT y entornos domésticos inteligentes ha sido anticipada por trabajos relevantes y recientes en la literatura como [3], [17] y [12], que informan también los desafíos de introducir estos mecanismos. sobre arquitecturas distribuidas y dispositivos restringidos, propios del IoT.

En este artículo presentamos UCIoT (Control de uso en Internet de las cosas), un marco que tiene como objetivo llevar el control de uso en arquitecturas de IoT de una manera fluida, configurable y dinámica. El marco está diseñado para arquitecturas heterogéneas y distribuidas de dispositivos conectados, evaluando y aplicando políticas de seguridad o de propósito general mediante la explotación del expresivo lenguaje U-XACML [5]. En particular, este trabajo se centra en una implementación de UCIoT en un entorno doméstico inteligente, presentando políticas específicas del entorno e informando experimentos en un banco de pruebas comparable en rendimiento y dimensión a un entorno doméstico inteligente.

La contribución de este documento se resume a continuación: •

Definimos UCIoT, un marco distribuido, descentralizado, tolerante a fallas e independiente de la infraestructura para implementar funcionalidades de control de uso en sistemas de IoT; • Discutimos una implementación del marco UCIoT en un entorno doméstico inteligente, introduciendo también dos políticas para el ahorro de energía y la seguridad física que pueden aplicarse a través de UCIoT; • Informamos detalles sobre la implementación del marco UCIoT, que explota la tabla hash distribuida Apache Cassandra basada en CHORD para permitir la comunicación distribuida y el almacenamiento de datos, informando los desafíos y las opciones de implementación para descentralizar las funciones de UCON.

• Presentamos un conjunto de experimentos para medir el rendimiento del marco propuesto, para medir la sobrecarga de rendimiento causada por el marco UCIoT en una implementación real realizada en dispositivos Raspberry PI-32.

El resto del artículo está organizado de la siguiente manera. En la Sección II se describen los componentes de UCON y las fases del flujo de trabajo. La Sección III describe el marco UCIoT, detallando los

1Gartner Inc., http://www.gartner.com/newsroom/id/3598917

2https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

2324-9013/17 $31,00 © 2017 IEEE
DOI 10.1109/Trustcom/BigDataSE/ICES.2017.352

componentes y el flujo de trabajo y las opciones de implementación para adaptar los componentes UCON a una arquitectura distribuida. La Sección IV informa la evaluación del desempeño de UCIoT documentando la implementación en un banco de pruebas de dispositivos reales. La Sección V informa sobre un conjunto de trabajos relacionados sobre el control de acceso y uso en entornos de IoT. Finalmente, la Sección VI concluye brevemente proponiendo algunas direcciones futuras.

II. EL MODELO DE CONTROL DE USO

El modelo de Control de Uso (UCON) [14], [16] amplía los modelos tradicionales de control de acceso introduciendo atributos mutables y nuevos factores de decisión además de las autorizaciones: obligaciones y condiciones. Los atributos mutables representan características de sujetos, recursos y entorno que cambian sus valores como consecuencia del funcionamiento normal del sistema [15].
Por ejemplo, algunos atributos mutables cambian sus valores porque la política incluye declaraciones de actualización de atributos que se ejecutan antes (antes de la actualización), durante (durante la actualización) o después (después de la actualización) de la ejecución del acceso. Por ejemplo, el saldo de la billetera electrónica es un atributo del sujeto que la política podría reducir cada vez que el sujeto realiza un nuevo acceso a un recurso.

Dado que los atributos mutables cambian sus valores durante el uso de un objeto, el modelo de control de uso permite definir políticas que se evalúan antes (predecisión) y continuamente durante el acceso al objeto (decisión continua).

La evaluación continua de la política cuando el acceso está en curso tiene como objetivo ejecutar contramedidas adecuadas (como interrumpir el acceso) cuando el derecho de ejecución ya no es válido, con el fin de reducir el riesgo de mal uso de los recursos.
Por lo tanto, en el modelo de control de uso es crucial poder recuperar continuamente los valores actualizados de los atributos mutables, para poder realizar la evaluación continua de la política y reaccionar rápidamente al cambio de atributo tomando las acciones adecuadas, por ejemplo, interrumpiendo aquellos accesos en curso que ya no estén autorizados.

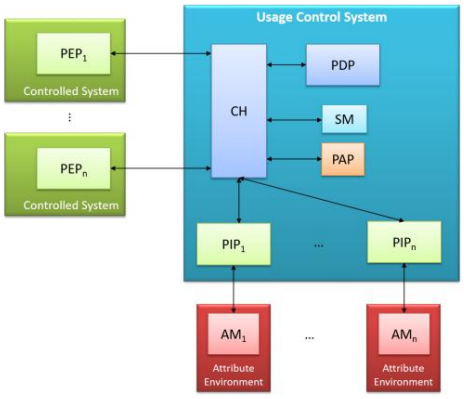

Fig. 1: Arquitectura del sistema de control de uso.

Este artículo tiene en cuenta los sistemas de control de uso basados en la arquitectura de referencia XACML [13], con especial referencia al que presentamos en [2], [10], que se muestra en la Figura 1. En la arquitectura de referencia XACML, los puntos de aplicación de políticas (PEP) integrados en el sistema controlado interceptan la ejecución de operaciones relevantes para la seguridad e invocan el controlador de contexto (CH), que es la interfaz del sistema de control de uso. Los Puntos de Información de Política (PIP) son los componentes invocados por el CH para recuperar los atributos requeridos por el Punto de Decisión de Política (PDP) para la ejecución del proceso de decisión, es decir, para evaluar la política recuperada del Almacén de Políticas (PS). Los atributos son administrados por administradores de atributos (AM), a veces llamados proveedores de atributos o almacenes de atributos, que proporcionan las interfaces para recuperar y, en el caso de atributos mutables, actualizar sus valores actuales. Cada escenario específico en el que se explota el sistema de control de uso requiere su propio conjunto de AM para gestionar los atributos necesarios para la evaluación de la política. Por lo tanto, los PIP están configurados adecuadamente para poder consultar los AM específicos adoptados en el escenario de interés para recuperar y actualizar atributos. En particular, cada PIP implementa el protocolo específico requerido para interactuar con el AM relacionado y explota los mecanismos proporcionados para proteger las comunicaciones. El modelo de control de uso enfatiza el papel de los PIP porque introduce la aplicación continua de políticas mientras el acceso está en progreso para hacer frente a atributos mutables. En particular, el PIP también es responsable de detectar cuándo cambia el valor de un atributo para desencadenar la reevaluación de la política para los accesos en curso involucrados, que son administrados por el Administrador de Sesión (SM). Para detectar cambios de atributos, el PIP podría aprovechar el mecanismo de suscripción proporcionado por el AM o el PIP debe emularlo si no es compatible con el AM.

Las fases del proceso de decisión de Control de Uso están reguladas por las interacciones entre el PEP y los sistemas de Control de Uso de la siguiente manera (derivado de [19]): TryAccess: es la fase previa a la decisión, que comienza cuando el mensaje Tryaccess es enviado por el PEP al sistema de Control de Uso porque un sujeto solicita ejecutar el acceso. La fase TryAccess finaliza cuando el sistema de Control de Uso envía la respuesta al PEP. Las posibles respuestas son: PERMITIR, permitir el acceso, o NEGAR;

StartAccess: es la primera parte de la fase de decisión en curso, que comienza cuando el mensaje StartAccess es enviado por el PEP al sistema de Control de Uso porque el acceso recién comienza y finaliza cuando se evalúa la política y se envía la respuesta de regreso al PEP;

RevokeAccess: esta es la segunda parte de la fase de decisión en curso. Esta fase se ejecuta cada vez que un atributo cambia de valor. Esta fase comienza cuando un atributo cambia de valor. Finaliza cuando se ha evaluado la política y, si se produce una violación de la política, el sistema de control de uso envía el mensaje RevokeAccess al PEP.

Cuando el sujeto intenta ejecutar una acción relevante para la seguridad a, el PEP suspende su ejecución y recupera la información relacionada con este acceso (ID del sujeto y del recurso, etc.). La PEP envía el mensaje TryAccess con los datos previamente recopilados al sistema del sistema de Control de Uso, que realiza el proceso de predecisión y devuelve el resultado a la PEP, quien lo hace cumplir. Si se permite la ejecución de una, el PEP

envía el mensaje StartAccess al sistema de control de uso tan pronto como se inicia, para iniciar la fase de decisión. Nuevamente, el sistema de Control de Uso realiza la primera evaluación de la política de control de uso. A partir de este momento, mientras la acción a esté en curso, el servicio de Control de Uso evalúa la política de Control de Uso cada vez que un atributo cambia su valor, y a esta fase la llamamos RevokeAccess. Si se viola la política, el sistema de Control de Uso envía el mensaje RevokeAccess al PEP, para que tome las contramedidas adecuadas.

III. EL MARCO DE LA UCIOT

Esta sección describe el marco propuesto que implementa el control de uso en los dispositivos IoT instalados en un escenario de hogar inteligente. Aunque el paradigma de Control de Uso ya se ha aplicado con éxito en varios escenarios, trayendo, por ejemplo, el Sistema de Control de Uso (UCS) en la Nube [2] y en dispositivos Android [11], el presente trabajo enfrenta el desafío de descentralizar las funcionalidades UCS proponen una arquitectura distribuida Peer-to-Peer (P2P) para dispositivos restringidos, como los de Smart Home. En particular, la mayoría de los dispositivos que estamos considerando (cámaras inteligentes, televisores inteligentes, termostatos inteligentes, luces inteligentes y hornos inteligentes) tienen capacidades de almacenamiento y potencia computacional limitadas, por lo que no son adecuados para manejar y evaluar todas las solicitudes que puedan ser emitido en un entorno doméstico inteligente, ni para almacenar la información necesaria para la reevaluación de políticas para todas las sesiones activas. Como se detallará a continuación, UCIoT replica parcialmente las funcionalidades del UCS en cada nodo P2P. Además, cualquier evaluación de políticas puede ser realizada por cualquiera de los nodos que puedan explotar los atributos recopilados de los otros nodos. Estas dos características permiten, respectivamente, tolerancia a fallas y una mayor complejidad para las políticas aplicadas.

Otra ventaja relevante de un sistema distribuido de este tipo se refiere a los aspectos de tolerancia a fallos, porque el fallo de uno (o varios) dispositivos puede tolerarse siempre que el sistema pueda ejecutarse en los restantes.

Además, al estar distribuido y explotar varios dispositivos, UCIoT permite que el marco de control de uso explote los datos recopilados de los sensores de todos estos dispositivos, permitiendo así definir políticas más complejas.

A. Arquitectura

La arquitectura del marco UCIoT, representada en la Figura 2, es un marco P2P distribuido, donde cada nodo representa un dispositivo inteligente conectado lógicamente a todos los demás a través de una tabla hash distribuida (DHT) [1]. En particular, el DHT explotado por UCIoT es el protocolo Cassandra [8], basado en una versión modificada del CHORD DHT, que también incluye una base de datos distribuida no relacional, utilizada por varias aplicaciones populares como Facebook.

Consideramos dispositivo inteligente cualquier electrodoméstico, sensor, actuador, placa, computadora, teléfono inteligente u otro dispositivo que ejecute un sistema operativo capaz de instalar y ejecutar aplicaciones de terceros. Cada dispositivo inteligente será considerado así un nodo P2P, mientras que los dispositivos conectados que no cumplan estas especificaciones serán considerados periféricos inteligentes o periféricos y

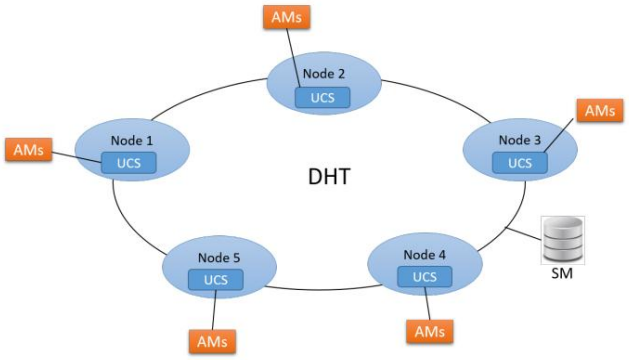

Fig. 2: Arquitectura lógica de UCIoT.

nodo específico, lógicamente incorporado en el propio nodo. Algunos elementos del entorno de los dispositivos inteligentes podrían incluso tener también una potencia computacional aceptable, una autonomía energética constante, aún con espacio de almacenamiento limitado. Dadas las capacidades de los dispositivos inteligentes, la UCS puede instalarse como una aplicación de terceros y ejecutarse en cualquier dispositivo inteligente en UCIoT.

En la capa de red y de enlace de datos, el protocolo de comunicación utilizado es el inalámbrico ad-hoc que, al estar completamente distribuido, no depende de la presencia de un único nodo enrutador, que podría representar un único punto de falla en la red. Así, gracias al protocolo DHT y a la conexión inalámbrica ad-hoc, la comunicación entre dispositivos inteligentes seguirá siendo posible incluso si un subconjunto limitado está apagado o no funciona correctamente. El protocolo de red ignora por completo las aplicaciones que se ejecutan en los dispositivos inteligentes, por lo que los dispositivos inteligentes intercambiarán mensajes como si estuvieran conectados a la misma LAN inalámbrica. En Como se anticipó, el

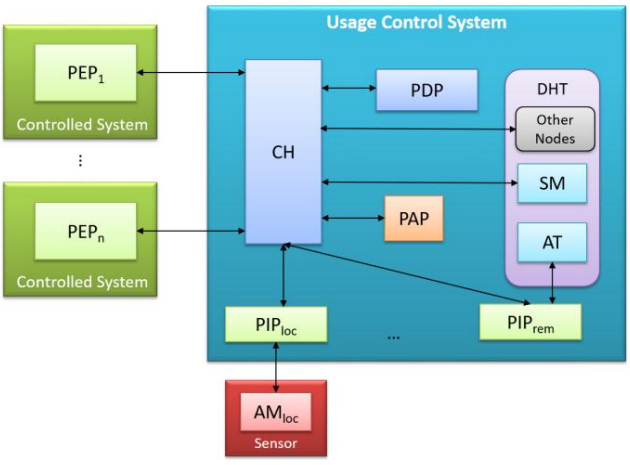

Fig. 3: Arquitectura lógica de un nodo UCIoT.

Las funcionalidades de UCS se replican en cada dispositivo, que así puede decidir permitir o denegar el acceso/uso a una operación o recurso que controla. Esta solicitud puede ser emitida por usuarios físicos, otros dispositivos inteligentes o periféricos y

se compara con las políticas de control de uso almacenadas localmente por el UCS. Más específicamente, cada dispositivo inteligente ejecuta (i) el Punto de administración de políticas (PAP) para almacenar políticas, (ii) una instancia del Punto de decisión de políticas (PDP) para hacer coincidir la solicitud y las políticas y decidir permitir o denegar el uso de recursos, (iii) un conjunto de puntos de información de política (PIP) para consultar atributos locales, (iv) un controlador de contexto (CH) para gestionar la interacción entre los distintos componentes. La principal diferencia con la arquitectura UCON presentada en nuestros trabajos anteriores radica en el Session Manager, para el cual UCIoT aprovecha la base de datos distribuida que ofrece Cassandra. De hecho, dado que los dispositivos inteligentes tienen un almacenamiento de memoria limitado y la cantidad de sesiones administradas probablemente no estará equilibrada entre los dispositivos en la configuración de un hogar inteligente, el conjunto de sesiones activas se guarda en el DHT, para utilizar de manera eficiente el almacenamiento de memoria global que ofrecen todos los dispositivos. Además, gracias al factor de replicación configurable, el SM no es propenso a problemas de punto único de falla. La otra diferencia principal con las arquitecturas UCON anteriores es la presencia de atributos remotos. Un atributo se considera local para un nodo cuando uno de los PIP que pertenecen al UCS de ese nodo consulta directamente al Administrador de atributos (AM) (consulte la Figura 2). Sin embargo, en UCIoT es posible que no todos los atributos necesarios para evaluar una política sean locales, es decir, algunos atributos son locales para otros dispositivos inteligentes y, por lo tanto, se consideran remotos. Para abstraer este procedimiento al UCS local del dispositivo de evaluación, se agrega un componente abstracto llamado PIP remoto a todos los UCS y se encarga de interactuar con el dispositivo inteligente remoto para recuperar el atributo. El PIP remoto recupera el identificador del nodo conectado físicamente al AM mediante la tabla de atributos almacenada en el DHT, luego recupera el atributo de acuerdo con el procedimiento descrito en la siguiente subsección. r, definimos dos estrategias distintas para recopilar los atributos remotos necesarios para realizar el proceso de decisión.

B. Flujo de

trabajo Las operaciones realizadas por UCIoT son equivalentes a las realizadas por el marco UCON descrito en la Sección II. La principal diferencia en el flujo de trabajo la introduce la presencia de atributos remotos y el Administrador de sesión distribuido. UCIoT ha sido diseñado con el objetivo de reducir al máximo las diferencias en el flujo de trabajo respecto al flujo de trabajo estándar UCON, a pesar de la distributividad de la arquitectura donde se aplica.

Como se anticipó, el SM se almacena en la base de datos Apache Cassandra DHT, lo que brinda las ventajas de un mayor espacio de almacenamiento, abordando así el problema de los dispositivos con memoria limitada, y garantiza la tolerancia a fallas gracias al factor de replicación, almacenando datos de sesiones relacionadas con no disponibles temporalmente. -Dispositivos compatibles. La base de datos DHT también enmascara la arquitectura distribuida subyacente de la base de datos al nivel de la aplicación, dejando así las funcionalidades del SM sin modificar.

Por otro lado, los atributos remotos introducen algunos desafíos adicionales, por lo que requieren una pequeña alteración del flujo de trabajo habitual y de la arquitectura UCON. Como se anticipó, el problema de los atributos remotos consiste en recolectar el valor de los atributos de los AM conectados a un nodo UCS diferente al quien evalúa la política. Para ello introducimos en cada nodo un componente que abstrae el procedimiento de recogida de los atributos remotos, denominado PIP remoto. Este PIP explota la tabla de atributos almacenada en el DHT que memoriza, para cada atributo, el nodo a cuyo AM está conectado físicamente. Por lo tanto, cuando se necesita un atributo remoto, el CH local llega a través del DHT al CH del nodo interesado (CH rem en la Figura 4), que le indicará al PIP local (PIP loc en la Figura 4) que recupere el valor del atributo, que luego se devuelve al CH local.

El flujo de trabajo completo de la recuperación de un atributo remoto se muestra en la Figura 4. En particular, la Figura 4 representa el

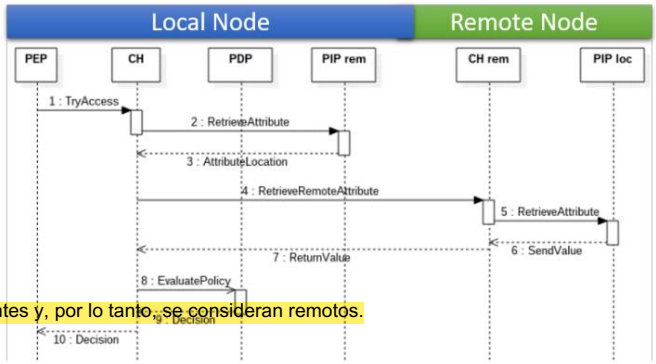

Fig. 4: Diagrama de secuencia de TryAccess con recuperación remota de atributos.

flujo de trabajo de la solicitud TryAccess emitida por el PEP, detallando el intercambio de mensajes entre los distintos componentes. En aras de la simplicidad y claridad de la representación, no se recuperan atributos locales en el flujo de trabajo representado. La recuperación remota de atributos se realiza también para StartAccess, que es idéntico a TryAccess en el flujo de trabajo, excepto por la adición de la sesión iniciada al SM distribuido y por la suscripción del CH remoto a los atributos mutables remotos. Por lo tanto, si un atributo remoto cambia su valor, se notifica al CH remoto y el nuevo valor se envía de vuelta al CH local para una reevaluación de la política y una posible revocación.

C. Implementación

La implementación actual del marco UCIoT consiste en una aplicación Java, enviada en forma de archivo jar, que se puede instalar en cualquier dispositivo que ejecute un JRE. Cuando se instala, la aplicación creará una instancia en los dispositivos del DHT para la base de datos distribuida y para manejar la comunicación con otros nodos que pertenecen a la misma red. La aplicación instalada incluye el UCS completo, donde el CH y el PDP no se modifican respecto al modelo UCON estándar.

El SM se instala en la base de datos distribuida manejada por el DHT, que también se encargará de la replicación de datos. La base de datos distribuida también alberga la tabla de atributos, que mantiene la correspondencia entre los atributos y el nodo al que está conectado físicamente el AM, por lo que cualquier nodo puede consultarla y posiblemente actualizarla. Como sucede en el marco estándar UCON, los PIP y PEP serán específicos del dispositivo,

para interconectarse con actuadores y sensores propios del dispositivo específico. Las instancias únicas de la aplicación UCIoT se pueden configurar dinámicamente y el código relacionado con PIP y PEP específicos se puede cargar en tiempo de ejecución aprovechando la reflexión de Java.

D. Casos de uso relevantes

Se pueden abordar con éxito varios escenarios distintos de Hogares Inteligentes aprovechando el marco propuesto. De hecho, se pueden definir y aplicar varios tipos de políticas a través del marco propuesto, como políticas de seguridad y protección, políticas de ahorro de energía, etc. Por ejemplo, se puede definir y aplicar la siguiente política de seguridad con el sistema propuesto: "El quemador de un horno inteligente puede estar encendido si el sistema está en buen estado y si hay al menos un adulto en la cocina cuando también hay un niño presente. "En la Figura 5 se muestra una representación esquemática de este caso de uso. Esta política tiene como objetivo proteger

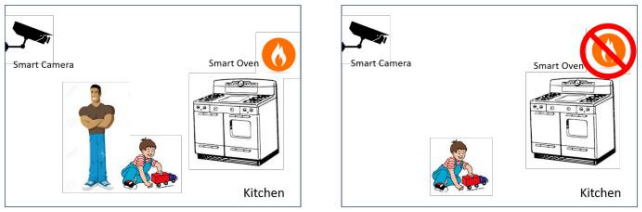

Fig. 5: Caso de uso de control parental y seguridad del horno inteligente.

la casa de fugas de gas y reduciendo la posibilidad de que un niño se lastime o queme al tocar el horno, debido a la falta de vigilancia de un adulto. Esta política de seguridad requiere la presencia de un horno inteligente[3] capaz de comprobar que no existen fallas físicas como fugas de gas, lo que hace cumplir la política de control de uso impidiendo el encendido de los quemadores y apagándolos, en caso de que la política no se cumpla. ya no.

También se requiere la presencia de una o más cámaras inteligentes en la cocina, para detectar quién está presente, lo que también es capaz de discernir entre niños y adultos. La implementación de UCIoT en este escenario específico considerará tanto el horno como la cámara inteligente como nodos de la arquitectura UCIoT. En particular, la política antes mencionada será evaluada y aplicada en el nodo del horno que incluirá así el PEP, cuyo actuador es la válvula controladora del quemador. Para evaluar la política será necesario un atributo local, es decir, el control sanitario para evitar fugas de gas, y un par de atributos remotos, es decir, el número de personas presentes en la cocina y el número de niños, que son proporcionados por la cámara.

Un ejemplo de política de ahorro de energía podría ser el siguiente: "La refrigeración por aire puede estar encendida si hay personas en la casa y no hay ventanas abiertas". Una representación esquemática se muestra nuevamente en la Figura 6. En este caso imaginamos como sensores un conjunto de cámaras y/o sensores de presencia para verificar que alguien está efectivamente presente en la casa, y sensores de ventanas para verificar si una o más ventanas están abiertas. Para implementar UCIoT en este escenario, la UCS será

[3]por ejemplo: http://www.dacor.com/Products/Ranges

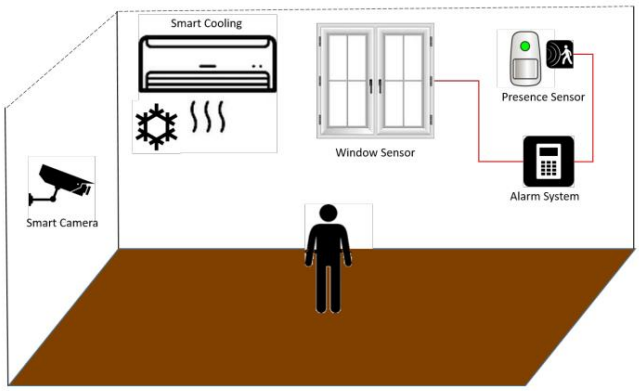

Fig. 6: Configuración de la política de ahorro de energía basada en el control inteligente de refrigeración por aire.

instalado dentro del sistema de refrigeración por aire, el sistema antirrobo de la casa y posiblemente en cámaras inteligentes. Por lo tanto, el PEP interactuará directamente con el controlador de refrigeración por aire, evitando que se encienda o apague cuando la política no coincida. El antirrobo recogerá la información de los sensores de presencia y de ventanas para comprobar que no hay ventanas abiertas y que efectivamente hay personas en la casa. Se proporcionará información adicional sobre la presencia de personas, según anticiparon las cámaras. Así, la política será evaluada y aplicada por el UCS local en el sistema de refrigeración por aire (aire acondicionado), explotando los atributos remotos recopilados por los PIP instalados en las cámaras y antirrobo.

IV. EVALUACIÓN EXPERIMENTAL

Como se muestra, al descentralizar las funcionalidades de UCON, permitimos la evaluación y aplicación continua de UCON.

políticas en dispositivos restringidos. La ejecución del marco UCON en dichos dispositivos y, en particular, el proceso de recuperación de atributos remotos de los otros nodos y datos de sesión del DHT, introduce una sobrecarga. Para cuantificar dichos gastos generales y evaluar su impacto en el uso en el sistema real, realizamos en nuestro banco de pruebas de referencia una serie de experimentos destinados a medir el rendimiento del sistema propuesto y los resultados se analizan a continuación.

A. El banco de pruebas

Como se anticipó, el banco de pruebas consta de cinco Raspberries PI 3 Modelo B y se muestra en la Figura 7. Como se muestra, tres dispositivos están equipados con el módulo Raspberry SenseHat, que se suma a las capacidades de detección de la placa simple, incorporando sensores de temperatura, humedad y presión. , magnetómetro, acelerómetro y giroscopio. El SenseHat también cuenta con una matriz de LED que se puede utilizar para mostrar mensajes. Los dos dispositivos restantes han sido equipados con un módulo de cámara Pi v2, añadiendo así la capacidad de recopilar imágenes y vídeos. La Raspberry PI 3 Modelo B tiene las siguientes características: 1 GB de RAM, procesador ARM de 1,2 GHz, núcleo de gráficos 3D VideoCore IV y tiene varias interfaces para comunicación inalámbrica, a saber, 802.11n WLAN, Bluetooth 4.0 y Bluetooth Low Energy (BLE). Para

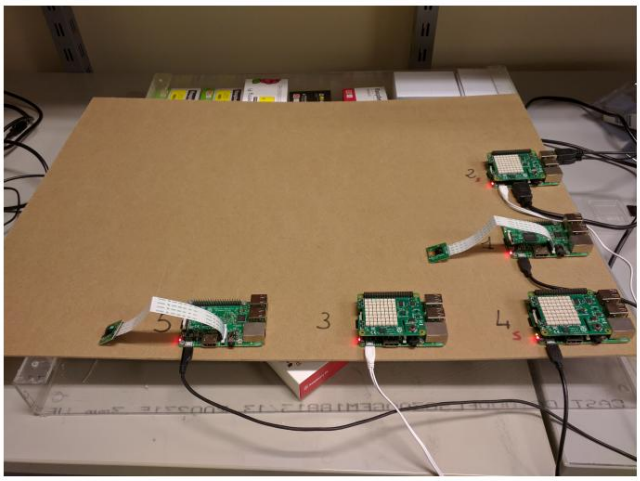

Fig. 7: Cinco Raspberries PI 3 utilizadas como banco de pruebas equipadas con SenseHat y Pi Camera.

En nuestro conjunto de experimentos, los dispositivos se interconectan a través de una red WiFi ad-hoc, constituyendo así un WANET con protocolo de enrutamiento AODV [4]. La elección de utilizar WiFi en lugar de Bluetooth se debe a la mayor fiabilidad y mayor velocidad del WiFi, que además no requiere fases de emparejamiento como ocurre en el Bluetooth. Además, al estar todos los dispositivos conectados a una fuente de alimentación, en el caso de uso del hogar inteligente no es necesario utilizar protocolos de bajo consumo. Esta hipótesis es sólida ya que en una casa inteligente, los electrodomésticos generalmente están conectados a la fuente de alimentación y no dependen de baterías. Sin embargo, vale la pena señalar que el marco UCIoT es completamente independiente del protocolo de enrutamiento y enlace de datos.

En nuestro banco de pruebas se utilizan sensores y cámaras como AM, que proporcionan los valores de los atributos que se utilizarán en las políticas. La matriz de LED se utiliza en cambio como actuador, por lo que es comandada por el PEP, mostrando una “P” cada vez que se inicia una sesión, es decir, el PDP devuelve la decisión PERMISO para un StartAccess y una “D” si el acceso es denegado o revocado.

B. Evaluación local

El primer conjunto de experimentos, cuyo desempeño temporal se muestra en la Figura 8, evalúa el tiempo requerido para ejecutar las fases del proceso de evaluación de políticas, a saber: Try-Access, Startaccess y RevokeAccess (ver Sección II). Los experimentos se han realizado aplicando políticas que incluyen atributos locales únicamente, variando el número de atributos requeridos para el proceso de decisión de 2 a 50. El tiempo de la fase TryAccess se mide desde el momento en que la solicitud de acceso es enviada por el PEP al CH hasta el momento en que la PEP reciba la respuesta (permitir/negar) a esta solicitud. Este tiempo afecta la experiencia del usuario porque es el retraso introducido por nuestro marco en la utilización del dispositivo doméstico inteligente. El tiempo de la fase StartAccess se mide desde el momento en que se activa la evaluación de la política por parte de la PEP porque recién se inició el acceso, hasta el momento en que la respuesta ha sido recibida por la PEP.

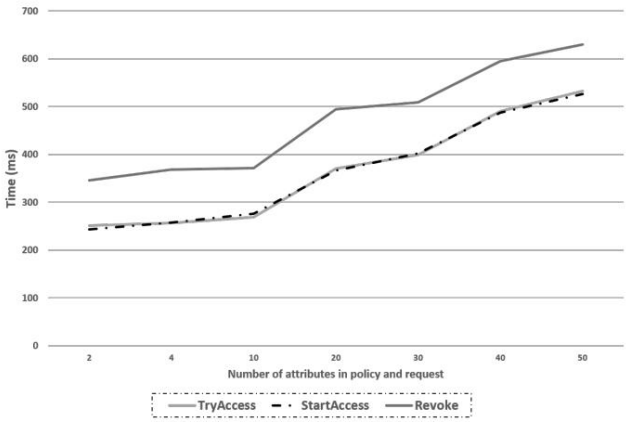

Fig. 8: Rendimiento con atributos locales.

El tiempo para realizar la fase RevokeAccess se mide desde el momento en que un determinado atributo cambia su valor hasta el momento en que la PEP recibe la respuesta de revocación.
Por lo tanto, también incluye el tiempo requerido por el PIP para detectar que un atributo cambió su valor. Las fases StartAccess y RevokeAccess no introducen directamente ningún retraso en la utilización del recurso porque se ejecutan mientras el acceso al recurso ya está en curso. Cuando se detecta una infracción de política, el tiempo requerido por la fase StartAccess o por la RevokeAccess representa el intervalo durante el cual se ha utilizado el recurso sin tener el derecho relacionado.

El eje X informa el número de atributos locales, nLA, mientras que el eje Y informa el tiempo en milisegundos. Para medir el tiempo que lleva la fase RevokeAccess, supusimos que solo una de las sesiones almacenadas en el DHT requiere la reevaluación de la política.

Los resultados de nuestros experimentos muestran que, en el caso de 2 atributos locales, el tiempo necesario para realizar la fase TryAccess es de 252 ms, la fase StartAccess tarda 247 ms mientras que la fase RevokeAccess requiere 346 ms. En el caso de 50 atributos locales, en cambio, la fase TryAccess tarda 533, la fase StartAccess requiere 528 ms, mientras que el tiempo para ejecutar la fase RevokeAccess es 630 ms. En primer lugar, notamos que el tiempo para ejecutar la fase TryAccess es bastante similar al tiempo requerido para la fase StartAccess. La razón es que los flujos de trabajo de las dos fases son muy similares, es decir, realizan prácticamente las mismas operaciones en el mismo orden. Por lo tanto, en el caso de atributos locales, el retraso introducido por el marco UCIoT no afectaría la experiencia del usuario y el uso del recurso se revocaría en poco tiempo en caso de infracción de la política. Además, la diferencia entre el tiempo del RevokeAccess y del TryAccess o StartAccess, para todos los valores de los atributos, se debe principalmente al tiempo necesario para detectar que un atributo cambia de valor.

C. Recuperación remota de atributos

El siguiente conjunto de experimentos, que se muestra en la Figura 9, tiene como objetivo evaluar el tiempo necesario para ejecutar TryAccess y RevocarAcceder a las fases del proceso de evaluación de políticas en caso

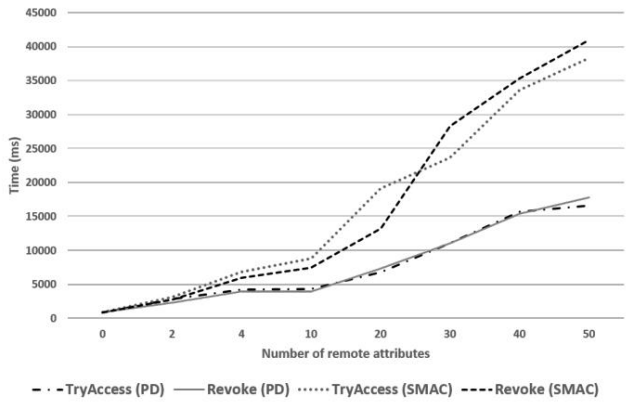

Fig. 9: Rendimiento con atributos locales y remotos.

de políticas que incluyen atributos remotos. Omitimos el tiempo requerido para la fase StartAccess ya que los resultados del primer conjunto de experimentos muestran que es bastante el mismo que el tiempo de la fase TryAccess. En este conjunto de experimentos, la política siempre incluye 50 atributos y variamos la proporción entre atributos locales y remotos. La política está escrita de tal manera que el valor de los 50 atributos siempre debe recopilarse y evaluarse para realizar el proceso de decisión, es decir, el PDP de UCIoT no puede realizar la fase de decisión explotando solo un subconjunto de los 50 atributos. Recordamos que las políticas reales incorporarían un número menor de atributos, por lo que esto puede considerarse como la prueba del peor de los casos. Por ejemplo, la primera política descrita en la Sección III-D incorpora solo 3 atributos.

En el eje X del gráfico de la Figura 9 informamos el número de atributos remotos en la política aplicada, nRA. Por tanto, el número de atributos locales nLA viene dado por (50 − X). También en este caso, para medir el tiempo que lleva la ejecución de la fase RevokeAccess, se supone que sólo una sesión requiere la reevaluación de la política. Los experimentos se han realizado aprovechando dos estrategias distintas para recuperar los atributos remotos descritos, a saber, impulsado por políticas (PD) y colección secuencial de atributos faltantes (SMAC). La primera estrategia consiste en concienciar al CH de la política a analizar, para tener una lista de los atributos que serán consultados por los distintos PIP. Por lo tanto, la lista de atributos se compara con la tabla de atributos almacenada por el control remoto PIP, para encontrar la lista de atributos remotos, que luego se consultan temporalmente.

En cambio, la estrategia SMAC no realiza la búsqueda en la tabla de atributos, simplemente envía la solicitud a los PIP locales para enriquecerla y luego le pide al PDP que evalúe la solicitud enriquecida. Por lo tanto, el PDP devolverá eventuales atributos faltantes presentes en la política, que se consultan mediante el control remoto PIP. Por tanto, esta estrategia tiene la ventaja de posponer la búsqueda en la tabla de atributos, lo que no se realiza en absoluto si todos los atributos necesarios son locales. En cambio, el vigía siempre está presente en la estrategia PD, que sin embargo muestra un mejor rendimiento cuando el número de atributos remotos es considerable. De hecho, en la estrategia SMAC, el PDP devuelve un atributo faltante cada vez, por lo que varias búsquedas y reevaluaciones de políticas importantes. Esta diferencia en el rendimiento se muestra bien en la Figura 9.

Observamos que, cuando todos los atributos son locales (es decir, nLA = 50,nRA = 0), el tiempo requerido para la ejecución de las fases TryAccess y RevokeAccess es, respectivamente, de aproximadamente 533 y 630 ms, sin diferencias mensurables entre los dos estrategias. Por tanto, el tiempo de vigilancia AT es insignificante.

En el caso de políticas con 10 atributos remotos (es decir, nLA = 40,nRA = 10), si adoptamos la estrategia PD el tiempo requerido para la ejecución de la fase TryAccess es de 4,2 segundos, mientras que el tiempo para la fase RevokeAccess es 3,9 segundos. Adoptando la estrategia SMAC los tiempos necesarios para la ejecución de las dos fases son, respectivamente, 8,8 y 7,7 segundos.
En cambio, cuando la política incorpora atributos remotos únicamente (es decir, nLA = 0, nRA = 50), el tiempo requerido para la ejecución de las fases StartAccess y RevokeAccess es, respectivamente, aproximadamente 16,6 y 17,8 segundos si adoptamos la estrategia PD, mientras que son, respectivamente, 38,3 y 40,9 segundos si adoptamos la estrategia SMAC.

D. Evaluación del retraso de la red

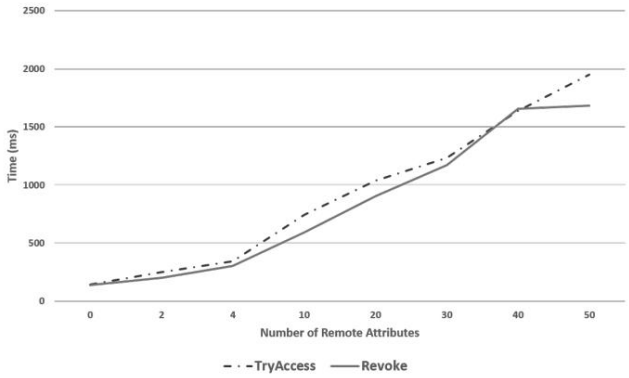

Fig. 10: Rendimiento en dispositivos emulados.

El objetivo del tercer conjunto de experimentos, que se muestra en la Figura 10, es evaluar la sobrecarga introducida por las comunicaciones a través de la red. En particular, instalamos los 5 nodos en distintas máquinas virtuales que se ejecutan en la misma máquina física y medimos el tiempo de ejecución de las fases TryaAccess y RevokeAccess en el caso de políticas que incluyan atributos remotos. En estos experimentos adoptamos la estrategia PD para la recopilación de atributos remotos.

Los resultados muestran que, incluso en el caso de 50 atributos remotos, el tiempo para realizar las fases TryAccess y RevokeAccess es inferior a 2 segundos. En comparación con los resultados de la Figura 9, estos experimentos confirman que el tiempo requerido para las comunicaciones a través de la red es el factor principal que afecta la sobrecarga introducida por el marco UCIoT.

V. TRABAJOS RELACIONADOS

La necesidad de introducir control de acceso en entornos IoT ha sido discutida recientemente por V. Cerf en [3], donde también propuso algunas direcciones basadas en la idea de introducir

Mecanismos de control y verificación en dispositivos de borde.
UCIoT amplía esta visión al introducir el modelo UCON más expresivo y presentar una arquitectura que se puede integrar fácilmente tanto en dispositivos internos como de borde, dado que tienen requisitos de dispositivos inteligentes. El autor de [18] presenta un modelo para incluir control de acceso en dispositivos restringidos para ser utilizados en IoT. El trabajo se centra principalmente en la introducción en el protocolo COAP de una superposición para garantizar el acceso autenticado. Por otro lado, UCIoT se centra en el control de uso actuando a nivel de aplicación, sin modificar el protocolo de comunicación estándar. En [17], se analiza una serie de desafíos para la seguridad y la privacidad en IoT. El documento no propone soluciones a los problemas presentados, pero respalda la afirmación de que una solución distribuida, como UCIoT, sería más eficaz en el entorno de IoT. El trabajo en [9] presenta una arquitectura distribuida y descentralizada para el control del uso de datos. A diferencia de UCIoT, este trabajo se centra en controlar el derecho de acceso y uso de datos en un entorno multidominio, intentando garantizar que las políticas se respeten incluso después de que los datos se muevan en un entorno diferente con diferentes mecanismos de control. El trabajo en [6] presenta una metodología para imponer el control del uso de datos en IoT, para permitir el intercambio de información protegida por políticas, explotando tecnologías web semánticas para derivar el contexto actual. Sin embargo, en este trabajo no se presenta una implementación real que, a diferencia de UCIoT, no se centra en el control de operaciones no relacionadas con el intercambio de datos. En cambio, en [7] se presenta una arquitectura que presenta UCS distribuido, donde diferentes UCS alojados en diferentes sistemas cooperan para encontrar el valor más confiable de un atributo común, cuyo AM no está disponible temporalmente.
En el artículo se presenta una aplicación al entorno de las ciudades inteligentes, aunque el enfoque de este artículo está en el algoritmo de reputación utilizado para medir la confiabilidad de la información intercambiada.

VI. CONCLUSIÓN Y TRABAJO FUTURO

Introducir la seguridad en IoT es una tarea desafiante, que debe hacer frente a la naturaleza distribuida, descentralizada y restringida de las arquitecturas de IoT. En este artículo, hemos propuesto a UCIoT un marco para introducir el control de uso en arquitecturas de IoT, que aprovecha la naturaleza distribuida de los sistemas de IoT para proporcionar un marco flexible, dinámico, tolerante a fallas y adaptable capaz de hacer cumplir las políticas de control de uso. El documento mostró cómo se puede aprovechar UCON para implementar políticas de seguridad y ahorro de energía y demostró su viabilidad mediante la implementación en dispositivos reales y experimentos de rendimiento. Varias direcciones de trabajo futuras podrían surgir de este prototipo; en particular, planeamos optimizar aún más el rendimiento para que el sistema sea capaz de manejar políticas con varios atributos remotos, al paralelizar el proceso de recuperación de atributos. Además, se realizarán experimentos adicionales en un entorno más grande y heterogéneo, evaluando el rendimiento con diferentes estrategias de enrutamiento y casos de uso de múltiples saltos.

RECONOCIMIENTO

Este trabajo ha sido financiado parcialmente por los proyectos financiados por la UE H2020 C3ISP, GA #700294, H2020 NeCS, GA #675320 y EIT Digital HII sobre gestión confiable de la nube.

REFERENCIAS

[1] Hari Balakrishnan, M. Frans Kaashoek, David Karger, Robert Morris e Ion Stoica. Búsqueda de datos en sistemas p2p. Comunitario. ACM, 46(2):43–48, febrero de 2003.

[2] Enrico Carniani, Davide D'Arenzo, Aliaksandr Lazouski, Fabio Martinelli y Paolo Mori. Control de uso en sistemas en la nube. Compensación de generación futura. Syst., 63:37–55, 2016.

[3] VG Cerf. Control de acceso e internet de las cosas. Internet IEEE Computing, 19(5):96–c3, septiembre de 2015.

[4] ID Chakeres y EM Belding-Royer. Diseño de implementación del protocolo de enrutamiento Aodv. En talleres de la 24ª Conferencia Internacional sobre Sistemas de Computación Distribuida, 2004. Actas., páginas 698–703, marzo de 2004.

[5] Maurizio Colombo, Aliaksandr Lazouski, Fabio Martinelli y Paolo Mori. Una propuesta para mejorar XACML con funciones de control de uso continuo. En Grids, P2P and Services Computing [Actas del taller del grupo de trabajo CoreGRID ERCIM sobre Grids, P2P and Service Computing, 24 de agosto de 2009, Delft, Países Bajos]., páginas 133–146, 2009.

[6] PK Das, S. Narayanan, NK Sharma, A. Joshi, K. Joshi y T. Finin. Seguridad basada en políticas sensibles al contexto en Internet de las cosas. En la Conferencia internacional IEEE sobre informática inteligente de 2016 (SMARTCOMP), páginas 1 a 6, mayo de 2016.

[7] Mario Faiella, Fabio Martinelli, Paolo Mori, Andrea Saracino y Mina Sheikhalishahi. Recuperación colaborativa de atributos en entornos con administradores de atributos defectuosos. En 11.ª Conferencia Internacional sobre Disponibilidad, Fiabilidad y Seguridad, ARES 2016, Salzburgo, Austria, 31 de agosto - 2 de septiembre de 2016, páginas 296–303, 2016.

[8] Dietrich Featherston. cassandra: Principios y aplicación. Departamento de Ciencias de la Computación Universidad de Illinois en Urbana-Champaign, 2010.

[9] Florian Kelbert y Alexander Pretschner. Una infraestructura de aplicación del control del uso de datos totalmente descentralizada. En Criptografía aplicada y seguridad de redes: 13.ª Conferencia Internacional, ACNS 2015, Nueva York, NY, EE. UU., 2 al 5 de junio de 2015, artículos seleccionados revisados, páginas 409 a 430, 2015.

[10] A. Lazouski, G. Mancini, F. Martinelli y P. Mori. Control de uso en sistemas en la nube. En la Séptima Conferencia Internacional sobre Tecnología de Internet y Transacciones Garantizadas (ICITST-2012), páginas 202–207, 2012.

[11] Aliaksandr Lazouski, Fabio Martinelli, Paolo Mori y Andrea Saracino. Control de uso de datos con estado para dispositivos móviles Android. Revista internacional de seguridad de la información, páginas 1 a 25, 2016.

[12] A. Mordeno y B. Russell. Gestión de identidades y acceso en Internet de las cosas: guía resumida, 2017. https://cloudsecurityalliance.org/download/identity-and-access-management-for-the-iot/.

[13] OÁSIS. Lenguaje de marcado de control de acceso extensible (XACML) versión 3.0, enero de 2013.

[14] J. Park y R. Sandhu. El modelo de control de uso de la UCONABC. Transacciones ACM sobre seguridad de sistemas e información, 7(1):128–174, 2004.

[15] J. Park, X. Zhang y R. Sandhu. Mutabilidad de atributos en el control de uso. En Direcciones de investigación en seguridad de datos y aplicaciones XVIII, IFIP TC11/WG 11.3 Decimoctava conferencia anual sobre seguridad de datos y aplicaciones, páginas 15 a 29, 2004.

[16] A. Pretschner, M. Hilty y DA Basin. Control de uso distribuido. Comunicaciones de la ACM, 49(9):39–44, 2006.

[17] Rodrigo Román, Jianying Zhou y Javier López. Sobre las características y desafíos de la seguridad y la privacidad en el Internet distribuido de las cosas. Computer Networks, 57(10):2266 – 2279, 2013. Hacia una ciencia de la seguridad cibernéticaArquitectura de seguridad e identidad para la futura Internet.

[18] Denis Sitenkov, Supervisores-Ludwig Seitz, Shahid Raza y Goran ¨ Selander. Control de acceso en el internet de las cosas. Tesis de maestría, 2014.

[19] Xinwen Zhang, Francesco Parisi-Presicce, Ravi Sandhu y Jaehong Park. Modelo formal y especificación de políticas de control de uso. Transacciones ACM sobre seguridad de sistemas e información, 8(4):351–387, 2005.