

DOTS
Internet-Draft
Intended status: Standards Track
Expires: September 9, 2019

K. Nishizuka
NTT Communications
M. Boucadair
Orange
T. Reddy
McAfee
T. Nagata
Lepidum
March 8, 2019

Controlling Filtering Rules Using DOTS Signal Channel
draft-nishizuka-dots-signal-control-filtering-05

Abstract

This document specifies an extension to the DOTS signal channel to control the filtering rules when an attack mitigation is active.

Particularly, this extension allows a DOTS client to activate or deactivate existing filtering rules during a DDoS attack. The characterization of these filtering rules is supposed to be conveyed by a DOTS client during peace time by means of DOTS data channel.

Editorial Note (To be removed by RFC Editor)

Please update these statements within the document with the RFC number to be assigned to this document:

- o "This version of this YANG module is part of RFC XXXX;"
- o "RFC XXXX: Controlling Filtering Rules Using DOTS Signal Channel";
- o reference: RFC XXXX
- o [RFCXXXX]

Please update these statements with the RFC number to be assigned to the following documents:

- o "RFC SSSS: Distributed Denial-of-Service Open Threat Signaling (DOTS) Signal Channel Specification" (used to be [[I-D.ietf-dots-signal-channel](#)])
- o "RFC DDDD: Distributed Denial-of-Service Open Threat Signaling (DOTS) Data Channel Specification" (used to be [[I-D.ietf-dots-data-channel](#)])

Please update the "revision" date of the YANG module.

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 September 9, 2019.

Copyright Notice

Copyright (c) 2019 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 Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
1.1.	The Problem	3
1.2.	The Solution	4
2.	Notational Conventions and Terminology	4
3.	Controlling Filtering Rules	4
3.1.	Binding the Data and Signal Channels	4
3.2.	DOTS Signal Channel Extension	5
3.2.1.	Filtering Control	5
3.2.2.	Sample Examples	7
3.2.2.1.	Conflict Handling	7
3.2.2.2.	Activate an Accept-List Filter	11
3.2.2.3.	Activate Rate-Limit or Drop Filters	12

3.2.3.	DOTS Signal Filtering Control Module	15
3.2.3.1.	Tree Structure	15
3.2.3.2.	YANG Module	16
4.	IANA Considerations	18
4.1.	DOTS Signal Channel CBOR Mappings Registry	19
4.2.	DOTS Signal Control Filtering YANG Module	19
5.	Security Considerations	20
6.	Acknowledgements	20
7.	References	20
7.1.	Normative References	20
7.2.	Informative References	21
	Authors' Addresses	21

1. Introduction

1.1. The Problem

The DOTS data channel protocol [[I-D.ietf-dots-data-channel](#)] is used for bulk data exchange between DOTS agents to improve the coordination of all the parties involved in the response to the DDoS attack. Filter management is one of its tasks which enables a DOTS client to retrieve DOTS server filtering capabilities and to manage filtering rules. Filtering rules are used for dropping or rate-limiting unwanted traffic, and permitting accept-listed traffic.

Unlike the DOTS signal channel, the DOTS data channel is not expected to deal with attack conditions. As such, an issue that might be encountered in some deployments is when filters installed by means of DOTS data channel protocol may not function as expected during DDoS attacks or exacerbate an ongoing DDoS attack. The DOTS data channel cannot be used then to change these filters, which may complicate DDoS mitigation operations [[Interop](#)].

A typical case is a DOTS client which configures during 'idle' time (i.e., no mitigation is active) filtering rules using DOTS data channel to permit traffic from accept-listed sources, but during a volumetric DDoS attack the DDoS mitigator identifies the source addresses/prefixes in the accept-listed filtering rules are attacking the target. For example, an attacker can spoof the IP addresses of accept-listed sources to generate attack traffic or the attacker can compromise the accept-listed sources and program them to launch DDoS attack.

[[I-D.ietf-dots-signal-channel](#)] is designed so that the DDoS server notifies the conflict to the DOTS client ('conflict-cause' set to 2 (Conflicts with an existing accept list)), but the DOTS client may not be able to withdraw the accept-list rules during the attack period due to the high-volume attack traffic saturating the inbound

link. In other words, the DOTS client cannot use the DOTS data channel to withdraw the accept-list filters when the DDoS attack is in progress. This assumes that this DOTS client is the owner of the filtering rule.

1.2. The Solution

This specification addresses the problems discussed in [Section 1.1](#) by adding the capability of managing filtering rules using the DOTS signal channel, which enables a DOTS client to request the activation or de-activation of filtering rules during a DDoS attack.

The DOTS signal channel protocol [[I-D.ietf-dots-signal-channel](#)] is designed to enable a DOTS client to contact a DOTS server for help even under severe network congestion conditions. Therefore, extending the DOTS signal channel protocol to manage the filtering rules during a attack will enhance the protection capability offered by DOTS protocols. Sample examples are provided in [Section 3.2.2](#).

Note: The experiment at the IETF103 hackathon [[Interop](#)] showed that even when the incoming link is saturated by DDoS attack traffic, the DOTS client can signal mitigation requests using the DOTS signal channel over the saturated link.

Conflicts that are induced by filters installed by other DOTS clients of the same domain are not discussed in this specification.

2. Notational Conventions and Terminology

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.

The reader should be familiar with the terms defined in [[I-D.ietf-dots-requirements](#)].

3. Controlling Filtering Rules

3.1. Binding the Data and Signal Channels

The filtering rules eventually managed using the DOTS signal channel must be created a priori by the same DOTS client using the DOTS data channel. Managing conflicts with filters installed by other DOTS clients of the same domain is out of scope.

As discussed in Section 4.4.1 of [[I-D.ietf-dots-signal-channel](#)], a DOTS client must use the same 'cuid' for both the signal and data channels. This requirement is meant to facilitate binding channels used by the same DOTS client.

The DOTS signal and data channels from a DOTS client may or may not use the same DOTS server. Nevertheless, the scope of the mitigation request, alias, and filtering rules are not restricted to the DOTS server but to the DOTS server domain. To that aim, DOTS servers within a domain are assumed to have a mechanism to coordinate the mitigation requests, aliases, and filtering rules to coordinate their decisions for better mitigation operation efficiency. The exact details about such mechanism is out of scope of this document.

A filtering rule controlled by the DOTS signal channel is identified by its Access Control List (ACL) name. Note that an ACL name unambiguously identifies an ACL bound to a DOTS client, but the same name may be used by distinct DOTS clients.

The activation or de-activation of an ACL by the signal channel overrides the 'activation-type' (defined in [Section 7.2](#) [[I-D.ietf-dots-data-channel](#)]) a priori conveyed with the filtering rules using the DOTS data channel.

[3.2.](#) DOTS Signal Channel Extension

[3.2.1.](#) Filtering Control

This specification extends the mitigation request defined in Section 4.4.1 of [[I-D.ietf-dots-signal-channel](#)] to convey the intended control of the configured filtering rules. The DOTS client conveys the following parameters in the CBOR body of the mitigation request:

acl-name: A name of an access list defined in the data channel.

As a reminder, an ACL is an ordered list of Access Control Entries (ACE). Each Access Control Entry has a list of match criteria and a list of actions [[I-D.ietf-dots-data-channel](#)]. The list of configured ACLs can be retrieved using the DOTS data channel during 'idle' time.

This is an optional attribute.

activation-type: Indicates the activation type of an ACL overriding the existing 'activation-type' installed by the DOTS client using the DOTS data channel.

This attribute can be set to 'deactivate', 'immediate', or 'activate-when-mitigating' defined [[I-D.ietf-dots-data-channel](#)].

Note that both 'immediate' and 'activate-when-mitigating' have an immediate effect when a mitigation request is being processed by the DOTS server.

If this attribute is not provided, the DOTS server MUST use 'activate-when-mitigating' as the default value.

This is an optional attribute.

The JSON/YANG mapping to CBOR for 'activation-type' is shown in Table 1.

Parameter Name	YANG Type	CBOR Key	CBOR Major Type & Information	JSON Type
activation-type	enumeration	0x0031 (TBD1)	0 unsigned	String

Table 1: JSON/YANG mapping to CBOR for 'activation-type'

When acl-* attributes are to be included in a request with an existing 'mid', the DOTS client MUST repeat all the other parameters as sent in the original mitigation request (i.e., having that 'mid') apart from a possible change to the lifetime parameter value.

If the DOTS server does not find the ACL name conveyed in the mitigation request in its configuration data for this DOTS client, it MUST respond with a "4.04 (Not Found)" error response code.

It is RECOMMENDED for a DOTS client to subscribe to asynchronous notifications of the attack mitigation, as detailed in Section 4.4.2.1 of [[I-D.ietf-dots-signal-channel](#)]. If not, the polling mechanism in Section 4.4.2.2 of [[I-D.ietf-dots-signal-channel](#)] has to be followed by the DOTS client.

A DOTS client MUST NOT use the filtering control over DOTS signal channel if no attack (mitigation) is active; such requests MUST be discarded by the DOTS server with 4.00 (Bad Request). By default, ACL-related operations are achieved using the DOTS data channel [[I-D.ietf-dots-data-channel](#)] when no attack is ongoing.

A DOTS client relies on the information received from the DOTS server and/or local information to the DOTS client domain to trigger a filter control request. Obviously, only filters that are pertinent for an ongoing mitigation should be controlled by a DOTS client using the DOTS signal channel.

This specification does not require any modification to the efficacy update, the retrieval of mitigation requests, and the withdrawal procedures defined in [[I-D.ietf-dots-signal-channel](#)]. In particular, ACL-related clauses are not included in a PUT request used to send an efficacy update, GET responses, and DELETE requests.

[3.2.2.](#) Sample Examples

This section provides sample examples to illustrate the behavior specified in [Section 3.2.1](#). These examples are provided for illustration purposes; they should not be considered as deployment recommendations.

[3.2.2.1.](#) Conflict Handling

Let's consider a DOTS client which contacts its DOTS server during 'idle' time to install an accept-list allowing for UDP traffic issued from 2001:db8:1234::/48 with a destination port number 443 to be forwarded to 2001:db8:6401::2/127. It does so by sending, for example, a PUT request shown in Figure 1.


```
PUT /restconf/data/ietf-dots-data-channel:dots-data\
  /dots-client=paL8p4Zqo4SLv64TLPXrxA/acls\
  /acl=an-accept-list HTTP/1.1
Host: {host}:{port}
Content-Type: application/yang-data+json
{
  "ietf-dots-data-channel:acls": {
    "acl": [
      {
        "name": "an-accept-list",
        "type": "ipv6-acl-type",
        "activation-type": "activate-when-mitigating",
        "aces": {
          "ace": [
            {
              "name": "test-ace-ipv6-udp",
              "matches": {
                "ipv6": {
                  "destination-ipv6-network": "2001:db8:6401::2/127",
                  "source-ipv6-network": "2001:db8:1234::/48"
                },
                "udp": {
                  "destination-port": {
                    "operator": "eq",
                    "port": 443
                  }
                }
              },
              "actions": {
                "forwarding": "accept"
              }
            }
          ]
        }
      }
    ]
  }
}
```

Figure 1: DOTS Data Channel Request to Create a Filtering

Some time later, consider that a DDoS attack is detected by the DOTS client on 2001:db8:6401::2/127. Consequently, the DOTS client sends a mitigation request to its DOTS server as shown in Figure 2.


```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=paL8p4Zqo4SLv64TLPXrxA"
Uri-Path: "mid=123"
Content-Format: "application/dots+cbor"
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:6401::2/127"
        ],
        "target-protocol": [
          17
        ],
        "lifetime": 3600
      }
    ]
  }
}
```

Figure 2: DOTS Signal Channel Mitigation Request

The DOTS server accepts immediately the request by replying with 2.01 (Created) (Figure 3).

```
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "mid": 123,
        "lifetime": 3600
      }
    ]
  }
}
```

Figure 3: Status Response

Assuming the DOTS client subscribed to asynchronous notifications, when the DOTS server concludes that some of the attack sources belong to 2001:db8:1234::/48, it sends a notification message with 'status' code set to '1 (Attack mitigation is in progress)' and 'conflict-cause' set to '2' (conflict-with-acceptlist) to the DOTS client to indicate that this mitigation request is in progress, but a conflict is detected.

Upon receipt of the notification message from the DOTS server, the DOTS client sends a PUT request to deactivate the "an-accept-list" ACL as shown in Figure 4.

The DOTS client can also decide to send a PUT request to deactivate the "an-accept-list" ACL, if suspect traffic is received from an accept-listed source (2001:db8:1234::/48). The structure of that PUT is the same as the one shown in Figure 4.

```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=paL8p4Zqo4SLv64TLPXrxA"
Uri-Path: "mid=123"
Content-Format: "application/dots+cbor"
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:6401::2/127"
        ],
        "target-protocol": [
          17
        ],
        "acl-list": [
          {
            "acl-name": "an-accept-list",
            "activation-type": "deactivate"
          }
        ]
      }
    ]
  }
}
```

Figure 4: PUT for Deactivating a Conflicting Filter

Then, the DOTS server deactivates "an-accept-list" ACL and replies with 2.04 (Changed) response to the DOTS client to confirm the successful operation.

3.2.2.2. Activate an Accept-List Filter

Let's consider a DOTS client which contacts its DOTS server during 'idle' time to install an accept-list allowing for UDP traffic issued from 2001:db8:1234::/48 to be forwarded to 2001:db8:6401::2/127. It does so by sending, for example, a PUT request shown in Figure 5. The DOTS server installs this filter with a "deactivated" state.

```
PUT /restconf/data/ietf-dots-data-channel:dots-data\
    /dots-client=ioiuLoZqo4SLv64TLPXrxA/acls\
    /acl=my-accept-list HTTP/1.1
Host: {host}:{port}
Content-Type: application/yang-data+json
{
  "ietf-dots-data-channel:acls": {
    "acl": [
      {
        "name": "my-accept-list",
        "type": "ipv6-acl-type",
        "activation-type": "deactivate",
        "aces": {
          "ace": [
            {
              "name": "an-ace",
              "matches": {
                "ipv6": {
                  "destination-ipv6-network": "2001:db8:6401::2/127",
                  "source-ipv6-network": "2001:db8:1234::/48",
                  "protocol": 17
                }
              },
              "actions": {
                "forwarding": "accept"
              }
            }
          ]
        }
      }
    ]
  }
}
```

Figure 5: DOTS Data Channel Request to Create an Accep-List Filter

Sometime later, consider that a UDP DDoS attack is detected by the DOTS client on 2001:db8:6401::2/127 but the DOTS client wants to let the traffic from 2001:db8:1234::/48 to be accept-listed to the DOTS

client domain. Consequently, the DOTS client sends a mitigation request to its DOTS server as shown in Figure 6.

```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=ioiuLoZqo4SLv64TLPXrxA"
Uri-Path: "mid=4879"
Content-Format: "application/dots+cbor"
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:6401::2/127"
        ],
        "target-protocol": [
          17
        ],
        "acl-list": [
          {
            "acl-name": "my-accept-list",
            "activation-type": "immediate"
          }
        ]
      }
    ]
  }
}
```

Figure 6: DOTS Signal Channel Mitigation Request with a Filter Control

The DOTS server activates "my-accept-list" ACL and replies with 2.01 (Created) response to the DOTS client to confirm the successful operation.

3.2.2.3. Activate Rate-Limit or Drop Filters

This section describes a scenario in which a DOTS client activates a drop-list or a rate-limit filter during an attack.

Consider a DOTS client that contacts its DOTS server during 'idle' time to install an accept-list that rate-limits all (or a part thereof) traffic to be forwarded to 2001:db8:123::/48 as a last resort countermeasure whenever required. It does so by sending, for

example, a PUT request shown in Figure 7. The DOTS server installs this filter with a "deactivated" state.

```
PUT /restconf/data/ietf-dots-data-channel:dots-data\
  /dots-client=0opPisZqo4SLv64TLPXrxA/acls\
  /acl=my-ratelimit-list HTTP/1.1
Host: {host}:{port}
Content-Type: application/yang-data+json
{
  "ietf-dots-data-channel:acls": {
    "acl": [
      {
        "name": "my-ratelimit-list",
        "type": "ipv6-acl-type",
        "activation-type": "deactivate",
        "aces": {
          "ace": [
            {
              "name": "my-ace",
              "matches": {
                "ipv6": {
                  "destination-ipv6-network": "2001:db8:123::/48"
                }
              },
              "actions": {
                "forwarding": "accept",
                "rate-limit": "20.00"
              }
            }
          ]
        }
      }
    ]
  }
}
```

Figure 7: DOTS Data Channel Request to Create a Rate-Limit Filter

Consider now that a DDoS attack is detected by the DOTS client on 2001:db8:123::/48. Consequently, the DOTS client sends a mitigation request to its DOTS server (Figure 8).


```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=0opPisZqo4SLv64TLPXrxA"
Uri-Path: "mid=85"
Content-Format: "application/dots+cbor"
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:123::/48"
        ],
        "lifetime": 3600
      }
    ]
  }
}
```

Figure 8: DOTS Signal Channel Mitigation Request to Activate a Rate-Limit Filter

For some reason (e.g., the DOTS server, or the mitigator, is lacking a capability or capacity), the DOTS client is still receiving the attack traffic which saturates available links. To soften the problem, the DOTS client decides to activate the filter that rate-limits the traffic destined to the DOTS client domain. To that aim, the DOTS client sends the mitigation request to its DOTS server shown in Figure 9.


```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=0opPisZqo4SLv64TLPXrxA"
Uri-Path: "mid=85"
Content-Format: "application/dots+cbor"
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:123::/48"
        ],
        "acl-list": [
          {
            "acl-name": "my-ratelimit-list",
            "activation-type": "activate"
          }
        ]
      }
    ]
  }
}
```

Figure 9: DOTS Signal Channel Mitigation Request to Activate a Rate-Limit Filter

Then, the DOTS server activates "my-ratelimit-list" ACL and replies with 2.04 (Changed) response to the DOTS client to confirm the successful operation.

3.2.3. DOTS Signal Filtering Control Module

3.2.3.1. Tree Structure

This document augments the "dots-signal-channel" DOTS signal YANG module defined in [[I-D.ietf-dots-signal-channel](#)] for managing the filtering rules.

This document defines the YANG module "ietf-dots-signal-control-filter", which has the following tree structure:


```
module: ietf-dots-signal-control-filter
  augment /ietf-signal:dots-signal/ietf-signal:message-type
    /ietf-signal:mitigation-scope/ietf-signal:scope:
  +--rw acl-list* [acl-name] {control-filtering}?
    +--rw acl-name
      |   -> /ietf-data:dots-data/dots-client/acls/acl/name
    +--rw activation-type?   activation-type
```

3.2.3.2. YANG Module

<CODE BEGINS> file "ietf-dots-signal-control-filter@2019-02-15.yang"

```
module ietf-dots-signal-control-filter {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-dots-signal-control-filter";
  prefix signal-control-filter;

  import ietf-dots-signal-channel {
    prefix ietf-signal;
    reference
      "RFC SSSS: Distributed Denial-of-Service Open Threat
        Signaling (DOTS) Signal Channel Specification";
  }
  import ietf-dots-data-channel {
    prefix ietf-data;
    reference
      "RFC DDDD: Distributed Denial-of-Service Open Threat
        Signaling (DOTS) Data Channel Specification";
  }

  organization
    "IETF DDoS Open Threat Signaling (DOTS) Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/dots/>
     WG List:  <mailto:dots@ietf.org>

    Author:   Konda, Tirumaleswar Reddy
              <mailto:TirumaleswarReddy_Konda@McAfee.com>

    Author:   Mohamed Boucadair
              <mailto:mohamed.boucadair@orange.com>

    Author:   Kaname Nishizuka
              <mailto:kaname@nttv6.jp>

    Author:   Takahiko Nagata
```


<<mailto:nagata@lepidum.co.jp>>;

description

"This module contains YANG definition for the signaling messages exchanged between a DOTS client and a DOTS server for the DOTS signal channel controlling the filtering rules configured using the DOTS data channel.

Copyright (c) 2019 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 (<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 2019-02-15 {

description

"Initial revision.";

reference

"RFC XXXX: Controlling Filtering Rules Using DOTS Signal Channel ";

}

feature control-filtering {

description

"This feature means that DOTS signal channel is able to manage the filtering rules created by the same DOTS client using the DOTS data channel.";

}

typedef activation-type {

type enumeration {

enum "activate-when-mitigating" {

value 1;

description

"The ACL is installed only when a mitigation is active.
The ACL is specific to this DOTS client.";

}

enum "immediate" {

value 2;

description

"The ACL is immediately activated.";


```
    }
    enum "deactivate" {
        value 3;
        description
            "The ACL is maintained by the DOTS server, but it is
            deactivated.";
    }
}
description
    "Set the activation type of an ACL.";
}

augment "/ietf-signal:dots-signal/ietf-signal:message-type/" +
    "ietf-signal:mitigation-scope/ietf-signal:scope" {
    if-feature control-filtering;

    description "ACL name and activation type";

    list acl-list {
        key "acl-name";
        description
            "List of ACLs as defined in the DOTS data
            channel. These ACLs are uniquely defined by
            cuid and name.";
        leaf acl-name {
            type leafref {
                path "/ietf-data:dots-data/ietf-data:dots-client/" +
                    "ietf-data:acls/ietf-data:acl/ietf-data:name";
            }
            description
                "Reference to the ACL name bound to a DOTS client.";
        }
        leaf activation-type {
            type activation-type;
            default activate-when-mitigating;
            description
                "Set the activation type of an ACL.";
        }
    }
}
}
<CODE ENDS>
```

[4.](#) IANA Considerations

4.1. DOTS Signal Channel CBOR Mappings Registry

This specification registers the 'activation-type' parameter in the IANA "DOTS Signal Channel CBOR Key Values" registry established by [\[I-D.ietf-dots-signal-channel\]](#).

The 'activation-type' is a comprehension-required parameter. The 'acl-list' and 'acl-name' parameters are defined as comprehension-required parameters in Table 6 in [\[I-D.ietf-dots-signal-channel\]](#). Following the rules in [\[I-D.ietf-dots-signal-channel\]](#), if the DOTS server does not understand the 'acl-list' or 'acl-name' or 'activation-type' attributes, it responds with a "4.00 (Bad Request)" error response code.

- o Note to the RFC Editor: Please delete (TBD1) once the CBOR key is assigned from the (0x0001 - 0x3FFF) range.

Parameter Name	CBOR Key Value	CBOR Major Type	Change Controller	Specification Document(s)
activation-type	0x0031 (TBD1)	0	IESG	[RFCXXXX]

4.2. DOTS Signal Control Filtering YANG Module

This document requests IANA to register the following URI in the "IETF XML Registry" [\[RFC3688\]](#):

URI: urn:ietf:params:xml:ns:yang:ietf-dots-signal-control-filter
 Registrant Contact: The IESG.
 XML: N/A; the requested URI is an XML namespace.

This document requests IANA to register the following YANG module in the "YANG Module Names" registry [\[RFC7950\]](#).

name: ietf-dots-signal-control-filter
 namespace: urn:ietf:params:xml:ns:yang:ietf-dots-signal-control-filter
 maintained by IANA: N
 prefix: signal-control-filter
 reference: RFC XXXX

5. Security Considerations

The security considerations discussed in [I-D.ietf-dots-signal-channel] and [I-D.ietf-dots-data-channel] need to be taken into account.

6. Acknowledgements

Thank you to Takahiko Nagata, Wei Pan, and Xia Liang for the comments.

7. References

7.1. Normative References

- [I-D.ietf-dots-data-channel]
Boucadair, M. and R. K, "Distributed Denial-of-Service Open Threat Signaling (DOTS) Data Channel Specification", [draft-ietf-dots-data-channel-27](#) (work in progress), February 2019.
- [I-D.ietf-dots-signal-channel]
K, R., Boucadair, M., Patil, P., Mortensen, A., and N. Teague, "Distributed Denial-of-Service Open Threat Signaling (DOTS) Signal Channel Specification", [draft-ietf-dots-signal-channel-30](#) (work in progress), March 2019.
- [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>>.
- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", [RFC 7950](#), DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.
- [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>>.

7.2. Informative References

- [I-D.ietf-dots-requirements]
Mortensen, A., K, R., and R. Moskowitz, "Distributed Denial of Service (DDoS) Open Threat Signaling Requirements", [draft-ietf-dots-requirements-20](#) (work in progress), February 2019.
- [Interop] Nishizuka, K., Shallow, J., and L. Xia , "DOTS Interop test report, IETF 103 Hackathon", November 2018, <<https://datatracker.ietf.org/meeting/103/materials/slides-103-dots-interop-report-from-ietf-103-hackathon-00>>.

Authors' Addresses

Kaname Nishizuka
NTT Communications
GranPark 16F 3-4-1 Shibaura, Minato-ku
Tokyo 108-8118
Japan

Email: kaname@nttv6.jp

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Tirumaleswar Reddy
McAfee, Inc.
Embassy Golf Link Business Park
Bangalore, Karnataka 560071
India

Email: kondtir@gmail.com

Takahiko Nagata
Lepidum
Japan

Email: nagata@lepidum.co.jp

