Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 20, 2017

E. Lear
R. Droms
Cisco Systems
D. Romascanu
Avaya
August 19, 2016

Manufacturer Usage Description Specification
draft-ietf-opsawg-mud-00

Abstract

This memo specifies the necessary components to implement manufacturer usage descriptions (MUD). This includes two YANG modules, IPv4 and IPv6 DHCP options, an LLDP TLV, a URL suffix specification, an X.509 certificate extension and a means to sign and verify the descriptions.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 20, 2017.

Table of Contents

1. Introduction

The Internet has largely been constructed on general purpose computers; those devices that may be used for a purpose that is specified by those who buy the device. [RFC1984] presumed that an end device would be most capable of protecting itself. This made sense when the typical device was a workstation or a mainframe, and it continues to make sense for general purpose computing devices today, including laptops, smart phones, and tablets.

[RFC7452] discusses design patterns for, and poses questions about, smart objects. Let us then posit a group of objects that are specifically not general purpose computers. These devices therefore have a purpose to their use. By definition, therefore, all other purposes are NOT intended. The combination of these two statements can be restated as a manufacturer usage description (MUD) that can be applied at various points within a network. Although this memo may seem to stress access requirements, usage intent also consists of quality of service needs a device may have.

We use the notion of "manufacturer" loosely in this context, to simply mean the entity or organization that will state how a device is intended to be used. In the context of a lightbulb, this might indeed be the lightbulb manufacturer. In the context of a smarter device that has a built in Linux stack, it might be integrator of that device. The key points are that the device itself is expected to serve a limited purpose, and that there may exist an organization in the supply chain of that device that will take responsibility for informing the network about that purpose.

The converse statement holds that general computing systems will benefit very little from MUD, as their manufacturers cannot envision a specific communication pattern to describe.

The intent of MUD is to therefore solve for the following problems:

- Substantially reduce the threat surface on a device entering a network to those communications intended by the manufacturer.

- Provide for a means to scale network policies to the ever-increasing number types of devices in the network.

- Provide a means to address at least some vulnerabilities in a way that is faster than it might take to update systems. This will be particularly true for systems that are no longer supported by their manufacturer.

- Keep the cost of implementation of such a system to the bare minimum.

No matter how good a MUD-enabled network is, it will never replace the need for manufacturers to patch vulnerabilities. It may, however, provide network administrators with some additional protection when those vulnerabilities exist.

1.1. A Simple Example

A light bulb is intended to light a room. It may be remotely controlled through the network; and it may make use of a rendezvous service of some form that an app on smart phone accesses. What we can say about that light bulb, then, is that all other network access is unwanted. It will not contact a news service, nor speak to the refrigerator, and it has no need of a printer or other devices. It has no Facebook friends. Therefore, an access list applied to it that states that it will only connect to the single rendezvous service will not impede the light bulb in performing its function, while at the same time allowing the network to provide both it and other devices an additional layer of protection.

1.2. Determining Intended Use

The notion of intended use is in itself not new. Network administrators apply access lists every day to allow for only such use. This notion of white listing was well described by Chapman and Zwicky in [FW95]. Programmatically profiling systems have existed for years as well. These systems make use of heuristics that take at least some time to assert what a system is.

A system could just as easily tell the network what sort of protection it requires without going into what sort of system it is. This would, in effect, be the converse of [RFC7488]. In seeking a general purpose solution, however, we assume that a device has so few capabilities that it will implement the least necessary capabilities to function properly. This is a basic economic constraint. Unless the network would refuse access to such a device, its developers would have no reason to implement such an approach. To date, such an assertion has held true.

1.3. Finding A Policy: The MUD URL

Our work begins, therefore, with the device emitting a Universal Resource Locator (URL) [RFC3986]. This URL may serves both to classify the device type and to provide a means to locate a policy file.

In this memo three means are defined to emit the MUD URL. One is a DHCP option[RFC2131],[RFC3315] that the DHCP client uses to inform the DHCP server. The DHCP server may take further actions, such as retrieve the URL or otherwise pass it along to network management system or controller. The other method defined is an X.509 constraint. The IEEE has developed [IEEE8021AR] that provides a certificate-based approach to communicate device characteristics, which itself relies on [RFC5280]. The MUD URL extension is non-critical, as required by IEEE 802.1AR. Finally, an LLDP frame is defined.

1.4. Types of Policies

When the MUD URL is resolved, the MUD controller retrieves a file that describes what sort of communications a device is designed to have. The manufacturer may specify either specific hosts for cloud based services or certain classes for access within an operational network. An example of a class might be "devices of a specified manufacturer type", where the manufacturer type itself is indicated simply by the authority of the MUD URL. Another example might to allow or disallow local access. Just like other policies, these may be combined. For example:

```
Allow access to host controller.example.com with QoS AF11
Allow access to devices of the same manufacturer
Allow access to and from controllers who need to speak COAP
Allow access to local DNS/DHCP
Deny all other access
```

To add a bit more depth that should not be a stretch of anyone's imagination, one could also make use of port-based access lists. Thus a printer might have a description that states:

```
Allow access for port IPP or port LPD
Allow local access for port HTTP
Deny all other access
```

In this way anyone can print to the printer, but local access would be required for the management interface.

The files that are retrieved are intended to be closely aligned to existing network architectures so that they are easy to deploy. We make use of YANG [RFC6020] because of the time and effort spent to develop accurate and adequate models for use by network devices. JSON is used as a serialization for compactness and readability, relative to XML.

The YANG modules specified here are extensions of [I-D.ietf-netmod-acl-model]. The extensions to this model allow for a manufacturer to express classes of systems that a manufacturer would find necessary for the proper function of the device. Two modules are specified. The first module specifies a means for domain names to be used in ACLs so that devices that have their controllers in the cloud may be appropriately authorized with domain names, where the mapping of those names to addresses may rapidly change.

The second module abstracts away IP addresses into certain classes that are instantiated into actual IP addresses through local processing. Through these classes, manufacturers can specify how the device is designed to communicate, so that network elements can be configured by local systems that have local topological knowledge. That is, the deployment populates the classes that the manufacturer specifies.

Because manufacturers do not know who will be using their devices, it is important for functionality referenced in usage descriptions to be relatively ubiquitous, and therefore, mature. Therefore, only a a limited subset YANG-based configuration of is permitted in a MUD file.

1.5. Terminology

MUD: manufacturer usage description.

MUD file: a file containing YANG-based JSON that describes a recommended behavior.

MUD file server: a web server that hosts a MUD file.

MUD controller: the system that requests and receives the MUD file from the MUD server. After it has processed a MUD file it may direct changes to relevant network elements.

MUD URL: a URL that can be used by the MUD controller to receive the MUD file.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.6. The Manufacturer Usage Description Architecture

With these components laid out we now have the basis for an archicture. This leads us to ASCII art.

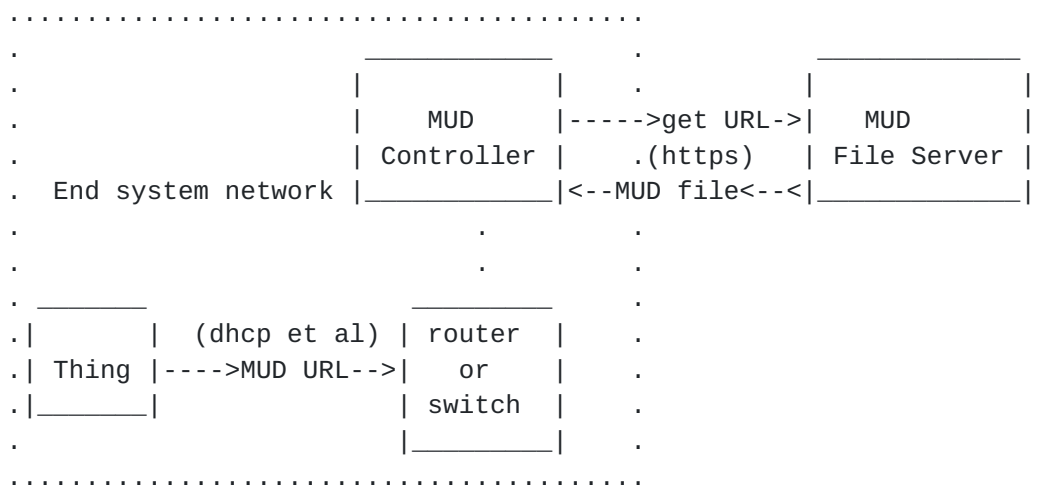


Figure 1: MUD Architecture

In the above diagram, the switch or router collects MUD URLs and forwards them to the network management system for processing. This happens in different ways, depending on how the URI is communicated. For instance, in the case of DHCP, the DHCP server might receive the URI and then process it. In the case of IEEE 802.1X, the switch would tunnel the URI to the authentication server, who would then process it.

The information returned by the web site is valid for the duration of the device's connection, or as specified in the description. Thus if the device is mobile, when it moves on, any configuration in the switch is removed. Similarly, from time to time the description may be refreshed, based on new capabilities or communication patterns or vulnerabilities.

The web site is typically run by or on behalf of the manufacturer. Its domain name is that of the authority found in the MUD URL. For legacy cases where Things cannot emit a URL, if the switch is able to determine the appropriate URI, it may proxy it, the trivial cases being a map between some registered device or port and a URL.

2. The MUD Model and Semantic Meaning

A MUD file consists of JSON based on a YANG model. For purposes of MUD, the elements that can be modified are access lists as augmented by this model. The MUD file is limited to the serialization of a small number of YANG schema, including the models specified in the following documents:

o [I-D.ietf-netmod-acl-model]

o [RFC6991]

Publishers of MUD files MUST NOT include other elements except as described in Section 13, and MUST only contain information relevant to the device being described. Devices parsing MUD files MUST cease processing if they find other elements.

This module is structured into three parts. The first container holds information that is relevant to retrieval and validity of the MUD file itself. The second container augments the access list to indicate direction the ACL is to be applied. The final container augments the matching container of the ACL model to add several elements that are relevant to the MUD URL, or other otherwise abstracted for use within a local environment.

```
  module: ietf-mud
    +--rw meta-information
        +--rw last-update?         yang:date-and-time
        +--rw previous-mud-file?   yang:uri
        +--rw cache-validity?      uint32
        +--rw masa-server?         inet:uri
        +--rw is-supported?        boolean
  augment /acl:access-lists/acl:acl:
     +--rw packet-direction?   direction
  augment /acl:access-lists/acl:acl
          /acl:access-list-entries/acl:ace/acl:matches:
     +--rw manufacturer?          inet:host
     +--rw same-manufacturer?     empty
     +--rw model?                 string
     +--rw local-networks?        empty
     +--rw controller?            inet:uri
     +--rw direction-initiated?   direction
```

3. Element Definitions

The following elements are defined.

3.1. last-update

This is a date-and-time value of the last time the MUD file was updated. This is akin to a version number. Its form is taken from [RFC6991] which, for those keeping score, turn was taken from Section 5.6 of [RFC3339], which was taken from [ISO.8601.1988].

3.2. previous-mud-file

This is a URL that should point to the previous MUD URL for auditing purposes. Because it should not be necessary to resign a MUD file when a new one is released, the archival location of a current MUD file should be identified prior to its release. Note the signature file MUST also be available. For example, if previous-mud-file is set to "https://example.com/.mud/v1/xxx", the corresponding signature would be found at "https://example.com/.mud/v1/xxx.p7s".

3.3. cache-validity

This uint32 is the period of time in hours that a network management station MUST wait since its last retrieval before checking for an update. It is RECOMMENDED that this value be no less than 24 and no more than 1440 for any device that is supported.

3.4. masa-server

This optional element refers to the URL that should be used to resolve the location any MASA service, as specified in [I-D.ietf-anima-bootstrapping-keyinfra].

3.5. is-supported

This boolean is an indication from the manufacturer to the network administrator as to whether or not the device is supported. In this context a device is said to be supported if the manufacturer might issue an update to the device or if the manufacturer might update the MUD file.

3.6. packet-direction

[I-D.ietf-netmod-acl-model] describes access-lists but does not attempt to indicate where they are applied as that is handled elsewhere in a configuration. However, in this case, a MUD file must be explicit in describing the communcation pattern of a device, and that includes indicating what is to be permitted or denied in either direction of communication. This element takes a single value of either "to-device" or "from-device", based on a typedef "direction".

3.7. manufacturer

This element consists of a hostname that would be matched against the authority section of another device's MUD URL.

3.8. same-manufacturer

This is an equivalent for when the manufacturer element is used to indicate the authority that is found in another device's MUD URL matches that of the authority found in this device's MUD URL.

3.9. model

This string matches the one and only segment following the authority section of the MUD URL. It refers to a model that is unique within the context of the authority. It may also include product version information. Thus how this field is constructed is entirely a local matter for the manufacturer.

3.10. local-networks

This null-valued element expands to include local networks. Its default expansion is that packets must not traverse toward a default route that is received from the router.

3.11. controller

This URI specifies a value that a controller will register with the network management station. The element then is expanded to the set of hosts that are so registered.

In addition, some meta information is defined in order to determine when a usage description should be refreshed.

3.12. direction-initiated

When applied this matches packets when the flow was initiated in the corresponding direction. [RFC6092] provides guidance for IPv6 guidance best practices. While that document is scoped specifically to IPv6, its contents are applicable for IPv4 as well. When this flag is set, and the system has no reason to believe a flow has been initiated it MUST drop the packet. This match SHOULD be applied with specific transport parameters, such as protocol.

4. Processing of the MUD file

To keep things relatively simple in addition to whatever definitions exist, we also apply two additional default behaviors:

- Anything not explicitly permitted is denied.
- Local DNS, DHCP, and NTP are, by default, permitted to and from the device.

5. What does a MUD URL look like?

To begin with, MUD takes full advantage of both the https: scheme and the use of .well-known. HTTPS is important in this case because a man in the middle attack could otherwise harm the operation of a class of devices. .well-known is used because we wish to add additional structure to the URL. And so the URL appears as follows:

```
mud-url   = "https://" authority  "/.well-known/mud/" mud-rev
            "/" model ( "?" extras )
            ; authority is from RFC3986
mud-rev   = "v1"
model     = segment  ; from RFC3986
extras    = query    ; from RFC3986
```

mud-rev signifies the version of the manufacturer usage description file. This memo specifies "v1" of that file. Later versions may permit additional schemas or modify the format.

"model" represents a device model as the manufacturer wishes to represent it. It could be a brand name or something more specific. It also may provide a means to indicate what version the product is. Specifically if it has been updated in the field, this is the place where evidence of that update would appear. The field should be changed when the intended communication patterns of a device change. While from a controller standpoint, only comparison and matching operations are safe, it is envisioned that updates will require some administrative review. Processing of this URL occurs as specified in [RFC2818] and [RFC3986].

6. The MUD YANG Model

```
<CODE BEGINS>file "ietf-mud@2016-07-20.yang";

module ietf-mud {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud";
```

```
  prefix "ietf-mud";

  import ietf-access-control-list {
    prefix "acl";
  }

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-inet-types {
    prefix "inet";
  }

  organization
    "IETF OPSAWG (Ops Area) Working Group";

  contact
       "WG Web: http://tools.ietf.org/wg/opsawg/
       WG List: opsawg@ietf.org
       WG Chair: Warren Kumari
       warren@kumari.net
       WG Chair: Zhou Tianran
       zhoutianran@huawei.com
       Editor: Eliot Lear
       lear@cisco.com
       Editor: Ralph Droms
       rdroms@cisco.com
    ";

  description
    "This YANG module defines a component that augments the
     IETF description of an access list.  This specific module
     focuses on additional filters that include local, model,
     and same-manufacturer.

    Copyright (c) 2016 IETF Trust and the persons identified as
    the document authors.  All rights reserved.
    Redistribution and use in source and binary forms, with or
    without modification, is permitted pursuant to, and subject
    to the license terms contained in, the Simplified BSD
    License set forth in Section 4.c of the IETF Trust's Legal
    Provisions Relating to IETF Documents
    (http://trustee.ietf.org/license-info).
    This version of this YANG module is part of RFC XXXX; see
    the RFC itself for full legal notices.";

  revision "2016-07-20"  {
```

```
    description "Base version of MUD extensions to ACL model";
    reference "RFC XXXX: Manufacturer Usage Description Specification";
  }

  typedef direction {
     type enumeration {
         enum to-device {
           description "packets or flows destined to the target device";
         }
         enum from-device {
           description "packets or flows destined from
                        the target device";
           }
         }
     description "Which way are we talking about?";
  }

  container meta-information {

    description "Information about when support end(ed), and
                 when to refresh";

    leaf last-update {
      type yang:date-and-time;
      description "This is intended to be the time and date that
                   the MUD file was generated.";
    }

    leaf previous-mud-file {
       type inet:uri;
       description "Use to find the previous MUD file location
                    for auditing purposes.";
    }

    leaf cache-validity {
      type uint32;
      description "The information retrieved from the MUD server is
                  valid for these many hours, after which it should
                  be refreshed.";
    }

    leaf masa-server {
      type inet:uri;
      description "The URI of the MASA server that network
      elements should forward requests to for this device.";
    }

    leaf is-supported {
```

```
      type boolean;
      description "The element is currently supported
                   by the manufacturer.";
    }
  }

  augment "/acl:access-lists/acl:acl"   {
    description "add inbound or outbound.  Normally access lists
                 are applied in an inbound or outbound direction
                 separately from their definition.  This is not
                 possible with MUD.";
    leaf packet-direction
    {
      type direction;
      description "inbound or outbound ACL.";
    }
  }


  augment "/acl:access-lists/acl:acl/" +
     "acl:access-list-entries/acl:ace/" +
     "acl:matches" {
    description "adding abstractions to avoid need of IP addresses";

    leaf manufacturer {
      type inet:host;
      description "authority component of the manufacturer URI";
    }

    leaf same-manufacturer {
      type empty;
      description "expand to ACEs for each device
                   with the same origin";
    }

    leaf model {
        type string;
        description "specific model (including version) for a
                     given manufacturer";
    }

    leaf local-networks {
      type empty;
      description "this string is used to indicate networks
                   considered local in a given environment.";
    }
    leaf controller {
      type inet:uri;
```

```
      description "expands to one or more controllers for a
                   given service that is codified by inet:uri.";
    }
    leaf direction-initiated {
      type direction;
      description "which direction a flow was initiated";
    }
  }
}
```

<CODE ENDS>

7. The Domain Name Extension to the ACL Model

This module specifies an extension to IETF-ACL model such that domain names may be referenced by augmenting the "matches" element. Different implementations may deploy differing methods to maintain the mapping between IP address and domain name, if indeed any are needed. However, the intent is that resources that are referred to using a name should be authorized (or not) within an access list.

The structure of the change is as follows:

```
augment
/acl:access-lists/acl:acl/acl:access-list-entries
   /acl:ace/acl:matches/acl:ace-type/acl:ace-ip:
   +--rw src-dnsname?        inet:host
   +--rw dst-dnsname?        inet:host
```

The choice of this particular point in the access-list model is based on the assumption that we are in some way referring to IP-related resources, as that is what the DNS returns. A domain name in our context is defined in [RFC6991].

The following elements are defined.

7.1. source-dnsname

The argument corresponds to a domain name of a source as specified by inet:host. Depending on how the model is used, it may or may not be resolved, as required by the implementation and circumstances.

7.2. destination-dnsname

The argument corresponds to a domain name of a destination as specified by inet:host. Depending on how the model is used, it may or may not be resolved, as required by the implementation and circumstances.

7.3. The ietf-acldns Model

```
 <CODE BEGINS>file "ietf-acldns@2007-07020.yang";

 module ietf-acldns {
   yang-version 1.1;
   namespace "urn:ietf:params:xml:ns:yang:ietf-acldns";
   prefix "ietf-acldns";

   import ietf-access-control-list {
     prefix "acl";
   }

   import ietf-inet-types {
     prefix "inet";
   }

   organization
     "IETF OPSAWG (Ops Area) Working Group";

   contact
        "WG Web: http://tools.ietf.org/wg/opsawg/
        WG List: opsawg@ietf.org
        WG Chair: Warren Kumari
        warren@kumari.net
        WG Chair: Zhou Tianran
        zhoutianran@huawei.com
        Editor: Eliot Lear
        lear@cisco.com
        Editor: Ralph Droms
        rdroms@cisco.com
     ";


   description
     "This YANG module defines a component that augments the
      IETF description of an access list to allow dns names
      as matching criteria.";

   revision "2016-07-20" {
     description "Base version of dnsname extension of ACL model";
     reference "RFC XXXX: Manufacturer Usage Description Specification";
   }

   augment "/acl:access-lists/acl:acl/" +
      "acl:access-list-entries/acl:ace/" +
      "acl:matches/acl:ace-type/acl:ace-ip" {
     description "adding domain names to matching";
```

```
    leaf src-dnsname {
      type inet:host;
      description "domain name to be matched against";
    }
    leaf dst-dnsname {
      type inet:host;
      description "domain name to be matched against";
    }
  }
}

<CODE ENDS>
```

8. MUD File Example

This example contains two access lists that are intended to provide outbound access to a cloud service on TCP port 443.

```
{
   "ietf-mud:support-information": {
      "last-update": "2016-05-18T20:00:50Z",
      "cache-validity": 1440
      },
   "ietf-access-control-list:access-lists":  {
     "acl": [ {
      "acl-name": "inbound-stuff",
      "acl-type" : "ipv4-acl",
      "ietf-mud:direction" : "to-device",
      "access-list-entries": {
         "ace": [
            {
               "rule-name": "access-cloud",
               "matches": {
                 "ietf-acldns:src-dnsname":
                     "lighting-system.example.com",
                  "protocol" : 6,
                  "source-port-range" : {
                     "lower-port" : 443,
                     "upper-port" : 443
                     }
                  },
               "actions" : {
                 "permit" : [null]
                 }
              }
            ]
          }
         },
```

```
           {
         "acl-name": "outbound-stuff",
         "acl-type" : "ipv4-acl",
         "ietf-mud:direction" : "from-device",
         "access-list-entries": {
            "ace": [
               {
                  "rule-name": "access-cloud",
                  "matches": {
                     "ietf-acldns:dst-dnsname":
                          "lighting-system.example.com",
                      "protocol" : 6,
                      "destination-port-range" : {
                        "lower-port" : 443,
                        "upper-port" : 443
                       }
                   },
                  "actions" : {
                      "permit" : [null]
                    }
                }
               ]
             }
           }
        }
    }
```

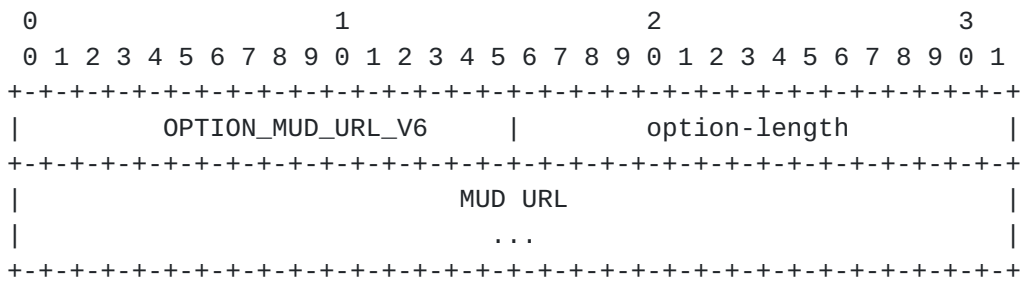
9. The MUD URL DHCP Option

The IPv4 MUD URL client option has the following format:

```
  +------+-----+-----------------------------
  | code | len |  MUD URL
  +------+-----+-----------------------------
```

Code OPTION_MUD_URL_V4 (TBD) is assigned by IANA. len is a single octet that indicates the length of the URL in octets. MUD URL is a URL. The length of a MUD URL does not exceed 255 bytes.

The IPv6 MUD URL client option has the following format:



OPTION_MUD_URL_V6 (TBD; assigned by IANA).

option-length contains the length of the URL in octets. The length MUST NOT exceed 255 octets.

The intent of this option is to provide both a new device classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview of the network system as managed by the network administrator to decide what to do with this information. The key function of this option is simply to identify the type of device to the network in a structured way such that the policy can be easily found with existing toolsets.

9.1. Client Behavior

A DHCP client MAY emit either a DHCPv4 or DHCPv6 option or both. These options singletons, as specified in [RFC7227]. Because clients are intended to have at most one MUD URL associated with them, they may emit at most one MUD URL option via DHCPv4 and one MUD URL option via DHCPv6. In the case where both v4 and v6 DHCP options are emitted, the same URL MUST be used.

Clients SHOULD log or otherwise report improper acknowledgments from servers, but they MUST NOT modify their MUD URL configuration based on a server's response. The server's response is only an acknowledgment that the server has processed the option, and promises no specific network behavior to the client. In particular, it may not be possible for the server to retrieve the file associated with the MUD URL, or the local network administration may not wish to use the usage description. Neither of these situations should be considered in any way exceptional.

9.2. Server Behavior

A DHCP server may ignore these options or take action based on receipt of these options. For purposes of debugging, if a server successfully parses the option and the URL, it MUST return the option

with the same URL as an acknowledgment. Even in this circumstance, no specific network behavior is guaranteed. When a server consumes this option, it will either forward the URL and relevant client information to a network management system (such as the giaddr), or it will retrieve the usage description by resolving the URL.

DHCP servers may implement MUD functionality themselves or they may pass along appropriate information to a network management system or controller. A DHCP server that does process the MUD URL MUST adhere to the process specified in [RFC2818] and [RFC5280] to validate the TLS certificate of the web server hosting the MUD file. Those servers will retrieve the file, process it, create and install the necessary configuration on the relevant network element. Servers SHOULD monitor the gateway for state changes on a given interface. A DHCP server that does not provide MUD functionality and has forwarded a MUD URL to a network management system MUST notify the network management of any corresponding change to the DHCP state of the client (such as expiration or explicit release of a network address lease).

9.3. Relay Requirements

There are no additional requirements for relays.

10. The Manufacturer Usage Description (MUD) URL X.509 Extension

[RFC7299] provides a procedure and means to specify extensions to X.509 certificates. The MUD URL is a non-critical Certificate extension that points to an on-line Manufacturer Usage Description concerning the certificate subject. This extension contains a single Uniform Resource Identifier (URI). Internationalized Resource Identifiers must be represented as URI's in the way described in RFC 5280, section 7.4.

The choice of id-pe is based on guidance found in Section 4.2.2 of [RFC5280]:

> These extensions may be used to direct applications to on-line information about the issuer or the subject.

The MUD URL is precisely that: online information about the particular subject.

The new extension is identified as follows:

- The MUD URL extension id-pe-mud-url OBJECT IDENTIFER ::= { id-pe TBD }

The extension returns a single value:

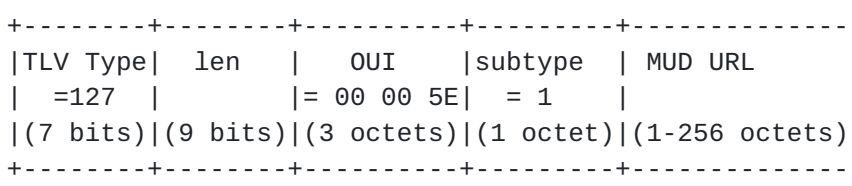
mudURLSyntax ::= IA5String - for use with MUD architecture.

The semantics of the URI are defined Section 5.

11. The Manufacturer Usage Description LLDP extension

The IEEE802.1AB Link Layer Discovery Protocol (LLDP) [IEEE8021AB] is a one hop vendor-neutral link layer protocols used by end hosts network devices for advertising their identity, capabilities, and neighbors on an IEEE 802 local area network. Its Type-Length-Value (TLV) design allows for 'vendor-specific' extensions to be defined. IANA has a registered IEEE 802 organizationally unique identifier (OUI) defined as documented in [RFC7042]. The MUD LLDP extension uses a subtype defined in this document to carry the MUD URL.

The LLDP vendor specific frame has the following format:



where:

- TLV Type = 127 indicates a vendor-specific TLV
- len - indicates the TLV string length
- OUI = 00 00 5E is the organizationally unique identifier of IANA
- subtype = 1 (to be assigned by IANA for the MUD URL)
- MUD URL - the length MUST NOT exceed 256 octets (consistent with the DHCP option defined in Section 9)

The intent of this extension is to provide both a new device classifier to the network as well as some recommended configuration to the routers that implement policy. However, it is entirely the purview of the network system as managed by the network administrator to decide what to do with this information. The key function of this extension is simply to identify the type of device to the network in a structured way such that the policy can be easily found with existing toolsets.

Hosts, routers, or other network devices that implement this option are intended to have at most one MUD URL associated with them, so they may transmit at most one MUD URL value.

Hosts, routers, or other network devices that implement this option may ignore these options or take action based on receipt of these options. For example they may fill in information in the respective extensions of the LLDP Management Information Base (LLDP MIB). LLDP operates in a one-way direction. LLDPDUs are not exchanged as information requests by one device and response sent by another device. The other devices do not acknowledge LLDP information received from a device. No specific network behavior is guaranteed. When a device consumes this extension, it may either forward the URL and relevant remote device information to a network management system, or it will retrieve the usage description by resolving the URL.

12. Creating and Processing of Signed MUD Files

Because MUD files contain information that may be used to configure network access lists, they are sensitive. To insure that they have not been tampered with, it is important that they be signed. We make use of DER-encoded Cryptographic Message Syntax (CMS) [RFC5652] for this purpose.

12.1. Creating a MUD file signature

A MUD file MUST be signed using CMS as an opaque binary object. In order to make successful verification more likely, intermediate certificates SHOULD be included. If the device that is being described supports IEEE 802.1AR, its manufacturer certificate and the certificate in the MUD file MUST share a common trust anchor in order to insure that manufacturer of the device is also the provider of the MUD file. The signature is stored at the same location as the MUD URL but with the suffix of ".p7s". Signatures are transferred using content-type "Application/pkcs7-signature".

For example:

```
% openssl cms -sign -signer mancertfile -inkey mankey \
              -in mudfile -binary -outform DER - \
              -certfile intermediatecert -out mudfile.p7s
```

Note: A MUD file may need to be resigned if the signature expires.

12.2. Verifying a MUD file signature

Prior to retrieving a MUD file the MUD controller SHOULD retrieve the MUD signature file using the MUD URL with a suffix of ".p7s". For example, if the MUD URL is "https://example.com/.well-known/v1/modela", the MUD signature URL will be "https://example.com/.well-known/v1/modela.p7s".

Upon retrieving a MUD file, a MUD controller MUST validate the signature of the file before continuing with further processing. A MUD controller SHOULD produce an error and it MUST cease all processing of that file if the signature cannot be validated. If the MUD controller has received the MUD URL via IEEE 802.1AR containing an IDevID (a manufacturer certificate), it MUST further confirm that the manufacturer certificate and that of the MUD file share a common trust anchor.

For Example:

```
% openssl cms -verify -in mudfile.p7s -inform DER -content mudfile
```

Note the additional step of verifying the common trust root.

13. Extensibility

One of our design goals is to see that MUD files are able to be understood by as broad a cross-section of systems as is possible. Coupled with the fact that we have also chosen to leverage existing mechanisms, we are left with no ability to negotiate extensions and a limited desire for those extensions in any event. A such, a two-tier extensibility framework is employed, as follows:

1. At a coarse grain, a protocol version is included in a MUD URL. This memo specifies MUD version 1. Any and all changes are entertained when this version is bumped. Transition approaches between versions would be a matter for discussion in future versions.

2. At a finer grain, only extensions that would not incur additional risk to the device are permitted. Specifically, augmenting of the meta-information container is permitted with the understanding that such additions may be ignored. In addition, augmentation of the ACL model is permitted so long as it remains safe for a given ACE to be ignored by the MUD Controller or the network elements it configures. Most specifically, is is not permitted to include as an augmentation that modifies "deny" behavior without bumping the version. Furthermore, implementations that are not able to parse a component of the ACE

array MUST ignore the entire array entry (e.g., not the entire array) and MAY ignore the entire MUD file.

14. Security Considerations

Based on the means a URL is procured, a device may be able to lie about what it is, thus gaining additional network access. There are several means to limit risk in this case. The most obvious is to only believe devices that make use of certificate-based authentication such as IEEE 802.1AR certificates. When those certificates are not present, devices claiming to be of a certain manufacturer SHOULD NOT be included in that manufacturer grouping without additional validation of some form. This will occur when it makes use of primitives such as "manufacturer" for the purpose of accessing devices of a particular type.

Network management systems SHOULD NOT deploy a usage description for a device with the same MAC address that has indicated a change of authority without some additional validation (such as review of the class). New devices that present some form of unauthenticated MUD URL SHOULD be validated by some external means when they would be otherwise be given increased network access.

It may be possible for a rogue manufacturer to inappropriately exercise the MUD file parser, in order to exploit a vulnerability. There are three recommended approaches to address this threat. The first is to validate the signature of the MUD file. The second is to have a system do a primary scan of the file to ensure that it is both parseable and believable at some level. MUD files will likely be relatively small, to start with. The number of ACEs used by any given device should be relatively small as well. Second, it may be useful to limit retrieval of MUD URLs to only those sites that are known to have decent web reputations.

Use of a URL necessitates the use of domain names. If a domain name changes ownership, the new owner of that domain may be able to provide MUD files that MUD controllers would consider valid. There are a few approaches that can mitigate this attack. First, MUD file servers SHOULD cache certificates used by the MUD file server. When a new certificate is retrieved for whatever reason, the MUD controller should check to see if ownership of the domain has changed. A fair programmatic approximation of this is when the name servers for the domain have changed. If the actual MUD file has changed, the controller MAY check the WHOIS database to see if registration ownership of a domain has changed. If a change has occured, or if for some reason it is not possible to determine whether ownership has changed, further review may be warranted. Note, this remediation does not take into account the case of a

device that was produced long ago and only recently fielded, or the case where a new MUD controller has been installed.

15. IANA Considerations

15.1. DHCPv4 and DHCPv6 Options

IANA is requested to allocated the DHCPv4 and v6 options as specified in Section 9.

15.2. PKIX Extensions

The IANA is requested to assign a value for id-pe-mud-uri in the "SMI Security for PKIX Certificate Extension" Registry. Its use is specified in Section 10.

15.3. Well Known URI Suffix

The IANA is requested to register the URL suffix of "mud" as follows:

o URI Suffix: "mud" o Specification documents: this document o Related information: n/a

15.4. MIME Media-type Registration for MUD files

The following media-type is defined for transfer of MUD file:

- Type name: application
- Subtype name: mud+json
- Required parameters: n/a
- Optional parameters: n/a
- Encoding considerations: 8bit; application/mud+json values are represented as a JSON object; UTF-8 encoding SHOULD be employed.
- Security considerations: See {{secon}} of this document.
- Interoperability considerations: n/a
- Published specification: this document
- Applications that use this media type: MUD controllers as specified by this document.
- Fragment identifier considerations: n/a
- Additional information:

    Magic number(s): n/a
    File extension(s): n/a
    Macintosh file type code(s): n/a

- Person & email address to contact for further information: Eliot Lear <lear@cisco.com>, Ralph Droms <rdroms@cisco.com>
- Intended usage: COMMON
- Restrictions on usage: none

- Author: Eliot Lear <lear@cisco.com>, Ralph Droms <rdroms@cisco.com>
- Change controller: IESG
- Provisional registration? (standards tree only): No.

15.5. LLDP IANA TLV Subtype Registry

IANA is requested to create a new registry for IANA Link Layer Discovery Protocol (LLDP) TLV subtype values. The recommended policy for this registry is Expert Review. The maximum number of entries in the registry is 256.

IANA is required to populate the initial registry with the value:

LLDP subtype value = 1

Description = the Manufacturer Usage Description (MUD) Uniform Resource Locator (URL)

Reference = < this document >

16. Acknowledgments

The authors would like to thank Einar Nilsen-Nygaard, Bernie Volz, Tom Gindin, Brian Weis, Sandeep Kumar, Thorsten Dahm, John Bashinski, Steve Rich, Jim Bieda, and Dan Wing for their valuable advice and reviews. The remaining errors in this work are entirely the responsibility of the author.

17. References

17.1. Normative References

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., and S. Bjarnason, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-03 (work in progress), June 2016.

[I-D.ietf-netmod-acl-model]
Bogdanovic, D., Koushik, K., Huang, L., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-08 (work in progress), July 2016.

[IEEE8021AB]
Institute for Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks--Station and Media Access Control Connectivity Discovery", n.d..

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <http://www.rfc-editor.org/info/rfc2131>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC3315] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, DOI 10.17487/RFC3315, July 2003, <http://www.rfc-editor.org/info/rfc3315>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <http://www.rfc-editor.org/info/rfc5652>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6092] Woodyatt, J., Ed., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, DOI 10.17487/RFC6092, January 2011, <http://www.rfc-editor.org/info/rfc6092>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <http://www.rfc-editor.org/info/rfc6991>.

[RFC7227] Hankins, D., Mrugalski, T., Siodelski, M., Jiang, S., and S. Krishnan, "Guidelines for Creating New DHCPv6 Options", BCP 187, RFC 7227, DOI 10.17487/RFC7227, May 2014, <http://www.rfc-editor.org/info/rfc7227>.

[RFC7299] Housley, R., "Object Identifier Registry for the PKIX Working Group", RFC 7299, DOI 10.17487/RFC7299, July 2014, <http://www.rfc-editor.org/info/rfc7299>.

17.2. Informative References

[FW95] Chapman, D. and E. Zwicky, "Building Internet Firewalls", January 1995.

[IEEE8021AR] Institute for Electrical and Electronics Engineers, "Secure Device Identity", 1998.

[ISO.8601.1988]
International Organization for Standardization, "Data elements and interchange formats - Information interchange - Representation of dates and times", ISO Standard 8601, June 1988.

[RFC1984] IAB and IESG, "IAB and IESG Statement on Cryptographic Technology and the Internet", BCP 200, RFC 1984, DOI 10.17487/RFC1984, August 1996, <http://www.rfc-editor.org/info/rfc1984>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <http://www.rfc-editor.org/info/rfc3339>.

[RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <http://www.rfc-editor.org/info/rfc7042>.

[RFC7452] Tschofenig, H., Arkko, J., Thaler, D., and D. McPherson, "Architectural Considerations in Smart Object Networking", RFC 7452, DOI 10.17487/RFC7452, March 2015, <http://www.rfc-editor.org/info/rfc7452>.

[RFC7488] Boucadair, M., Penno, R., Wing, D., Patil, P., and T. Reddy, "Port Control Protocol (PCP) Server Selection", RFC 7488, DOI 10.17487/RFC7488, March 2015, <http://www.rfc-editor.org/info/rfc7488>.

Appendix A. Changes from Earlier Versions

RFC Editor to remove this section prior to publication.

Draft -03 to -04: * add LLDP extension. * add Dan Romascanu as co-author.

Draft -02 to -03: * incorporate domain name model. * discuss extensibility. * leave placeholder for LLDP TLV.

Draft -01 to -02:

- XML->JSON
- Remove device versioning information from URL
- Add PKIX and DHCP options

- Add Content-type information
- Clean up IANA considerations to match registration templates
- Ralph Droms carried over as author from DHCP option.
- Signing information
- Expanded Security Considerations
- Add directionality for both packets and flows.
- add previous-mud-file

Draft -00 to -01:

- Add MASA server element

Authors' Addresses

Eliot Lear
Cisco Systems
Richtistrasse 7
Wallisellen CH-8304
Switzerland

Phone: +41 44 878 9200
Email: lear@cisco.com

Ralph Droms
Cisco Systems
55 Cambridge Parkway
Cambridge 1057
United States

Phone: +1 617 621 1904
Email: rdroms@cisco.com

Dan Romascanu
Avaya
26, HaRokhmim Str., Bldg. D
Holon 5885849
Israel

Phone: +972-3-645-8414
Email: dromasca@avaya.com