SDN Enabled Secure IoT Architecture

Kallol Krishna Karmakar*, Vijay Varadharajan*, Surya Nepal†, Uday Tupakula*,
*Advanced Cyber Security Engineering Research Centre
The University of Newcastle, Australia -2308
†Data 61, CSIRO,
Australia, Marsfield NSW 2122
kallolkrishna.karmakar@uon.edu.au*,[vijay.varadharajan,uday.tupakula]@newcastle.edu.au*,Surya.Nepal@data61.csiro.au†

***Abstract*—The Internet of Things (IoT) is increasingly being used in applications ranging from precision agriculture to critical national infrastructure by deploying a large number of resource-constrained devices in hostile environments. These devices are being exploited to launch attacks in cyber systems. As a result, security has become a significant concern in the design of IoT based applications. In this paper, we present a security architecture for IoT networks by leveraging the underlying features supported by Software Defined Networks (SDN). Our security architecture restricts network access to authenticated IoT devices. We use fine granular policies to secure the flows in the IoT network infrastructure and provide a lightweight protocol to authenticate IoT devices. Such an integrated security approach involving authentication of IoT devices and enabling authorized flows can help to protect IoT networks from malicious IoT devices and attacks.**

***Index Terms*—Internet of Things (IoT) Security, Software Defined Network (SDN) Security, Policy based Secure IoT Architecture, IoT Authentication and Access Control.**

I. Introduction

The Internet of Things (IoT) is being touted as the next big technological trend by both academic researchers and industry players. The IoT refers to the connection of 'things' in the real world via the Internet, so that these 'things' can communicate with each other as well as other Internet services. Examples of devices range from smart light bulbs, to door locks, to traffic sensors, to baby monitors and healthcare devices. These IoT devices can be small and resource constrained as well as embedded in other real world objects. According to an industry estimate, the number of IoT devices is expected to reach 50 billion by 2020 [1]. IoT enables the connection of the physical world with the cyber world, allowing remote monitoring and control of physical objects from the cyber world [2, 3]. The heterogeneity of the IoT devices, underlying communication infrastructure and the different types of protocols used by these devices increase the complexity of the IoT infrastructure and make it vulnerable to security attacks. As has been the case with the Internet, security has become an after-thought, and many IoT devices and applications are already in operation without any security mechanism. A major challenge is how to provide security in such IoT infrastructures [4, 5].

Due to the resource constrained nature of IoT devices, an attacker is able to target them easily by sending forged requests, intercepting and illegally manipulating valuable sensor data in transit or capturing a physical device and transforming it to a zombie to launch attacks on other systems. Denial of service (DoS) and energy depletion attacks are the most common IoT attacks [6, 7]. Often it is not possible to implement security on these devices using traditional defence mechanisms as they are located in open environments and they would incur extra computational load on small IoT devices. One main issue is how to ensure that only authenticated data from legitimate IoT devices is used in cloud processing and in service provision. In this paper, we develop a network based architecture solution that can help to secure the IoT infrastructure. The main idea behind our solution is that it enables only authenticated data from legitimate IoT devices that are able to discover and use the network services, as well as enabling only the authorized, network services to consume data only from authenticated devices.

Our security architecture makes use of Software Defined Networks (SDN) to manage the IoT infrastructures securely. In SDN infrastructure, the Controller has visibility over its network domain, the applications running in the Controller can manage the security of the underlying IoT network infrastructure. The use of SDN to manage the IoT devices is not in itself new; other works [8, 9] have used such an approach to detect malicious devices and detect attacks. However, in our architecture, SDN Controller acts as a security policy decision authority, and the network switches and IoT gateways enforce the security policies in the IoT network infrastructure. Such a policy-driven approach provides the capability to achieve secure management of network flows in an IoT infrastructure in a dynamic manner, and deal with security attacks in a proactive manner.

In this paper, we describe a policy-driven security architecture using SDN for an IoT network infrastructure. The security architecture uses fine grained policies to control and manage services and IoT devices as well as secure the corresponding flows in the IoT infrastructure. For instance, policies can enforce the creation of secure channels for data from particular authenticated IoT devices through specific gateways to the cloud. This can help to achieve secure flow management in the IoT network infrastructure. Our security architecture also provides device authentication and data integrity enabled IoT services as well as device authorization for network services. A lightweight elliptic curve cryptography (ECC) based protocol has been deployed to achieve authentication of IoT devices and integrity of IoT data. Authorization for network services requested by different IoT devices is achieved using Open Authorization (OAuth) protocol [10]. The proposed architecture has been implemented using ONOS SDN Controller.

The rest of the paper is structured as follows. Section II gives an overview of IoT infrastructure. Section III describes the SDN based security architecture for IoT infrastructure. Finally, Section IV concludes the paper.

II. Internet of Things (IoT)

In this section, we give a summary of the different layers in an IoT architecture and also explain why we have chosen SDN infrastructure for managing IoT network.

A. IoT Infrastructure:

An IoT infrastructure can be thought of as consisting of four architecture layers, namely Perception, Network, Middleware and Application Layers [11]. The Perception Layer acts as an interface to the physical world and consists of actuators and sensors. This layer transfers the raw data to the Network Layer. The Network Layer processes and routes the data through the network infrastructure. Further processing of the data happens at the Middleware Layer. This layer helps to perform actions pre-encoded by third-party applications running on the Application Layer. The Application Layer provides the interface for third-parties to develop and execute their applications for storage and further processing of the device data.

B. Why SDN for IoT Security?

The features of SDN technology which make it a suitable platform for securing IoT infrastructure are as follows:
Separation of Control Plane from Data Plane: This is useful for designing our policy based security architecture at the SDN Controller in the control plane and enforcing the security policies in the network switches and IoT devices in the data plane. The SDN Controller communicates with the switches in the data plane using open and standardised interfaces and protocols (OpenFlow), which is useful for securing the communications between the policy decision authority in the Controller and the enforcement mechanisms in the IoT devices and switches.
Network Domain View: The SDN Controller has visibility over the whole network domain under its jurisdiction. This can be used by our architecture to achieve secure management of IoT devices and flows in the network infrastructure. The Controller maintains a topological information database which logs information about all the forwarding devices connected to the Controller. This will be useful in the specification of path based security policies in our architecture.
SDN Northbound Applications: The SDN provides flexible northbound API which enables us to develop secure applications or use third-party applications to securely monitor and control the behaviour of IoT devices and network nodes in the SDN network domain. Our security architecture has a secure application running on top of the Controller (developed using its northbound API) that provides security services in the IoT infrastructure.

III. SDN based Security Architecture for IoT Network Infrastructure

We use Software Defined Networks (SDN) to develop our security architecture for IoT network infrastructure.

A. Security Architecture for IoT Network Infrastructure

Our proposed SDN based security architecture uses policies to control and manage IoT devices, services and network entities (switches, nodes and gateways). Figure 1 shows the proposed SDN based security architecture. The heart of the security architecture is the SDN Controller where security policies reside and are evaluated. The IoT actuators and sensors are the end devices and connect to the IoT Nodes. These IoT Nodes are connected to the IoT Gateways either via wired or wireless networks. The IoT Gateways are connected to the SDN Controller. In some cases, OpenFlow switches can act as IoT Gateways/Nodes. Our implementation considers OpenFlow switches as IoT Gateways. Users are connected to the OpenFlow switches.
The IoT devices are of a heterogeneous nature. The IoT devices can use different network protocols, authentication mechanism, can have different operation and application platforms. Furthermore, there can be a large number of IoT devices. Hence, a scalable solution is needed, recognizing the individual capabilities of the connected IoT devices. In our security architecture, we have created a device provisioning mechanism for each IoT device groups using a template based approach. Each template consists of variables explaining device capabilities like protocol, manufacturer, authentication mechanisms and other features. This device provisioning functionality is integrated with the core application of the Controller, enabling it to provision the IoT devices at runtime. In our implementation, we have developed two secure applications that control and manage the behaviour of IoT devices in the network. These applications runs over an SDN Controller. The first one is the *Policy based Security Application (PbSA)*, and the second one is called *IoT Security Applications (ISA)*. *ISA* has two sub-modules, namely *IoT Authentication Authority* and *IoT Authorization Authority*. Before describing in detail the functions of each of these applications and modules, we first provide a high-level overview of the security interactions with the IoT devices in the network infrastructure.
The operation of our security architecture can be viewed as involving two phases. In the first phase, we have interactions that are associated with authentication. New IoT devices that are to be connected to the network need to be authenticated. IoT Gateways forward IoT device information (such as device ID) using *Packet_in* messages to the SDN Controller. The *IoT Authentication Authority* module in the ISA application carries out the authentication process using a lightweight elliptic curve cryptography (ECC) based authentication protocol. This process of IoT authentication will in turn make the network domain and services visible to authenticated IoT Devices.
In the second phase, the PbSA checks the network service request from authenticated IoT devices against the specified security policies. The PbSA has fine grained policies that control the behaviour of the IoT devices; for instance, a user "A" accessing via OpenFlow Switch 3 (*SW3*) is allowed to read data from Sensor 2 connected through *Gateway I (SW1 & Node NII)*. We describe the security policies and their implementation in detail in Section III-B. If the service request is permissible according to *PbSA* security policies, then our security architecture uses the OAuth protocol to provide a

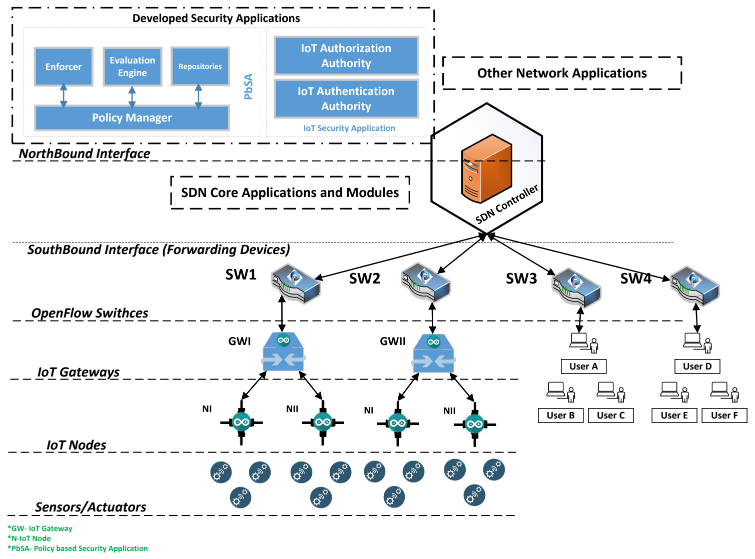

Fig. 1: Security Architecture for IoT Network Infrastructure

token to the IoT device. The IoT device uses this token for further communication in the network. This process is jointly performed by *PbSA* and *IoT Authorization Authority*.

B. Policy Based Security Architecture

Our IoT network infrastructure consists of OpenFlow switches/ IoT Gateways and end hosts (IoT sensors/ actuators). We have used a single SDN Controller to manage the IoT network infrastructure. The network devices (OF devices) forward the packets generated by the IoT devices/ users, which are then subjected to the policies specified in the SDN Controller for transfer in the network. Figure 1 shows the Policy-based Security Architecture (PbSA) for securing the IoT Infrastructure using SDN. As *PbSA* is designed to be modular, the components of *PbSA* can be implemented on a single host or can be distributed over multiple hosts. Here, we provide a detailed description of different modules of *PbSA*.

PbSA consists of four major modules, namely a) Policy Manager, b) Evaluation Engine, c) Repositories & d) Policy Enforcer.

- **Policy Manager:** It is the core component of the security architecture, as it manages every operation such as extracts IoT device flow attributes, updates the Topology Repository, and instructs the Policy Enforcer to enforce the policies at the OpenFlow IoT Gateways. It also communicates with the *ISA* application for the transfer of OAuth token after checking the network service request from the IoT devices.
- **Evaluation Engine:** It evaluates the service request against the relevant policies stored in the Policy Repository.
- **Repositories:** Our architecture has two repositories: a) Topology Repository and b) Policy Repository. The Topology Repository contains the network topology of IoT devices and end hosts/users. SDN controller has its own device Topology Repository. We are using the same Topology Repository for this purpose. The Policy Repository contains the Policy Expressions (PE) associated with the various IoT devices and the associated flow attributes. The attributes in PEs also include security parameters such as security labels associated with the OpenFlow IoT Gateways.
- **Policy Enforcer:** Fetches the required information from the OpenFlow IoT Gateways and enforces the flow rules obtained from the Policy Manager.

Security Policy

The security policies are expressed as Policy Expressions (PE), which specify whether packets and flows from IoT devices and end hosts follow a particular path or paths in the network, and the conditions under which the packets and flows follow these paths. The PE syntax uses an enhanced version of RFC1102 [12]. They are fine-grained and specify a range of policies using various attributes of IoT devices and flows; for instance, these attributes include different types of devices, source and destination attributes, flow attributes and constraints, requested services, security services and security labels. The attributes are: **(a) Flow Attributes:** Flow ID, sequence of packets associated with the Flow, type of packets, Security Profile indicating the set of security services that are to be associated with the packets in the Flow; **(b) Device Attributes:** ID specific to IoT sensor/actuator; **(c) Switch Attributes:** Identities of the Switches and Security Label of the Switches (In our case, these are OpenFlow IoT Gateways); **(d) Host Attributes:** Identities of Hosts such as Source/Destination Host ID; **(e) Flow and Domain Constraints:** Constraints such as Flow Constraints (FlowCons) and Domain Constraints (DomCons) associated with a specific device Flow; **(f) Services:** For which the PE applies (e.g. FTP storage access); **(g) Time Validity:** The period for which the PE remains valid; and **(h) Path:** Indicates a specific sequence of switches any particular flow from specific IoT devices/users should traverse.

The Constraints (Flow and Domain) are conditions that apply to specific flows from any IoT devices. For instance, a constraint might specify the flow from a specific type of sensor, should only go through a set of switches that can provide a gurrented bandwidth. From a security point of view, a constraint could be that a flow should only go through OpenFlow switches that have a particular Security Label. The PEs support wildcards for attributes, enabling it to set policies for a group of IoTs/services. A simplified Policy Expression template is as follows:

PE_i= $< FlowID, IoTDeviceID, SourceAS, DestAS, SourceHostIP, DestHostIP, SourceMAC, DestMAC, User, FlowCons, DomCons, Services, Sec-Profile, Seq-Path >:< Actions >$
where i is the Policy Expression number.

C. IoT Security Application (ISA)

In this section, we briefly explain the security protocols used in our security architecture. These are implemented by the *IoT Security Application (ISA)*.

In our architecture, each IoT device has a unique ID and connects to IoT Gateways using wireless networks. The IoT Gateways are basically OpenFlow Switches/ Access Points (OF-AP) which support OpenFlow protocol. As mentioned earlier, our security protocol has two phases, namely the authentication phase followed by the policy application and the OAuth protocol phase.

IoT devices are authenticated in the first phase of the security protocol. We have used a lightweight ECC protocol [13] to authenticate the IoT devices. An IoT device sends a “Hello” message with its ID (Message X in Figure 2) requesting for a Network Service (NS). The OF-APs sends the ID and Network Service (NS) request via $Packet\_in$ messages to the *IoT Authentication Authority*. *IoT Authentication Authority* performs authentication of the IoT devices using ECC based protocol. At the end of the authentication phase, *IoT Authentication Authority* authenticates the IoT device and establishes a common secret key with the IoT device that can be used for further secure communications.

In the authorization phase, the *PbSA* checks the Network Service (NS) request from the IoT device using the security policy specifications described above. The *IoT Authentication Authority* uses the key established by the ECC protocol in the authorization request to *PbSA*. If the device and flow attributes for the Network Service (NS) satisfy the Policy Expressions, then the PbSA requests the *IoT Authorization Authority* to generate the OAuth Token for the particular IoT device for the specific Network Service(s) requested. The token is then used by the authenticated IoT device to access the required Network Service(s). This process is illustrated in Figure 2.

Lightweight ECC based Authentication for IoT Devices

The Elliptic Curve Cryptography (ECC) based public key system uses the algebraic structure of elliptic curves as their finite points. It is computationally faster than other public key cryptosystems such as RSA. Hence it is more suitable for computationally constrained IoT devices. It can be used to achieve key agreement as well as encryption and digital signature. In our security architecture, the IoT devices are authenticated using a lightweight ECC based authentication protocol proposed in [13]. Then we use the key established using this protocol to encrypt the data from the authenticated IoT devices. This lightweight security protocol works in conjunction with the OAuth protocol.

The ECC based authentication protocol used in our architecture has 3 stages: Setup, Installation and Key Agreement. Algorithm 1 below gives an outline of the 3 stages. The Key Agreement (Stage 3) is illustrated in Figure 3. The performance of the proposed protocol (in terms of both computation and communication costs) is lower than the previously proposed protocols. Hence it is particularly suitable for the IoT environment as it shifts the processing load from the resource-limited devices to the more powerful servers in the Controller.

Algorithm 1: ECC Protocol

```
1 Step1 (Setup): a) The IoT Authentication Authority
    chooses a prime number $q$, computes $F_q, E/F_q, G_q, P$;
  b) The IoT Authentication Authority chooses a master
    key $x$ and calculates public key $P_{pub} = xP \in E/F_q$;
  c) It then computes two hashes $H1$ and $H2$;
  d) $F_q, E/F_q, G_q, P, P_{pub}, H1, H2$ are made public.;
2 Step2 (Installation): IoT Authentication Authority and
    devices separately generate and validate their private
    and public keys.;
  $R_{gt} = r_{gt}P$, where $r_{gt}$ is a random number $r_{gt} \in Z_q^*$.
  $y_{gt} = H_1(ID_{gt}, R_{gt}).x$
  $R_d = r_dP$, where $r$ is a random number $r_i \in Z_q^*$.
  $y_d = H_2(ID_d, y_{gt}).x$
  $S_d = r_d + y_d$
  $S_d$ and $R_d$ are the private and public values of $d$.
  At the end of this stage each device has its own
    $S_d, R_d, y_d, r_d$ and IoT Authentication Authority has
    $(y_{gt}, r_{gt})$;
3 Step3 (Key Agreement): Between IoT Authentication
    Authority and IoT Devices;
```

In [13], the authors analyze the security of the ECC authentication scheme and show that it is secure against active attackers who are capable of eavesdropping, modifying and injecting messages in the protocol.

Authorization for Network Services using OAuth Protocol

The authorization service using the OAuth protocol [10, 14] works as follows: It consists of four actors: i) Client, ii) Resource Owner, iii) Resource Server and iv) Authorization Server. The client contacts the Resource Owner of the resource. The Resource Owner grants the access to the client by sending an authorization code. The client delivers the received authorization code to the Authorization Server. The Authorization Server verifies the authorization code and releases a token containing the details of the consent provided to the client (time limit, scope, and so on). The client forwards the token to the Resource Server. The Resource Server checks the validity of the received token, and in the affirmative case provides access to the protected resource.

In our SDN-IoT architecture, network services are the resources. Figure 2 shows the OAuth Token handover process (marked with a continuous green line). First, the *IoT Authentication Authority* authenticates the IoT devices using ECC protocol mentioned above. After the authentication phase, it requests access to network services on behalf of the authenticated IoT devices. The PbSA checks the policy repository and if the network service request is valid, then it issues an *Authorization Permission (Kp)*, which is forwarded to the

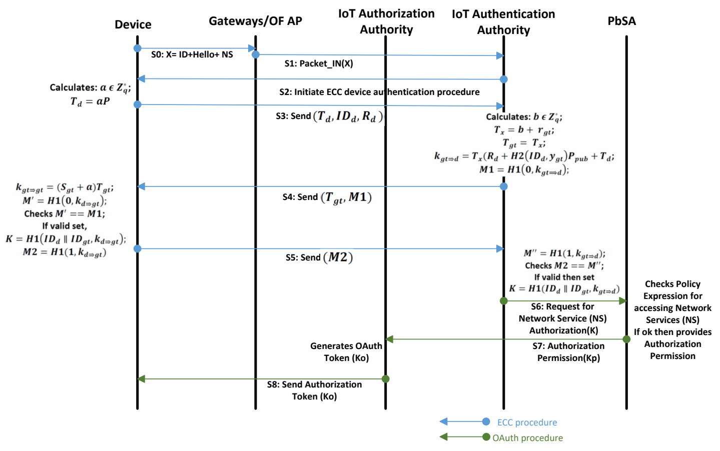
Device
Gateways/OF AP
IoT Authorization Authority
IoT Authentication Authority
PbSA
S0: X= ID+Hello+ NS
S1: Packet_IN(X)
S2: Initiate ECC device authentication procedure
Calculates: $a \in Z_q^*$; $T_d = aP$
S3: Send (T_d, ID_d, R_d)
Calculates: $b \in Z_q^*$; $T_x = b + r_{gt}$; $T_{gt} = T_x$; $k_{gt\Rightarrow d} = T_x(R_d + H2(ID_d, y_{gt})P_{pub} + T_d$; $M1 = H1(0, k_{gt\Rightarrow d})$;
S4: Send $(T_{gt}, M1)$
$k_{gt\Rightarrow gt} = (S_{gt} + a)T_{gt}$; $M' = H1(0, k_{d\Rightarrow gt})$; Checks $M' == M1$; If valid set, $K = H1(ID_d \parallel ID_{gt}, k_{d\Rightarrow gt})$; $M2 = H1(1, k_{d\Rightarrow gt})$
S5: Send $(M2)$
$M'' = H1(1, k_{gt\Rightarrow d})$; Checks $M2 == M''$; If valid then set $K = H1(ID_d \parallel ID_{gt}, k_{gt\Rightarrow d})$
S6: Request for Network Service (NS) Authorization(K)
Checks Policy Expression for accessing Network Services (NS) If ok then provides Authorization Permission
S7: Authorization Permission(Kp)
Generates OAuth Token (Ko)
S8: Send Authorization Token (Ko)
ECC procedure
OAuth procedure

Fig. 2: Security Protocol process diagram

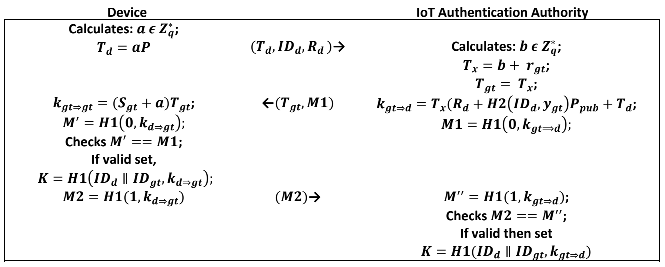
Device
IoT Authentication Authority
Calculates: $a \in Z_q^*$; $T_d = aP$
$(T_d, ID_d, R_d)\rightarrow$
Calculates: $b \in Z_q^*$; $T_x = b + r_{gt}$; $T_{gt} = T_x$;
$k_{gt\Rightarrow gt} = (S_{gt} + a)T_{gt}$; $M' = H1(0, k_{d\Rightarrow gt})$; Checks $M' == M1$; If valid set, $K = H1(ID_d \parallel ID_{gt}, k_{d\Rightarrow gt})$;
$\leftarrow(T_{gt}, M1)$
$k_{gt\Rightarrow d} = T_x(R_d + H2(ID_d, y_{gt})P_{pub} + T_d$; $M1 = H1(0, k_{gt\Rightarrow d})$;
$M2 = H1(1, k_{d\Rightarrow gt})$
$(M2)\rightarrow$
$M'' = H1(1, k_{gt\Rightarrow d})$; Checks $M2 == M''$; If valid then set $K = H1(ID_d \parallel ID_{gt}, k_{gt\Rightarrow d})$

Fig. 3: Key agreements between Devices and *IoT Authentication Authority*

IoT Authorization Authority. The *IoT Authorization Authority* generates the *OAuth Token (Ko)*. We have used JSON Web Token (JWT) for this purpose. The *IoT Authorization Authority* forwards the OAuth Token to the IoT devices. The *OAuth Token (Ko)* contains the user ID, IoT device ID, device type (e.g. sensor or actuator) and expiry time (network service access time). The IoT device integrates this OAuth Token in their network packet while accessing the network services.
In our current implementation, we have signed the OAuth Token using the private key of *IoT Authentication Authority* S_gt, which was used in the ECC authentication phase. The signature is verified by the network service using the public key of *IoT Authentication Authority* R_gt before service provision. (In a more general distributed architecture where *IoT Authorization Authority* and *IoT Authentication Authority* are not co-located, the ECC authentication protocol will be used to generate public and private values for *IoT Authorization Authority* using a similar process as for an IoT device and this private key will be used to sign the OAuth Token).

IV. Conclusion

Threats towards the IoT network infrastructure are increasing on a daily basis. Due to resource constraints and diversity of devices, it is hard to deploy existing legacy security measures in IoT networks. In this paper, we focused on proposing a security solution for IoT network infrastructure. Our proposed security architecture uses SDN to provide authentication and authorization services to IoT network infrastructure. Our proposed security architecture aligns with the emerging Software Defined Perimeter (SDP) paradigm which aims to restrict network access and connections between allowed elements using a software controlled architecture [15]. Our SDN-IoT security architecture allows communication for IoT devices that have been authenticated (IoT Authentication Authority) and whose requests for network services are authorized against a set of policies specified in our PbSA which determines whether the security constraints have been met. The separation between control and data planes has been achieved using the SDN architecture whereby the ability to control access is managed separately from the transmission of data and the communications between the IoT devices are virtualised using overlays that represent logical paths over a physical network. We believe such a combination of SDN based policy-driven security architecture together with lightweight authentication protocol and OAuth authorization framework offers a novel approach for secure provision and management of services in an IoT network infrastructure.

References

[1] R. Bogue, "Towards the trillion sensors market," *Sensor Review*, vol. 34, no. 2, pp. 137–142, 2014.
[2] J. Gubbi et al., "Internet of things: A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29-7, pp. 1645–1660, 2013.
[3] S. D. T. Kelly et al., "Towards the implementation of iot for environmental condition monitoring in homes," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3846–3853, 2013.
[4] R. H. Weber, "Internet of things–new security and privacy challenges," *Computer law & security review*, vol. 26, no. 1, pp. 23–30, 2010.
[5] M. Medwed, "Iot security challenges and ways forward," in *Proceedings of the 6th International Workshop on Trustworthy Embedded Devices*. ACM, 2016, p. 55.
[6] L. A. B. Pacheco et al., "Evaluation of distributed denial of service threat in the internet of things," in *Network Computing and Applications (NCA), 2016 IEEE 15th International Symposium on*. IEEE, 2016, pp. 89–92.
[7] X. Cao et al., "Ghost-in-zigbee: Energy depletion attack on zigbee-based wireless networks," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 816–829, 2016.
[8] M. Miettinen et al., "Iot sentinel: Automated device-type identification for security enforcement in iot," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 2177–2184.
[9] L. Galluccio et al., "Sdn-wise: Design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks," in *Computer Communications (INFOCOM), 2015 IEEE Conference on*. IEEE, 2015, pp. 513–521.
[10] D. Hardt, "The oauth 2.0 authorization framework," 2012.
[11] L. Da Xu et al., "Internet of things in industries: A survey," *IEEE Transactions on industrial informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.
[12] D. Clark, "Policy routing in internet protocols. request for comment rfc-1102," *Network Information Center*, 1989.
[13] A. Mohammadali et al., "A novel identity-based key establishment method for advanced metering infrastructure in smart grid," *IEEE Trans. on Smart Grid*, 2016.
[14] S. Sciancalepore et al., "Oauth-iot: An access control framework for the internet of things based on open standards," in *Computers and Communications (ISCC), 2017 IEEE Symposium on*. IEEE, 2017, pp. 676–681.
[15] D. Conde, "Software-defined perimeters: An architectural view of sdp," https://sdn.ieee.org/newsletter/march-2017/software-defined-perimeters-an-architectural-view-of-sdp, March 2017.