# Throwing MUD into the FOG: Defending IoT and Fog by expanding MUD to Fog network

Vafa Andalibi
*Indiana University Bloomington*

DongInn Kim
*Indiana University Bloomington*

Jean Camp
*Indiana University Bloomington*

## Abstract

Manufacturer Usage Description (MUD) is a proposed IETF standard enabling local area networks (LAN) to automatically configure their access control when adding a new IoT device based on the recommendations provided for that device by the manufacturer. MUD has been proposed as an isolation-based defensive mechanism with a focus on devices in the home, where there is no dedicated network administrator. In this paper, we describe the efficacy of MUD for a generic IoT device under different threat scenarios in the context of the Fog. We propose a method to use rate limiting to prevent end devices from participating in denial of service attacks (DDoS), including against the Fog itself. We illustrate our assumptions by providing a possible real world example and describe the benefits for MUD in the Fog for various stakeholders.

## 1 Introduction

The accelerating trend of Internet of Things (IoT), creates networks over billions of devices and produces significant amounts of data. IoT has driven companies to divert parts of the data processing and computations nodes closer to the end devices. The advantages of this include but are not limited to minimizing latency, increasing scalability, load reductions for cloud and data centers, and offloading cloud traffic. Conversely, the diversity and complexity of fog and edge computing correspondingly great.

IoT devices monitor, process data, and trigger actions at the farthest edge of the network while Fog nodes do the exact same thing inside the Fog network between the cloud and IoT devices. The main difference is between the type of the actions: IoT devices interact with humans at one end and other devices and machines on the other end, whereas Fog nodes usually interact with machines and devices on both ends. Examples of actions by Fog nodes in response IoT sensors include broadcasting a message to other IoT devices, changing device configuration, or acting on the physical environment (e.g. opening a locked door).

There have been numerous studies that have tried to apply Fog computing on new applications, several of which are extremely sensitive and raise serious security and privacy concerns. The included domains are health care [8], transportation [4] and smart cities [2]. Even limited Distributed Denial of Service (DDoS) attacks on the Fog nodes supporting such applications may cause a limited service outage with serious consequences. Similarly, a compromised Fog node might allow an attacker to implement attacks on end users that are connected to the same node via traffic manipulating or denial of service. Note that defending Fog nodes is a current research challenge, even in the case of encrypted data at rest [7].

The main purpose of the Manufacturer Usage Description (MUD) standard is to reduce the attack surface on devices, scale network policies to large numbers of devices, and address vulnerabilities in systems that are no longer supported or where patching the system is infeasible [6]. By isolating the IoT devices in Local Area Networks (LAN), MUD can also defend IoT devices against other compromised devices. Several previous studies have focused on identifying and defending the Fog networks specifically against DDoS attacks [1, 3, 9, 10, 11]. To the knowledge of authors, MUD has not been considered as a defense in Fog nodes.

In this work we try to leverage MUD for that purpose. While the initial design of MUD focused on end users; its operational benefits can be leveraged to defend a manufacturer's infrastructure. Specifically, we focus on minimizing the damage of malicious Fog nodes and preventing DDoS attacks targeting either the user's network or the Fog nodes.

The rest of the paper is organized as follows. In Section 2 we describe the MUD workflow in a home and how MUD can isolate the IoT devices. In Section 3 we describe how the MUD workflow would be different in a Fog network. This is followed by a description of MUD protective mechanisms for both Fog networks and end-users. Finally, the conclusion and discussion are provided in Sections 4 and 5 respectively, including the questions we hope to engage in the workshop.

## 2 MUD Workflow in LAN

Understanding the proposed role of MUD in the Fog requires some understanding of MUD. MUD provides isolation for devices, implementing white lists when feasible for dedicated IoT devices, and is capable of implementing black lists as well. The MUD workflow is illustrated in Figure 1. There are six main components to a MUD implementation. First there is the **MUD-File**: A file created by the manufacturer that describes the device and its expected network behavior. Normally this is a YANG-based JSON file (RFC 7951) signed with a public key signature from the device manufacturer.[1] This file is hosted in the **MUD file server**. The location of the file is embedded as a uniform resource locator, the **MUD-URL**, that is used to retrieve the MUD file from manufacturer's server when a device is added to a home network. The **Authentication server** (AAA) is responsible for enforcing the traffic rules for a device. The Authentication server implements access control and thus must be between the IoT device and the open Internet. In current implementations are built on LAN routers.

The **MUD-Manager** is the core of a MUD instantiation. The MUD-Manager extracts the MUD-URL, retrieves the MUD file, and sends the resulting configuration to the AAA server. An example instantiation of a MUD-Manager is available to public in GitHub (https://github.com/CiscoDevNet/MUD-Manager).

The sixth and final component is the **Network Access Device** (NAD). The NAD is primarily responsible for routing, however, there is usually an internal Firewall component which is used by MUD-Manager via AAA server to control the traffic and enforce rules.

MUD workflow in the network begins with the IoT device transmitting a MUD-URL as in step 1 in Figure 1. The device will authenticate with a X.509 certificate; although DHCP and LLDP are also available depending on the implementation. The MUD-Manager will receive the MUD-URL via a router and AAA server and, in case of X.509 authentication, it will validate the signature.

After delivery and authentication of the MUD-URL the MUD Manager will retrieve the MUD-file from the manufacturer's file server (step 3). Then AAA server and subsequently NAD will be configured accordingly. NAD by default blocks outbound connections of all internal devices. Depending on the MUD configuration, it might block inbound connections as well. After receive of the MUD-File and NAD reconfiguration, only those connections that are explicitly included in the MUD-file will be allowed. In the aforementioned example, the MUD-Manager will restrict the communication of all smart devices based on the MUD-Files that it receives from each of those devices' manufacturers.

[1] As we write this there is an option for Dynamic Host Configuration Protocol (DHCP) local authentication instead of robust cryptography. We assume the cryptographic implementation in this work unless otherwise specified. Analysis of the DHCP or Link Layer Discovery Protocol (LLDP) options is beyond the scope of this work.

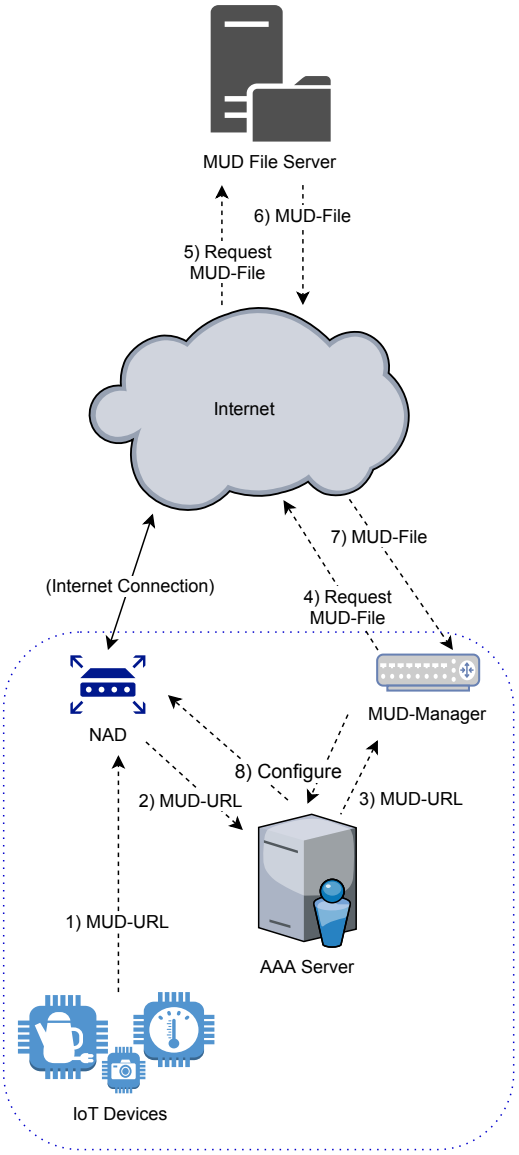


Figure 1: The workflow in a LAN with a MUD-compliance AAA: the blue dotted line indicates the LAN boundary.

## 3 MUD in the Fog

As illustrated, the defense that MUD provides to IoT in home networks or enterprise environments is based on isolation. In this section we describe how we can leverage MUD capabilities in the context of Fog nodes.

Note that in our model, we assume that the manufacturer deploys some implementation of Fog nodes and has control over them. Also, the Fog nodes are completely transparent from the customer's point of view. So the customer should not

notice any change in a device's behavior or usability resulting from the deployment of MUD in the Fog. (Note that we describe the adopters of Fog as *customers* to distinguish them from those interaction with IoT at home who are *users* in the current MUD standard.) We provide a solution by introducing the Fog MUD-Manager and a parameter used by this manager (the Peak request rate).

- **Fog MUD-Manager**: This is a standalone hardware or software component adjacent to the firewall, inside the Fog network. The Fog MUD-Manager is required for proper configuration of the firewall. It applies filtering for the IoT devices at the remote networks as these try to communicate with the Fog. The workflow, described in section 3.1 , slightly differs from the MUD-Manager at the end-user's network.

- **Peak request rate**: In any case it should contain 1) the maximum message size, 2) the maximum rate of transmission, and 3) the maximum value over some time period. The peak request rate could be added by the manufacturer in the MUD-File. For most of the low-end IoT devices that need to communicate with the manufacturer's services, the request rate can be calculated based on the characteristic of the device. For instance, getting 100 requests per second from a smart security camera would indeed be unusual since it is not physically likely for 100 moving objects to pass the smart security camera in 1 second.

Having described the standard and workflow of MUD in the previous section, in the next section we describe the proposed extension. We describe how the Fog MUD-Manager can prevent DDoS attacks using the peak request rate.

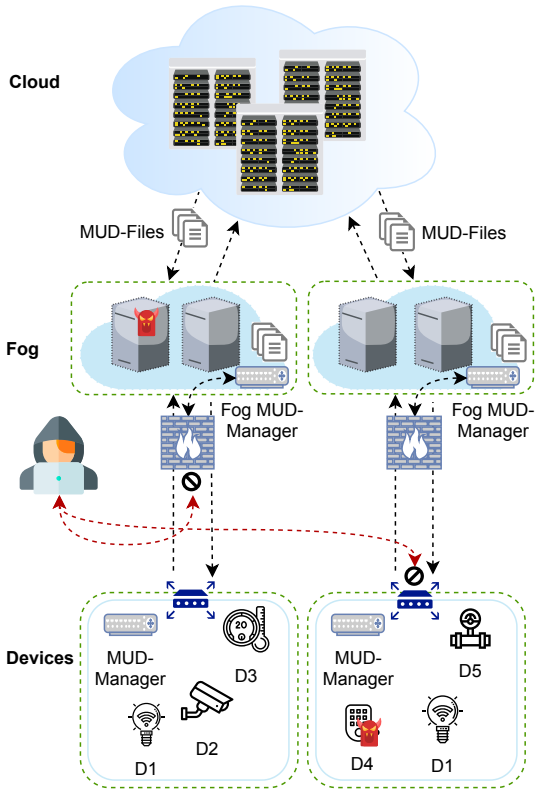


Figure 2: MUD's workflow in the Fog: the green dotted line indicates the MUD protecting the network. The red dotted line indicates the attacker trying to access the backdoor on the compromised IoT device or Fog node.

## 3.1 MUD workflow in the Fog network

The workflow of Fog MUD-Manager is slightly different from the MUD-Manager in LAN. Here we describe the main workflow differences.

**Accessing the MUD-File:** As mentioned previously, the IoT device directly advertises its MUD-URL to the MUD-Manager in the LAN. This request cannot reach beyond the local network since the device needs to authenticate using the local MUD-Manager before connecting to the outside network. Therefore, another mechanism is required for the Fog MUD-Manager to access the MUD-File. Thus, when the MUD-Manager in a LAN requests the MUD-File associated with one of the IoT devices in the network from the manufacturer's cloud, the manufacturer transfers a copy of the MUD-File to the Fog MUD-Manager in the Fog network communicating to the user's network.

The Fog MUD-Manager will use the same MUD-File to configure the firewall in the Fog network. It will discover the device type as a component of the MUD-File, and track the public IP address of the IoT device for configuring the firewall. Note that the IoT device might be behind a NAT or there might be more than one instance of a device. This is still manageable on the Fog network assuming that the manufacturer clearly recognizes the traffic of its devices. Regardless of how many instance of a a device exist in the user's network, the access controls would be identical and therefore possible to enforce, without the need to distinguish which instance of the device that traffic belongs to.

**Traffic Direction:** Normally, the MUD-Manager enforces access control based on the assumption that the outgoing traffic is generated by the devices inside the network. This is not the case in Fog network, as Fog traffic is not generated in the Fog network. Accordingly, the access control of the devices also differs from the LAN. For example, a normal MUD-Manager in a LAN will block all outgoing traffic from a smart bulb except for those that are going to manufacturer's domain. However a Fog MUD-Manager allows incoming smart bulb's traffic because it is necessary to provide the service to the local user. Moreover, all the communications indicated in the MUD-File that are within the manufacturer's Fog network will be ignored by the For MUD-Manager. Note that it is

indeed possible that a manufacturer communicates with a 3rd-party service either as a part of normal operation. In these cases, we assume that the manufacturer applies the rules separately on the firewall, yet we suggest that the manufacturer use a private channel for such communications.

In summary, the Fog MUD-Manager workflow is as follows: A request for a MUD-File is received by the manufacturer via a Fog network. The MUD-File is transferred to the user's network via the Fog network and the same MUD-File is parsed by Fog MUD-Manager. The IP address of the device as well as its type are recognized by the manufacturer's Fog network. Then the appropriate firewall rules for incoming connection from that device and outgoing connections to the same network are applied to the Fog network's Firewall. All other communications are blocked by default. This process is illustrated in Figure 2.

### 3.2 MUD's Benefit to the Manufacturer

In this section we will review the MUD's benefits from the perspective of the manufacturer and Fog server. The majority of DDoS mitigation solutions are based on filtering the problematic traffic which are not legitimate to the normal services if it is the network-oriented attack. The attackers can avoid such a mitigation by flooding the service with the distributed botnets which can make the legitimate requests.

The presented model, can help the manufacturer addressing this issue. In the scenario presented in section 2, suppose the smart security camera is infected by a botnet malware, flooding the Fog network with 100 requests per second. The Fog service would not be aware of the malware being source of the requests, and from the service's point of view they are legitimate requests coming from authenticated devices.

Using the newly introduced Peak request rate which is a metric suggested by the manufacturer, the MUD-Manager knows the reasonable range of requests an IoT device can send. Based on the device type, which is inferred from the MUD-File request, device address, and peak request rate. Both the MUD-Manager in the user's network as well as the Fog MUD-Manager would be able to configure the firewall properly, allowing it to drop the excessive requests. Naturally the effect of blocking the flooding requests on the user side is more effective since the traffic will not even reach the outside network it will offload the traffic on the link between user network and manufacturer network.

The ability to maintain Fog nodes enables support in the field without interacting with individual homes. The ability to observe aggregate traffic enables controls on both where traffic is sent and the volume. In addition to DDoS filtering the other capacities of Fog remain. For example, the Sony may be able to sell Aibo in Illinois if Fog enabled filtering out all facial recognition data; and if Cloud Pets could have implemented access control these would not have been removed from the market. It is also likely that the manufacturer or service provider would be more able to secure Fog nodes than the home user would be to secure their own network; thus avoiding a security debacle.

### 3.3 MUD's Benefit to the End-User

The DDoS mitigation discussed in the previous section naturally benefits both manufacturer and end-user. Other benefits accrue only to the user. For example, reconsider the scenario from section 2: the smart camera sends the photograph of a person who has entered the property to the Fog nodes for processing. The Fog node will respond with the most likely identification to associate with the image. Imagine the person is identified as the home owner by the Fog nodes. Then the Fog node will open a the lock via a query to the local network. If the Fog node is compromised, the attacker could admit anyone. However, such an attack most certainly requires communication with the command and control address of the attacker. Deploying MUD on the Fog side could defend the Fog network against such compromised Fog node as described below.

The Fog-MUD-Manager will use the original MUD-File to configure the firewall as shown in Figure 2. In the smart security camera example, the manufacturer knows that the device is allowed to communicate with smart lock. For the sake of simplicity we assume complete interoperability, e.g. both of the products are manufactured by the same company. If the smart lock is in the same network, the MUD-Manager and the firewall are configured to expect the smart lock from the same network to communicate with the Fog nodes. Therefore, both the Fog-MUD-Manager and a MUD-Manager would allow the smart security camera's traffic to pass. Thus having a MUD between the Fog node and the network is needed to ensure isolation.

With dual MUD-Manager configuration both the customer and manufacturer are protected from a variety of attacks that need communication with adversary. Other potentially malicious risk that would be mitigated includes variations of backdoor, keylogger, spyware, and reconnaissance malware to name instances where a command and control architecture is needed by the attacker. For example, in case of the smart security camera, outgoing traffic is only allowed when these are departing for other IoT end-devices which are isolated with MUD. Even if the attacker takes control of the other end-devices, it would not be feasible to communicate to other servers since they are only allowed to send the traffic to manufacturer's Fog network when both sides of communication are isolated.

## 4 Conclusions

In this paper, we describe both the potential of MUD if integrated with Fog as well as the possible extension for an edge defense that enables the extension of the protection model of

MUD to Fog nodes. In addition to the potential protection of Fog nodes via ACL, we also propose an addition to the MUD-File information to mitigate the threat of subverted IoT Fog-enabled devices from participating in DDoS attack. Using the presented model both the end-user and manufacturer could benefit from implementation of MUD in Fog and End-User's network.

We propose a solution for DDoS attacks on Fog nodes using MUD which is applicable even when a device has no meaningful ACL. For example, a smart television may expect to be able to access any IP space in which content would be hosted; however, the Fog-MUD proposal here would mitigate the harm from an individual device.

We propose to implement this by specifying a new vector to be included in the MUD which can include peak request rate and peak request volume. Such variables are part of the design of a produce and could be provided by the manufacturers based on their knowledge of the devices. Providing an estimated peak request rate for low-end IoT devices, e.g. a smart bulb, would be trivial. However even in cases of less constrained devices, such as an Smart TV, estimating such metric is likely best done by the manufacturer. While for a general use device the capacity would depend on the services desired by the customer, MUD leverages the fact that IoT and SCADA devices are designed for specific purposes, and not general computing devices.

Threat modeling of the MUD and the proposed MUD-Fog architecture is an extensive task which is ongoing alongside with implementation of this project. For example, when an the attacker is also performing a man in the middle attack on one of the routers, it would still be possible to send out traffic or communicate with the backdoor using a covert channel. Also, the attacker would still be able to manipulate the processing result on the compromised Fog node. For instance, in the smart security camera scenario, suppose the real output of a set of data arriving from the camera would be a confirmation of a face recognition process. The attacker would then be able to manipulate the result of the algorithm by intercepting the program execution, aka process hooking. The presented defense also does not prevent any malware with payload that causes system corruption on local triggers; for example, a logic bomb.

We have previously tested the MUD implementation using X.509 constraints on a local network. Currently we are working on expanding that implementation to support DHCP in a secure way using two factor authentication and device activation logging. We will then move towards testing our implementation on Fog Mud-Manager and build a proof of the concept MUD implementation on a compromised node acting as a Fog node.

## 5 Discussion

A core assumption in the proposed architecture is that that the manufacturer has full control over Fog nodes and that these reside close to the edge as shown in Figure 2. This is not an unlikely expectation for any small manufacturer (or home user) who contracts to their internet service provider for network operations support. Also we assume that it is feasible for the consumer to install a stand-alone device representing Fog MUD-Manager, and/or that a mirroring manager is available in the Fog network.

One discussion point is the existence an estimate of traffic that is high enough not to disrupt normal communications 99.9999% of the time but low enough to provide meaningful mitigation of DDoS attacks. Is such a traffic rating possible? Limits on capacity have proven difficult in other domains, for example in spam [5]. Should functionality be added to the Fog MUD-Manager so that it can log the request rate of the devices and find out the maximum over some period of time? One consideration of this approach is how this might impinge user privacy.

One of the difficulty of implementation of our models would be in the case where a manufacturer supports IoT devices of other companies. In that case there is the classic challenges of interoperability. These include both trust across the domain of control and communication between the LAN and the edge network for transferring the MUD file from the other company to the edge network of another company. Alternatively, companies can directly retrieve each others' MUD-File, removing the user from the communication cycle.

In contrast, one of the advantages of the MUD architecture is, if adopted, it provides a mechanism for isolating deployed IoT devices without requiring access to the device or the code. Certainly the question remains of who pays; however, a Fog/MUD configuration makes this much more manageable. Another advantage to this approach is that all the mechanism are common approaches some with mature code bases is a strength of this approach, given the complexity of the challenges of the IoT.

The reliability of our proposed architecture significantly depends on the fact that the MUD-File and MUD-Manager are well protected. Taking control of the MUD-Managers, or modifying the MUD-Files are both equivalent to controlling the firewall rules.

## Acknowledgments

This research was supported by Cisco Research Award funding. This material is based upon work supported by the National Science Foundation under Grant CNS 1814518 and CNS 1565375. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF. We thank our colleague Eliot Lear from Cisco Systems whose

insight and expertise enhanced our work; mistakes are entirely our own.

## References

[1] Ketan Bhardwaj, Joaquin Chung Miranda, and Ada Gavrilovska. Towards iot-ddos prevention using edge computing. In {*USENIX*} *Workshop on Hot Topics in Edge Computing (HotEdge 18)*, 2018.

[2] Ning Chen and Yu Chen. Smart city surveillance at the network edge in the era of iot: opportunities and challenges. In *Smart Cities*, pages 153–176. Springer, 2018.

[3] Clinton Dsouza, Gail-Joon Ahn, and Marthony Taguinod. Policy-driven security management for fog computing: Preliminary framework and a case study. In *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014)*, pages 16–23. IEEE, 2014.

[4] M Muzakkir Hussain, Mohammad Saad Alam, and MM Sufyan Beg. Fog computing model for evolving smart transportation applications. *Fog and Edge Computing: Principles and Paradigms*, 2019.

[5] Ben Laurie and Richard Clayton. Proof-of-work proves not to work; version 0.2. In *Workshop on Economics and Information, Security*, 2004.

[6] Eliot Lear, Ralph Droms, and Dan Romascanu. Manufacturer Usage Description Specification. RFC 8520, March 2019.

[7] Mithun Mukherjee, Rakesh Matam, Lei Shu, Leandros Maglaras, Mohamed Amine Ferrag, Nikumani Choudhury, and Vikas Kumar. Security and privacy in fog computing: Challenges. *IEEE Access*, 5:19293–19304, 2017.

[8] Ammar Awad Mutlag, Mohd Khanapi Abd Ghani, N Arunkumar, Mazin Abed Mohamed, and Othman Mohd. Enabling technologies for fog computing in healthcare iot systems. *Future Generation Computer Systems*, 90:62–78, 2019.

[9] Mert Özçelik, Niaz Chalabianloo, and Gürkan Gür. Software-defined edge defense against iot-based ddos. In *2017 IEEE International Conference on Computer and Information Technology (CIT)*, pages 308–313. IEEE, 2017.

[10] Amandeep Singh Sohal, Rajinder Sandhu, Sandeep K Sood, and Victor Chang. A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments. *Computers & Security*, 74:340–354, 2018.

[11] Qiao Yan, Wenyao Huang, Xupeng Luo, Qingxiang Gong, and F Richard Yu. A multi-level ddos mitigation framework for the industrial internet of things. *IEEE Communications Magazine*, 56(2):30–36, 2018.