SOFTWARE: PRACTICE AND EXPERIENCE
Softw. Pract. Exper. (2016)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/spe.2437

Adaptable secure communication for the Cloud of Things

Valter Vasić[1,2], Aleksandar Antonić[1,*,†], Krešimir Pripužić[1], Miljenko Mikuc[1] and Ivana Podnar Žarko[1]

[1]*University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia*
[2]*Ericsson Nikola Tesla d.d., Krapinska 45, HR-10000 Zagreb, Croatia*

SUMMARY

Cloud of Things (CoT) is a novel concept driven by the synergy of the Internet of Things (IoT) and cloud computing paradigm. The CoT concept has expedited the development of smart services resulting in the proliferation of their real world deployments. However, new research challenges arise because of the transition of research-driven and proof-of-concept solutions to commercial offerings, which need to provide secure, energy-efficient, and reliable services. An open research issue in the CoT is to provide a satisfactory level of security between various IoT devices and the cloud. Existing solutions for secure CoT communication typically use devices with pre-loaded and pre-configured parameters, which define a static setup for secure communication. In contrast to existing pre-configured solutions, we present an adaptable model for secure communication in CoT environments. The model defines six secure communication operations to enable CoT entities to autonomously and dynamically agree on the security protocol and cryptographic keys used for communication. Further on, we focus on device agreement and present an original solution, which uses the Agile Cryptographic Agreement Protocol in the context of CoT. We verify our solution by a prototype implementation of CoT device agreement based on required security level, which takes into account the capabilities of communicating devices. Our experimental evaluation compares the average processing times of the proposed secure communication operations demonstrating the viability of the proposed solution in real-world deployments. Copyright © 2016 John Wiley & Sons, Ltd.

Received 8 October 2015; Revised 27 May 2016; Accepted 27 July 2016

KEY WORDS: Internet of Things; secure communication agreement; adaptable communication; secure communication model; cloud of things

1. INTRODUCTION

We are witnessing the rise of a novel concept, the Cloud of Things (CoT), which arises from the convergence of the Internet of Things (IoT) with the cloud computing paradigm. Networked devices, sensors, actuators, and wearables connect nowadays to the Internet, either directly or through intermediaries such as smartphones or gateways, to create smart spaces and heterogeneous environments of interlinked devices. IoT devices represent rich data sources about their surrounding environment with a potential to produce vast amounts of data that need to be processed and analyzed, often in real-time, emphasizing the need for novel solutions in the domain of big data processing [1]. Because cloud computing offers a utility-driven service model, which enables dynamic use of computing resources and adapts well to the requirements of a particular service [2], it represents an optimal environment for hosting back-end IoT platform components. Furthermore, the IoT and cloud benefit from the synergy of the two environments to facilitate utility-driven sensing and actuation services (*Sensing-as-a-Service*) [3] where devices are mapped to their virtual representations while many applications can share the available virtualized resources.

*Correspondence to: Aleksandar Antonić, University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia.
†E-mail: aleksandar.antonic@fer.hr

Transparent and secure access and usage of available IoT resources, as well as secure communication of IoT devices with cloud infrastructure is crucial to preserve citizen security and privacy in the CoT context. However, the integration of IoT spaces with the cloud has created a new set of challenges with regards to the control of data flow within CoT environments. IoT resource virtualization and the opening of data to third parties create new questions regarding data source ownership, verifiable data origin and data trustworthiness. Another serious problem is eavesdropping and monitoring without the knowledge of the observed person. Thus, security is a challenging problem for the CoT, a prerequisite that needs to be dealt with high caution before CoT platforms can reach a level of full economical exploitation.

Devices in the CoT can interconnect in an ad hoc fashion while also communicating with a back-end cloud. The communication needs to be secure, which requires an agreement protocol to enable communicating parties to agree on a cryptographic algorithm and keys used to protect the exchanged messages. Flexible and adaptable agreement mechanisms are needed in CoT environments because IoT devices have limited computing and bandwidth resources and thus cannot store all possible keys and support dynamical choice of the best cryptographic algorithm. Because CoT services are driven by the underlying data sources, it is critical to validate a sensor as a credible data source. For example, an electrical company needs to measure household energy consumption in a credible way and thus needs to validate data readings received from household smart meters. For some data readings, it is not even sufficient to validate their origin, but rather their content needs to be secured because messages contain private information, which should not be revealed to third parties. A good example are medical readings, which should be accessible only to physicians or patients.

In this paper, we present two major contributions: a model for secure communication within a CoT environment and prototype implementation of the agreement operation of the model. This model defines a minimal set of objects and operations that are needed to establish secure communication between various IoT devices and the cloud. It is based on six secure communication operations, namely agreement, sign, verify signature, hash-based message authentication code (HMAC), encrypt, and decrypt. The model describes devices by the following security properties: a set of available communication interfaces, hardware capabilities, public and private key pairs, and supported cryptographic algorithms. Note that trust establishment between devices and privacy related issues are out of scope of this paper. We rather focus on establishing a secure communication within the CoT. The model is in accordance with a comprehensive list of security risks given by the Open Web Application Security Project foundation for the year 2014 [4]. We address the following security risks in the paper: insufficient authentication, lack of transport encryption, and insufficient security configurability.

The agreement operation uses the Agile Cryptographic Agreement Protocol (ACAP) [5] to implement an adaptable security management agreement operation as defined in the secure CoT communication model.

ACAP is designed to be secure, adaptable, lightweight, and opportunistic. It provides the means for devices to securely exchange data while dynamically adapting to the required security level by using a cryptographic algorithm agreement procedure. This procedure ranks cryptographic algorithms based on valuation functions, which take into account both algorithm and device/environment properties. The solution is opportunistic in a sense that it does not require additional configuration before the start of communication. ACAP has been modeled and formally verified by using the Scyther verification tool [6, 7] to demonstrate the soundness of the implemented operation, consequently preventing security attacks on the proposed solution. To showcase that ACAP is usable in practice, we have verified its prototype implementation on a number of different devices. The performed experimental evaluation compares the average processing times of the proposed secure communication operations and demonstrates the viability of the proposed solution in real-world deployments.

The paper is structured in the following way: Section 2 provides description of the CoT environment with an emphasis on security issues. Section 3 presents the secure communication model for the CoT. The implementation details of the agreement operation as well as experimental evaluation of the six proposed secure communication operations is provided in Section 4. Section 5 provides a

DOI: 10.1002/spe

brief overview of related work addressing the field of CoT/IoT security, while Section 6 concludes the paper and gives directions for future work.

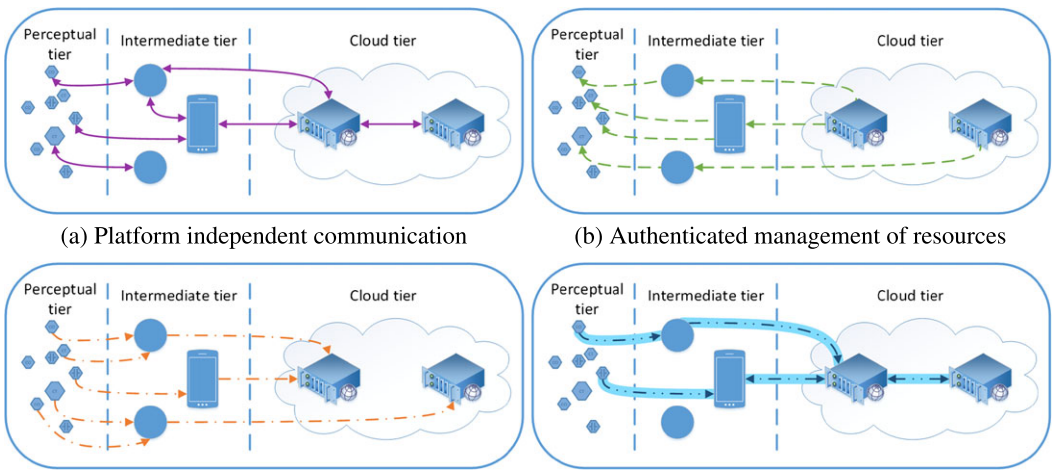
2. SECURE COMMUNICATION SCENARIOS IN THE CLOUD OF THINGS

The CoT ecosystem can be described as tiered architecture composed of various devices interconnected through different networked environments. The architecture is depicted by a number of scenarios in Figure 1 and represents instances of a tiered IoT architecture integrated with the cloud presented in [8]. The architecture consists of the following three main tiers:

- *Perceptual tier (sensors and actuators)* represents a data source level composed of low-power constrained devices that can sense their surroundings to generate data or execute simple tasks.
- *Intermediate tier (sink nodes and rendezvous collection points)* is a data forwarding level composed of devices with limited processing capabilities that collect and aggregate data from the perceptual tier, perform preprocessing tasks before forwarding the data to the cloud tier, and remotely manage devices of the perceptual tier.
- *Cloud tier (database and application servers)* represents a service level which offers computing resources as a utility to perform the processing/storage of data collected at the perceptual tier and preprocessed at the intermediary tier. The data at the cloud tier can be classified as big data, which is used to create information and knowledge, that can trigger further actions at both the intermediary and perceptual tiers, for example, delivery of information to smartphones or actuation tasks.

A wide adoption of IoT solutions and their integration with the cloud puts security issues in the spotlight of any CoT system design. We identify key communication scenarios that affect the security of the entire CoT ecosystem and focus on the process of establishing secure communication between devices, which 1) integrates the functionality to verify a data source and 2) enables the exchange of data between devices while respecting the required privacy level. The identified communication scenarios are shown in Figure 1.

Platform independent communication. Our secure communication model for CoT enables secure message exchange between devices running on different IoT platforms. A platform independent and adaptable agreement protocol is required to secure the communication between different devices across platforms. Such agreement protocol should exchange cryptographic algorithms and keys to enable ad hoc secure communication in a dynamic networked environment. In other words, devices should support the possibility to dynamically change the length of cryptographic keys



(a) Platform independent communication (b) Authenticated management of resources

(c) Reliable data collection (d) Sensitive data protection

Figure 1. Cloud of Things communication scenarios.

DOI: 10.1002/spe

(in order to achieve the required level of security) and also support the replacement of vulnerable cryptographic algorithms. Additionally, each message should be stand-alone and provide all needed information (e.g., a message that reports a single sensor reading) and protection within itself. In our secure communication model, the messages can either be exchanged between two parties or routed through the entire communication architecture as shown in Figure 1(a) (e.g., messages from the cloud tier can be forwarded through the intermediate tier to their destinations in the perceptual tier).

Authenticated management of resources. All messages used for management of resources, either at the perceptual or intermediate tier, should be authenticated in order to properly administer a large IoT deployment. These messages are usually sent from a higher to lower tier as demonstrated in Figure 1(b). While exchanging management messages, a producer (i.e., sender) needs to digitally sign its messages so that a receiver can verify the origin. A digital signature gives proof of both message integrity and origin thus enabling non-repudiation of management messages. Because digital signatures require costly computations, they should be used only for secuting critical operations. The examples of such operations are as follows: (i) a sensor reading request which is important for charging an IoT service; (ii) power-saving instructions that change a reading frequency of a sensor node so that malicious users cannot tamper with the data acquisition process; and (iii) support of cryptographic algorithm configuration updates that need to be received from a verified source.

Reliable data collection. Usually, when the data flow is directed from lower tiers towards higher tiers, it is possible for an attacker to send false or inaccurate data to higher tier devices. In our CoT architecture, this is prevented by ensuring the integrity of sent data, using digital signatures or HMAC [9] protection. Therefore, one of the main benefits of our CoT architecture is the data trustworthiness, which is enabled by reliable data collection, shown in Figure 1(c). Because digital signing is a costly operation, we recommend HMAC operation as a lightweight integrity protection mechanism for devices from the perceptual tier, which have limited hardware capabilities and prefer lightweight operations due to energy conservation. For example, HMAC operation can be used for data collection from sensors that do not measure sensitive information (e.g., air or water quality measurements, which are of public interest but have to be verifiable). The HMAC operation ensures integrity of data from its source to destination but does not explicitly provide proof of origin. However, because devices from the perceptual tier are designated data sources, this operation implicitly provides the proof of origin between different tiers of our CoT architecture.

Sensitive data protection. Sometimes the collected data can be sensitive and therefore should be protected from information leakage as illustrated in Figure 1(d). This can happen in communication within or between all tiers in the CoT architecture. Although sensors usually collect non-sensitive data, certain use cases, for example, health monitoring applications, must not leak any measured information. Moreover, certain corporate applications could also have a requirement that all data exchanged within a CoT environment needs to be secret. Furthermore, communication between cloud instances usually includes sensitive information about users (e.g., user preferences or recommendations). In our CoT architecture, sensitive data protection is achieved by using symmetric encryption with a previously exchanged shared key.

3. SECURE CLOUD OF THINGS COMMUNICATION MODEL

The secure communication model for the CoT is defined as a sextuple S_{CoT}:

$$S_{CoT} = \{D, K, L, C, A, O\}, \tag{1}$$

where D is a set of all devices, K is a set of all key pairs (public and private keys) of all devices, L is a set of all layers on which devices can communicate, C is a set of all possible device hardware capabilities, A is a set of cryptographic algorithms, which devices can use, and O is a set of secure communication operations between devices.

We distinguish three different types of cryptographic algorithms in the set $A = \{A^h, A^s, A^p\}$: cryptographic hash algorithms A^h, secret key algorithms A^s, and public/private key algorithms A^p.

DOI: 10.1002/spe

A device D_i in our model is represented with its set of supported communication layers L_i, set of its hardware capabilities C_i, set of supported cryptographic algorithms A_i, and set of public and private key pairs K_i:

$$D_i \in D : \{D_i = \{L_i, C_i, A_i, K_i\}, L_i \subseteq L, C_i \subset C, A_i \subseteq A, K_i \subset K\}, \tag{2}$$

where $A_i = \{A_i^h, A_i^s, A_i^p\}$ such that $A_i^h \subseteq A^h$, $A_i^s \subseteq A^s$ and $A_i^p \subseteq A^p$.

Each key pair $k(a_i^p) \in K_i$ is generated to match the corresponding public/private key algorithm $a_i^p \in A_i^p$. A key pair $k(a_i^p)$ contains a private key $pri(a_i^p)$ and corresponding public key $pub(a_i^p)$:

$$k(a_i^p) = \{pri(a_i^p), pub(a_i^p)\}. \tag{3}$$

A set of device's public keys must be transferred to the other device when initiating secure communication between them. The set of public keys PK_i of a device D_i is defined as

$$PK_i = \{pub(a_i^p) \in k(a_i^p) : k(a_i^p) \in K_i\}. \tag{4}$$

In our model, two devices D_i and D_j can communicate only if the following holds, $L_i \cap L_j \neq \emptyset$. This means that devices D_i and D_j share a common communication layer, which can be used for secure communication. A communication layer defines the type of interface (wired or wireless) and network protocols that can be used for data exchange between devices (Bluetooth, Zigbee, WLAN, Ethernet, TCP/IP, etc.). Note that a typical device from the perceptual tier has a wireless interface (e.g., Bluetooth or Zigbee) for interaction with an intermediate tier device, while the latter device uses a high speed communication interface (e.g., WLAN or Ethernet) to communicate with the cloud tier.

If two devices share a common communication layer, they can communicate independently of the tiers they belong to. Tiers are used to classify devices based on their capabilities. They are not meant to impose any limitations other than the shared communication layer.

To secure communication between devices in our model, we define a set of operations O, which is composed of the six secure communication operations that are used for agreement, data integrity, authentication, and data secrecy.

$$O = \{agreement, sign, verify\_signature, hmac, encrypt, decrypt\} \tag{5}$$

The selection of a cryptographic algorithm and shared secret agreement operation ($agreement$) is a prerequisite for secure communication between an initiator D_i and responder D_j and needs to be conducted before all other operations. As a result of this operation, both devices agree upon a cryptographic algorithm triplet ($\tilde{A}_{ij}$) to be used on both devices, shared secret key (S_{ij}) that can be used to protect data and public key for each device:

$$agreement : (D_i, D_j) \longmapsto \{\tilde{A}_{ij}, S_{ij}, pub(a_i^p), pub(a_j^p)\}, \tag{6}$$

where $\tilde{A}_{ij} = \{\tilde{A}_{ij}^h, \tilde{A}_{ij}^s, \tilde{A}_{ij}^p\}$ such that $\tilde{A}_{ij}^h \in A_i^h \cap A_j^h$, $\tilde{A}_{ij}^s \in A_i^s \cap A_j^s$, and $\tilde{A}_{ij}^p \in A_i^p \cap A_j^p$.

In other words, the agreed cryptographic hash, secret key, and public/private key algorithms must be supported by both devices. The public keys $pub(a_i^p)$ and $pub(a_j^p)$ correspond to the selected public/private key algorithm $\tilde{A}_{ij}^p$ so that $a_i^p = a_j^p = \tilde{A}_{ij}^p$. The shared secret key S_{ij} is calculated by using the Diffie–Hellman (DH) key exchange, which is described in Section 4.

Our model is flexible as it supports different procedures for selecting $\tilde{A}_{ij}$ among cryptographic algorithms supported by both devices. The selection procedure depends on a desired criterion such as the required level of security, as we explain in Section 4.2.

The model supports the following secure communication operations, which can be performed only after a successful $agreement$ operation:

- Data integrity and authentication operation ($sign$) is used to digitally sign $data$, which is exchanged between a sender D_i and receiver D_j

DOI: 10.1002/spe

$$sign : [data, \tilde{A}^h_{ij}, \tilde{A}^p_{ij}, pri(a^p_i)] \longmapsto \{signature\}. \quad (7)$$

The signature is derived from data by calculating its hash using the selected hash algorithm $\tilde{A}^h_{ij}$ and then by signing the calculated hash using the algorithm $a^p_i = \tilde{A}^p_{ij}$ and its corresponding private key $pri(a^p_i)$. The device D_j verifies the received signature using the $verify\_signature$ operation:

$$verify\_signature : [\{data, signature\}, \tilde{A}^h_{ij}, \tilde{A}^p_{ij}, pub(a^p_i)] \longmapsto [true, false]. \quad (8)$$

The receiver D_j verifies the received $signature$ deriving the hash from $signature$ using the algorithm $a^p_j = \tilde{A}^p_{ij}$ and corresponding public key $pub(a^p_i)$, and then compares it to the hash value calculated on device D_j from received $data$ using the algorithm $\tilde{A}^h_{ij}$.

- Hash-based message authentication code data authentication and integrity operation (*hmac*) is based on HMAC [9] function and is used as a lightweight data protection mechanism when exchanging $data$ between a sender D_i and receiver D_j:

$$hmac : (data, \tilde{A}^h_{ij}, S_{ij}) \longmapsto \{authentication\_code\} \quad (9)$$

The $authentication\_code$ is calculated using the selected hash algorithm $\tilde{A}^h_{ij}$ on $data$ and previously agreed secret key S_{ij}. The device D_j compares the received $authentication\_code$ with the $authentication\_code$ calculated using the same *hmac* operation (as the sender D_i). The verification is successful when the calculated and received $authentication\_code$ match.

- Data secrecy operations *encrypt* and *decrypt* are used to ensure that the $data$ exchanged between a sender D_i and receiver D_j is known only to them. The $encrypt$ operation uses the agreed secret algorithm $\tilde{A}^s_{ij}$ and key S_{ij} to encrypt $data$:

$$encrypt : (data, \tilde{A}^s_{ij}, S_{ij}) \longmapsto secret\_data. \quad (10)$$

The original $data$ can be decrypted from the received $secret\ data$ using the $decrypt$ operation:

$$decrypt : (secret\_data, \tilde{A}^s_{ij}, S_{ij}) \longmapsto data. \quad (11)$$

The secure CoT communication model represents a minimal set of objects and secure communication operations, which are required to secure communication between different devices and/or tiers in the CoT environment. Even though this model does not explicitly define an end-to-end secure communication, this is achieved by combining multiple, point-to-point, communication channels between two devices. It requires a that a single device is the initiator in one channel and the receiver in an other channel. In a standard setup, a sensor would be the initiator of the communication with a responding intermediate tier device, whereas the intermediate tier device would also be the initiator for transferring data to a cloud tier device. The same channels but with reversed roles would be needed for transferring an authenticated management message from the cloud tier.

4. PROTOTYPE IMPLEMENTATION AND EVALUATION

In this section, we first present a prototype implementation of the $agreement$ operation in the form of the ACAP [5]. Additionally, we explain the cryptographic algorithm agreement procedure (CAAP), which is a key part of the $agreement$ operation. This procedure extends our universal communication model S_{CoT} with the ability to adapt to the required security level taking into account current device capabilities. After that, we give a short security analysis of the protocol on which the agreement is based. Finally, we experimentally evaluate the processing time of all six proposed secure communication operations and comment the results.

DOI: 10.1002/spe

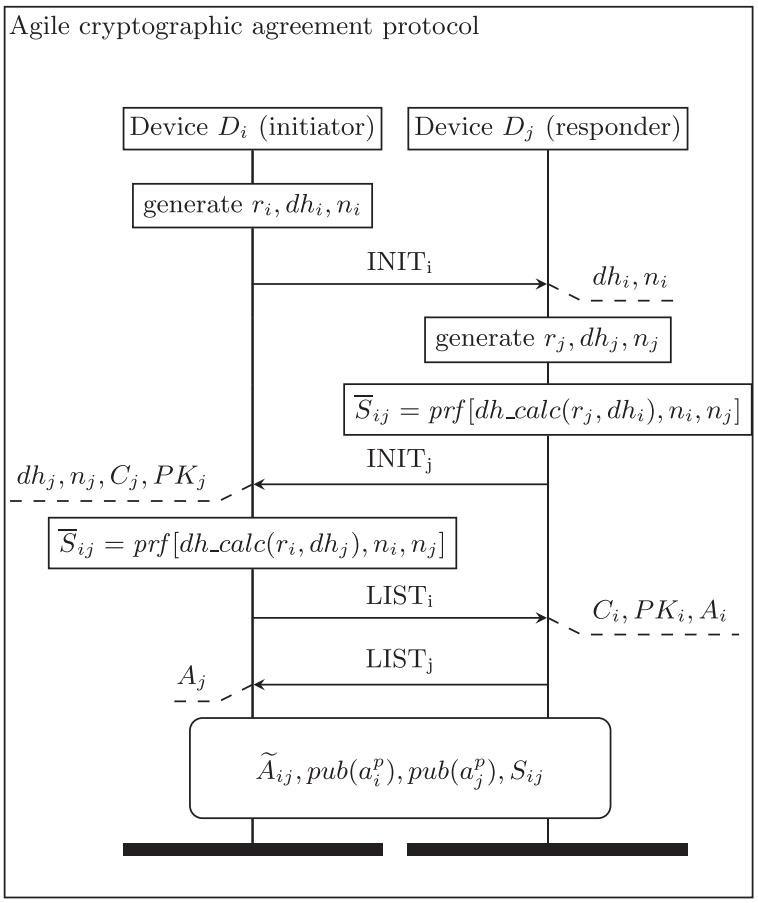


Figure 2. Agile cryptographic agreement protocol message exchange diagram.

4.1. Agreement operation implementation

As previously stated, when two devices D_i and D_j want to communicate securely, they first need to perform the $agreement$ operation. While performing this operation, they agree upon a cryptographic algorithm triplet $\tilde{A}_{ij}$, a shared secret S_{ij} that can be used to protect data and a public key for each of them, as defined in relation (6).

During this operation, four messages need to be exchanged between an initiator D_i and responder D_j, as shown in Figure 2. The figure also shows the following: (i) the data flow between devices; (ii) functions that need to be performed during the exchange (shown in rectangles); and (iii) the final result of the $agreement$ operation (shown in the rounded rectangle at the bottom).

Device D_i initiates the $agreement$ operation by sending an INIT$_\mathrm{i}$ message to device D_j. This message contains a public part of DH[‡] dh_i and nonce[§] n_i. A DH public part dh_i is calculated from an appropriate random number[¶] r_i, which was previously generated by D_i. Nonce n_i is an arbitrary number that may only be used for a single $agreement$ operation.

Upon receipt of an INIT$_\mathrm{i}$ message, device D_j generates[‖] its own (appropriate) random number r_j, calculates a DH public part dh_j from r_j and generates a nonce n_j. After that, device D_j can

[‡]A Diffie–Hellman key exchange is used to exchange a shared secret between devices. [10]
[§]Cryptographic nonce is used to differentiate between two separate agreements and must not be reused.
[¶]Note that different constraints on the key generation process can be enforced, based on the type of DH key exchange. (i.e., standard discrete logarithm DH or elliptic curve DH)
[‖]The generation can be performed periodically in the background to make the agreement faster and less vulnerable to resource exhaustion attacks.

DOI: 10.1002/spe

perform a pseudo-random function *prf*[**] to create a session key $\bar{S}_{ij}$ that will be used to secure the rest of the $agreement$ operation:

$$\begin{aligned} dh\_secret &= dh\_calc(r_j, dh_i) = dh\_calc(r_i, dh_j) \\ \bar{S}_{ij} &= prf(dh\_secret, n_i, n_j), \end{aligned} \tag{12}$$

where dh_calc represents a Diffie-Hellman calculation function that produces a DH shared secret dh_secret. A session key $\bar{S}_{ij}$ is calculated with *prf* function on dh_secret and nonces n_i, n_j.

After that, device D_j sends the INIT_j message, which contains a DH public part dh_j, nonce n_j, list of its hardware capabilities C_j, and list of its public keys PK_j.

Upon receipt of an INIT_j message, device D_i can calculate the same shared DH secret and $agreement$ operation session key $\bar{S}_{ij}$ as device D_j, as shown in Equation (12). Device D_i then sends a LIST_i message that contains a list of its hardware capabilities C_i, list of its public keys PK_i, and list of supported algorithms A_i. Upon the receipt of this message, device D_j replies with a LIST_j message which contains the list of supported algorithms for device D_j.

During the message exchange, both devices obtain a list of supported cryptographic algorithms $\tilde{A}_{ij}$, which is a result of the $algorithm\_agreement$ function on A_i and A_j:

$$algorithm\_agreement(A_i, A_j) = \tilde{A}_{ij}. \tag{13}$$

The resulting keys $pub(a_i^p)$ and $pub(a_j^p)$ are selected from PK_i and PK_j, respectively to match the agreed public key algorithm $a_i^p = a_j^p = \tilde{A}_{ij}^p \in \tilde{A}_{ij}$.

Furthermore, devices D_i and D_j obtain a shared secret key S_{ij} that is different (and computationally independent) from agreement session key $\bar{S}_{ij}$ because it is calculated using some additional agreement data (e.g., the resulting list of agreed algorithms $\tilde{A}_{ij}$):

$$S_{ij} = prf(dh\_secret, n_i, n_j, \tilde{A}_{ij}). \tag{14}$$

Note that the length of secret key S_{ij} must match the agreed secret key algorithm $\tilde{A}_{ij}^s \in \tilde{A}_{ij}$.

Agreement messages. The four messages exchanged in the $agreement$ operation are defined as follows:

$\text{INIT}_\text{i}(D_i \rightarrow D_j)$: dh_i, n_i

$\text{INIT}_\text{j}(D_j \rightarrow D_i)$: $dh_j, n_j, C_j, PK_j, sign\{(dh_j), \bar{A}^h, \bar{A}^p, pri(a_j^p)\}$, $hmac\{(C_j, PK_j, n_i, n_j), \bar{A}^h, \bar{S}_{ij}\}$

$\text{LIST}_i(D_i \rightarrow D_j)$: $C_i, PK_i, A_i, sign\{(A_i, dh_i, dh_j), \bar{A}^h, \bar{A}^p, pri(a_i^p)\}$, $hmac\{(C_i, PK_i, n_j, n_i), \bar{A}^h, \bar{S}_{ij}\}$,

$\text{LIST}_j(D_j \rightarrow D_i)$: $A_j, sign\{(dh_i, dh_j, n_i, n_j, A_j), \bar{A}^h, \bar{A}^p, pri(a_j^p)\}$.

As we can see, besides required data that is transferred to the other side, these messages also contain results of $hmac$ and $sign$ operations introduced in relations (7) and (9), which are necessary to secure the communication between devices D_i and D_j.

To protect the $agreement$ operation from network attacks, a default cryptographic hash algorithm $\bar{A}^h$ and public key algorithm $\bar{A}^p$ are used in combination with the $agreement$ session key $\bar{S}_{ij}$ introduced in Equation (12). Our model requires that both default algorithms, $\bar{A}^h$ and $\bar{A}^p$, are supported across all devices in the architecture:

$$\forall D_i \in D : (\bar{A}^h \in A_i) \wedge (\bar{A}^p \in A_i). \tag{15}$$

Additionally, it requires that keys $pri(a_i^p)$ and $pri(a_j^p)$ used in $sign$ operations, while creating the messages, must match the default public key algorithm $a_i^p = a_j^p = \bar{A}^p$.

[**] A function that uses iterative hashing to produce a pseudo-random result, based on a given input (i.e., seed).

DOI: 10.1002/spe

4.2. Cryptographic algorithm agreement procedure (CAAP)

The CAAP, as a part of the $agreement$ operation, is defined in Algorithm 1. The CAAP procedure takes algorithm lists of both devices (line 1), that is, A_i for D_i and A_j for D_j. After that, for each algorithm $type$[††] (lines 2–12), it selects the algorithm with the highest score (lines 3–8) given by function $score(a^{type}, C_i, C_j, slev)$, based on the hardware capabilities C_i for device D_i, C_j for device D_j, and the required security level $slev$. Finally, it returns a triplet of selected cryptographic algorithms $\tilde{A}_{ij}$ (line 13) or an empty set otherwise (line 10).

Algorithm 1 Cryptographic algorithm agreement procedure

```
1:  procedure CRYPTO–AGREEMENT(C_i, C_j, A_i, A_j, slev)
2:      for each type ∈ {h, s, p} do
3:          for each a^type ∈ (A_i^type ∩ A_j^type) do
4:              if (Ã_ij^type == ∅) or [score(a^type, C_i, C_j, slev) > score(Ã_ij^type, C_i, C_j, slev)]
    then
5:                  Ã_ij^type = a^type
6:                  break
7:              end if
8:          end for
9:          if Ã_ij^type == ∅ then
10:             return ∅                          ▷ The negotiation was unsuccessful.
11:         end if
12:     end for
13:     return Ã_ij = {Ã_ij^h, Ã_ij^s, Ã_ij^p}
14: end procedure
```

The reason for making decisions with regard to scoring of supported cryptographic algorithms is in determining which of them are of higher value compared with others in terms of their suitability for a secure communication task. The score of an algorithm may be directly linked to the required level of security of a communication task and hardware capabilities of devices because of their limited computing, bandwidth, and power resources.

Hence, the $scores$ of different cryptographic algorithms may be calculated using the so-called algorithm valuation functions, which are used to rank algorithms from 'best' to 'worst'. More formally, we consider the overall score of algorithm a to be $score(a, C_i, C_j, slev)$, expressed as a weighted combination of multiple value dimensions, namely: level of security $slev$, algorithm complexity $ac(a)$, and currently available battery power bp, bandwidth bw, and processing power pp of both devices:

$$score(a, C_i, C_j, slev) = f[w_0 \cdot slev, w_1 \cdot ac(a), w_2 \cdot \min(bp_i, bp_j), w_3 \cdot \min(bw_i, bw_j), w_4 \cdot \min(pp_i, pp_j)], \tag{16}$$

where C_i and C_j are defined as $\{bp_i, bw_i, pp_i\}$ and $\{bp_j, bw_j, pp_j\}$, respectively. Different weights w_x indicate the relative importance of a given value dimension contributing to the overall algorithm score and may be considered to be application specific.

The security level is determined by the current communication scenario and the environment in which the communication is performed. Communication in the cloud tier has a greater security level than the communication in the perceptual tier because the amount of information transferred is much larger and because the platform permits the use of more secure cryptographic algorithms. The security level in inter-tier communication is also decided by the communication scenario and affected by the distance the data needs to travel, because it increases the attack possibilities.

[††] Where types h, s, and p indicate cryptographic hash algorithms A^h, secret key algorithms A^s, and public/private key algorithms A^p, respectively.

DOI: 10.1002/spe

4.3. Agreement protocol verification

The main requirement for a secure agreement protocol is cryptographic agility. Cryptographic agility represents the possibility to refresh the cryptographic keys and algorithms used during communication. Most modern secure communication protocols like IPsec [11], SSL [12]/TLS [13], and SSH [14] implement cryptographic agility and provide secure communication between two parties. However, these protocols are fairly complex and consist of a number of components that need to interoperate in order to provide secure communication channels. Even though solutions implementing these protocols exist on more capable devices, implementation and deployment of such complex protocols cannot be performed on perceptual tier devices in an energy and resource efficient way. Furthermore, secure channels are not a requirement in the CoT environment because all communication can be conducted by using message based communication. In order to provide a unified solution for all tiers in the CoT architecture, we used ACAP in complement with other secure operations which are based on safe cryptographic primitives like cryptographic hashes, digital signatures, and symmetric encryption.

Agile cryptographic agreement protocol [5] is a lightweight solution that is completely formally verified,[‡‡] easy to implement, and deployed even on hardware with limited resources. It consists of only four messages and is based on the SIGMA (sign and mac) principle [15] for securing the message exchange. ACAP employs various security mechanisms both from the design and implementation perspective. It has been designed to prevent the man in the middle and late replay attacks, which is proved by the performed formal security verification. From the implementation perspective, it provides perfect forward secrecy by using computationally independent keys for negotiation and traffic protection, and it mitigates denial of service attacks by precomputing the most computationally expensive message parts. Additionally, the trust establishment can be performed by using the public key infrastructure or by previously distributing public keys in a controlled environment.

4.4. Experimental evaluation

We have implemented and evaluated the performance of all six secure communication operations, which are defined in the secure CoT communication model, on devices from different tiers. For measuring each operation duration, we have used 1KB (1024 bytes) of exchanged data and performed 1000 iterations of the experiment. The following devices were used in our experiments: Arduino Yún for the perceptual tier, Raspberry Pi for the intermediate tier, and a virtual machine instance running on a XenServer platform for the cloud tier.

The Arduino Yún tests were performed on the Linux MIPS coprocessor Atheros AR9330 @ 400 Mhz, Raspberry Pi tests were run on an ARM v6 @ 700 Mhz, whereas the XenServer instance tests were performed on an Intel Xeon E5-2680 @ 2.7 Ghz. The prototype was implemented in Python, and the same code was used across all platforms. The average duration of the six operations defined in Section 3 are shown in Figure 3.

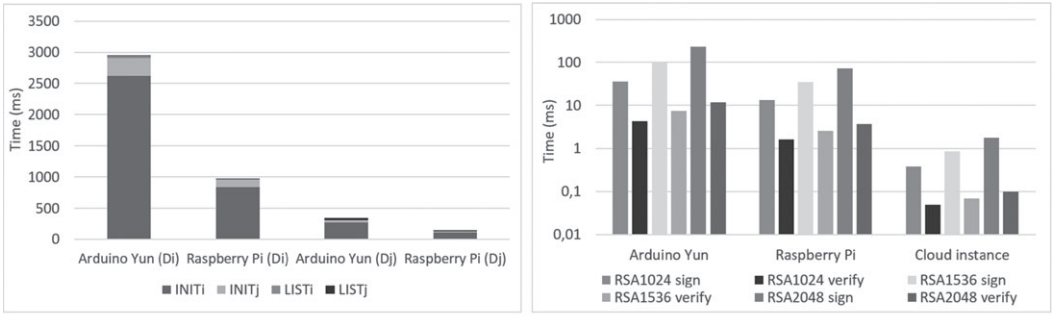
Figure 3(a) shows the average duration of the $agreement$ procedure for Arduino and Raspberry Pi. We see that there is a significant difference in operation duration when a device is an initiator (i.e., D_i) of communication compared with situations in which it is a responder (i.e., D_j). This is mainly caused by the complexity of generating a good DH value on the initiator side. Note that the depicted time includes the time spent on processing (i.e., creating a message before sending and parsing it upon receipt) of all defined agreement messages, but it does not include the transmission time between devices.

As the entire agreement process on average lasts less than 3 seconds for perceptual tier devices, which have the most constrained hardware capabilities, it does not represent a significant overhead in typical IoT deployments and can, in general, be performed as often as required by specific deployments.

Figure 3(b) shows the average processing time of the $sign$ and $verify\_signature$ operations. The processing time is considerable for lower tier devices and thus this operation should be avoided

[‡‡]The entire model and verification procedure are available at http://public.tel.fer.hr/acap.

DOI: 10.1002/spe



(a) $agreement$ for devices D_i and D_j

(b) $sign$ and $verify$ for different key sizes

(c) $hmac$ for two hash algorithms

(d) $encrypt$ and $decrypt$ for different key sizes

Figure 3. Average operation duration across different tiers.

in order to preserve the power on such devices. We propose the usage of the $sign$ operation only for important sensor management operations, such as sending device software and configuration updates from the higher tiers (cloud or intermediate to the lower tiers (sensors and actuators). In such cases, the lower tier devices only perform the $verify\_signature$ operation which requires less processing time than the original $sign$ operation. In this figure, we can also see that the increase in key size (i.e., from 1024 bits to 1536 or 2048 bits), which offers a greater level of security, also prolongs the average processing time of $sign$ and $verify\_signature$ operations.

We propose the $hmac$ operation as a replacement to the $sign$ operation in case of lower tier devices because it requires substantially less processing time for the same amount of exchanged data in comparison with the $sign$ operation, as shown in Figure 3(c). Moreover, because the $hmac$ operation does not reduce the level of security when compared with the $sign$ operation, it should be used for ensuring data integrity during the data collection process, as explained in Section 2.

The average processing times of the $encrypt$ and $decrypt$ operations are shown in Figure 3(d). These times are comparable with the $hmac$ operation time, and thus these operations can be performed as often as needed in order to provide sensitive data protection throughout all tiers of our secure CoT communication model. Note that there is only a small increase in the processing time of these operations when using longer keys (i.e., 256 instead of 128 bits), which provide a higher level of security.

5. RELATED WORK

Research efforts in the area of security for the CoT can be divided into two categories: (i) privacy related and (ii) secure communication challenges. Privacy is defined as the guarantee that users maintain control over the release of their sensitive information [16], while secure communication is defined as the fulfillment of primary security requirements (i.e., confidentiality, integrity, and availability) in network communication [17]. The latter paper presents the Agile Cryptographic Negotiation Protocol (ACNP), which was the first version of the our agreement protocol that was not formally verified and did not satisfy the needed security requirements.

DOI: 10.1002/spe

An overview of privacy related issues and open challenges in participatory sensing, which is commonly used in the CoT environment, is presented in [18]. The paper [19] outlines several critical security and privacy threats for the mobile crowd sensing paradigm, a novel approach to ubiquitous computing especially popularized through growth of the IoT. The analysis of nine challenges regarding privacy and security for the opportunistic sensing, an alternative to the participatory sensing in the crowd sensing paradigm, is presented in [20], alongside with conceptual solution for each challenge. In our paper, we deal with the following 2 out of 9 challenges: data authenticity and system integrity. Our work follows the proposed conceptual solution for data authenticity because it provides the required computational and bandwidth-efficient authentication. Additionally, we deal with system integrity by providing data integrity from the source to destination.

A security architecture for the CoT/IoT, compatible with our security model, is presented in [21] where the authors identify security issues at different CoT tiers. A similar approach is used in [22] where the authors provide a review of security features, requirements, and technologies for the IoT. The authors of the latter paper have recognized the importance of lightweight cryptographic protocols for reducing the energy consumption of devices in the perceptual tier. To achieve energy savings, our CAAP procedure automatically adapts to hardware capabilities (e.g., current energy levels) of devices and uses lightweight cryptographic protocols when necessary.

An extensive analysis of security issues and possible attacks is provided in [23] in which the authors also discuss promising approaches for specific challenges depending on deployment architecture of an IoT system. The paper [24] analyzes emerging security problems in the IoT and discusses possible counter measures. Another work which explains an IoT architecture from the security and privacy point of view, and also includes a brief overview of the EU legislation in the privacy and security area, is [25]. The work presented in [26] discusses new regulatory approaches for privacy and security requirements in the IoT. The survey paper [27] compares security issues between the IoT and traditional networks, and also provides practical solutions for a large number of identified security challenges in the IoT environment.

In our paper, we do not cover trust establishment and management. For details on these topics, we refer an interested reader to a detailed survey on the trust management for the IoT, presented in [28].

6. CONCLUSION

The paper presents a novel adaptable communication model for secure communication in the CoT. The CoT is a heterogeneous environment, which is built on the interoperability of various devices and platforms that should support secure communication in the case of different security scenarios, such as authenticated device management, reliable data delivery, and sensitive data protection. In this paper, we propose a generic solution that provides the essential building blocks (i.e., operations) for secure communication in the CoT with a goal to offer flexible communication mechanisms which can adapt well to the available resources and context of a CoT environment. We also present a practical implementation of the six proposed secure communication operations (i.e., agreement, sign, verify signature, hmac, encrypt, and decrypt) with a special emphasis on the agreement operation, which is the basis of our adaptable secure communication model. Our experimental evaluation demonstrates promising runtime performance of the proposed solution for different security scenarios and devices in the CoT, and thus represents a valuable contribution to the current state of art particularly in terms of the practical security implementations.

ACKNOWLEDGEMENTS

This work has been supported in part by the Croatian Science Foundation under the project number 8065 (Human-centric Communications in Smart Networks). This work is supported in part by the H2020 symbIoTe project, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 688156.

DOI: 10.1002/spe

REFERENCES

1. Sagiroglu S, Sinanc D. Big data: A review. *2013 International Conference on Collaboration Technologies and Systems (CTS)*, San Diego, CA, USA, 2013; 42–47.
2. Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. *Communications of the ACM* 2010; **53**(4):50–58.
3. Soldatos J, Kefalakis N, Serrano M, Hauswirth M. Design principles for utility-driven services and cloud-based computing modelling for the internet of things. *IJWGS* 2014; **10**(2/3):139–167. DOI:10.1504/IJWGS.2014.060254.
4. OWASP T. *10 2014 The Ten Most Critical Internet of Things Security Risks*, 2014. Available from: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=Top_10_IoT_Vulnerabilities__282014_29 [last accessed September 12th, 2016].
5. Vasić V, Mikuc M, Vuković M. Lightweight and adaptable solution for security agility. *KSII Transactions on Internet & Information Systems* 2016; **10**:1212–1228. DOI:10.3837/tiis.2016.03.015.
6. Cremers CJF. *Scyther: Unbounded Verification of Security Protocols*. ETH, Department of Computer Science: Zurich, Switzerland, 2007.
7. Cremers CJF. Scyther user manual, 2014. Available from: http://www.cs.ox.ac.uk/people/cas.cremers/scyther/index.html [last accessed September 12th, 2016].
8. Perera C, Zaslavsky A, Christen P, Georgakopoulos D. Context aware computing for the internet of things: A survey. *Communications Surveys & Tutorials, IEEE* 2014; **16**(1):414–454. Available from: http://arxiv.org/pdf/1305.0982.pdf [last accessed September 12th, 2016].
9. Bellare M, Canetti R, Krawczyk H. Keying hash functions for message authentication. In *Advances in Cryptology – CRYPTO '96, Lecture Notes in Computer Science*, Vol. 1109, Koblitz N (ed.). Springer: Berlin Heidelberg, 1996; 1–15. DOI:10.1007/3-540-68697-5_1.
10. Diffie W, Hellman ME. New directions in cryptography. *IEEE Transactions on Information Theory* 1976; **22**(6):644–654.
11. Frankel S, Krishnan S. IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap. RFC 6071 (Informational), Feb 2011. Available from: http://www.ietf.org/rfc/rfc6071.txt [last accessed September 12th, 2016].
12. Freier A, Karlton P, Kocher P. The Secure Sockets Layer (SSL) Protocol Version 3.0. RFC 6101 (Historic), Aug 2011. Available from: http://www.ietf.org/rfc/rfc6101.txt [last accessed September 12th, 2016].
13. Dierks T, Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), Aug 2008. Available from: http://www.ietf.org/rfc/rfc5246.txt [last accessed September 12th, 2016], updated by RFCs 5746, 5878, 6176.
14. Ylonen T, Lonvick C. *The Secure Shell (SSH) Protocol Architecture. RFC 4251 (Proposed Standard)*, Jan 2006. Available from: http://www.ietf.org/rfc/rfc4251.txt [last accessed September 12th, 2016].
15. Krawczyk H. Sigma: The 'sign-and-mac'approach to authenticated diffie-hellman and its use in the ike protocols. *Advances in Cryptology-Crypto 2003*, Springer: Berlin, Heidelberg, 2003; 400–425.
16. Christin D, Reinhardt A, Kanhere SS, Hollick M. A survey on privacy in mobile participatory sensing applications. *Journal of Systems and Software* 2011; **84**(11):1928–1946.
17. Vasic V, Mikuc M. Security agility solution independent of the underlaying protocol architecture. *Proceedings of the First International Conference on Agreement Technologies*, Dubrovnik, Croatia, 2012.
18. Li Q, Cao G. Privacy-preserving participatory sensing. *Communications Magazine, IEEE* 2015; **53**(8):68–74.
19. Yang K, Zhang K, Ren J, Shen X. Security and privacy in mobile crowdsourcing networks: Challenges and opportunities. *Communications Magazine, IEEE* 2015; **53**(8):75–81.
20. Kapadia A, Kotz D, Triandopoulos N. Opportunistic sensing: Security challenges for the new paradigm. *Communication Systems and Networks and Workshops, 2009. Comsnets 2009. First International*, Piscataway, NJ, USA, 2009; 1–10.
21. Zhang B, Ma XX, Qin ZG. Security architecture on the trusting internet of things. *Journal of Electronic Science and Technology* 2011; **9**(4):364–367.
22. Suo H, Wan J, Zou C, Liu J. Security in the internet of things: A review. *2012 International Conference on Computer Science and Electronics Engineering (ICCSEE)*, Vol. 3, 2012; 648–651.
23. Roman R, Zhou J, Lopez J. On the features and challenges of security and privacy in distributed internet of things. *Computer Networks* 2013; **57**(10):2266–2279. DOI: 10.1016/j.comnet.2012.12.018. Towards a Science of Cyber SecuritySecurity and Identity Architecture for the Future Internet.
24. Zhao K, Ge L. A survey on the internet of things security. *2013 9th International Conference on Computational Intelligence and Security (CIS)*, 2013; 663–667.
25. Kozlov D, Veijalainen J, Ali Y. Security and privacy threats in IoT architectures. *Proceedings of the 7th International Conference on Body Area Networks*, BodyNets '12, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 2012; 256–262.
26. Weber RH. Internet- of things - new security and privacy challenges. *Computer Law & Security Review* 2010; **26**(1):23–30. Available from: http://www.sciencedirect.com/science/article/pii/S0267364909001939 [last accessed September 12th, 2016].
27. Jing Q, Vasilakos AV, Wan J, Lu J, Qiu D. Security of the internet of things: Perspectives and challenges. *Wireless Networks* 2014; **20**(8):2481–2501. DOI: 10.1007/s11276-014-0761-7.
28. Yan Z, Zhang P, Vasilakos AV. A survey on trust management for internet of things. *Journal of Network and Computer Applications* 2014; **42**:120–134. Available from: http://www.sciencedirect.com/science/article/pii/S1084804514000575.