International Journal of Pure and Applied Mathematics
Volume 118 No. 20 2018, 3173-3180
ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ijpam.eu

Secure Multicasting using CoAP- Group Keying Approach

M.P.Anuradha[1] S.Ramya[2] Manivannan Doraipandian[2*]
[1,] Department of Computer Science, Cauvery College for Women, Trichy, India.
[2] School of Computing, SASTRA University, India.

Abstract:
Internet of Things (IoT) includes multiple industries and use cases that balance from a tiny constrained device to enormous multi-domain deployments of embedded technologies and cloud systems concerning in real-time. CoAP (Constrained Application Protocol) is a restful protocol, best for self-consciousembedded devices and networks, and dedicated for D2D, M2M applications. Communication protocols are used to interconnect and that consent to all end devices, gateways and servers to integrate each other in an innovative and more consistent ways. The IoT end device frequently communicates over a practical network with gateways and this communication needs to be securely managed and also requires continuous monitoring. The secret key is used to establish a secure communication among all end devices and gateway in an IoT Framework. To ensure that, no IoT end devices, gateways and communication channels can be attacked by an external attacker. In order to overcome the pitfalls, in this work, hybrid key management framework is proposed and all the secret keys are conversedusing CoAP protocol over the network. A total number of rekey messages generated while new device joining, leaving, the number of keys required in all device and total numbers of keys stored in the entire system are measured and conferred. The proposed scheme is better compared to the other methods in the existing schemes.

Keywords: Internet of Things, CoAP,Key management.

Introduction:

Embedded devices are over and over again constrained in their resources related to memory, computational power, and communication overheads and are consequently incapable to manage in typical IP (Internet protocols). An embedded system generally integrates a number of tiny devices to perform some smart applications like smart home automation, smart city, smarthealth care applications. CoAP [9] is an embedded web transfer protocol and it's an alternative standard protocol as the Hypertext Transfer Protocol (HTTP). CoAP is mainly used to connect uncertain microelectronic devices to interconnect over the Internet.Naming and addressing of embedded devices in IoT application field is a significant constraint and also to be addressed individually and also grouped. Embedded tiny devices have to be addressed in groups in addition to being addressable independently. CoAP trailssignificant approaches [8]to manage group communication among the end devices in IoT applications.

Now a day, the number of embedded devices connected to the Internet is promptly increasing. On the other hand, the number of devices and the variety of devices are connected to the network are gradually increasing. TCP and UDP protocols provide high complexity and also support optional acknowledgments system in embedded communication.CoAP in contrast with HTTP [11] requires smaller data packets for transmission and it also eliminates packet loss by communicating all the

possible data at first attempt overcoming the drawbacks of TCP. It uses a multicast group and auto discovery of clients without any prior knowledge database in the server, which facilitates fewer resource utilization compared to other traditional techniques. Over UDP, CoAPdoes not [12,13] store any information about state connections, if any packets are discarded during transmission, the information about the transmission is not lost. CoAP is a Restful protocol, this protocol is ideal for constrained devices and networks, and this protocol is specialized for M2M applications, it follows asynchronous transaction model.CoAP based Key Agreement approach is described in Figure1.

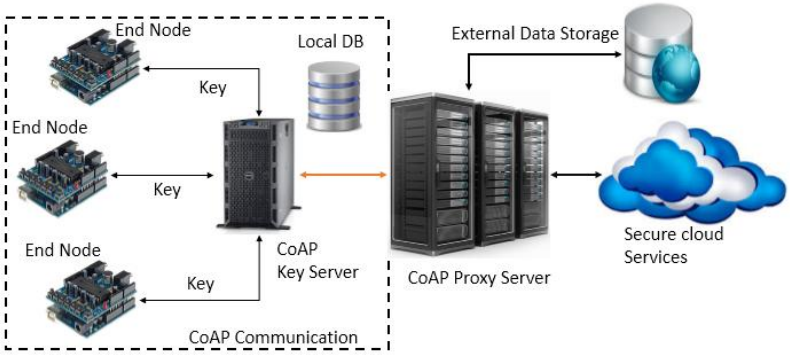


Figure 1: CoAP based Key Agreement

In current communication scenario, more and number of embedded devices are connected together and to process the combined operation in IoT framework. The tiny devices in network processing are able to handle smart data transaction with network/internet applications.All the tiny devices are also involve data communication overcommunal networks that require protection from external and unlicensed access and thus the security requirements in communication devices have become an essential one. In a device to device communication, the data can be securely transmitted over the network by using the cryptographic techniques.Different number of crypto techniques involves a different number of keys which needs to be maintained security among the devices. Hence there is need to generate, stored and distribute keys secretly among all the sensor nodes. Since in group communication, there are many devices involved and each may use separate keys because no two devices can have the same key to maintain data confidentiality and integrity. Hence key management is an important issues in secure group communication. In secure multi-casting, involves n number of tiny devices want to securely shares its data using crypto algorithms with secret keys. Loading of secret keys, sharing and update of secret keys for all the devices seems an interesting task of an embedded applications.IoT [7] with its growing importance has created the need for secure communication with optimum resource utilization and overall reduced network overhead. Multicast routing technique and key sharing approached requires lightweight server/client approaches. In order to accommodate with HTTP and Restful webservers CoAP protocol [8,9,10] is specific towards constrained devices and supports multicast routing techniques. It is designed in such a way to interoperate with native internet technology. CoAP utilizes UDP connectionless datagram instead of TCP which in-turn reduced network overhead with small data packets and allows devices for a longer sleep time. In addition, it is built on top of IPv6 which inherently supports multicast addresses along with the normalIPv6 address. Hence, CoAP[8,9,10] can be utilized for effective multicast routing as well as multiple key sharing techniques. In this work, CoAP based secure multicasting with hybrid group keying approach is proposed and theoretical analysis and simulation are carried out. Figure 1 describes the CoAP based key agreement services.

Among WSN nodes, CoAP protocol [8,9,10] is used for data transmission and it's designed to easily translate to HTTP for simplified integration with the web, it supports multicasting with less overhead and simplicity. CoAP provides a better solution than the HTTP stack for WSN but it has some shortcomings due to preserving interoperability services with HTTP systems [11,12,13]. IoT devices [7] are more vulnerable towards different IoT bugs i.e., there can be no information in advance whether IoT devices are being attacked or not by any internal and external attackers. Hence lightweight key management protocol is proposed and securely established among all the end devices. Using secure key generation mechanism, 32bit, 64-bit keys are generated and preloaded into the embedded device, whenever any end devices want to add or leave from the network, the corresponding keys are securely updated.

Related Works:

In device key management [1,2], communication channel is established among all nodes that must exchange data securely, the device may be added and discarded at any time. Unauthorized devices should be restricted to have communication with nodes in the network. The forward and backward secrecy among all the devices in the network is maintained. The computational cost of key generation, storage overhead,and communication overhead are the important key issues in key management. Pair-wise keying, probabilistic methods,and the hierarchical methods are the various key management protocols used in WSN. The pair-wise keying scheme [3] does not scale well when the number of devices becomes very large. The probabilistic methods have large storage overheads. The hierarchical methods are less storage overhead and also provide scalable solutions. Star tree key management, Logical key hierarchy(LKH) and One-way function tree(OFT) are the important existing hierarchical key-based protocols. In this work, a dynamic hybrid hierarchical key management protocol is proposed.

In Star-Tree key management [4, 5] has one centralized server which is responsible for generating the Group Key (GK) and the individual keys. Each user has a pre-deployed key which is maintained in the database of the server. Whenever the user wants to join the network the server authenticates the identity of the user based on the pre-deployed key. In OFT scheme is a bottom-up approach in which the keys of the group are generated by the members of the group. The server is responsible for authenticating the members before granting permission to join the group. The server follows a hierarchical method i.e., the tree structure in dealing with the addition and deletion of nodes. In LKH there is a centralized server which is responsible for generating the keys. The whole system is divided into a hierarchical structure as in N-array tree, where d is the degree of the tree. Each node is divided into d branches which is the degree of the tree. The members are added as a leaf node. All the leaf nodes and the all the internal nodes are given a key which is generated by the server. A subgroup key is common to all its group members. The group key of the root is common to all its members. Scalability and Resilience are the two important metrics are used to evaluate the efficiency of the Key distribution strategies. An embedded network supports both pre-deployment and post-deployment strategies for device management. Using CoAP protocol [13,14,15], the key is securely distributed among the end devices and gateways.

System Model:

In this work, hierarchical, hybrid and decentralized key management approach [5,6 is proposed, in which the control does not solely with the centralized server. An embedded device like Raspberry Pi will act as a master gateway.The masterserver can select other embedded devices to act as a gateway. Each gateway acts as anindividual server for its group members. Whenever a device joins the group, it sends a request to the master gateway, which looks up the database and sends the

member to appropriate sub-gateway (SG). SG then provides the member with a secret key (DK_i) and sub-gateway key (SGK_i).Each group acts as a separate independent group. Addition/Deletion of devices in the group does not affect the other embedded nodes in other groups. If n is the group size and m is the sub-group size then, m<<n. The number of rekey messages is reduced from O(n) to O(m). Though each sub-group has 'm' members with the complexity of O(m), there still exists " 1 affects m" phenomenon. There is the possibility of affecting all m members in each sub-group. This can still be reduced to O(log m) by implementing a binary tree inside each subgroup.Instead of the simple binary tree inside each sub-group, the concept of level homogeneous trees has been implemented and the rekeying and storage cost has been analyzed. The degree of communication will play a significant role to estimate the communication cost in the network. In this work, between cloud servers and the gateway, only one degree of communication and the between gateway and sub-gateways has the degree of 4 and the sub gateways to end nodes have the degree 2.Tree-based structure are established for key management among the n number of end devices, gateways, master gateways and network server.

When the degree of communication of the tree has increased the height of the tree is reduced. This implies that the number of keys stored in each member is also reduced. Hybrid (N–array) tree [6] is optimal in terms of a number of rekey messages and also storage. But maintaining an N-array tree structure is difficult. The binary tree, on the other hand, has greater tree height and the number of keys stored in each member and also in the server is higher. But the number of rekey messages is same as in N-array tree. The advantage of the binary tree is that maintaining a binary tree is very easy. The proposed protocol combines the advantages of both the binary and N-array trees. In this work, the number of rekey messages can still be reduced by implementing the concept of one-way key derivation. The storage is lesser than the binary tree since the upper levels have an N-array tree. Since the node addition takes place only in the lower levels the tree structure can be easily updated because it is a binary tree.Network Model with end devices, sub gateways and Master Gateway is described in figure 2.

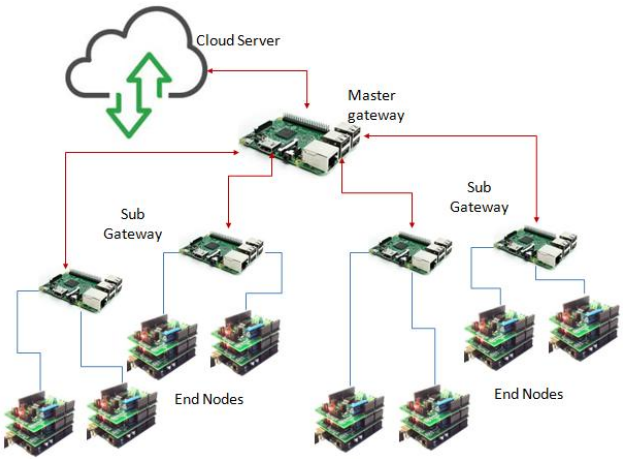


Figure 2: Network Model with end devices, sub gateways and Master Gateway

The generalized formula to find the number of rekey messages and the total number of keys [5, 6]in each member and also the total number of keys in the system has been derived and explained using the following example which consists of 08 end devices, two sub gateways,and one master gateways. For rekeying the concept of One-Way Key Derivation [4, 5] is used. During the leave, one of the non-leaving devices is chosen at each level and the information about the chosen devices is broadcast along with a salt value (K_g). The selected devices calculate the rekey using a one-way

hash function of the original key XOR'[ed] with the salt value. Only for the remaining members, the rekey message is being sent. Suppose Node 8 leaves the group, and then the keys to be updated are K_5, K_6, K_7, K_8, $K_{5\text{-}8}$, and $K_{1\text{-}8}$. One among the non-leaving device which uses the keys is chosen at each level and is broadcast by the server.A total number of rekey messages generated during device leaving are $\log_2 (m/d^2) + (d\text{-}1) \log_d d^2$, Total number of keys stored in each device is $\log_2 (m/d^2) + 2$ and Total number of keys stored in the entire system is $g(1+d-d^2+2m)+1$. Where d is the degree of N-array tree, m is the number of devices in each group and g is the total number of groups.Proposed key management framework with end devices and gateways are described in figure 3. The comparative declaration is presented in table1.

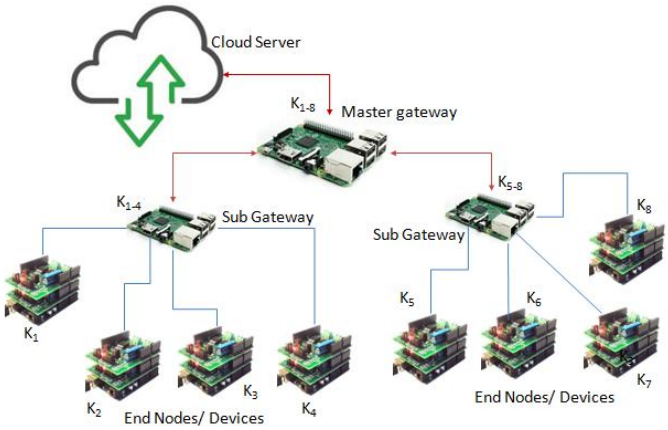


Figure 3: Proposed key management framework.

Table1: Analysis outcome

Category	STN	OFT	LKH	This Work
Total no of keys stored in group	Minimum	Maximum	Reasonable	Marginal
No. of keys stored in each member	Minimum	Maximum	Reasonable	Marginal
No. of rekeys generated during join	Minimum	Maximum	Reasonable	Marginal
No. of rekeys generated during leave	Minimum	Maximum	Reasonable	Marginal

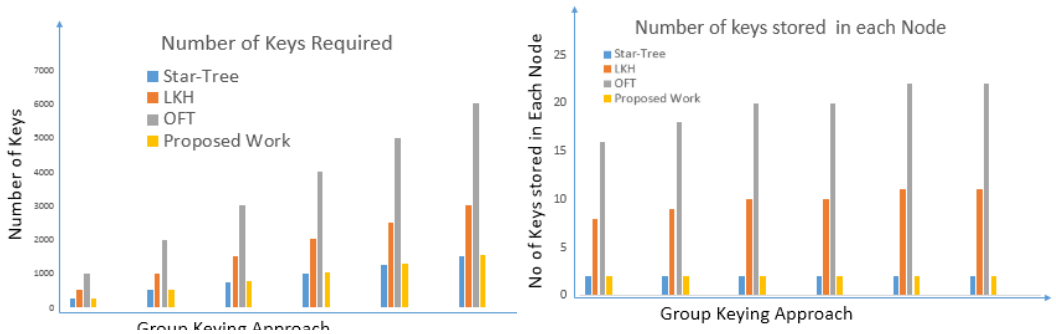


Figure 4: Number of Keys, Number of Keys stored in each node Vs. Group Keying approach

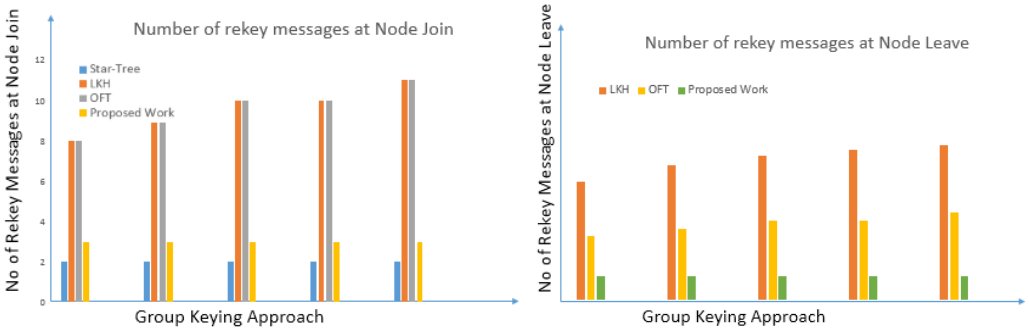


Figure 5: Number of Rekey Message at Node Join and Node LeaveVs. Group Keying approach

Discussion and Conclusion:

Security is a significant territory to defend the data communication amongtiny devices. CoAP is now becoming the customary practice for IoT applications. CoAP is fashioned on HTTP protocol and is intended for constrained resource devices like embedded devices. Hierarchical, hybrid and decentralized key management approach is also proposed and implemented with CoAP service. The outcome of this work are, a lightweight key management system is implemented which intern reduces the complexity results in effective power management.The proposed system when integrated with an IoT framework will provide enhanced security along with reduced storage space. Further, the overall system follows CoAP protocol which facilitates IoT integration, reduced memory utilization, enhance security and flexibility for adapting to all IoT applications.

References:

1. L. Eschenauer, V.D. Gligor, A key-management scheme for distributed sensor networks, in: Proceedings of the 9th ACM Conference on Computer and Communications security, ACM, 2002 pp.41-47.
2. B.Dutertre, S,Cheung, J.Levy, Lightweight key management in wireless sensor networks by leveraging initial trust, Technical Report SRI-SDL-04-02, SRI International, 2004.
3. M. Eltoweissy, M. Moharrum, R. Mukkamala, Dynamic key management in sensor networks, IEEE Commun. Mag. 44 (4), 122-130, 2006.
4. O. Catakoglu, A. Levi, Uneven kwy pre-distribution scheme for multi-phase wireless sensor networks, in: Information Sciences and Systems 2013, Springer, pp. 359-368, 2013.
5. R. Roman, C. Alcaraz, J. Lopez, N. Sklavos, Key management systems for sensor networks in the context of the internet of things, Cimput. Electr. Eng. 37(2), 147-159, 2011.
6. J. Zhang, V. Varadharajan, Wireless sensor network key management survey and taxonomy, J. Netw.Comput. Appl. 33(2) 63-75, 2010.
7. C. Lerche and K. Hartke and M. Kovatsch. Industry adoption of the Internet of Things: A constrained application protocol survey. IEEE International Conference on Emerging Technologies and Factory Automation, ETFA , 2012.
8. B.C. Villaverde and D. Pesch and R. De Paz Alberola and S. Fedor and M. Boubekeur. Constrained Application Protocol for Low Power Embedded Networks: A Survey. Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing , 2012

9. Zhang X, He J, Wei Q. EDDK: energy-efficient distributed deterministic key management for wireless sensor networks. EURASIP Journal on Wireless Communications and Networking 2011.
10. N. De Caro and W. Colitti and K. Steenhaut and G. Mangino and G. Reali. Comparison of two lightweight protocols for smartphone-based sensing. IEEE SCVT 2013 - Proceedings of 20th IEEE Symposium on Communications and Vehicular Technology in the BeNeLux , 0-5, 2013.
11. Levä, T. Comparing the cost-efficiency of CoAP and HTTP in Web of Things applications, 2014. 1–41.
12. D. Thangavel and X. Ma and A. Valera and H.X. Tan and C.K. Tan. Performance evaluation of MQTT and CoAP via a common middleware. IEEE ISSNIP 2014 - 2014 IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Conference Proceedings , 21-24, 2014.
13. Perera, C., Zaslavsky, A., Liu, C., Compton, M., Christen, P., Georgakopoulos, D.. Sensor Search techniques for Sensing as a Service Architecture for the Internet of Things. Sensors Journal, IEEE 2014; 14 (2):406–420. doi:10.1109 / JSEN.2013.2282292.
14. AElmangoush and R. Steinke and T. Magedanz and A.A. Corici and A. Bourreau and A. Al-Hezmi. Application-derived communication protocol selection in M2M platforms for smart cities. International Conference on Intelligence in Next Generation Networks, ICIN , 2015.
15. Talaminos-Barroso and M.A. Estudillo-Valderrama and L.M. Roa and J. Reina-Tosina and F. Ortega-Ruiz. A Machine-to-Machine protocol benchmark for eHealth applications - Use case: Respiratory rehabilitation. Computer Methods and Programs in Biomedicine , 1-11, 2016.