DOTS
Internet-Draft
Intended status: Standards Track
Expires: June 21, 2018

T. Reddy, Ed.
McAfee
M. Boucadair, Ed.
Orange
K. Nishizuka
NTT Communications
L. Xia
Huawei
P. Patil
Cisco
A. Mortensen
Arbor Networks, Inc.
N. Teague
Verisign, Inc.

December 18, 2017

Distributed Denial-of-Service Open Threat Signaling (DOTS) Data Channel

draft-ietf-dots-data-channel-11

Abstract

The document specifies a Distributed Denial-of-Service Open Threat Signaling (DOTS) data channel used for bulk exchange of data not easily or appropriately communicated through the DOTS signal channel under attack conditions.

This is a companion document to the DOTS signal channel specification.

Editorial Note (To be removed by RFC Editor)

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

- "This version of this YANG module is part of RFC XXXX;"
- "RFC XXXX: Distributed Denial-of-Service Open Threat Signaling (DOTS) Data Channel";
- reference: RFC XXXX

Status of This Memo

This Internet-Draft will expire on June 21, 2018.

Copyright Notice

Table of Contents

1. Introduction

A distributed denial-of-service (DDoS) attack is an attempt to make machines or network resources unavailable to their intended users. In most cases, sufficient scale can be achieved by compromising enough end-hosts and using those infected hosts to perpetrate and amplify the attack. The victim in such attack can be an application server, a router, a firewall, an entire network, etc.

As discussed in [I-D.ietf-dots-requirements], the lack of a common method to coordinate a real-time response among involved actors and network domains inhibits the speed and effectiveness of DDoS attack mitigation. From that standpoint, DDoS Open Threat Signaling (DOTS) [I-D.ietf-dots-architecture] defines an architecture enabling requests for DDoS attack mitigation, reducing attack impact, and contributing to more efficient defensive strategies. To that aim, DOTS defines two channels: signal and data channels (Figure 1).

```
+---------------+                                 +---------------+
|               | <------- Signal Channel ------> |               |
|  DOTS Client  |                                 |  DOTS Server  |
|               | <=======  Data Channel  ======> |               |
+---------------+                                 +---------------+
```

Figure 1: DOTS Channels

The DOTS signal channel is used to convey that a network is under a DDoS attack to an upstream DOTS server so that appropriate mitigation actions are undertaken on the suspect traffic. The DOTS signal channel is further elaborated in [I-D.ietf-dots-signal-channel].

The DOTS data channel is used for infrequent bulk data exchange between DOTS agents in the aim to significantly augment attack response coordination. Section 2 of [I-D.ietf-dots-architecture] identifies that the DOTS data channel is used to perform the following tasks:

o Creating identifiers, such as names or aliases, for resources for which mitigation may be requested.

A DOTS client may submit to its DOTS server a collection of prefixes which it would like to refer to by an alias when requesting mitigation. The DOTS server can respond to this request with either a success or failure response (see Section 2 in [I-D.ietf-dots-architecture]).

Refer to Section 6 for more details.

o Filter management, which enables a DOTS client to request the installation or removal of traffic filters, dropping or rate-limiting unwanted traffic, and permitting white-listed traffic.

Sample use cases for populating black- or white-list filtering rules are detailed hereafter:

* If a network resource (DOTS client) detects a potential DDoS attack from a set of IP addresses, the DOTS client informs its servicing DOTS gateway of all suspect IP addresses that need to be blocked or black-listed for further investigation. The DOTS client could also specify a list of protocols and port numbers in the black-list rule.

  The DOTS gateway in-turn propagates the black-listed IP addresses to a DOTS server which will undertake appropriate actions so that traffic from these IP addresses to the target network (specified by the DOTS client) is blocked.

* A network, that has partner sites from which only legitimate traffic arrives, may want to ensure that the traffic from these sites is not penalized during DDoS attacks. The DOTS client uses the DOTS data channel to convey the white-listed IP prefixes of the partner sites to its DOTS server.

  The DOTS server uses this information to white-list flows from such IP prefixes reaching the network.

Refer to Section 7 for more details.

2. Notational Conventions and Terminology

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

The reader should be familiar with the terms defined in [I-D.ietf-dots-architecture].

The terminology for describing YANG data modules is defined in [RFC7950]. The meaning of the symbols in tree diagrams is defined in [I-D.ietf-netmod-yang-tree-diagrams].

For simplicity, all of the examples in this document use "/restconf" as the discovered RESTCONF API root path. Many protocol header lines and message-body text within examples throughout the document are split into multiple lines for display purposes only. When a line ends with backslash ('\') as the last character, the line is wrapped for display purposes. It is to be considered to be joined to the next line by deleting the backslash, the following line break, and the leading whitespace of the next line.

3. DOTS Data Channel: Design Overview

Unlike the DOTS signal channel [I-D.ietf-dots-signal-channel], which must operate nominally even when confronted with signal degradation due to packets loss, the DOTS data channel is not expected to be constructed to deal with DDoS attack conditions. The requirements for DOTS data channel protocol are documented in [I-D.ietf-dots-requirements].

This specification does not require an order of contact nor the time interval between DOTS signal and data channel creations. These considerations are implementation- and deployment-specific.

As the primary function of the data channel is data exchange, a reliable transport is required in order for DOTS agents to detect data delivery success or failure. This document uses RESTCONF [RFC8040] over TLS [RFC5246] over TCP as the DOTS data channel protocol (Figure 2).

> Note: RESTCONF is a protocol based on HTTP [RFC7230] to provide CRUD (create, read, update, delete) operations on a conceptual datastore containing YANG data. Concretely, RESTCONF is used for configuring data defined in YANG version 1 [RFC6020] or YANG version 1.1 [RFC7950], using the datastore concepts defined in the Network Configuration Protocol (NETCONF) [RFC6241]. RESTCONF combines the simplicity of the HTTP protocol with the predictability and automation potential of a schema-driven API. RESTCONF offers a simple subset of NETCONF functionality and provides a simplified interface using REST-like API which addresses the needs of the DOTS data channel and hence an optimal choice.

```
+-------------------+
| DOTS Data Channel |
+-------------------+
|      RESTCONF     |
+-------------------+
|        TLS        |
+-------------------+
|        TCP        |
+-------------------+
|        IP         |
+-------------------+
```

Figure 2: Abstract Layering of DOTS Data Channel over RESTCONF over TLS

The HTTP POST, PUT, PATCH, and DELETE methods are used to edit data resources represented by DOTS data channel YANG data modules. These basic edit operations allow the DOTS data channel running configuration to be altered by a DOTS client.

DOTS data channel configuration data and state data can be retrieved with the GET method. An HTTP status-line header field is returned for each request to report success or failure for RESTCONF operations (Section 5.4 of [RFC8040]).

The DOTS client performs the root resource discovery procedure discussed in Section 3.1 of [RFC8040] to determine the root of the RESTCONF API. After discovering the RESTCONF API root, the DOTS client uses this value as the initial part of the path in the request URI, in any subsequent request to the DOTS server. The DOTS server may support retrieval of the YANG modules it supports (Section 3.7 in [RFC8040]), for example, a DOTS client may use RESTCONF to retrieve the company proprietary YANG modules supported by the DOTS server.

JavaScript Object Notation (JSON) [RFC7159] payload is used to propagate data channel specific payload messages that convey request parameters and response information such as errors. This specification uses the encoding rules defined in [RFC7951] for representing DOTS data channel configuration data defined using YANG (Section 5) as JSON text.

A DOTS client registers itself to its DOTS server(s) in order to set up DOTS data channel related configuration data and receive state data (i.e., non-configuration data) from the DOTS server(s).

A single DOTS data channel between DOTS agents can be used to exchange multiple requests and multiple responses. To reduce DOTS client and DOTS server workload, DOTS client SHOULD re-use the same

TLS session. While the communication to the DOTS server is quiescent, the DOTS client MAY probe the server to ensure it has maintained cryptographic state. Such probes can also keep alive firewall and/or NAT bindings. A TLS heartbeat [RFC6520] verifies the DOTS server still has TLS state by returning a TLS message.

In deployments where one or more translators (e.g., NAT44, NAT64, NPTv6) are enabled between the client's network and the DOTS server, DOTS data channel messages forwarded to a DOTS server must not include internal IP addresses/prefixes and/or port numbers; external addresses/prefixes and/or port numbers as assigned by the translator must be used instead. This document does not make any recommendation about possible translator discovery mechanisms. The following are some (non-exhaustive) deployment examples that may be considered:

o Port Control Protocol (PCP) [RFC6887] or Session Traversal Utilities for NAT (STUN) [RFC5389] may be used to retrieve the external addresses/prefixes and/or port numbers. Information retrieved by means of PCP or STUN will be used to feed the DOTS data channel messages that will be sent to a DOTS server.

o A DOTS gateway may be co-located with the translator. The DOTS gateway will need to update the DOTS messages, based upon the local translator's binding table.

When a DOTS gateway is involved in DOTS data channel exchanges, the same considerations for manipulating 'client-identifier' parameter as specified in [I-D.ietf-dots-signal-channel] MUST be followed by DOTS agents. This specification includes examples to illustrate sample messages without any 'client-identifier' parameter, messages with 'client-identifier' parameter having one single value, and messages with 'client-identifier' parameter listing multiple values.

A DOTS server may detect conflicting filtering requests from the same or distinct DOTS clients which belong to the same domain. For example, a DOTS client would request to blacklist a prefix, while another DOTS client would request to whitelist that same prefix. It is out of scope of this specification to recommend the behavior to follow for handling conflicting requests (e.g., reject all, reject the new request, notify an administrator for validation). DOTS servers SHOULD support a configuration parameter to indicate the behavior to follow when a conflict is detected. Section 7.1 specifies the behavior when no instruction is supplied to a DOTS server.

4. DOTS Server(s) Discovery

This document assumes that DOTS clients are provisioned with the reachability information of their DOTS server(s) using a variety of means (e.g., local configuration, or dynamic means such as DHCP). These means are out of scope of this document.

Likewise, it is out of scope of this document to specify the behavior to follow by a DOTS client to place its requests (e.g., contact all servers, select one server among the list) when multiple DOTS servers are provisioned.

5. DOTS Data Channel YANG Module

5.1. Identifier YANG Tree Structure

The YANG module (ietf-dots-data-channel) allows to create identifiers, such as names or aliases, for resources for which mitigation may be requested. Such identifiers may be used in subsequent DOTS signal channel exchanges to refer more efficiently to the resources under attack. The tree structure for DOTS identifiers is as follows:

```
    +--rw identifier
       +--rw client-identifier*   binary
       +--rw alias* [alias-name]
          +--rw alias-name            string
          +--rw target-prefix*        inet:ip-prefix
          +--rw target-port-range* [lower-port upper-port]
          |  +--rw lower-port    inet:port-number
          |  +--rw upper-port    inet:port-number
          +--rw target-protocol*      uint8
          +--rw target-fqdn*          inet:domain-name
          +--rw target-uri*           inet:uri
```

This structure is aligned with [I-D.ietf-dots-signal-channel].

5.2. Filter YANG Tree Structure

This document augments the Access Control List (ACL) YANG module [I-D.ietf-netmod-acl-model] for managing DOTS filtering rules. The notion of ACL is explained in Section 1 of [I-D.ietf-netmod-acl-model].

Examples of ACL management in a DOTS context include, but not limited to:

- o Black-list management, which enables a DOTS client to inform a DOTS server about sources from which traffic should be suppressed.

- o White-list management, which enables a DOTS client to inform a DOTS server about sources from which traffic should always be accepted.

- o Filter management, which enables a DOTS client to request the installation or removal of traffic filters, dropping or rate-limiting unwanted traffic and permitting white-listed traffic.

This document defines the DOTS Data Channel YANG to augment the "ietf-access-control-list" module to support filters based on the client identifier (client-identifier), to support rate-limit action (rate-limit), and to handle fragmented packets (fragments).

Filtering fragments adds an additional layer of protection against a DoS attack that uses only non-initial fragments. When there is only Layer 3 information in the ACL entry and the fragments keyword is present, for non-initial fragments matching the ACL entry, the deny or permit action associated with the ACL entry will be enforced and for initial or non-fragment matching the ACL entry, the next ACL entry will be processed. When there is both Layer 3 and Layer 4 information in the ACL entry and the fragments keyword is present, the ACL action is conservative for both permit and deny actions. The actions are conservative to not accidentally deny a fragmented portion of a flow because the fragments do not contain sufficient information to match all of the filter attributes. In the deny action case, instead of denying a non-initial fragment, the next ACL entry is processed. In the permit case, it is assumed that the Layer 4 information in the non-initial fragment, if available, matches the Layer 4 information in the ACL entry.

The tree structure for DOTS filtering rules is as follows:

```
  augment /ietf-acl:access-lists:
    +--rw client-identifier*   binary
  augment /ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace/ietf-
acl:actions:
    +--rw rate-limit?   decimal64
  augment /ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace/ietf-
acl:matches/ietf-acl:ipv4-acl:
    +--rw fragments?   empty
  augment /ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces/ietf-acl:ace/ietf-
acl:matches/ietf-acl:ipv6-acl:
    +--rw fragments?   empty
  augment /ietf-acl:access-lists:
    +--rw dots-acl-order
```

```
       +--rw acl-set* [set-name type]
          +--rw set-name    -> /ietf-acl:access-lists/acl/acl-name
          +--rw type        -> /ietf-acl:access-lists/acl/acl-type
```

5.3. YANG Module

```
<CODE BEGINS> file "ietf-dots-data-channel@2017-12-18.yang"

module ietf-dots-data-channel {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-dots-data-channel";

  prefix "data-channel";

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

  organization "IETF DDoS Open Threat Signaling (DOTS) Working Group";

  contact
    "WG Web:   <https://datatracker.ietf.org/wg/dots/>
     WG List:  <mailto:dots@ietf.org>

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

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

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

     Author:  Liang Xia
              <mailto:frank.xialiang@huawei.com>

     Author:  Prashanth Patil
              <mailto:praspati@cisco.com>

     Author:  Andrew Mortensen
              <mailto:amortensen@arbor.net>

     Author:  Nik Teague
              <mailto:nteague@verisign.com>";

  description
    "This module contains YANG definition for configuring
     identifiers for resources and filtering rules using DOTS
     data channel.

     Copyright (c) 2017 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 2017-12-18 {
       description
         "Initial revision.";
       reference
         "RFC XXXX: Distributed Denial-of-Service Open Threat
                    Signaling (DOTS) Data Channel";
     }

     container identifier {
       description "Top level container for identifiers";

       leaf-list client-identifier {
         type binary;
         description
           "A client identifier conveyed by a
            server-side DOTS gateway to a remote DOTS server.";
         reference
           "I-D.itef-dots-signal-channel: Distributed Denial-of-Service
                         Open Threat Signaling (DOTS) Signal Channel";
       }

       list alias {
         key alias-name;
         description
           "List of identifiers";

         leaf alias-name {
           type string;
           description "alias name";
         }

         leaf-list target-prefix {
           type inet:ip-prefix;
           description
             "IPv4 or IPv6 prefix identifying the target.";
         }

         list target-port-range {
```

```
          key "lower-port upper-port";

          description
            "Port range. When only lower-port is
             present, it represents a single port.";

          leaf lower-port {
            type inet:port-number;
            mandatory true;
            description
              "Lower port number.";
          }

          leaf upper-port {
            type inet:port-number;
            must ". >= ../lower-port" {
               error-message
                 "The upper port number must be greater than
                  or equal to lower port number.";
            }
            description
              "Upper port number.";
          }
        }

        leaf-list target-protocol {
          type uint8;
          description
            "Identifies the target protocol number.

             The value '0' means 'all protocols'.

             Values are taken from the IANA protocol registry:
             https://www.iana.org/assignments/protocol-numbers/
             protocol-numbers.xhtml

             For example, 6 for a TCP or 17 for UDP.";
        }

        leaf-list target-fqdn {
          type inet:domain-name;
          description
            "FQDN identifying the target.";
        }

        leaf-list target-uri {
          type inet:uri;
          description
```

```
            "URI identifying the target.";
        }
      }
    }

    augment "/ietf-acl:access-lists" {
      description
        "client-identifier parameter.";

      leaf-list client-identifier {
        type binary;
        description
          "A client identifier conveyed by a server-side DOTS
          gateway to a remote DOTS server.";
      }
    }

    augment "/ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces" +
            "/ietf-acl:ace/ietf-acl:actions" {
      description
        "rate-limit action";
      leaf rate-limit {
        when "/ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces/" +
             "ietf-acl:ace/ietf-acl:actions/" +
             "ietf-acl:forwarding = 'ietf-acl:accept'" {
        description
          "rate-limit valid only when accept action is used";
        }
        type decimal64 {
          fraction-digits 2;
        }
       description
         "rate-limit traffic";
       }
    }

    augment "/ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces" +
            "/ietf-acl:ace/ietf-acl:matches/ietf-acl:ipv4-acl" {
      description
        "Handle non-initial and initial fragments for IPv4 packets.";

      leaf fragments {
        type empty;
        description
          "Handle fragments.";
      }
    }
```

```
    augment "/ietf-acl:access-lists/ietf-acl:acl/ietf-acl:aces" +
            "/ietf-acl:ace/ietf-acl:matches/ietf-acl:ipv6-acl" {
      description
        "Handle non-initial and initial fragments for IPv6 packets.";

      leaf fragments {
        type empty;
        description
          "Handle fragments.";
      }
    }

    augment "/ietf-acl:access-lists" {
      description
        "Handle ordering of ACLs from a DOTS client";

      container dots-acl-order {
        description
          "Enclosing container for ordering
           the ACLs from a DOTS client";

         list acl-set {
          key "set-name type";
          ordered-by user;
          description
            "List of ACLs";

          leaf set-name {
            type leafref {
              path "/ietf-acl:access-lists/ietf-acl:acl" +
                   "/ietf-acl:acl-name";
            }
            description
              "Reference to the ACL set name";
          }
           leaf type {
            type leafref {
              path "/ietf-acl:access-lists/ietf-acl:acl" +
                   "/ietf-acl:acl-type";
            }
            description
              "Reference to the ACL set type";
          }
        }
      }
    }
  }
   <CODE ENDS>
```

6. DOTS Identifiers

6.1. Create Identifiers

A POST request is used to create identifiers, such as names or aliases, for resources for which a mitigation may be requested. Such identifiers may be used in subsequent DOTS signal channel exchanges to refer more efficiently to the resources under attack (Figure 3).

DOTS clients within the same domain can create different aliases for the same resource.

```
 POST /restconf/data/ietf-dots-data-channel HTTP/1.1
 Host: {host}:{port}
 Content-Type: "application/yang-data+json"
 {
  "ietf-dots-data-channel:identifier": {
    "client-identifier": [
         "string"
    ],
    "alias": [
      {
        "alias-name": "string",
        "target-prefix": [
          "string"
        ],
        "target-port-range": [
          {
            "lower-port": integer,
            "upper-port": integer
          }
        ],
        "target-protocol": [
          integer
        ],
        "target-fqdn": [
          "string"
        ],
        "target-uri": [
          "string"
        ]
      }
    ]
  }
}
```

Figure 3: POST to create identifiers

The parameters are described below:

client-identifier: This attribute has the same meaning, syntax, and processing rules as the 'client-identifier' attribute defined in [I-D.ietf-dots-signal-channel].

This is an optional attribute.

alias-name: Name of the alias.

This is a mandatory attribute.

target-prefix: Prefixes are separated by commas. Prefixes are represented using Classless Inter-domain Routing (CIDR) notation [RFC4632]. As a reminder, the prefix length must be less than or equal to 32 (resp. 128) for IPv4 (resp. IPv6).

This is an optional attribute.

target-port-range: A range of port numbers.

The port range is defined by two bounds, a lower port number (lower-port) and an upper port number (upper-port).

When only 'lower-port' is present, it represents a single port number.

For TCP, UDP, Stream Control Transmission Protocol (SCTP) [RFC4960], or Datagram Congestion Control Protocol (DCCP) [RFC4340], the range of port numbers can be, for example, 1024-65535.

This is an optional attribute.

target-protocol: A list of protocols. Values are taken from the IANA protocol registry [proto_numbers].

The value '0' has a special meaning for 'all protocols'.

This is an optional attribute.

target-fqdn: A list of Fully Qualified Domain Names (FQDNs). An FQDN is the full name of a resource, rather than just its hostname. For example, "venera" is a hostname, and "venera.isi.edu" is an FQDN.

This is an optional attribute.

target-uri: A list of Uniform Resource Identifiers (URIs) [RFC3986].

This is an optional attribute.

In the POST request at least one of the attributes 'target-prefix' or 'target-fqdn' or 'target-uri' MUST be present. DOTS agents can safely ignore Vendor-Specific parameters they don't understand.

Figure 4 shows a POST request to create alias called "https1" for HTTPS servers with IP addresses 2001:db8:6401::1 and 2001:db8:6401::2 listening on port 443.

```
POST /restconf/data/ietf-dots-data-channel HTTP/1.1
Host: www.example.com
Content-Type: "application/yang-data+json"
{
  "ietf-dots-data-channel:identifier": {
    "alias": [
      {
        "alias-name": "https1",
        "target-protocol": [
          6
        ],
        "target-prefix": [
          "2001:db8:6401::1/128",
          "2001:db8:6401::2/128"
        ],
        "target-port-range": [
          {
            "lower-port": 443
          }
        ]
      }
    ]
  }
}
```

Figure 4: POST to create identifiers

The DOTS server indicates the result of processing the POST request using status-line codes. Status codes in the range "2xx" codes are success, "4xx" codes are some sort of invalid requests and "5xx" codes are returned if the DOTS server has erred or it is incapable of accepting the alias.

"201 Created" status-line is returned in the response if the DOTS server has accepted the alias.

If the request is missing one or more mandatory attributes or if the request contains invalid or unknown parameters, then "400 Bad Request" status-line MUST be returned in the response. The HTTP response will include the JSON body received in the request.

A DOTS client MAY use the PUT request (Section 4.5 in [RFC8040]) to create or modify the aliases in the DOTS server.

6.2. Retrieve Installed Identifiers

A GET request is used to retrieve one or all installed identifiers by a DOTS client from a DOTS server (Section 3.3.1 in [RFC8040]). If no 'alias-name' parameter is included in the request, this is an indication the request is about retrieving all identifiers instantiated by the DOTS client.

Figure 5 shows an example to retrieve all the identifiers that were instantiated by the DOTS client. The content parameter and its permitted values are defined in Section 4.8.1 of [RFC8040].

```
GET /restconf/data/ietf-dots-data-channel:identifier\
    /client-identifier=dz6pHjaADkaFTbjr0JGBpw?\
    content=config HTTP/1.1
Host: {host}:{port}
Accept: application/yang-data+json
```

Figure 5: GET to retrieve all the installed identifiers

Figure 6 shows an example of response message body that includes all the identifiers that are maintained by the DOTS server for DOTS client identified by the 'client-identifier' parameter.

```
{
 "ietf-dots-data-channel:identifier": {
    "client-identifier": [
       "dz6pHjaADkaFTbjr0JGBpw"
    ],
    "alias": [
      {
        "alias-name": "Server1",
        "traffic-protocol": [
          6
        ],
        "target-prefix": [
          "2001:db8:6401::1/128",
          "2001:db8:6401::2/128"
        ],
        "target-port-range": [
          {
            "lower-port": 443
          }
        ]
      },
      {
        "alias-name": "Server2",
        "target-protocol": [
          6
        ],
        "target-prefix": [
          "2001:db8:6401::10/128",
          "2001:db8:6401::20/128"
        ],
        "target-port-range": [
          {
            "lower-port": 80
          }
        ]
      }
    ]
  }
}
```


Figure 6: Response body

If 'alias-name' parameter is included in the request, but the DOTS server does not find that alias name in its configuration data, it MUST respond with a "404 Not Found" status-line.

6.3. Delete Identifiers

A DELETE request is used to delete identifiers maintained by a DOTS server.

In RESTCONF, URI-encoded path expressions are used. A RESTCONF data resource identifier is encoded from left to right, starting with the top-level data node, according to the 'api-path' rule defined in Section 3.5.3.1 of [RFC8040]. The data node in the path expression is a YANG list node and MUST be encoded according to the rules defined in Section 3.5.1 of [RFC8040].

If the DOTS server does not find the alias name conveyed in the DELETE request in its configuration data, it MUST respond with a "404 Not Found" status-line.

The DOTS server successfully acknowledges a DOTS client's request to remove the identifier using "204 No Content" status-line in the response.

Figure 7 shows an example of a request to delete an alias.

```
  DELETE /restconf/data/ietf-dots-data-channel:identifier\
         /client-identifier=dz6pHjaADkaFTbjr0JGBpw,\
         iAYmCNPmrYoKoqzgFMiobw/alias-name=Server1 HTTP/1.1
  Host: {host}:{port}
```

Figure 7: DELETE an identifier

7. DOTS Filtering Rules

The DOTS server either receives the filtering rules directly from the DOTS client or via a DOTS gateway.

If the DOTS client signals the filtering rules via a DOTS gateway, the DOTS gateway validates first if the DOTS client is authorized to signal the filtering rules. If the client is authorized, it propagates the rules to the DOTS server. Likewise, the DOTS server validates if the DOTS gateway is authorized to signal the filtering rules. To create or purge filters, the DOTS client sends HTTP requests to its DOTS gateway. The DOTS gateway validates the rules in the requests and proxies the requests containing the filtering rules to a DOTS server. When the DOTS gateway receives the associated HTTP response from the DOTS server, it propagates the response back to the DOTS client.

The following sub-sections define means for a DOTS client to configure filtering rules on a DOTS server.

7.1. Install Filtering Rules

A POST request is used to push filtering rules to a DOTS server.

Figure 8 shows a POST request example to block traffic from 192.0.2.0/24, destined to 198.51.100.0/24. The ACL JSON configuration for the filtering rule is generated using the ACL YANG module (Section 4.3 of [I-D.ietf-netmod-acl-model]).

```
POST /restconf/data/ietf-dots-data-channel HTTP/1.1
Host: www.example.com
Content-Type: "application/yang-data+json"
{
 "ietf-dots-data-channel:access-lists": {
    "client-identifier": [
     "dz6pHjaADkaFTbjr0JGBpw"
    ],
    "acl": [
        {
             "acl-name": "sample-ipv4-acl",
             "acl-type": "ipv4-acl",
             "aces": {
                 "ace": [
                     {
                         "rule-name": "rule1",
                         "matches": {
                           "ipv4-acl": {
                             "source-ipv4-network": "192.0.2.0/24",
                             "destination-ipv4-network": "198.51.100.0/24"
                           }
                          },
                          "actions": {
                             "forwarding" : "drop"
                          }
                      }
                  ]
             }
        }
    ]
 }
}
```

Figure 8: POST to install filtering rules

The parameters defined in [I-D.ietf-netmod-acl-model] are discussed below:

acl-name: The name of access-list. This is a mandatory attribute.

acl-type: Indicates the primary intended type of match criteria (e.g., IPv4, IPv6). This is a mandatory attribute.

protocol: Internet Protocol numbers. This is an optional attribute.

source-ipv4-network: The source IPv4 prefix. This is an optional attribute.

destination-ipv4-network: The destination IPv4 prefix. This is an optional attribute.

actions: Actions in the forwarding ACL category can be "drop" or "accept" or "rate-limit". "accept" action is used to white-list traffic. "drop" action is used to black-list traffic. "rate-limit" action is used to rate-limit traffic, the allowed traffic rate is represented in bytes per second indicated in IEEE floating point format [IEEE.754.1985]. This is an optional attribute.

The DOTS server indicates the result of processing the POST request using status-line header. "2xx" codes are success, 4xx codes are some sort of invalid requests, and 5xx codes are returned if the DOTS server has erred or it is incapable of configuring the filtering rules. Concretely, "201 Created" status-line MUST be returned in the response if the DOTS server has accepted the filtering rules. If the request is missing one or more mandatory attributes or contains invalid or unknown parameters, then "400 Bad Request" status-line MUST be returned in the response.

If the request is conflicting with an existing filtering, the DOTS server returns "409 Conflict" status-line to the requesting DOTS client. The error-tag "invalid-value" is used in this case.

The "insert" query parameter discussed in Section 4.8.5 of [RFC8040] MAY be used to specify how a ACE is inserted within an ACL and how a ACL is inserted within an ACL list.

The DOTS client MAY use the PUT request to create or modify the filtering rules in the DOTS server.

7.2. Retrieve Installed Filtering Rules

The DOTS client periodically queries the DOTS server to check the counters for installed filtering rules. A GET request is used to retrieve filtering rules from a DOTS server.

If the DOTS server does not find the access list name and access list type conveyed in the GET request in its configuration data, it responds with a "404 Not Found" status-line.

Figure 9 shows how to retrieve all the filtering rules programmed by the DOTS client and the number of matches for the installed filtering rules.

```
  GET /restconf/data/ietf-dots-data-channel:access-lists\
      /client-identifier=dz6pHjaADkaFTbjr0JGBpw?\
      content=all HTTP/1.1
  Host: {host}:{port}
  Accept: application/yang-data+json
```

Figure 9: GET to retrieve the configuration data and state data for the filtering rules

7.3. Remove Filtering Rules

A DELETE request is used to delete filtering rules from a DOTS server.

If the DOTS server does not find the access list name and access list type conveyed in the DELETE request in its configuration data, then it responds with a "404 Not Found" status-line. The DOTS server successfully acknowledges a DOTS client's request to withdraw the filtering rules using "204 No Content" status-line, and removes the filtering rules as soon as possible.

Figure 10 shows an example of a request to remove the IPv4 ACL named "sample-ipv4-acl". This request is being relayed by a DOTS gateway as hinted by the presence of the 'client-identifier' parameter.

```
  DELETE /restconf/data/ietf-dots-data-channel:access-lists\
         /client-identifier=dz6pHjaADkaFTbjr0JGBpw\
         /acl-name=sample-ipv4-acl&\
         acl-type=ipv4-acl HTTP/1.1
  Host: {host}:{port}
```

Figure 10: DELETE to remove the filtering rules

8. IANA Considerations

8.1. DOTS Data Channel JSON Attribute Mappings Registry

The document requests IANA to create a new registry, entitled "DOTS Data Channel JSON Attribute Mappings Registry". The structure of this registry is provided in Section 8.1.1.

The registry is initially populated with the values in Section 8.1.2.

Values from that registry MUST be assigned via Expert Review [RFC8126].

8.1.1. Registration Template

JSON Attribute:
JSON attribute name.

Description:
Brief description of the attribute.

Change Controller:
For Standards Track RFCs, list the "IESG". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.

Specification Document(s):
Reference to the document or documents that specify the parameter, preferably including URIs that can be used to retrieve copies of the documents. An indication of the relevant sections may also be included but is not required.

8.1.2. Initial Registry Contents

- JSON Attribute: "client-identifier"
- Description: Client identifier.
- Change Controller: IESG
- Specification Document(s): this document

- JSON Attribute: "alias-name"
- Description: Name of alias.
- Change Controller: IESG
- Specification Document(s): this document

- JSON Attribute: "target-protocol"
- Description: Internet protocol numbers.
- Change Controller: IESG
- Specification Document(s): this document

- JSON Attribute: "target-port-range"
- Description: The port range, lower-port for lower port number and upper-port for upper port number. For TCP, UDP, SCTP, or DCCP: a range of ports can be, e.g., 80 to 8080.
- Change Controller: IESG
- Specification Document(s): this document

- o JSON Attribute: "lower-port"
- o Description: Lower port number for the port range.
- o Change Controller: IESG
- o Specification Document(s): this document

- o JSON Attribute: "upper-port"
- o Description: Upper port number for the port range.
- o Change Controller: IESG
- o Specification Document(s): this document

- o JSON Attribute: "target-prefix"
- o Description: IP prefix
- o Change Controller: IESG
- o Specification Document(s): this document

- o JSON Attribute: "target-fqdn"
- o Description: Fully Qualified Domain Name, is the full name of a system, rather than just its hostname. For example, "venera" is a hostname, and "venera.isi.edu" is an FQDN.
- o Change Controller: IESG
- o Specification Document(s): this document

- o JSON Attribute: "target-uri"
- o Description: Uniform Resource Identifier (URI).
- o Change Controller: IESG
- o Specification Document(s): this document

8.2. 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-data-channel
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-data-channel
namespace: urn:ietf:params:xml:ns:yang:ietf-dots-data-channel
prefix: data-channel
reference: RFC XXXX
```

9. Contributors

The following individuals have contributed to this document:

Dan Wing

Email: dwing-ietf@fuggles.com

10. Security Considerations

RESTCONF security considerations are discussed in [RFC8040]. In particular, DOTS agents MUST follow the security recommendations in Sections 2 and 12 of [RFC8040] and the mutual authentication TLS profile discussed in Section 7.1 of [I-D.ietf-dots-signal-channel].

Authenticated encryption MUST be used for data confidentiality and message integrity. The interaction between the DOTS agents requires Transport Layer Security (TLS) with a cipher suite offering confidentiality protection and the guidance given in [RFC7525] MUST be followed to avoid attacks on TLS.

An attacker may be able to inject RST packets, bogus application segments, etc., regardless of whether TLS authentication is used. Because the application data is TLS protected, this will not result in the application receiving bogus data, but it will constitute a DoS on the connection. This attack can be countered by using TCP-AO [RFC5925]. If TCP-AO is used, then any bogus packets injected by an attacker will be rejected by the TCP-AO integrity check and therefore will never reach the TLS layer.

In order to prevent leaking internal information outside a client-domain, client-side DOTS gateways SHOULD NOT reveal the identity of internal DOTS clients (client-identifier) unless explicitly configured to do so.

Special care should be taken in order to ensure that the activation of the proposed mechanism won't have an impact on the stability of the network (including connectivity and services delivered over that network).

All data nodes defined in the YANG module which can be created, modified, and deleted (i.e., config true, which is the default) are considered sensitive. Write operations applied to these data nodes without proper protection can negatively affect network operations. Appropriate security measures are recommended to prevent illegitimate users from invoking DOTS data channel primitives. Nevertheless, an attacker who is able to access to a DOTS client can undertake various attacks, such as:

- o Set an arbitrarily low rate-limit that may lead to discarding legitimate traffic to be forwarded (rate-limit).
- o Set an arbitrarily high rate-limit that may lead to allowing illegitimate DDoS traffic to be forwarded (rate-limit).
- o Communicate invalid aliases to the server (alias) that will lead to failure to associate both data and signal channels.
- o Set invalid ACL entries that may lead to discard legitimate traffic from being forwarding. Likewise, invalid ACL entries may lead to forward DDoS traffic.

11. Acknowledgements

Thanks to Christian Jacquenet, Roland Dobbins, Roman Danyliw, Ehud Doron, Russ White, Jon Shallow, Gilbert Clark, and Nesredien Suleiman for the discussion and comments.

12. References

12.1. Normative References

[I-D.ietf-dots-architecture]
Mortensen, A., Andreasen, F., Reddy, T., christopher_gray3@cable.comcast.com, c., Compton, R., and N. Teague, "Distributed-Denial-of-Service Open Threat Signaling (DOTS) Architecture", draft-ietf-dots-architecture-05 (work in progress), October 2017.

[I-D.ietf-dots-signal-channel]
Reddy, T., Boucadair, M., Patil, P., Mortensen, A., and N. Teague, "Distributed Denial-of-Service Open Threat Signaling (DOTS) Signal Channel", draft-ietf-dots-signal-channel-13 (work in progress), December 2017.

[I-D.ietf-netmod-acl-model]
Jethanandani, M., Huang, L., Agarwal, S., and D. Blair, "Network Access Control List (ACL) YANG Data Model", draft-ietf-netmod-acl-model-14 (work in progress), October 2017.

[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>.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, DOI 10.17487/RFC4632, August 2006, <https://www.rfc-editor.org/info/rfc4632>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/info/rfc5925>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[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>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

12.2. Informative References

[I-D.ietf-dots-requirements]
Mortensen, A., Moskowitz, R., and T. Reddy, "Distributed Denial of Service (DDoS) Open Threat Signaling Requirements", draft-ietf-dots-requirements-08 (work in progress), December 2017.

[I-D.ietf-netmod-yang-tree-diagrams]
Bjorklund, M. and L. Berger, "YANG Tree Diagrams", draft-ietf-netmod-yang-tree-diagrams-02 (work in progress), October 2017.

[IEEE.754.1985]
Institute of Electrical and Electronics Engineers, "Standard for Binary Floating-Point Arithmetic", August 1985.

[proto_numbers]
"IANA, "Protocol Numbers"", 2011, <http://www.iana.org/assignments/protocol-numbers>.

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

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <https://www.rfc-editor.org/info/rfc4340>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, DOI 10.17487/RFC5389, October 2008, <https://www.rfc-editor.org/info/rfc5389>.

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

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6520] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", RFC 6520, DOI 10.17487/RFC6520, February 2012, <https://www.rfc-editor.org/info/rfc6520>.

[RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <https://www.rfc-editor.org/info/rfc6887>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <https://www.rfc-editor.org/info/rfc7159>.

[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>.

Authors' Addresses

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

Email: kondtir@gmail.com

Mohamed Boucadair (editor)
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

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

Email: kaname@nttv6.jp

Liang Xia
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, Jiangsu 210012
China

Email: frank.xialiang@huawei.com

Prashanth Patil
Cisco Systems, Inc.

Email: praspati@cisco.com

Andrew Mortensen
Arbor Networks, Inc.
2727 S. State St
Ann Arbor, MI 48104
United States

Email: amortensen@arbor.net

Nik Teague
Verisign, Inc.
United States

Email: nteague@verisign.com