A framework for building secure software-defined wireless sensor networks

Cézar Murilo G. de Toledo, Marcos A. Simplicio Jr. and Cintia Borges Margi

***Abstract*—Software-Defined Wireless Sensor Network (SDWSN) refers to a paradigm that brings the flexibility of Software Defined Networking (SDN) to Wireless Sensor Network (WSN). Basically, an SDWSN consists in a architecture where a logically centralized controller manages data flows according to high level policies. Although this is a promising approach for giving administrators further control over the networked sensors' behavior, it also creates some security issues. In this paper, we identify attacks that are particular to the SDWSN paradigm, aiming to analyze the main threat vectors to which the architecture is exposed. Then, we present a security framework designed to mitigate such attacks and discuss its effectiveness.**

I. Introduction

Software Defined Networking (SDN) creates a network logic decoupled from the underlying hardware and managed by a remote control plane. The SDN paradigm considers three logically connected planes: data, control, and application. The data plane comprises the switches, responsible for forwarding packets. The control plane is run by the controller, which defines how switch should handle packets. The application plane is where users interact with the controller, developing applications and managing the network.

The SDN paradigm has become more prevalent in the last years, leading to many network deployments with support to the Southbound-based OpenFlow protocol. Unfortunately, however, the design of OpenFlow (OF) is not oriented toward wired networks [1]. In particular, it assumes high-speed switches that (1) can contact the network controller as often as necessary, and (2) can store (possibly large) tables with flow rules. Hence, OpenFlow does not always cope with the needs of resource-constrained environments like Wireless Sensor Networks (WSNs). This leads to the need of solutions able to deal with low speed network protocols (e.g., IEEE 802.15.4) and also with limited processing power and memory availability on network nodes. Indeed, these gaps have led to many research efforts focused on creating a Software-Defined Wireless Sensor Network (SDWSN) environment [2], [3].

Basically, SDWSNs aim at providing dynamic and scalable routing algorithms for accommodating the different needs of Internet-of-Things (IoT) applications running on the same WSN environment. Albeit promising, SDWSNs are prone to security vulnerabilities inherited from SDNs, such as link spoofing, access control abuse, denial of service (DoS), and man-in-the-middle (MitM). At the same time, they are also prone to WSN-related attacks, including sink holes and additional forms of DoS and MitM. Hence, the widespread adoption of SDWSN technologies depends at least in part on a robust security framework for addressing such threats. In the next section we identify relevant attacks against SDWSNs, using the open-source IT-SDN [2] as a basis for our analysis.

Cézar M. G. Toledo, Marcos A. Simplicio Jr., Cintia B. Margi. Universidade de São Paulo, Brasil. E-mail: {cmurilo, mjunior, cbmargi}@larc.usp.br. This work was funded by CAPES (Finance Code 001), FAPESP (grants 13/25977-7 and 2018/12579-7) and CNPq (grant 301198/2017-9).

II. SDWSNs vulnerabilities: attacks against IT-SDN

One vulnerability of SDWSNs, also present in IT-SDN, is the absence of authentication and authorization mechanisms. This means that new nodes (including attackers') can be added at will. This leads to three types of attacks, as follows.

A first issue is Man-in-the-middle (MitM) attacks, in which an intruder controls all traffic between two honest nodes. The intruder can then inject fake data into the communication, as well as eavesdrop the exchanged packets. This violates not only the communications' integrity and authenticity, but also break the confidentiality of application data (e.g., sensor readings) and control information (e.g., the network's topology and policies). In particular, in source-routed frameworks [2] this issue leaks information about existing network links, as well as about preferential routes for each network source and/or destination. The attacker can then build an accurate network fingerprint, which is useful for building subsequent attacks: e.g., the attacker can isolate one node by taking down its neighbors, thus affecting the network's availability. Those node's data can then be replaced with forged data.

Another attack consists in exploiting the underlying neighbor discovery protocol for impersonating the network controller. Specifically, if a malicious node announces itself as the controller, the victims are unable to notice that this information is fake. Besides creating a DoS effect from the original controller's perspective, the malicious controller can install any flow rules in the network's sensor nodes. The attacker can then steal application data and/or stop nodes from receiving data.

Finally, sink nodes can be falsified whenever the SDWSN routing approach is such that the controller adjusts routing rules so that all packets are delivered to the nearest sink, usually relying flow identifiers (flow-id) for this purpose [2]. In this scenario, the lack of authentication mechanisms allows a false sink node to advertise itself as one of the targets for a specific flow-id. Consequently, such a malicious sink gains access to the application's intended data, creating a sinkhole.

III. Securing SDWSNs

To the best of our knowledge, WS3N [4] is one of the few existing works that deals with security in SDWSNs. However,

WS3N has three main shortcomings [4]: (1) it generates many control packets, leading to poor scalability (e.g., a packet delivery rate below 10% for 64 nodes); (2) its bootstrapping mechanism is vulnerable to spoofed MAC address attacks, so malicious nodes can obtain valid certificates; and (3) it tackles key establishment between sensors and controllers, but no end-to-end security is provided between sensor and sink nodes.

To address such limitations of WS3N, we describe a secure and scalable SDWSN framework that, building upon IT-SDN, provides data confidentiality, integrity and authenticity. The resulting system thwarts the attacks from Sec. II while avoiding the aforementioned limitations of WS3N. Our proposal builds upon iSMQV [5], an authenticated key agreement (AKA) protocol which was designed specifically with WSNs in mind. Is assumes that trusted nodes are (pre-)loaded with valid credentials, including a public key. Nodes carrying a certified public key can, when entering the network, run iSMQV to establish a shared key with the controller; such nodes are then considered authorized to participate in security-enabled routes. Nevertheless, non-authorized nodes may also be tolerated in network, acting as forwarding devices if desired. The application layer can then define which types of nodes can participate in the routes for a given flow, nodes responsible for that flow indicate their preference by setting the corresponding security flags in their packets. This preference is then enforced by the controller: if a secure route is required, the controller includes only authorized nodes in the corresponding routes. This enables a more flexible and efficient route management, in which security can be prioritized whenever needed.

To indicate the end-to-end security mode to be employed, we use two bits (out of 40) from the packet header employed in the IT-SDN framework. This enables four possible security modes for data packets sent by sensor nodes to sinks, and also for control flow messages exchanged between sensor nodes and controllers: (`00`) no security; (`01`) authentication only, meaning that a message authentication code is employed for all relevant messages; (`10`) authentication and encryption, in which case an authenticated-encryption mode is employed for relevant messages; (`11`) authentication and encryption in a route containing only authorized nodes. The flowchart that describes the system's behavior is shown in Fig. 1: after running a neighbor discovery protocol and installing flow rules for reaching the (possibly fake) controller, each node run the AKA protocol with its public key; control flow requests may then include security services, enforced by the controller when building a suitable route; source and sink nodes then run the AKA protocol, using the controller as a proxy to enable a sink-to-source communication;the source node encrypts and/or authenticates the packets with the established symmetric keys.

The aforementioned attacks are mitigated by the proposed approach as follows. First, MitM attacks are addressed via the secure route flag: only authorized nodes participate in the transmission of packets. Actually, even if non-authorized nodes are allowed to participate in the data transmission, encryption services prevent data leakage, while their authentication thwarts forgery attempts. Controllers are authenticated by authorized nodes as soon as the system is set-up. Hence, malicious controllers are quickly detected and prevented from configuring invalid routes. Finally, all provided security services rely on authorized nodes, which are authenticated by the source nodes during route construction (attacks that infect such nodes are out of scope).As a result, as long as authorized sinks are not compromised, no application data is leaked.

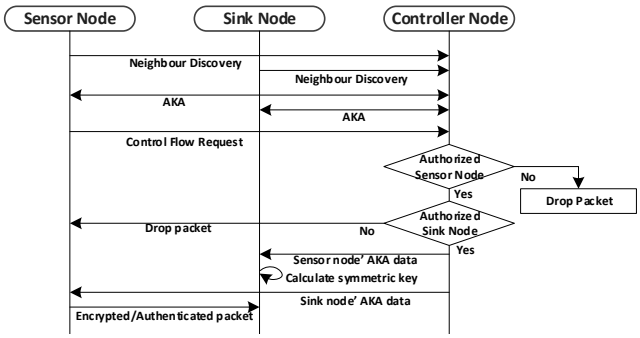

Fig. 1. Proposed flow for enabling a secure SDWSN.

IV. Conclusions

One challenge faced by SDWSN solutions, is that they end up combining the attack surface of both WSNs and SDNs. Despite the existence of many studies that deal with security vulnerabilities in either WSNs or SDNs, solutions capable of handling the security of SDWSNs are scarce. In this paper, we tackle this open issue first by listing some of the most relevant SDWSN-related attacks. We discuss how such attacks can lead to stolen or tampered data in a concrete SDWSN solution, the open-source IT-SDN framework. Then, we propose a solution that, building upon IT-SDN's properties and on lightweight key management protocols, enforces end-to-end authenticity, confidentiality and integrity to data and control packets. When compared to related works (e.g., [4]), our proposal provides on-demand establishment of security relationships between sensor nodes, as well as between nodes and controller. Also, the proposal allows nodes unable to establish such a security relationship to participate in delivering packets, as long as authorized by the corresponding application. Future works include the experimental analysis of the proposal's performance.

References

[1] I. Akyildiz, P. Wang, and S. Lin, "Softwater: Software-defined networking for next-generation underwater communication systems," *Ad Hoc Networks*, vol. 46, pp. 1–11, 2016.

[2] R. Alves, D. Oliveira, G. Núñez, and C. Margi, "IT-SDN: Improved architecture for SDWSN," in *XXXV Brazilian Symposium on Computer Networks and Distributed Systems*, 2017.

[3] O. Flauzac, C. Gonzalez, A. Hachani, and F. Nolot, "SDN based architecture for IoT and improvement of the security," in *29th Int. Conf. on Advanced Information Networking and Applications Workshops (WAINA)*. IEEE, 2015, pp. 688–693.

[4] R. Alves, D. Oliveira, G. Pereira, B. Albertini, and C. Margi, "WS3N: Wireless secure SDN-based communication for sensor networks," *Security and Communication Networks*, vol. 2018, 2018.

[5] M. Simplicio Jr, M. Silva, R. Alves, and T. Shibata, "Lightweight and escrow-less authenticated key agreement for the Internet of Things," *Computer Communications*, vol. 98, pp. 43–51, 2017.