

TAS-IoT: Trust-based Adaptive Security in the IoT

Hamed Hellaoui*, Abdelmadjid Bouabdallah[†], Mouloud Koudil*

*Ecole nationale Supérieure d'Informatique ESI, LMCS Laboratory

16309 Oued Smar, El Harrach, Algiers, Algeria. Email: {h_hellaoui, m_koudil}@esi.dz

[†]Sorbonne Universités, Université de Technologie de Compiègne UTC, CNRS, Heudiasyc UMR 7253
CS 60 319, 60 203 Compiègne cedex, France. Email: abdelmadjid.bouabdallah@hds.utc.fr

Abstract—Providing efficient security services in dynamic low-power environments as the Internet of Things (IoT) is a challenging task. The deployment of static security services will consume the energy even if it is not required in some situations, so this induces a waste of resources. In this paper, we introduce an efficient model for adaptive security in the IoT based on trust management. Most of existing adaptive security approaches lack of practical means to evaluate threats. On the other hand, trust management systems are designed to deal with selfish behaviors or internal attacks and not to assist cryptographic measures. Our solution evaluates the trust level related to the presence of security threats among nodes, and adapt consequently cryptographic measures. The obtained simulation results show that our solution reduces considerably energy consumption and remains yet secure.

Index Terms—Internet of Things (IoT), Adaptive security, Trust management.

I. INTRODUCTION

We are witnessing the emergence of a very important technology that connects physical objects to the network. The Internet of Things (IoT) opens new opportunities and is applied in many fields (health-care, transportation, manufacturing, etc.). Since the applications are generally sensitive, ensuring security services in the IoT is crucial for its development. However, highly secure mechanisms often consume a considerable amount of resources. Since the resources of the objects connected to the IoT can be exhaustible, as it is the case for energy, it is important to reduce consumption while ensuring the required security services.

Recently, there has been a focus on an interest concept known as adaptive security [2]–[5]. It consists in adjusting or maintaining security measures in varying situations [1]. Therefore, it can be used to reduce resource consumption in the IoT by adjusting security measures depending on threat levels. Indeed, static security considers always the worst case. The dynamic nature of the IoT makes that even threats can change over the time and over the place when objects are mobile. However, most of existing adaptive security approaches lack of efficient means to evaluate threats in order to take the adequate security decision.

Trust management, another approach to ensure security, is a process that allows evaluating the level of confidence between entities. A node trusts another one, means that the latter behaves as the former expects (the former is called ‘trustor’ and the latter ‘trustee’) [6]. The concept of trust has been widely used to detect malicious attacks that have already passed cryptographic barriers. It has also been used to detect selfish

behaviors, which cannot be done by cryptographic primitives [7]. As trust management provides a way for judging nodes, it could be a good candidate for evaluating threats in order to adapt security measures.

While existing approaches lack of practical means to evaluate threats, our basic idea is to use trust management to perform the evaluation. We formulate the adaptive security problem based on trust management in the following way. *Given a cryptographic security service, what adaptation could a node consider, depending on the trust level related to this service.* Several solutions have been proposed for the trust management problem in the IoT (surveys are provided in [6], [7]). However, existing trust management models are not designed for assisting adaptive security decisions and cannot be directly used for this problem. The solution we provide in this paper introduces an efficient adaptive security model for the IoT, based on trust management.

The rest of this paper is organized as follows. Section II reviews existing adaptive security approaches for the IoT. We present our trust-based model for IoT adaptive security in section III. A performance analysis is provided in section IV. Finally, section V concludes this paper.

II. RELATED WORKS

In [4], Taddeo et al. proposed an approach where a node delivers only essential packets after decreasing security measures. In [5], a node uses beacons to announce its current security measures, which helps the sender to choose its receiver depending on its security needs. However, adaptation in those works are triggered by the level of energy. The lack of evaluating threats makes that a malicious node can take advantage of decreasing security measures to make damage (e.g. inject wrong priority messages).

In [2], Hamdi and Abie proposed a game-based model for adaptive security in the IoT. They use the Markov game theory for the modeling of uncertain game variables in order to enable and disable security mechanisms. Nevertheless, the authors do not define how a node will determine if another one is compromised or not. They simulate an epidemic model of virus spread in WSN, which makes their approach more analytical. The same critic can be made for the work [3] proposed by Wang et al. In another work, Younis et al. [8] proposed a solution in which security is adapted depending on nodes’ confidence in terms of routing metrics (e.g. packet

drop rate, medium access collision). However, this does not allow evaluating threats as those metrics are not related to the cryptographic service to adapt.

III. TRUST MODEL FOR ADAPTIVE SECURITY

A. Security model

We consider a network of heterogeneous objects in terms of resource capabilities. Messages can be posted only by nodes legitimate to do so. An adversary (not legitimate to post messages) is interested in posting arbitrary messages in the network which will lead to consume node resources. For that, a legitimate node appends an authenticator to each message it generates and relay nodes have to authenticate that message (data origin authentication) so to reject bogus messages as soon as possible.

B. Our solution

Although the basic security model prevents a non desired node to inject packets in the network by authenticating data origin, it also induces energy consumption. Our adaptive security solution aims to reduce the authentication overhead by authenticating packets only when it is required. Indeed, if some nodes are compromised, the authentication has to be performed by nodes receiving packets from the compromised ones. In our proposal scheme, each node decides locally to authenticate the received message or not depending on the trust level that it associates to the message sender (neighbor that issued or relayed the message). Algorithm 1 shows when to perform authentication by a relay node. Here, T_{ij} refers to the trust level that node p_i (receiver) associates to node p_j (sender) and f the function that decides to apply the authentication or not. As we can see, the authentication of the received message, $Authenticate(m)$, is not applied in a static manner.

Algorithm 1 The proposed receive procedure for a relay node

```

procedure Receive( $m, p_j$ )
  if ( $f(T_{ij})$ ) then
    Send( $m, NextHop$ )
  else
    if ( $Authenticate(m)$ ) then
      Send( $m, NextHop$ )
    else
      Drop( $m$ )
    end if
  end if
end procedure

```

In the following, the trust evaluation model and the adaptive function f are explained in detail.

Trust evaluation

T_{ij} represents the trust level that p_i associates to p_j . It is a real number which takes value between 0 and 1. The value 0 means that no trust is given to the trustee node, and 1 refers to a complete trust. In order to compute its T_{ij} , p_i relies on three complementary components: its own experiences E_{ij} , its own observations O_{ij} , and recommendations R_{ij} received about p_j . This trust is given by the following formula (equation 1)

$$T_{ij} = \alpha E_{ij} + \beta O_{ij} + \gamma R_{ij} \quad (1)$$

where α, β , and γ are weighting parameters which serve two purposes. They keep the trust value between 0 and 1 by satisfying the relation $\alpha + \beta + \gamma = 1$, and they weigh the trust components. We explain in the following how to calculate each component: experiences, observations, and recommendations.

E_{ij} refers to the trust level perceived from the direct experience between the two nodes. Indeed, when p_i receives a packet from p_j , it may decide to authenticate the received message (depending on the function f). If the packet is authenticated, this would have a positive impact on E_{ij} . And if not, either p_j is compromised and has just sent an undesired message, or it is a clean node that retransmitted the message without authenticating it. The last situation is more possible when p_j is constrained in terms of resources.

A relation of order is used to represent the heterogeneity between nodes. We denote by c_i the category of node p_i . $c_i < c_j$ means that node p_i is more constrained than node p_j . We express the E_{ij} component in a recursive manner as it is shown in equation 2. The new value of E_{ij} depends on the old ones, E'_{ij} , and the actual experience. The latter can have the value 1 if the received message is authenticated, a value a ($0 < a < E'_{ij}$) to tolerate an unauthenticated message from constrained node that is trusted until now (t and c are thresholds), and the value 0 otherwise. It is clear that the tolerance option decreases E_{ij} , and consequently T_{ij} . Thus, even a trusty constrained node that persists to send unauthenticated messages will be untrusted. The parameter δ^e aims to keep E_{ij} between 0 and 1 ($\delta^e \in]0, 1[$) and to give a weight to old or the new values. A small δ^e means that E_{ij} relies more on new experiences than old ones. This can be used when old experiences may not be very relevant for evaluation (for example because the time between two experiences is long or the node change its behavior overtime). It is very easy to prove that when $\delta^e = \frac{NB-1}{NB}$, E_{ij} becomes the average of the NB experiences of p_i with p_j .

$$E_{ij} = \delta^e E'_{ij} + (1 - \delta^e) \times \begin{cases} 1 & \text{if } Authenticate(m) \\ a & \text{if } !Authenticate(m) \\ & \wedge T_{ij} > t \wedge c_j < c \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

O_{ij} represents the trust level that p_i associates to p_j from observing its behavior. Using the wireless transmission medium, p_i can overhear messages sent by p_j and some messages intended to this node that are in p_i 's communication range. When p_i sends an authenticated message to p_j for relaying, the latter should transmit it without changing its content. Likewise, when p_j receives a message from another node, it may not retransmit it because it could be unauthenticated, but if p_j retransmitted the received message, this should be without changing the message's content. If not so, p_i should decrease O_{ij} .

We denote by $Retrans_{ji}(m)$ a function that returns 1 if p_j retransmitted p_i 's message, m , without changing its content, 0 if it did not retransmitted it, and -1 if p_j retransmitted the message but changed its content. As it is shown in equation 3, computing O_{ij} depends on its old values, O'_{ij} , and the new observation. The latter can have the value 1 if p_j

retransmitted p_i 's message correctly, a value b ($0 < b < O'_{ij}$) for a constrained node that did not retransmit the message, and 0 if p_j is not constrained and did not transmitted p_i 's message or retransmitted a received message, from p_j or another node, but changed its content. δ^o is a weight parameter.

$$O_{ij} = \delta^o O'_{ij} + (1 - \delta^o) \times \begin{cases} 1 & \text{if } \text{Retrans}_{ji}(m) == 1 \\ b & \text{if } (\text{Retrans}_{ji}(m) == 0 \\ & \wedge c_j < c) \\ 0 & \text{if } (\text{Retrans}_{ji}(m) == 0 \\ & \wedge c_j \geq c) \vee \\ & (\text{Retrans}_{jk}(m) == -1) \end{cases} \quad (3)$$

R_{ij} represents recommendations of p_j 's neighbors and has to be calculated from notes sent by those neighbors. When p_i receives a recommendation r from p_k about p_j , it updates its R_{ij} . The latter depends on its old values R'_{ij} and the new one r (equation 4). The recommendation r is a real number that takes value between 0 and 1. δ^r represents a weight parameter.

$$R_{ij} = \delta^r R'_{ij} + (1 - \delta^r) r \quad (4)$$

When p_k sends recommendation to p_i about p_j , it uses its E_{kj} and O_{kj} components as they reflect a direct perception (equation 5). δ weighs between E_{kj} and O_{kj} .

$$r = \delta E_{kj} + (1 - \delta) O_{kj} \quad (5)$$

Adaptive function

The adaptive function f decides whether to authenticate or not the message sent by p_j , based on the trust level T_{ij} . A simplest way to construct f is to return *true* when T_{ij} exceeds a given threshold. However, this would create a vulnerability that can be used by malicious nodes. Let us suppose that a malicious node p_j behaves well to win the trust of its neighbors. Once it convinces one node, lets say p_i , it sends wrong packets only to that node. As the trust level T_{ij} exceeds the given threshold, p_i will not authenticate the message and consequently E_{ij} will not be changed. Because p_j continues to behave well with the other nodes, O_{ij} also will not be changed. Even recommendations sent by neighbors cannot contain negative reports about p_j , as the latter generates wrong messages to p_i and does not change what it retransmits. This means that R_{ij} will not be changed also. Therefore, f should take into consideration other parameters.

The adaptive function that we propose is shown in Algorithm 2. The main idea behind is the following : when T_{ij} does not exceed a given threshold, t , f returns directly *false*. If T_{ij} exceeds the threshold, f will return *false* in a periodic manner.

Algorithm 2 The adaptive function f

```

function  $f(T_{ij})$ 
  if  $(T_{ij} < t)$  then
    return false.
  else
    if  $(\text{period}_j == 0)$  then
       $\text{period}_j = \text{random}() \% q_j$ 
      return false
    else
       $\text{period}_j = \text{period}_j - 1$ 
      return true
    end if
  end if
end function

```

This keeps detecting a malicious node that has already won the trust of p_i remains yet possible. The period period_j , initially equal to 0, is made up in a random manner ($\text{random}() \% q_j$). q_j can be small at the beginning and increase each time the period is finished until achieving a maximum value. This increase can differ depending on node resources c_i . The more powerful node is, the less it would increase q_j as it has enough resources.

When f decides to apply the authentication, the node can have a feedback about the relevance of its decisions and make adaptation according to that return. If $T_{ij} > t$ and f returns *false* (period_j achieved), then the received message turns out to be unauthenticated, it decreases q_j rather than increases it. This can be a sign that p_j is malicious and just tried to win the trust of p_i (this is commonly known as on-off attack).

Also, if a node p_j proves to be malicious over a long time and send wrong messages, p_i decides to stop observing its behavior and sending recommendation about it (experiences remain considered). Indeed, if p_i authenticate always p_j 's message, observing its behavior and sending recommendations about it would consume more energy than the basic model. That's why a node keeps only the experience component in this case.

IV. TEST EVALUATIONS

To evaluate our proposal, we carried out some tests using Cooja, the Contiki OS simulator [9]. The library tinyDTLS [10] was used to produce the authenticator and perform data origin authentication (HMAC MD5 was used). The evaluation was performed using the random topology shown in Figure 1. The considered scenario consists in sending messages in a periodic manner from network nodes to the root and vice versa. Note that the used value of t was 9/10, the weight ($\delta^e, \delta^o, \delta^r$) were 1/2, and $q_j \in [0, 10]$.

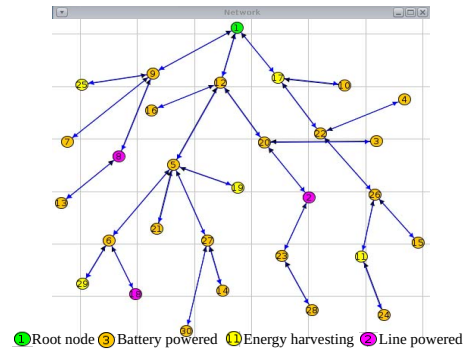


Fig. 1: Screenshot of the simulation window

To evaluate energy, we used the library "energest.h" which estimates the energy consumption by measuring the overall execution time. Figure 2 presents the results for node 5 during five minutes. This node has five neighbors they send packets to it. As we can see, if no neighbor is compromised there is a remarkable difference of execution time when using the proposed scheme. To measure the difference in terms of node life time, we define τ as $\tau = \frac{\tan_1}{\tan_2}$, where $\tan_1$ is the tangent for the execution time line without the proposal approach,

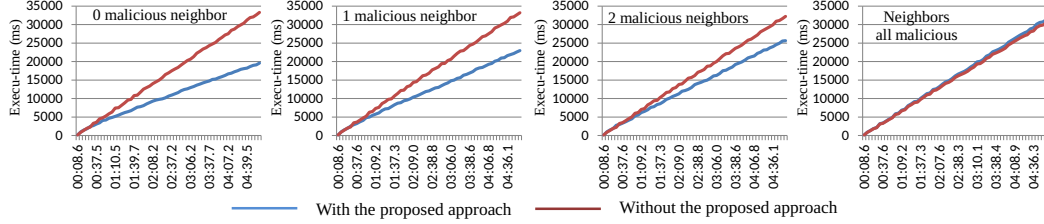


Fig. 2: Evolution of the overall execution time for node 5

and $\tan_2$ is the tangent for the execution time line using the proposal solution. Thus, this parameter means that the node life time is increased by τ using our trust-based model. When no neighbor is compromised, τ is about 1.7, which implies that the node 5 will reduce its energy consumption by 41.48%. Even when some neighbors are compromised, the gain of energy using the proposal approach is considerable (τ is about 1.4 in '1 malicious neighbor' case, and 1.2 in '2 malicious neighbors' case). Furthermore, as the node stops observing and sending recommendations about nodes that proved to be definitively malicious, energy consumption when all neighbors are compromised is almost unchanged (we note a light increase in the proposal scheme which is due to the computing of the trust at the beginning).

We also evaluated energy consumption for other network nodes. It shows always the same look as in Figure 2, but the difference is in terms of the parameter τ . Indeed, this difference is related to the number of received packets. For example, node 12 which received the biggest number of packets has a τ more than 2 when no neighbor is compromised. Table I summarizes values of the parameter τ for different nodes when neighbors are not compromised (when some neighbors are compromised τ is between 1 and that cited in the table).

Node	28	27	9	17	12
Nb of received packets	11	57	173	251	609
τ	1.00	1.06	1.32	1.58	2.06

TABLE I: Parameter τ of different nodes

To evaluate the proposal scheme in terms of transmitting authenticated messages, we considered a scenario where a node performs an on-off attack after being trusted. Figure 3 shows the trust evolution for node 6, the trustor (the trustee was node 5). We notice that the trustee can mislead the trustor and send unauthenticated messages. However, because the adaptive function returns *false* in a random manner, the trustee can't know when it will happen. Compared to existing approaches that are either impractical or do not really evaluate threats, our solution introduces a huge contribution by keeping tracing node's behavior. In addition, even if the trustee tries to regain the trust and send authenticated messages, the decision function considers small periods because the trustee is suspected to mount an on-off attack. The period will increase if the trustee behaves well for a long time. This discourage malicious node to switch between sending authenticated and unauthenticated message as it leads the adaptive function to return *false*.

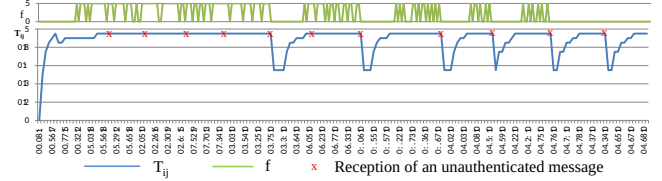


Fig. 3: Trust evaluation under on-off attack

V. CONCLUSION

In this paper we presented an efficient adaptive security model for the IoT, based on trust-management. It allows evaluating the trust related to the presence of security threats and performing, consequently, adaptive security decisions. The proposal trust evaluation method is composed of three complementary components: experiences, observations, and recommendations. Simulation results showed that our solution reduce resource consumption and remains yet secure.

As future work, it would be interesting to consider untrustworthy recommendations and perform more advanced evaluations (lossy networks, other topologies, etc.).

REFERENCES

- [1] E. Yuan, N. Esfahani, and S. Malek, "A systematic survey of self-protecting software systems," *ACM Trans. Auton. Adapt. Syst.*, vol. 8, no. 4, pp. 17:1–17:41, Jan. 2014.
- [2] M. Hamdi and H. Abie, "Game-based adaptive security in the internet of things for ehealth," in *2014 IEEE International Conference on Communications (ICC)*, June 2014, pp. 920–925.
- [3] E. K. Wang, T.-Y. Wu, C.-M. Chen, Y. Ye, Z. Zhang, and F. Zou, "Mdpas: Markov decision process based adaptive security for sensors in internet of things," in *Genetic and Evolutionary Computing*. Springer, 2015, pp. 389–397.
- [4] A. Taddeo, M. Mura, and A. Ferrante, "Qos and security in energy-harvesting wireless sensor networks," in *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, July 2010, pp. 1–10.
- [5] A. D. Mauro, X. Fafoutis, and N. Dragoni, "Adaptive security in odmac for multihop energy harvesting wireless sensor networks," *Int. J. Distrib. Sen. Netw.*, vol. 2015, pp. 68:68–68:68, Jan. 2015.
- [6] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for internet of things," *Journal of Network and Computer Applications*, vol. 42, pp. 120 – 134, 2014.
- [7] H. Yu, Z. Shen, C. Miao, C. Leung, and D. Niyato, "A survey of trust and reputation management systems in wireless communications," *Proceedings of the IEEE*, vol. 98, no. 10, pp. 1755–1772, 2010.
- [8] M. Younis, N. Krajewski, and O. Farrag, "Adaptive security provision for increased energy efficiency in wireless sensor networks," in *2009 IEEE 34th Conference on Local Computer Networks*, Oct 2009, pp. 999–1005.
- [9] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, Nov 2006, pp. 641–648.
- [10] O. Bergmann. (2015) Tinydtls: a library for datagram transport layer security. [Online]. Available: <https://sourceforge.net/projects/tinydtls/>