The Impact of Manufacturer Usage Description (MUD) on IoT Security

Zeno Heeb, Onur Kalinagac, Wissem Soussi and Gürkan Gür
Zurich University of Applied Sciences (ZHAW) InIT
Winterthur 8401, Switzerland
heebzen1@students.zhaw.ch, [name.surname]@zhaw.ch

***Abstract*—With the growing number of IoT (Internet of Things) devices and their particular characteristics compared to traditional systems, incumbent security mechanisms need to be advanced for secure and resilient IoT operation in current ICT systems. One particular standard, which tries to improve IoT security in that regard, is the Manufacturer Usage Description (MUD) by IETF. In this paper, as our main focus is to highlight the security gains of using MUD, we first discuss the critical threats to IoT devices based on available research. In the second step, we analyze the MUD technology to delineate where MUD is beneficial (or not) to address these security issues.**

I. Introduction

With the massive amount of IoT devices getting connected to networks, many vulnerabilities also emerge. This is a challenging situation mainly due to their cyberphysical nature as well as the constrained hardware and lightweight security mechanisms that come with the low price of such devices. Numerous research activities are underway to establish secure mechanisms that meet the requirements of such IoT devices. One research effort in that vein is the Manufacturer Usage Description (MUD) standard by IETF. With the MUD architecture, a device's communication should be limited to a predetermined expected behaviour, reducing the attack surface and making the network more secure. This paper considers some of the critical threats to an IoT device and discuss if the MUD architecture can alleviate the risk of these threats (*i.e.,* its utility).

II. IoT Security

The architecture of an IoT device can be described as a three or five-layer structure [1], [2]. While the three-layer architecture contains the perception, network, and application layers, the five-layer architecture extends it with two additional layers. In the literature, the naming and positions of these two additional layers are inconsistent. Since the three-layer architecture is adopted more and a more complex one is not substantially beneficial in grouping the security threats, we will rely on that architecture to consider IoT threats in a layer-specific setting.

a) Perception Layer: The perception layer is also known as the physical layer as it contains the physical device, sensors, and actors [2], [3]. This layer collects the data from the sensors and forwards them to the network layer. It also sends the actors the information received from the network layer.

b) Network Layer: The network layer is also called the transportation or communication layer [1], [2]. Its task is to exchange data between the perception layer in the device and the application layer in the cloud or server. Therefore, a wired or wireless connection is required.

c) Application Layer: The application layer feeds the received data from the network layer into services for users [1], [3]. Networked services for applications such as smart homes, smart cities, or intelligent health are defined in this layer. They provide what a user needs rather than just raw data.

A. IoT Security Challenges

IoT security differs from traditional ICT security due to the practical limitations of an IoT device which can be split into three groups: hardware-based, software-based, and network-based limitations.

a) Hardware limitations: Many IoT devices run on battery power and thus need hardware with energy consumption as low as possible [4]. Then, there is much less processing power and CPU-consuming cryptographic algorithms are not preferable in such a device. Limited memory is another issue. This is an evident challenge because most algorithms are not designed to occupy very low memory. An additional problem is that an IoT device must be tamper-resistant since it is a cyberphysical system and often located outside a controlled realm. This means an attacker can physically access the device and try to exfiltrate data or extract/replace security material such as cryptographic secrets.

b) Software limitations: IoT operating systems (OS) are supposed to have a lean network protocol stack and thus could lack capable security modules [4]. Another problem is the update/patching of an IoT device over its lifetime. To automatically update a device, a connection to the Internet or a local update service is needed. However, an IoT device is not essentially always connected to the Internet. Additionally, not all of these systems have a local update server, and some OSs can not fetch and integrate updates seamlessly.

c) Network limitations: A further challenge is the mobility of an IoT device where it joins to a proximal network without prior configuration [4]. However, most of the traditional security schemes are not scalable and lack feasibility. IoT devices use a broad range of media and multiple protocols like non-IP protocols for intra-network communication and IP protocols for communication with devices outside the local

DOI: 10.1109/6GNet54646.2022.9830354

```
{
 "ietf-mud:mud": {
   //General metadata are placed here
   "from-device-policy": {
       "access-lists": {
           "access-list": [ {"name": "mud-76100-v6fr"}]}}},
 "ietf-access-control-list:acls": {
   "acl": [{
       "name": "mud-76100-v6fr", "type": "ipv6-acl-type",
       "aces": { "ace": [{
                 "name": "cl0-frdev",
                 "matches": {
                   "ipv6": {
                     "ietf-acldns:dst-dnsname": "test.example.com",
                     "protocol": 6 },
                   "tcp": {
                     "ietf-mud:direction-initiated": "from-device",
                     "destination-port": {
                         "operator": "eq", "port": 443}}
                 },
                 "actions": {"forwarding": "accept"}
    ...
}
```

Fig. 1. Sample MUD file from the MUD RFC 8520 [5]

network. This multiple protocol characteristic of IoT devices is not adequately supported in traditional security schemes [4]. Additionally, existing security models do not cope well with a dynamic network topology with IoT devices repeatedly connecting/disconnecting and perhaps from different locations.

III. Manufacturer Usage Description (MUD)

In a nutshell, MUD specifies the expected behaviour of an IoT device. With traditional devices like a laptop, the device itself is responsible for protecting its system. However, for IoT devices with limited hardware, this approach should allow translating the protection away from the device. To enable this with so-called MUD files, the device behavior in the network is specified. Everything which is not explicitly listed in this file does not pertain to the expected behaviour of the device and should not be permitted [5]. The structure of this file is thereby a YANG model, which is serialized using JSON [6] in the first MUD version. In this file, any allowed connection for a device should be specified with source and destination IP addresses and ports, which protocol is used, and which endpoint initializes the connection [5], [7]. With this information, firewalls and switches can be configured to only permit exactly these connections. An example of such a MUD file is shown in Figure 1. In this example, one access control list (*"acl"*) defines the communication to a cloud service with the domain name 'service.bms.example-com' [5]. Only communications on port 443 using TCP are allowed. Moreover, the communication must be initiated by the IoT device and not the other way around. In a complete MUD file there should be another *acl* to define the rule to the device.

IV. How can MUD help with IoT Security Threats?

An overview of IoT threats and their security mechanisms with and without MUD is given in Table I and discussed below.

A. Perception layer

There are two essential threat groups in this layer. The first group contains different attacks against, for instance, the RFID technology, while the second focuses on the attacks against the physical replacement or the changes to the physical device. The MUD system cannot help preventing such attacks because it is based on rules that are applied and enforced in routers and switches. Changes to the physical device are undetectable by a switch and some communications, *e.g.,* RFID traffic, does not pass through such a device.

B. Transportation layer

Because this layer is the most interesting one and the MUD architecture is aimed at this layer, we will describe the threats considering MUD in more detail.

1) Eavesdropping: The goal of the MUD concept is to apply specific communication rules and prevent all non-compliant communications of a device. Nevertheless, these rules do not consider eavesdropping because no restricted communication is assumed to be established. Therefore, the illicit collection of information and data can also be done within the MUD architecture. Most often, the collected data is used to find a device-specific vulnerability. However, exploiting any discovered vulnerabilities for this purpose can be prevented by using MUD. For example, suppose an attacker finds a way to compromise a device by sending a particular order of commands. This can be prevented as the network node blocks the communication between the attacker and the device.

2) Wormhole: The rules defined in the MUD file are based on domain names and IP addresses. The wormhole attack is against the RPL (Routing Protocol for Low-Power and Lossy Networks) protocol, which tries to find the best path to send data to a target. In this attack, the target and sender IP addresses have not changed as only the underlying path is different. Consequently, MUD-based architecture cannot provide a capability to prevent such an attack. However, an attacker needs to compromise at least two nodes and this preliminary step can be blocked by the MUD rules.

3) Man in the Middle (MitM): MitM is more a concept than a specific attack because it just defines that an attacker is in the communication between two nodes. Therefore, there are many different MitM attacks on various protocols. One of the most famous ones is the Address Resolution Protocol (ARP) poisoning approach, which works on the data link layer. In this attack, an attacker sends ARP responses by spoofing the IP of a target, which results in wrong information about the target MAC address. After this step, the source devices with the false address on the target will send every packet to the attacker, which can then redirect them to the legitimate target. The MUD architecture is not helpful in this attack because neither the sender's nor the target's IP address changed, so the MUD-defined rules do not block the communication. In contrast, the MUD approach would work in an architecture where the data link layer connectivity (any-to-any) is replaced with an IP routing (one-to-any). In such a network, packets are sent to a specific port on the switch, which relies on the table with MAC/IP relations. Then the MUD rules can be applied to these requests to prevent unexpected connections. Overall, regarding the MitM attack technique, there are many different layers or protocol-based attacks where only a few of them can be blocked using MUD.

4) Spoofing: Spoofing is one of the most dangerous techniques against MUD since it is a basic technique rather than an attack. Moreover, it is used in multiple other attacks such as ARP poisoning. While the MUD architecture tries to enforce rules based on IP addresses or domain names, this technique can circumvent these rules by pretending to be another device with permissions to request malicious communication.

5) DDoS: DDoS is one of the main targets of the MUD architecture and, therefore, can be prevented in most scenarios. A DDoS attack can go in both directions, either by targeting one or multiple IoT devices to make them unavailable, or by abusing multiple IoT devices and amplifying an attack towards another target. This other target can be inside or outside the network. DDoS attacks can be blocked by the device that enforces the rules of the MUD file. One exception is if the MUD rules are not strict enough or if the connection to the device, which launches the attack, is necessary. This is the case if the device must be accessible from outside the network or if a trusted communication partner, *e.g.,* an update server, has been compromised. This also works in the other direction of a DDoS attack. Since a device that must be reachable from the Internet does not have to initiate connections to the outside, outgoing connections from such a device can be blocked. This reduces the possible abuse of IoT devices to generate or amplify DDoS attacks. A more granular restriction can be made on the used protocols and ports in case the device needs to start a connection with other nodes. By blocking all unnecessary ports and protocols, MUD reduces the attack surface that would lead to an IoT exploit and a consequent DDoS attack. Another advantage in this scenario is the lower load on the rule-enforcing network node. While in the former scenario, all requests from perhaps a huge number of devices must be handled by this device, in the latter, that node only has to bear the requests from a much smaller group of IoT devices inside this subnetwork. Because many well-known attacks recently used the second way of attack where multiple IoT devices are abused to attack a server of a targeted company, MUD is a pretty strong prevention technique. This was elaborated by Shutijser [8] and Morgese [9] where it is shown that most such attacks can be blocked using MUD.

6) Sinkhole: Like the wormhole attack, this is based on the RPL protocol, so it cannot be mitigated with MUD. The only thing that MUD can achieve in this attack is to make it harder for an attacker to compromise a device, thereby making the attack infeasible. However, similar to the wormhole attack, this is not a direct countermeasure.

7) Sleep deprivation: If the requests are not based on IP addresses, they cannot be blocked by a router or a switch, so no MUD rule can be applied. Otherwise, sleep deprivation attacks can be prevented by strict MUD rules applied on switches or routers except for two cases. The first exception is the spoofing issue where an attacker pretends to be someone else who has permission to communicate with the device. This leads to the other exception: the set of devices legitimately configured to communicate with the target device. If one of such devices executes an attack due to a bug (*i.e.,* involuntarily) or by being compromised, the MUD architecture cannot prevent it.

8) Sybil: The Sybil attack belongs to the same category as the MitM attack because it is a general concept rather than a specific attack. For example, if an attack uses the RPL, then it cannot be restricted by the MUD rules [10]. But if the Sybil attack is based on the IP protocol, the MUD rules are applied to these fake identities and therefore can be blocked. One example is if a device generates multiple identities to start TCP connections with another IoT device. This case probably results in a denial of service because the other device cannot handle all the opened TCP handshakes. But if there are strict MUD rules in use, these connections can be efficiently refused.

C. Application layer

The application layer is not essentially relevant to threat prevention using MUD. Most of the threats in this layer do not require a connection to another device at all or just to a controller or DB where the connection is allowed. One exception is the malicious propagation of viruses/worms. Although this threat is mainly located at the application layer, it tries to spread through the network. This behaviour can be prevented in some cases. One case is if a worm needs a port that is not already used for benign connections and therefore is blocked. Another criterion is the devices on which the worm can spread. If only user devices are in focus, very likely prevention cannot be achieved at all because they are not strictly regulated through the MUD rules, since a user wants to be able to communicate with a wide variety of servers or addresses.

A result of a combination of these two criteria can be that the worm is just able to spread over specific devices and if the connection between these devices is blocked at IP or port level, the worm cannot do much damage. But in the worst-case scenario, if we have a worm that can communicate over the ports, which are already used and not limited to some devices in the network, MUD cannot provide any improvement.

D. Other relevant aspects and limitations

1) Merge rules: The MUD standard does not define precisely how MUD rules should be handled and applied. Therefore it is important to mention that MUD rules need to be merged instead of overwriting existing ones. For instance, the MUD controller should not allow communication between two devices as long as this is not allowed from both devices. Otherwise, a malicious device with a MUD file that allows another target device to communicate to itself can circumvent MUD protection.

2) Compromised vendors and MUD-allowed sources: If a device is compromised, all the allowed communications applied to these devices can be abused and the MUD architecture cannot prevent that. Moreover, if the vendor is compromised, the allowed communication to the update server can also be abused, and the MUD files can even be maliciously adapted.

3) Changing IP addresses: As mentioned in [8], the rules based on domain names can lead to connectivity loss if not considered especially in the deployment. This is because the IP address belonging to a domain name can change over time

TABLE I

IoT threats and applicable security mechanisms with and without MUD (Perception layer is omitted since MUD is less relevant.)

Layer	Threat	Security without MUD	Security with MUD
Transportation layer	Eavesdropping	Use secure encryption standards, use discovery protocols to check the open accessible information.	Cannot prevent attack but can prevent some ways the data can be abused.
	Wormhole	Use an RSSI with a sensor node to detect it or use a machine learning approach.	Cannot prevent the attack but can make it harder to compromise a device needed for the attack.
	MitM	Device authorization, encrypted communication with signing, use a modified routing protocol or a system especially to detect MitM.	In an architecture like the one-to-any, the MUD rules can be applied. But other MitM attacks, *e.g.,* in an any-to-any architecture, MUD will not work.
	Spoofing	Verify the sender of a message.	-
	DDoS	Machine learning to detect and prevent such an attack.	Can reduce the risk and damage of such an attack to a low level.
	Sinkhole	Harden the device against compromises, a specialized system against this attack where heavy computation is outsourced to cloud or an edge node.	-
	Sleep deprivation	Use a rate limit approach, only allow authorized devices or use a behaviour supervising technique	Attacks using an IP-based request can be blocked with the exception of compromised or faulty devices and a spoofed device.
	Sybil	Check identities in the DIS messages, compare them with a whitelist or use a behaviour-based anomaly detection.	Only attacks based on IP/domains can be blocked.
Appl. layer	Data Leak	Data Loss Prevention (DLP) software	-
	Malicious code injection	Script detection mechanism	-
	Phishing	Use authentication and authorization, user training.	-
	Malicious virus/worm	Virus detection mechanism, a good firewall configuration and a hardened system	The possible communication paths of a worm can be reduced, whether this is effective depends on the worm.

and, as a result, the MUD rule is applied to the wrong address. [8] also proposes a solution to this problem.

4) Authentication of a MUD file: Because of the missing authentication, there is also the possibility of a compromised device pointing to a wrong MUD file. The proposed enhanced MUD (eMUD) solution aims to address this threat [11].

5) MUD controller: The MUD controller has a trust relationship with the IoT devices and is also a single point of failure [11]. This makes the MUD controller an attractive target for an attacker.

6) Enforcement of MUD rules: The MUD standard does not define how the rules should be applied and enforced [11]. This should be defined to handle the rules in a standardized, secured, and trusted way.

7) Manufacturers' Bankruptcy: An important point to consider is the bankruptcy of a manufacturer, especially considering the MUD file location [12]. The MUD files should be accessible and stored on a server, which is further available, or a mechanism should be defined to revoke these MUD URLs.

8) Supervision of MUD rules: To prevent compromised vendors or MUD files and increase security, MUD rules should be checked before being applied [12]. Because of scenarios like home networks, where users do not have the knowledge to do this, and because MUD is an automated system, it would be relevant to have an automated tool that performs general checks and supervises MUD rules.

V. Conclusion and future work

In this study, we described some critical threats to IoT devices and relevant prevention techniques categorized in the three-layer architecture. Naturally, this is not an exhaustive list of threats that are relevant to MUD. Most attacks in the perception and the application layer can not be prevented using the MUD architecture. But in the transportation layer, the attack surface can be drastically reduced. Two crucial risks to the MUD architecture are the spoofing and the lack of authentication of the MUD file. Without handling these two threats, most of the prevention gained using the MUD architecture can be circumvented.

Since MUD is mainly checked in IPv4 environments the advantages should also be confirmed in IPv6 networks. Further the enforcement of the MUD rules should be standardized.

Acknowledgment

The research leading to these results received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 871808 (5G PPP project INSPIRE-5Gplus). The paper reflects only the authors' views. The Commission is not responsible for any use that may be made of the information it contains.

References

[1] M. A. Jabraeil Jamali, B. Bahrami *et al.*, *IoT architecture*. Cham: Springer International Publishing, 2020, pp. 9–31.

[2] S. A. Al-Qaseemi, H. A. Almulhim *et al.*, "IoT architecture challenges and issues: Lack of standardization," in *2016 Future Technologies Conference (FTC)*. IEEE, Dec. 2016, pp. 731–738.

[3] M. Frustaci, P. Pace *et al.*, "Evaluating critical security issues of the IoT world: Present and future challenges," *IEEE Internet of Things Journal*, vol. 5, pp. 2483–2495, Aug. 2018.

[4] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the Internet of Things," *2015 IEEE World Congress on Services*, pp. 21–28, Aug. 2015.

[5] E. Lear, R. Droms, and D. Romascanu, "Manufacturer usage description specification," Internet Requests for Comments, RFC Editor, RFC 8520, Mar. 2019.

[6] E. Bjorklund, M., "The YANG 1.1 data modeling language," Internet Requests for Comments, RFC Editor, RFC 7950, Aug. 2016.

[7] A. Hamza, D. Ranathunga *et al.*, "Clear as MUD: Generating, validating and applying IoT behavioral profiles," in *Proceedings of the 2018 Workshop on IoT Security and Privacy*, ser. IoT S&P '18. New York, NY, USA: ACM, Aug. 2018, p. 8–14.

[8] C. Schutijser, "Towards automated DDoS abuse protection using MUD device profiles," http://essay.utwente.nl/76207/, Aug. 2018.

[9] L. Morgese, "Stepping out of the MUD : contextual network threat information for IoT devices with manufacturer-provided behavioural profiles," http://essay.utwente.nl/89157/, Dec. 2021.

[10] C. Pu, "Sybil attack in RPL-based internet of things: Analysis and defenses," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4937–4949, Jun. 2020.

[11] S. M. Sajjad, M. Yousaf *et al.*, "eMUD: Enhanced manufacturer usage description for IoT botnets prevention on home WiFi routers," *IEEE Access*, vol. 8, pp. 164 200–164 213, Sep. 2020.

[12] M. Souppaya, D. Montgomery *et al.*, *Securing Small-Business and Home Internet of Things (IoT) Devices: Mitigating Network-Based Attacks Using Manufacturer Usage Description (MUD)*. Special Publication, National Institute of Standards and Technology, 2021.