DOTS   K. Nishizuka
Internet-Draft   NTT Communications
Intended status: Standards Track   M. Boucadair
Expires: ~~October 25,~~ **November 17,** 2019   Orange
                                                                  T. Reddy
                                                                    McAfee
                                                                 T. Nagata
                                                                   Lepidum
                                                             ~~April 23,~~
                                                             May 16, 2019

     Controlling Filtering Rules Using Distributed Denial-of-Service Open
                     Threat Signaling (DOTS) Signal Channel
                     ~~draft-ietf-dots-signal-filter-control-00~~
                     draft-ietf-dots-signal-filter-control-01

Abstract

   This document specifies an extension to the DOTS signal channel
   protocol so that DOTS clients can control their filtering rules when
   an attack mitigation is active.

   Particularly, this extension allows a DOTS client to activate or de-
   activate existing filtering rules during a DDoS attack. The
   characterization of these filtering rules is supposed to be conveyed
   by a DOTS client during an idle time by means of the DOTS data
   channel protocol.

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 Distributed Denial-
       of-Service Open Threat Signaling (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

Table of Contents

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~~ **Distributed
   Denial-of-Service (DDoS)** attack. Filter management is one of its
   tasks which enables a DOTS client to retrieve the filtering
   capabilities of a DOTS server and to manage filtering rules. ~~These~~
   Typically, these Filtering rules are used for dropping or ~~rate-limiting~~ **rate-
   limiting** unwanted traffic, and permitting accept-listed traffic.

   Unlike the DOTS signal channel **protocol**
   [I-D.ietf-dots-signal-channel], the DOTS data channel **protocol** 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 **the** DOTS data channel protocol may not function as expected
   during DDoS attacks ~~or~~ **or, worse,** exacerbate an ongoing DDoS attack.
   The DOTS data channel **protocol** 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) some filtering rules using **the** DOTS
   data channel **protocol** 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 a DDoS attack.

   [I-D.ietf-dots-signal-channel] is designed so that the DDoS server
   notifies the conflict to the DOTS client (that is, 'conflict-cause'
   parameter 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.~~ **link to the DOTS client domain.** In other words, the DOTS
   client cannot use the DOTS data channel **protocol** to withdraw the
   accept-list filters when ~~the~~ **a** 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,~~ **channel protocol,** which enables a DOTS client to request the
   activation

~~or deactivation~~ **(or deactivation)** 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~~ **an** attack will enhance
the protection capability offered by DOTS protocols.

> Note: The experiment at the IETF103 hackathon [Interop] showed
> that even when the ~~incoming~~ **inbound** 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.

**An augment to the DOTS signal channel YANG module is defined in
Section 3.2.2.**

Sample examples are provided in Section 4, in particular:

o   Section 4.1 illustrates how the filter control extension is used
    when conflicts with ~~ACLs~~ **Access Control List (ACLs)** are detected **and
    reported** by a DOTS server.

o   Section 4.2 shows how a DOTS client can instruct a DOTS server to
    safely forward some specific traffic in 'attack' time.

o   Section 4.3 shows how a DOTS client can react if **the** DDoS traffic
    is still being forwarded to the DOTS client domain even if
    mitigation requests were sent to a DOTS server.

**The JavaScript Object Notation (JSON) encoding of YANG-modeled data
[RFC7951] is used to illustrate the examples.**

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].

The **terminology for describing YANG modules is defined in [RFC7950].
The** meaning of the symbols in **the** tree ~~diagrams~~ **diagram** is defined in
[RFC8340].

3.   Controlling Filtering Rules of a DOTS Client

3.1.   Binding ~~the~~ **DOTS** Data and Signal Channels

The filtering rules eventually managed using the DOTS signal channel
protocol are created a priori by the same DOTS client using the DOTS
data
~~channel.~~ **channel protocol.** 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 **DOTS** signal and
data channels. This requirement is meant to facilitate binding DOTS
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 **the** scope of this document.

A filtering rule controlled by the DOTS signal channel is identified
by its ~~Access Control List (ACL)~~ **ACL** name (Section 7.2 of [I-D.ietf-dots-data-channel]). 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 deactivation of an ACL by the **DOTS** signal channel
overrides the 'activation-type' (defined in Section 7.2 of

[I-D.ietf-dots-data-channel]) a priori conveyed with the filtering
rules using the DOTS data ~~channel.~~ **channel protocol.**

3.2.　DOTS Signal Channel Extension

3.2.1.　Parameters ~~&~~ **and** Behaviors

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.　Concretely, the DOTS
client conveys **'acl-list' attribute with** the following ~~parameters~~ **sub-attributes**
in the CBOR body of a mitigation ~~request:~~ **request (see the YANG-encoded
structure in Section 3.2.2.1):**

acl-name:　A name of an access list defined using the DOTS data
　　channel (Section 7.2 of [I-D.ietf-dots-data-channel]) that is
　　associated with the DOTS client.

　　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.

　　As a reminder, this attribute can be set to 'deactivate',
　　'immediate', or 'activate-when-mitigating' as defined in
　　[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.

　　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'

~~A DOTS client may include acl * attributes in a mitigation request
having a new or an existing 'mid'.　When acl * attributes~~

By default, ACL-related operations are ~~to be
included in a mitigation 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~~ **achieved using** the ~~lifetime parameter value.~~

~~It is RECOMMENDED for a~~ DOTS ~~client to subscribe to asynchronous
notifications of the~~ **data
channel protocol when no** 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~~ **is ongoing.** DOTS ~~client~~ **clients** MUST NOT
use the filtering control over DOTS signal channel in 'idle' time;
such requests MUST be discarded by ~~the~~ DOTS
~~server~~ **servers** 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~~

During an attack **time, DOTS clients may include 'acl-list', 'acl-
name', and 'activation-type' attributes in a mitigation request.
This request may be the initial mitigation request for a given
mitigation scope or a new one overriding an existing request.　In**

both cases, a new 'mid' MUST be used. Nevertheless, it is ~~ongoing.~~ **NOT
RECOMMENDED to include ACL attributes in an initial mitigation
request for a given mitigation scope or in a mitigation request
adjusting the mitigation scope.**

**As the attack evolves, DOTS clients can adjust the 'activation-type'
of an ACL conveyed in a mitigation request or control other filters
as necessary. This can be achieved by sending a PUT request with a
new 'mid' value.**

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

**As a reminder, the 'acl-list' and 'acl-name' parameters are defined
as comprehension-required parameters in Table 6 of
[I-D.ietf-dots-signal-channel]. Also, the 'activation-type' is
defined as a comprehension-required parameter (Section 5.1).
Following the rules in Section 6 of [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.**

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

If the DOTS server finds the ACL name for this DOTS client, and
assuming the request passed the validation checks in **Section 4.4.1 of**
[I-D.ietf-dots-signal-channel], the DOTS server MUST proceed with the
'activation-type' update. The update is immediately enforced by the
DOTS server and will be maintained as the new activation type for the
ACL name even after the termination of the mitigation request. In
addition, the DOTS server MUST update the lifetime of the
corresponding ACL similar to the update when a refresh request is
received using the DOTS data ~~channel.~~ **channel (Section 7.2 of
[I-D.ietf-dots-data-channel]). If, for some reason, the DOTS server
fails to apply the filter update, it MUST respond with 5.03 (Service
Unavailable) error response code and include the failed ACL update in
the diagnostic payload of the response (an example is shown in
Figure 1). Else, the DOTS server replies with the appropriate
response code defined in Section 4.4.1 of
[I-D.ietf-dots-signal-channel].**

```
{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "mid": 123,
        "ietf-dots-signal-control:acl-list": [
          {
            "ietf-dots-signal-control:acl-name": "an-accept-list",
            "ietf-dots-signal-control:activation-type": "deactivate"
          }
        ]
      }
    ]
  }
}
```

Figure 1: Example of a Diagnostic Payload Including Failed ACL Update

**If the DOTS client receives a 5.03 (Service Unavailable) with a
diagnostic payload indicating a failed ACL update as a response to an
initial mitigation or a mitigation with adjusted scope, the DOTS
client MUST immediately send a new request which repeats all the
parameters as sent in the failed mitigation request but without
including the ACL attributes. After the expiry of Max-Age returned
in the 5.03 (Service Unavailable) response, the DOTS client retries
with a new mitigation request (i.e., a new 'mid') that repeats all
the parameters as sent in the failed mitigation request.**

If, during an active mitigation, the 'activation-type' is changed at
the DOTS server (e.g., as a result of an external action) for an ACL

```
        bound to a DOTS client, the DOTS server notifies that DOTS client
        with the change by including the corresponding acl-* ACL parameters in an
        asynchronous notification (the DOTS client is observing the active
        mitigation) or in a response to a polling request (Section 4.4.2.2 of
        [I-D.ietf-dots-signal-channel]). (Section 4.4.2.2 of
        [I-D.ietf-dots-signal-channel]).

        If the DOTS signal and data channels of a DOTS client are not
        established with the same DOTS server of a DOTS server domain, the
        above request processing operations are undertaken using the
        coordination mechanism discussed in Section 3.1.

        This specification does not require any modification to the efficacy
        update 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 and
        DELETE requests.

3.2.2.  DOTS Signal Filtering Control Module

3.2.2.1.  Tree Structure

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

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

        module: ietf-dots-signal-control
          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?   ietf-data:activation-type

3.2.2.2.  YANG Module

        This module uses types defined in [I-D.ietf-dots-data-channel].

        <CODE BEGINS> file "ietf-dots-signal-control@2019-04-01.yang" "ietf-dots-signal-control@2019-05-13.yang"

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

          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
           to control, by means of the DOTS signal channel, 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 2019-04-01 2019-05-13 {
          description
            "Initial revision.";
          reference
            "RFC XXXX: Controlling Filtering Rules Using Distributed
                       Denial-of-Service Open Threat Signaling (DOTS)
                       Signal Channel";
        }

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

        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 using the DOTS data
               channel.  These ACLs bound to a DOTS client are uniquely defined
               identified by
               cuid and a 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 ietf-data:activation-type;
            default "activate-when-mitigating";
            description
              "Set
              "Sets the activation type of an ACL.";
          }
        }
      }
    }
    <CODE ENDS>
```

4. 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.

4.1. 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. 2.

```
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:~~ **2:** 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.~~ **3.**

```
  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:~~ **3:** DOTS Signal Channel Mitigation Request

The DOTS server accepts immediately the request by replying with 2.01
(Created) (Figure ~~3).~~ **4 depicts the message body of the response).**

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

```
          ]
       }
   }
}
```

                    Figure 3: **4:** Status Response (Message Body)

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. **5.**

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. **5.**

```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=paL8p4Zqo4SLv64TLPXrxA"
Uri-Path: "mid=123" "mid=124"
Content-Format: "application/dots+cbor"

{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:6401::2/127"
        ],
        "target-protocol": [
          17
        ],
        "acl-list":
        "ietf-dots-signal-control:acl-list": [
          {
            "acl-name":
            "ietf-dots-signal-control:acl-name": "an-accept-list",
            "activation-type":
            "ietf-dots-signal-control:activation-type": "deactivate"
          }
        ]
        "lifetime": 3600
      }
    ]
  }
}
```

            Figure 4: **5:** 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. The message body is similar to the one
depicted in Figure 3. **4.**

Once the attack is mitigated, the DOTS client may use the data
channel to retrieve its ACLs maintained by the DOTS server. As shown
in Figure 5, **6,** the activation type is set to 'deactivate' as set by the
signal channel (Figure 4) **5)** instead of the type initially set using the
data channel (Figure 1). **2).**

```
{
  "ietf-dots-data-channel:acls": {
    "acl": [
      {
        "name": "an-accept-list",
        "type": "ipv6-acl-type",
        "activation-type": "deactivate",
        "pending-lifetime": 10021,
        "aces": {
          "ace": [
            {
              "name": "test-ace-ipv6-udp",
              "matches": {
```
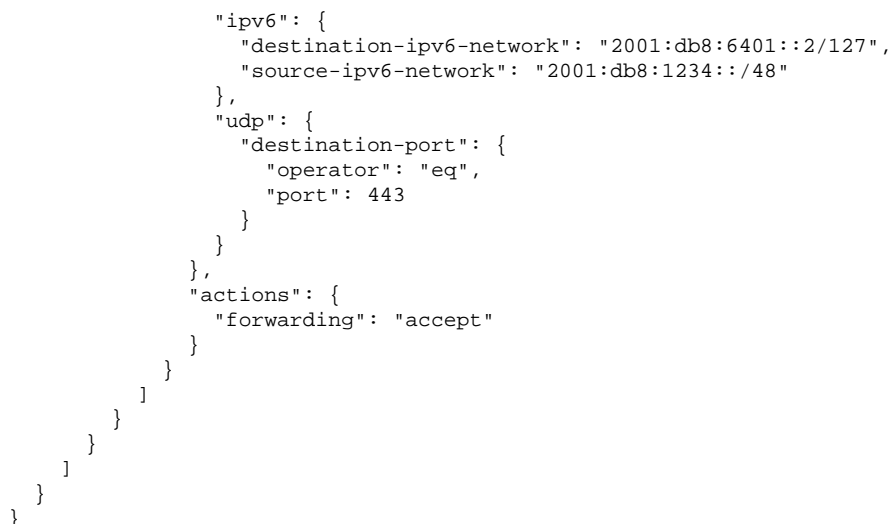
```
                        "ipv6": {
                          "destination-ipv6-network": "2001:db8:6401::2/127",
                          "source-ipv6-network": "2001:db8:1234::/48"
                        },
                        "udp": {
                          "destination-port": {
                            "operator": "eq",
                            "port": 443
                          }
                        }
                      },
                      "actions": {
                        "forwarding": "accept"
                      }
                    }
                  ]
                }
              }
            ]
          }
        }
```

           Figure ~~5:~~ **6:** DOTS Data Channel GET Response after Mitigation

4.2.   On-Demand Activation of 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 ~~6.~~ **7.**
   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 ~~6:~~ **7:** DOTS Data Channel Request to Create an Accept-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 ~~7.~~ **8.**

```
    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":
              "ietf-dots-signal-control:acl-list": [
                {
                  "acl-name":
                  "ietf-dots-signal-control:acl-name": "my-accept-list",
                  "activation-type":
                  "ietf-dots-signal-control:activation-type": "immediate"
                }
              ]
              "lifetime": 3600
            }
          ]
        }
      }
```

         Figure 7: 8: 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.

4.3. DOTS Servers/Mitigators Lacking Capacity

   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 8. 9. The DOTS server installs
   this filter with a "deactivated" state.

```
   PUT /restconf/data/ietf-dots-data-channel:dots-data\
       /dots-client=OopPisZqo4SLv64TLPXrxA/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 8: 9: 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 9). 10).

  Header: PUT (Code=0.03)
  Uri-Path: ".well-known"
  Uri-Path: "dots"
  Uri-Path: "mitigate"
  Uri-Path: "cuid=OopPisZqo4SLv64TLPXrxA"
  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 9: 10: 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 trafic 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 10. 11.

```
Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
Uri-Path: "cuid=OopPisZqo4SLv64TLPXrxA"
Uri-Path: mid=85 "mid=86"
Content-Format: "application/dots+cbor"

{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:123::/48"
        ],
          "acl-list":
        "ietf-dots-signal-control:acl-list": [
          {
              "acl-name":
            "ietf-dots-signal-control:acl-name": "my-ratelimit-list",
              "activation-type": "activate"
            "ietf-dots-signal-control:activation-type": "immediate"
          }
        ]
        "lifetime": 3600
      }
    ]
  }
}
```

Figure 10: 11: 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.

As the attack mitigation evolves, the DOTS client may decide to
deactivate the rate-limit policy (e.g., upon receipt of notification
status change from 'attack-exceeded-capability' to 'attack-
mitigation-in-progress'). Based on the mitigation status conveyed by
the DOTS server, the DOTS client can de-activate the rate-limit
action.). It does so by sending the request shown in Figure 12.

Header: PUT (Code=0.03)
Uri-Path: ".well-known"
Uri-Path: "dots"
Uri-Path: "mitigate"
```

```
Uri-Path: "cuid=OopPisZqo4SLv64TLPXrxA"
Uri-Path: "mid=87"
Content-Format: "application/dots+cbor"

{
  "ietf-dots-signal-channel:mitigation-scope": {
    "scope": [
      {
        "target-prefix": [
          "2001:db8:123::/48"
        ],
        "ietf-dots-signal-control:acl-list": [
          {
            "ietf-dots-signal-control:acl-name": "my-ratelimit-list",
            "ietf-dots-signal-control:activation-type": "deactivate"
          }
        ]
        "lifetime": 3600
      }
    ]
  }
}
```

         Figure 12: DOTS Signal Channel Mitigation Request to Deactivate a
                              Rate-Limit Filter

5. IANA Considerations

5.1. 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	CBOR	Change	Specification
	Key	Major	Controller	Document(s)
	Value	Type		
   +-------------------+--------+-------+-----------+--------------+
   | activation-type | 0x0031 | 0 | IESG | [RFCXXXX] |
   | | (TBD1) | | | |
   +-------------------+--------+-------+-----------+--------------+

5.2. DOTS Signal Filtering Control 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
     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
     Namespace: urn:ietf:params:xml:ns:yang:ietf-dots-signal-control
     Maintained by IANA: N
     Prefix: signal-control
     Reference: RFC XXXX

6. 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.

   A **DOTS client is entitled to access only to resources it creates. In
   particular, a DOTS client can not tweak filtering rules created by
   other DOTS clients of the same DOTS client domain.**

A compromised DOTS client can use the filtering control capability to exacerbate an ongoing attack. Likewise, such compromised DOTS client may abstain from reacting to an ACL conflict notification received from the DOTS server during attacks. These are not new attack vectors, but variations of threats discussed in [I-D.ietf-dots-signal-channel] and [I-D.ietf-dots-data-channel]. DOTS operators should carefully monitor and audit DOTS agents to detect ~~misbehavior~~ **misbehaviors** and to deter ~~misuse.~~ **misuses.**

7. Acknowledgements

   ~~Thank you~~

   Many thanks to Takahiko Nagata, Wei Pan, Xia Liang, Jon ~~Shollow,~~ **Shallow,** and Dan Wing for the comments.

8. References

8.1. 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-28~~
              draft-ietf-dots-data-channel-29 (work in progress), ~~March~~ **May**
              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-31~~
              ietf-dots-signal-channel-33 (work in progress), ~~March~~ **May** 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>.

8.2. 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-22 (work in
              progress), March 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>.

   **[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG",
              RFC 7951, DOI 10.17487/RFC7951, August 2016,
              <https://www.rfc-editor.org/info/rfc7951>.**

   [RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams",
              BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018,
              <https://www.rfc-editor.org/info/rfc8340>.

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