

Workgroup: Network Working Group

Internet-Draft:

draft-ietf-opsawg-sbom-access-04

Published: 5 January 2022

Intended Status: Standards Track

Expires: 9 July 2022

Authors: E. Lear                      S. Rose

         Cisco Systems        NIST

## **Discovering and Retrieving Software Transparency and Vulnerability Information**

### **Abstract**

To improve cybersecurity posture, automation is necessary to locate what software is running on a device, whether that software has known vulnerabilities, and what, if any recommendations suppliers may have. This memo specifies a model to provide access to this information. It may optionally be discovered through manufacturer usage 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 <https://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 9 July 2022.

### **Copyright Notice**

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in

Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

- [1. Introduction](#)
  - [1.1. Cases Not Addressed](#)
  - [1.2. How This Information Is Retrieved](#)
  - [1.3. Formats](#)
  - [1.4. Discussion points](#)
- [2. The well-known transparency endpoint set](#)
- [3. The mud-transparency extension model extension](#)
- [4. The mud-sbom augmentation to the MUD YANG model](#)
- [5. Examples](#)
  - [5.1. Without ACLS](#)
  - [5.2. SBOM Located on the Device](#)
  - [5.3. Further contact required.](#)
  - [5.4. With ACLS](#)
- [6. Security Considerations](#)
- [7. IANA Considerations](#)
  - [7.1. MUD Extension](#)
  - [7.2. YANG Registration](#)
  - [7.3. Well-Known Prefix](#)
- [8. Acknowledgments](#)
- [9. References](#)
  - [9.1. Normative References](#)
  - [9.2. Informative References](#)
- [Appendix A. Changes from Earlier Versions](#)
- [Authors' Addresses](#)

## 1. Introduction

A number of activities have been working to improve visibility to what software is running on a system, and what vulnerabilities that software may have[[E02021](#)].

Put simply, we seek to answer two classes of questions **at scale**:

\*Is this system vulnerable to a particular vulnerability?

\*Which devices in a particular environment contain vulnerabilities that require some action?

This memo doesn't specify the format of this information, but rather only how to locate and retrieve these objects.

Software bills of materials (SBOMs) are descriptions of what software, including versioning and dependencies, a device contains. There are different SBOM formats such as Software Package Data Exchange [[SPDX](#)] or CycloneDX[[CycloneDX12](#)].

System vulnerabilities may similarly be described using several data formats, including the aforementioned CycloneDX, Common Vulnerability Reporting Framework [CVRF], the Common Security Advisory Format [CSAF]. This information is typically used to report to customers the state of a system.

These two classes of information can be used in concert. For instance, a network management tool may discover that a system makes use of a particular software component that has a known vulnerability, and a vulnerability report may be used to indicate what if any versions of software correct that vulnerability, or whether the system exercises the vulnerable code at all.

Both classes of information elements are optional under the model specified in this memo. One can provide only an SBOM, only vulnerability information, or both an SBOM and vulnerability information.

Note that SBOM formats may also carry other information, the most common being any licensing terms. Because this specification is neutral regarding content, it is left for format developers such as the Linux Foundation, OASIS, and ISO to decide what attributes they will support.

This memo does not specify how vulnerability information may be retrieved directly from the endpoint. That's because vulnerability information changes occur at different rates to software updates. However, some SBOM formats may also contain vulnerability information.

SBOMs and vulnerability information are advertised and retrieved through the use of a YANG augmentation of the Manufacturer User Description (MUD) model [RFC8520]. Note that the schema creates a grouping that can also be used independently of MUD. Moreover, other MUD features, such as access controls, needn't be present.

The mechanisms specified in this document are meant to satisfy several use cases:

- \*A network-layer management system retrieving information from an IoT device as part of its ongoing lifecycle. Such devices may or may not have query interfaces available.

- \*An application-layer management system retrieving vulnerability or SBOM information in order to evaluate the posture of an application server of some form. These application servers may themselves be containers or hypervisors. Discovery of the topology of a server is beyond the scope of this memo.

To satisfy these two key use cases, objects may be found in one of three ways:

- \*on devices themselves

- \*on a web site (e.g., via URI)

- \*through some form of out-of-band contact with the supplier.

In the first case, devices will have interfaces that permit direct retrieval. Examples of these interfaces might be an HTTP, COAP or [OpenC2] endpoint for retrieval. There may also be private interfaces as well.

In the second case, when a device does not have an appropriate retrieval interface, but one is directly available from the manufacturer, a URI to that information MUST be discovered.

In the third case, a supplier may wish to make an SBOM or vulnerability information available under certain circumstances, and may need to individually evaluate requests. The result of that evaluation might be the SBOM or vulnerability itself or a restricted URL or no access.

To enable application-layer discovery, this memo defines a well-known URI [RFC8615]. Management or orchestration tools can query this well-known URI to retrieve a system's SBOM or vulnerability information. Further queries may be necessary based on the content and structure of the response.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 1.1. Cases Not Addressed

[ This section to be removed prior to publication ]

A separate use case may be addressed in future versions of this document:

- \*Related to the application layer, software as a service may involve multiple backend systems, depending on many factors. One example might be a large cloud-based service that offers spreadsheets, email, and document authoring and management. Depending on what service is being used, a different set of backend services may in turn be invoking different software that should be listed.

The reason why this use case isn't addressed here is that it may be better addressed inline within HTML. Further discussion is required.

## **1.2. How This Information Is Retrieved**

For devices that can emit a URL or can establish a well-known URI, the mechanism may be highly automated. For devices that have a URL in either their documentation or within a QR code on a box, the mechanism is semi-automated (someone has to scan the QR code or enter the URL).

Note that vulnerability and SBOM information is likely to change at different rates. The MUD semantics provide a way for manufacturers to control how often tooling should check for those changes through the cache-validity node.

## **1.3. Formats**

There are multiple ways to express both SBOMs and vulnerability information. When these are retrieved either directly from the device or directly from a web server, tools will need to observe the content-type header to determine precisely which format is being transmitted. Because IoT devices in particular have limited capabilities, use of a specific Accept: header in HTTP or the Accept Option in CoAP is NOT RECOMMENDED. Instead, backend tooling is encouraged to support all known formats, and SHOULD silently discard SBOM information sent with a media type that is not understood.

Some formats may support both vulnerability and software inventory information. When both vulnerability and software inventory information is available from the same location, both sbom and vuln nodes MUST indicate that. Network management systems retrieving this information MUST take note that the identical resource is being retrieved rather than retrieving it twice.

## **1.4. Discussion points**

The following is discussion to be removed at time of RFC publication.

- \*Is the model structured correctly?

- \*Are there other retrieval mechanisms that need to be specified?

- \*Do we need to be more specific in how to authenticate and retrieve SBOMs?

- \*What are the implications if the MUD URL is an extension in a certificate (e.g. an IDevID cert)?

## 2. The well-known transparency endpoint set

Two well known endpoints are defined:

`*"/.well-known/sbom"` retrieves an SBOM.

`*"/.well-known/openc2"` is the HTTPS binding to OpenC2.

As discussed previously, the precise format of a response is based on the Content-type provided.

## 3. The mud-transparency extension model extension

We now formally define this extension. This is done in two parts. First, the extension name "transparency" is listed in the "extensions" array of the MUD file. N.B., this schema extension is intended to be used wherever it might be appropriate (e.g., not just MUD).

Second, the "mud" container is augmented with a list of SBOM sources.

This is done as follows:

module: ietf-mud-transparency

```
augment /mud:mud:
  +--rw transparency
    +--rw (sbom-retrieval-method)?
      | +--:(cloud)
      | | +--rw sboms* [version-info]
      | |   +--rw version-info    string
      | |   +--rw sbom-url?       inet:uri
      | +--:(local-well-known)
      | | +--rw sbom-local-well-known? enumeration
      | +--:(sbom-contact-info)
      |   +--rw sbom-contact-uri    inet:uri
  +--rw (vuln-retrieval-method)?
    +--:(cloud)
    | +--rw vuln-url?               inet:uri
    +--:(vuln-contact-info)
      +--rw contact-uri             inet:uri
```

#### 4. The mud-sbom augmentation to the MUD YANG model

```

<CODE BEGINS>file "ietf-mud-transparency@2021-10-22.yang"
module ietf-mud-transparency {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-mud-transparency";
  prefix mudtx;

  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991";
  }
  import ietf-mud {
    prefix mud;
    reference "RFC 8520";
  }

  organization
    "IETF OPSAWG (Ops Area) Working Group";
  contact
    "WG Web: http://tools.ietf.org/wg/opsawg/
    WG List: opsawg@ietf.org

    Editor: Eliot Lear lear@cisco.com
    Editor: Scott Rose scott.rose@nist.gov";
  description
    "This YANG module augments the ietf-mud model to provide for
    reporting of SBOMs and vulnerability information.

    Copyright (c) 2020 IETF Trust and the persons identified as
    authors of the code. 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
    (https://trustee.ietf.org/license-info).

    This version of this YANG module is part of RFC XXXX
    (https://www.rfc-editor.org/info/rfcXXXX);
    see the RFC itself for full legal notices.

    The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
    NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
    'MAY', and 'OPTIONAL' in this document are to be interpreted as
    described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
    they appear in all capitals, as shown here. ";

  revision 2021-07-06 {
    description
      "Initial proposed standard.";
  }
}

```



```

reference
  "RFC XXXX: Discovering and Retrieving Software Transparency
    and Vulnerability Information";
}

grouping transparency-extension {
  description
    "Transparency extension grouping";
  container transparency {
    description
      "container of methods to get an SBOM.";
    choice sbom-retrieval-method {
      description
        "How to find SBOM information";
      case cloud {
        list sboms {
          key "version-info";
          description
            "A list of SBOMs tied to different s/w
              or h/w versions.";
          leaf version-info {
            type string;
            description
              "The version to which this SBOM refers.";
          }
          leaf sbom-url {
            type inet:uri;
            description
              "A statically located URI.";
          }
        }
      }
    }
  }
  case local-well-known {
    leaf sbom-local-well-known {
      type enumeration {
        enum http {
          description
            "Use http (insecure) to retrieve
              SBOM information.";
        }
        enum https {
          description
            "Use https (secure) to retrieve SBOM information.";
        }
        enum coap {
          description
            "Use COAP (insecure) to retrieve SBOM";
        }
        enum coaps {

```

```

        description
            "Use COAPS (secure) to retrieve SBOM";
    }
    enum openc2 {
        description
            "Use OpenC2 endpoint.
            This is https://{host}/.well-known/openc2";
    }
}
description
    "Which communication protocol to choose.";
}
}
case sbom-contact-info {
    leaf sbom-contact-uri {
        type inet:uri;
        mandatory true;
        description
            "This MUST be either a tel, http, https, or
            mailto uri schema that customers can use to
            contact someone for SBOM information.";
    }
}
}
choice vuln-retrieval-method {
    description
        "How to find vulnerability information";
    case cloud {
        leaf vuln-url {
            type inet:uri;
            description
                "A statically located URL.";
        }
    }
}
case vuln-contact-info {
    leaf contact-uri {
        type inet:uri;
        mandatory true;
        description
            "This MUST be either a tel, http, https, or
            mailto uri schema that customers can use to
            contact someone for vulnerability information.";
    }
}
}
}
}
}
}

augment "/mud:mud" {

```

```
    description
      "Add extension for software transparency.";
    uses transparency-extension;
  }
}
```

<CODE ENDS>

## 5. Examples

In this example MUD file that uses a cloud service, the modelX presents a location of the SBOM in a URL. Note, the ACLs in a MUD file are NOT required, although they are a very good idea for IP-based devices.

### 5.1. Without ACLS

This first MUD file demonstrates how to get SBOM and vulnerability information without ACLs.

```

{
  "ietf-mud:mud": {
    "mud-version": 1,
    "extensions": [
      "ol",
      "transparency"
    ],
    "ol": {
      "owners": [
        "Copyright (c) Example, Inc. 2022. All Rights Reserved"
      ],
      "spdx-tag": "0BSD"
    },
    "mudtx:transparency": {
      "sbom-local-well-known": "https",
      "vuln-url": "https://iot.example.com/info/modelX/csaf.json"
    },
    "mud-url": "https://iot.example.com/modelX.json",
    "mud-signature": "https://iot.example.com/modelX.p7s",
    "last-update": "2022-01-05T13:29:12+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "retrieving vuln and SBOM info via a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot.example.com/doc/modelX",
    "model-name": "modelX"
  }
}

```

The second example demonstrates that just SBOM information is included.

```

{
  "ietf-mud:mud": {
    "mud-version": 1,
    "extensions": [
      "ol",
      "transparency"
    ],
    "ol": {
      "owners": [
        "Copyright (c) Example, Inc. 2022. All Rights Reserved"
      ],
      "spdx-tag": "0BSD"
    },
    "mudtx:transparency": {
      "sbom-local-well-known": "https"
    },
    "mud-url": "https://iot.example.com/modelX.json",
    "mud-signature": "https://iot.example.com/modelX.p7s",
    "last-update": "2022-01-05T13:29:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "retrieving vuln and SBOM info via a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot.example.com/doc/modelX",
    "model-name": "modelX"
  }
}

```

## 5.2. SBOM Located on the Device

In this example, the SBOM is retrieved from the device, while vulnerability information is available from the cloud. This is likely a common case, because vendors may learn of vulnerability information more frequently than they update software.

```

{
  "ietf-mud:mud": {
    "mud-version": 1,
    "extensions": [
      "transparency"
    ],
    "mudtx:transparency": {
      "sbom-local-well-known": "https",
      "vuln-url": "https://iot-device.example.com/info/modelX/csaf.json"
    },
    "mud-url": "https://iot-device.example.com/modelX.json",
    "mud-signature": "https://iot-device.example.com/modelX.p7s",
    "last-update": "2022-01-05T13:25:14+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "retrieving vuln and SBOM info via a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot-device.example.com/doc/modelX",
    "model-name": "modelX"
  }
}

```

### 5.3. Further contact required.

In this example, the network manager must take further steps to retrieve SBOM information. Vulnerability information is still available.

```

{
  "ietf-mud:mud": {
    "mud-version": 1,
    "extensions": [
      "transparency"
    ],
    "ietf-mud-transparency:transparency": {
      "contact-info": "https://iot-device.example.com/contact-info.html",
      "vuln-url": "https://iot-device.example.com/info/modelX/csaf.json"
    },
    "mud-url": "https://iot-device.example.com/modelX.json",
    "mud-signature": "https://iot-device.example.com/modelX.p7s",
    "last-update": "2021-07-09T06:16:42+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "retrieving vuln and SBOM info via a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot-device.example.com/doc/modelX",
    "model-name": "modelX"
  }
}

```

#### **5.4. With ACLS**

Finally, here is a complete example where the device provides SBOM and vulnerability information, as well as access-control information.

```

{
  "ietf-mud:mud": {
    "mud-version": 1,
    "extensions": [
      "ol",
      "transparency"
    ],
    "ol": {
      "owners": [
        "Copyright (c) Example, Inc. 2022. All Rights Reserved"
      ],
      "spdx-tag": "0BSD"
    },
    "mudtx:transparency": {
      "sbom-local-well-known": "https",
      "vuln-url": "https://iot.example.com/info/modelX/csaf.json"
    },
    "mud-url": "https://iot.example.com/modelX.json",
    "mud-signature": "https://iot.example.com/modelX.p7s",
    "last-update": "2022-01-05T13:30:31+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "retrieving vuln and SBOM info via a cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot.example.com/doc/modelX",
    "model-name": "modelX",
    "from-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-65443-v4fr"
          }
        ]
      }
    },
    "to-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-65443-v4to"
          }
        ]
      }
    }
  },
  "ietf-access-control-list:acls": {
    "acl": [
      {
        "name": "mud-65443-v4to",

```



```

    "type": "ipv4-acl-type",
    "aces": {
      "ace": [
        {
          "name": "cl0-todev",
          "matches": {
            "ipv4": {
              "ietf-acldns:src-dnsname": "iotserver.example.com"
            }
          },
          "actions": {
            "forwarding": "accept"
          }
        }
      ]
    }
  },
  {
    "name": "mud-65443-v4fr",
    "type": "ipv4-acl-type",
    "aces": {
      "ace": [
        {
          "name": "cl0-frdev",
          "matches": {
            "ipv4": {
              "ietf-acldns:dst-dnsname": "iotserver.example.com"
            }
          },
          "actions": {
            "forwarding": "accept"
          }
        }
      ]
    }
  }
]
}
}

```

At this point, the management system can attempt to retrieve the SBOM, and determine which format is in use through the content-type header on the response to a GET request, independently repeat the process for vulnerability information, and apply ACLs, as appropriate.

## **6. Security Considerations**

SBOMs provide an inventory of software. If software is available to an attacker, the attacker may well already be able to derive this very same software inventory. Manufacturers MAY restrict access to SBOM information using appropriate authorization semantics within HTTP. In particular, if a system attempts to retrieve an SBOM via HTTP and the client is not authorized, the server MUST produce an appropriate error, with instructions on how to register a particular client. One example may be to issue a certificate to the client for this purpose after a registration process has taken place. Another example would involve the use of OAUTH in combination with a federations of SBOM servers.

Another risk is a skew in the SBOM listing and the actual software inventory of a device/container. For example, a manufacturer may update the SBOM on its server, but an individual device has not been upgraded yet. This may result in an incorrect policy being applied to a device. A unique mapping of a device's software version and its SBOM can minimize this risk.

To further mitigate attacks against a device, manufacturers SHOULD recommend access controls through the normal MUD mechanism.

Vulnerability information is generally made available to such databases as NIST's National Vulnerability Database. It is possible that vendor may wish to release information early to some customers. We do not discuss here whether that is a good idea, but if it is employed, then appropriate access controls and authorization would be applied to the vulnerability resource.

## **7. IANA Considerations**

### **7.1. MUD Extension**

The IANA is requested to add "transparency" to the MUD extensions registry as follows:

Extension Name: transparency

Standard reference: This document

## 7.2. YANG Registration

The following YANG module should be registered in the "YANG Module Names" registry:

Name: ietf-mud

URN: urn:ietf:params:xml:ns:yang:ietf-mud-transparency

Prefix: mudtx

Registrant contact: The IESG

Reference: This memo

## 7.3. Well-Known Prefix

The following well known URIs are requested in accordance with [\[RFC8615\]](#):

URI suffix: "sbom"

Change controller: "IETF"

Specification document: This memo

Related information: See ISO/IEC 19970-2 and SPDX.org

URI suffix: "openc2"

Change controller: "IETF"

Specification document: This memo

Related information: OpenC2 Project

## 8. Acknowledgments

Thanks to Russ Housley, Dick Brooks, Tom Petch, Nicolas Comstedt, who provided review comments.

## 9. References

### 9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/

RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

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

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <<https://www.rfc-editor.org/info/rfc8520>>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <<https://www.rfc-editor.org/info/rfc8615>>.

## 9.2. Informative References

[CSAF] OASIS, "Common Security Advisory Format", July 2021, <<https://github.com/oasis-tcs/csaf>>.

[CVRF] Santos, O., Ed., "Common Vulnerability Reporting Framework (CVRF) Version 1.2", September 2017, <<https://docs.oasis-open.org/csaf/csaf-cvrf/v1.2/csaf-cvrf-v1.2.pdf>>.

[CycloneDX12] cyclonedx.org, "CycloneDX XML Reference v1.2", May 2020.

[E02021] Biden, J., "Executive Order 14028, Improving the Nations Cybersecurity", May 2021.

[OpenC2] Lemire, D., Ed., "Specification for Transfer of OpenC2 Messages via HTTPS Version 1.0", July 2019, <<https://docs.oasis-open.org/openc2/open-impl-https/v1.0/open-impl-https-v1.0.html>>.

[SPDX] The Linux Foundation, "SPDX Specification 2.1", 2016.

## Appendix A. Changes from Earlier Versions

Draft -04: \* Address review comments

Draft -02:

\*include vulnerability information

Draft -01:

\*some modest changes

Draft -00:

\*Initial revision

#### **Authors' Addresses**

Eliot Lear  
Cisco Systems  
Richtistrasse 7  
CH-8304 Wallisellen  
Switzerland

Phone: [+41 44 878 9200](tel:+41448789200)

Email: [lear@cisco.com](mailto:lear@cisco.com)

Scott Rose  
NIST  
100 Bureau Dr  
Gaithersburg MD, 20899  
United States of America

Phone: [+1 301-975-8439](tel:+13019758439)

Email: [scott.rose@nist.gov](mailto:scott.rose@nist.gov)