

DOTS
Internet-Draft
Updates: [8782](#) (if approved)
Intended status: Standards Track
Expires: January 7, 2021

M. Boucadair
Orange
J. Shallow
July 6, 2020

A YANG Data Model for Distributed Denial-of-Service Open Threat
Signaling (DOTS) Signal Channel
draft-boucadair-dots-rfc8782-yang-update-00

Abstract

This document specifies an updated version of the Distributed Denial-of-Service Open Threat Signaling (DOTS) Signal Channel YANG data model. This updated version makes use of the new mechanisms for defining abstract data structures with YANG as specified in [RFC8791](#).

This document updates [RFC 8782](#).

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 January 7, 2021.

Copyright Notice

Copyright (c) 2020 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 | 2 |
| 2. | Summary of Changes From RFC8782 | 2 |
| 3. | Tree Structure | 3 |
| 4. | YANG Module | 6 |
| 5. | Security Considerations | 18 |
| 6. | IANA Considerations | 19 |
| 7. | Acknowledgements | 19 |
| 8. | References | 19 |
| 8.1. | Normative References | 19 |
| 8.2. | Informative References | 20 |
| | Authors' Addresses | 20 |

[1.](#) Introduction

As specified in [[RFC8782](#)], messages exchanged between DOTS agents are serialized using Concise Binary Object Representation (CBOR) [[RFC7252](#)]. CBOR-encoded payloads are used to carry signal channel-specific payload messages which convey request parameters and response information such as errors.

This document specifies a YANG module [[RFC7950](#)] for representing DOTS mitigation scopes, DOTS signal channel session configuration data, and DOTS redirected signaling. All parameters in the payload of the DOTS signal channel are mapped to CBOR types as specified in Table 5 of [[RFC8782](#)].

This YANG module is not intended to be used via NETCONF/RESTCONF for DOTS server management purposes; such a module is out of the scope of this document. It serves only to provide abstract data structures. This document uses the "structure" extension specified in [[RFC8791](#)].

The meaning of the symbols in YANG tree diagrams is defined in [[RFC8340](#)] and [[RFC8791](#)].

[2.](#) Summary of Changes From [RFC8782](#)

The main changes compared to the YANG version published in [[RFC8782](#)]

are as follows:

- o Follow the new YANG data structure specified in [\[RFC8791\]](#).

- o Add in "choice" to indicate the communication direction in which a data node applies. If no "choice" is indicated, a node can appear in both directions (i.e., from DOTS clients to DOTS servers and vice versa).
- o Remove "config" clauses. Note that "config" statements will be ignored (if present) anyway according to [Section 4 of \[RFC8791\]](#).
- o Remove "cuid", "cdid", and "sid" nodes from the structure because these nodes are included as Uri-Path options, not within the message body.
- o Remove the list keys for the mitigation scope message type (i.e., "cuid" and "mid"). "mid" is not indicated as a key because it is included as Uri-Path option for requests and in the message body for responses. Note that [Section 4 of \[RFC8791\]](#) specifies that a list does not require to have a key statement defined.

These changes are made with the constraint to avoid changes to the mapping table defined in Table 5 of [\[RFC8782\]](#). A DOTS signal channel attribute that may be present in both requests and responses will thus have the same CBOR key value and CBOR major type.

[3.](#) Tree Structure

This document defines the YANG module "ietf-dots-signal-channel", which has the following tree structure. A DOTS signal message can be a mitigation, a configuration, a redirect, or a heartbeat message. The use of these attributes is specified in [\[RFC8782\]](#).

This tree structure obsoletes the one described in [Section 5.1 of \[RFC8782\]](#).

```
module: ietf-dots-signal-channel
```

```
    structure dots-signal:
```

```

+-- (message-type)?
+--:(mitigation-scope)
|   +-- scope* []
|   |   +-- target-prefix*          inet:ip-prefix
|   |   +-- target-port-range* [lower-port]
|   |   |   +-- lower-port      inet:port-number
|   |   |   +-- upper-port?    inet:port-number
|   |   +-- target-protocol*      uint8
|   |   +-- target-fqdn*          inet:domain-name
|   |   +-- target-uri*           inet:uri
|   |   +-- alias-name*           string
|   |   +-- lifetime?             int32

```

```

|   +-- trigger-mitigation?          boolean
|   +-- (direction)?
|   +--:(server-to-client-only)
|   |   +-- mid?                    uint32
|   |   +-- mitigation-start?       uint64
|   |   +-- status?                 iana-signal:status
|   |   +-- conflict-information
|   |   |   +-- conflict-status?
|   |   |   |   iana-signal:conflict-status
|   |   |   +-- conflict-cause?
|   |   |   |   iana-signal:conflict-cause
|   |   +-- retry-timer?            uint32
|   |   +-- conflict-scope
|   |   |   +-- target-prefix*      inet:ip-prefix
|   |   |   +-- target-port-range* [lower-port]
|   |   |   |   +-- lower-port      inet:port-number
|   |   |   |   +-- upper-port?    inet:port-number
|   |   |   +-- target-protocol*    uint8
|   |   |   +-- target-fqdn*        inet:domain-name
|   |   |   +-- target-uri*         inet:uri
|   |   |   +-- alias-name*         string
|   |   |   +-- acl-list* [acl-name]
|   |   |   |   +-- acl-name        leafref
|   |   |   |   +-- acl-type?      leafref
|   |   |   +-- mid?                uint32
|   |   +-- bytes-dropped?
|   |   |   yang:zero-based-counter64
|   |   +-- bps-dropped?            yang:gauge64
|   |   +-- pkts-dropped?

```

```

|           |           yang:zero-based-counter64
|           +--- pps-dropped?           yang:gauge64
|           +--- attack-status?
|           iana-signal:attack-status
+---:(signal-config)
|   +--- mitigating-config
|   |   +--- heartbeat-interval
|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value?   uint16
|   |   |   |   |   +--- min-value?   uint16
|   |   |   |   |   +--- current-value?   uint16
|   |   +--- missing-hb-allowed
|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value?   uint16
|   |   |   |   |   +--- min-value?   uint16
|   |   |   |   |   +--- current-value?   uint16
|   |   +--- probing-rate

```

```

|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value?   uint16
|   |   |   |   |   +--- min-value?   uint16
|   |   |   |   |   +--- current-value?   uint16
|   |   +--- max-retransmit
|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value?   uint16
|   |   |   |   |   +--- min-value?   uint16
|   |   |   |   |   +--- current-value?   uint16
|   |   +--- ack-timeout
|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value-decimal?   decimal64
|   |   |   |   |   +--- min-value-decimal?   decimal64
|   |   |   |   |   +--- current-value-decimal?   decimal64
|   |   +--- ack-random-factor
|   |   |   +--- (direction)?
|   |   |   |   +---:(server-to-client-only)
|   |   |   |   |   +--- max-value-decimal?   decimal64
|   |   |   |   |   +--- min-value-decimal?   decimal64

```

```

| | +-- current-value-decimal?    decimal64
| +-- idle-config
| | +-- heartbeat-interval
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value?    uint16
| | | | +-- min-value?    uint16
| | | | +-- current-value?    uint16
| | +-- missing-hb-allowed
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value?    uint16
| | | | +-- min-value?    uint16
| | | | +-- current-value?    uint16
| | +-- probing-rate
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value?    uint16
| | | | +-- min-value?    uint16
| | | | +-- current-value?    uint16
| | +-- max-retransmit
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value?    uint16
| | | | +-- min-value?    uint16
| | | | +-- current-value?    uint16

```

```

| | +-- ack-timeout
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value-decimal?    decimal64
| | | | +-- min-value-decimal?    decimal64
| | | | +-- current-value-decimal?    decimal64
| | +-- ack-random-factor
| | | +-- (direction)?
| | | | +--:(server-to-client-only)
| | | | +-- max-value-decimal?    decimal64
| | | | +-- min-value-decimal?    decimal64
| | | | +-- current-value-decimal?    decimal64
| +--:(redirected-signal)
| | +-- (direction)?
| | | +--:(server-to-client-only)

```

| | | |
|-----------------|------------------------|-----------------|
| | +++ alt-server | string |
| | +++ alt-server-record* | inet:ip-address |
| +--:(heartbeat) | | |
| | +++ peer-hb-status | boolean |

4. YANG Module

This module uses the common YANG types defined in [\[RFC6991\]](#) and types defined in [\[RFC8783\]](#).

This version obsoletes the version described in [Section 5.3 of \[RFC8782\]](#).

```
<CODE BEGINS> file "ietf-dots-signal-channel@2020-07-02.yang"
module ietf-dots-signal-channel {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-dots-signal-channel";
  prefix signal;

  import ietf-inet-types {
    prefix inet;
    reference
      "Section 4 of RFC 6991";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "Section 3 of RFC 6991";
  }
  import ietf-dots-data-channel {
    prefix ietf-data;
    reference
```

```
      "RFC 8783: Distributed Denial-of-Service Open Threat Signaling
        (DOTS) Data Channel Specification";
  }
  import iana-dots-signal-channel {
    prefix iana-signal;
    reference
      "RFC 8782: Distributed Denial-of-Service Open Threat Signaling
        (DOTS) Signal Channel Specification";
```

```

}
import ietf-yang-structure-ext {
  prefix sx;
  reference
    "RFC 8791: YANG Data Structure Extensions";
}

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

  Editor:   Jon Shallow
            <mailto:supjps-ietf@jpshallow.com>

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

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

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

  Author:   Nik Teague
            <mailto:nteague@ironmountain.co.uk>";
description
  "This module contains YANG definition for the signaling
  messages exchanged between a DOTS client and a DOTS server.

  Copyright (c) 2020 IETF Trust and the persons identified as
  authors of the code. All rights reserved.

  Redistribution and use in source and binary forms, with or
  without modification, is permitted pursuant to, and subject
  to the license terms contained in, the Simplified BSD License

```


Relating to IETF Documents
(<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of [RFC 8782](#); see the RFC itself for full legal notices.";

```
revision 2020-07-02 {
  description
    "Updated revision to comply with RFC8791.";
  reference
    "RFC xxxx: A YANG Data Model for Distributed Denial-of-Service
      Open Threat Signaling (DOTS) Signal Channel";
}
revision 2020-05-28 {
  description
    "Initial revision.";
  reference
    "RFC 8782: Distributed Denial-of-Service Open Threat
      Signaling (DOTS) Signal Channel Specification";
}

/*
 * Groupings
 */

grouping mitigation-scope {
  description
    "Specifies the scope of the mitigation request.";
  list scope {
    description
      "The scope of the request.";
    uses ietf-data:target;
    leaf-list alias-name {
      type string;
      description
        "An alias name that points to a resource.";
    }
  }
  leaf lifetime {
    type int32;
    units "seconds";
    default "3600";
    description
      "Indicates the lifetime of the mitigation request.

      A lifetime of '0' in a mitigation request is an
      invalid value."
```

```
        A lifetime of negative one (-1) indicates indefinite
        lifetime for the mitigation request.";
    }
    leaf trigger-mitigation {
        type boolean;
        default "true";
        description
            "If set to 'false', DDoS mitigation will not be
            triggered unless the DOTS signal channel
            session is lost.";
    }
    choice direction {
        description
            "Indicates the communication direction in which the
            nodes can be included.";
        case server-to-client-only {
            description
                "These nodes appear only in a mitigation message
                sent from the server to the client.";
            leaf mid {
                type uint32;
                description
                    "Mitigation request identifier.

                    This identifier must be unique for each mitigation
                    request bound to the DOTS client.";
            }
        }
        leaf mitigation-start {
            type uint64;
            description
                "Mitigation start time is represented in seconds
                relative to 1970-01-01T00:00:00Z in UTC time.";
        }
        leaf status {
            type iana-signal:status;
            description
                "Indicates the status of a mitigation request.
                It must be included in responses only.";
        }
    }
    container conflict-information {
        description
            "Indicates that a conflict is detected.
            Must only be used for responses.";
        leaf conflict-status {
            type iana-signal:conflict-status;
            description
```

```
    "Indicates the conflict status.";
}
```

```
leaf conflict-cause {
    type iana-signal:conflict-cause;
    description
        "Indicates the cause of the conflict.";
}
leaf retry-timer {
    type uint32;
    units "seconds";
    description
        "The DOTS client must not resend the
        same request that has a conflict before the expiry of
        this timer.";
}
container conflict-scope {
    description
        "Provides more information about the conflict scope.";
    uses ietf-data:target {
        when "/dots-signal/scope/conflict-information/"
            + "conflict-cause = 'overlapping-targets'";
    }
    leaf-list alias-name {
        when "../..conflict-cause = 'overlapping-targets'";
        type string;
        description
            "Conflicting alias-name.";
    }
    list acl-list {
        when "../..conflict-cause ="
            + " 'conflict-with-acceptlist'";
        key "acl-name";
        description
            "List of conflicting ACLs as defined in the DOTS data
            channel. These ACLs are uniquely defined by
            cuid and acl-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 conflicting ACL name bound to
            a DOTS client.";
    }
    leaf acl-type {
        type leafref {
            path "/ietf-data:dots-data/ietf-data:dots-client/"
                + "ietf-data:acls/ietf-data:acl/ietf-data:type";
        }
    }

```

```

        description
            "Reference to the conflicting ACL type bound to
            a DOTS client.";
    }
}
leaf mid {
    when "../..//conflict-cause = 'overlapping-targets'";
    type uint32;
    description
        "Reference to the conflicting 'mid' bound to
        the same DOTS client.";
}
}
}
leaf bytes-dropped {
    type yang:zero-based-counter64;
    units "bytes";
    description
        "The total dropped byte count for the mitigation
        request since the attack mitigation was triggered.
        The count wraps around when it reaches the maximum value
        of counter64 for dropped bytes.";
}
leaf bps-dropped {
    type yang:gauge64;
    description
        "The average number of dropped bits per second for
        the mitigation request since the attack
        mitigation was triggered. This should be over
        five-minute intervals (that is, measuring bytes
        into five-minute buckets and then averaging these
        buckets over the time since the mitigation was

```

```

        triggered).";
    }
    leaf pkts-dropped {
        type yang:zero-based-counter64;
        description
            "The total number of dropped packet count for the
            mitigation request since the attack mitigation was
            triggered. The count wraps around when it reaches
            the maximum value of counter64 for dropped packets.";
    }
    leaf pps-dropped {
        type yang:gauge64;
        description
            "The average number of dropped packets per second
            for the mitigation request since the attack
            mitigation was triggered. This should be over

```

```

        five-minute intervals (that is, measuring packets
        into five-minute buckets and then averaging these
        buckets over the time since the mitigation was
        triggered).";
    }
    leaf attack-status {
        type iana-signal:attack-status;
        description
            "Indicates the status of an attack as seen by the
            DOTS client.";
    }
}
}
}
}

grouping config-parameters {
    description
        "Subset of DOTS signal channel session configuration.";
    container heartbeat-interval {
        description
            "DOTS agents regularly send heartbeats to each other
            after mutual authentication is successfully
            completed in order to keep the DOTS signal channel
            open.";
    }
}

```

```

choice direction {
  description
    "Indicates the communication direction in which the
    nodes can be included.";
  case server-to-client-only {
    description
      "These nodes appear only in a mitigation message
      sent from the server to the client.";
    leaf max-value {
      type uint16;
      units "seconds";
      description
        "Maximum acceptable heartbeat-interval value.";
    }
    leaf min-value {
      type uint16;
      units "seconds";
      description
        "Minimum acceptable heartbeat-interval value.";
    }
  }
}
leaf current-value {

```

```

    type uint16;
    units "seconds";
    default "30";
    description
      "Current heartbeat-interval value.

      '0' means that heartbeat mechanism is deactivated.";
  }
}
container missing-hb-allowed {
  description
    "Maximum number of missing heartbeats allowed.";
  choice direction {
    description
      "Indicates the communication direction in which the
      nodes can be included.";
    case server-to-client-only {
      description

```

```

        "These nodes appear only in a mitigation message
        sent from the server to the client.";
    leaf max-value {
        type uint16;
        description
            "Maximum acceptable missing-hb-allowed value.";
    }
    leaf min-value {
        type uint16;
        description
            "Minimum acceptable missing-hb-allowed value.";
    }
}
leaf current-value {
    type uint16;
    default "15";
    description
        "Current missing-hb-allowed value.";
}
}
container probing-rate {
    description
        "The limit for sending Non-confirmable messages with
        no response.";
    choice direction {
        description
            "Indicates the communication direction in which the
            nodes can be included.";
        case server-to-client-only {

```

```

    description
        "These nodes appear only in a mitigation message
        sent from the server to the client.";
    leaf max-value {
        type uint16;
        units "byte/second";
        description
            "Maximum acceptable probing-rate value.";
    }
    leaf min-value {
        type uint16;

```

```

        units "byte/second";
        description
            "Minimum acceptable probing-rate value.";
    }
}
leaf current-value {
    type uint16;
    units "byte/second";
    default "5";
    description
        "Current probing-rate value.";
}
}
container max-retransmit {
    description
        "Maximum number of retransmissions of a Confirmable
        message.";
    choice direction {
        description
            "Indicates the communication direction in which the
            nodes can be included.";
        case server-to-client-only {
            description
                "These nodes appear only in a mitigation message
                sent from the server to the client.";
            leaf max-value {
                type uint16;
                description
                    "Maximum acceptable max-retransmit value.";
            }
            leaf min-value {
                type uint16;
                description
                    "Minimum acceptable max-retransmit value.";
            }
        }
    }
}

```

```

}
leaf current-value {
    type uint16;
    default "3";
}

```



```

        description
            "Current max-retransmit value.";
    }
}
container ack-timeout {
    description
        "Initial retransmission timeout value.";
    choice direction {
        description
            "Indicates the communication direction in which the
            nodes can be included.";
        case server-to-client-only {
            description
                "These nodes appear only in a mitigation message
                sent from the server to the client.";
            leaf max-value-decimal {
                type decimal64 {
                    fraction-digits 2;
                }
                units "seconds";
                description
                    "Maximum ack-timeout value.";
            }
            leaf min-value-decimal {
                type decimal64 {
                    fraction-digits 2;
                }
                units "seconds";
                description
                    "Minimum ack-timeout value.";
            }
        }
    }
}
leaf current-value-decimal {
    type decimal64 {
        fraction-digits 2;
    }
    units "seconds";
    default "2";
    description
        "Current ack-timeout value.";
}
}
container ack-random-factor {

```

```
description
  "Random factor used to influence the timing of
  retransmissions.";
choice direction {
  description
    "Indicates the communication direction in which the
    nodes can be included.";
  case server-to-client-only {
    description
      "These nodes appear only in a mitigation message
      sent from the server to the client.";
    leaf max-value-decimal {
      type decimal64 {
        fraction-digits 2;
      }
      description
        "Maximum acceptable ack-random-factor value.";
    }
    leaf min-value-decimal {
      type decimal64 {
        fraction-digits 2;
      }
      description
        "Minimum acceptable ack-random-factor value.";
    }
  }
}
leaf current-value-decimal {
  type decimal64 {
    fraction-digits 2;
  }
  default "1.5";
  description
    "Current ack-random-factor value.";
}
}

grouping signal-config {
  description
    "DOTS signal channel session configuration.";
  container mitigating-config {
    description
      "Configuration parameters to use when a mitigation
      is active.";
    uses config-parameters;
  }
}
```

```
container idle-config {
```

```
    description
      "Configuration parameters to use when no mitigation
       is active.";
    uses config-parameters;
  }
}

grouping redirected-signal {
  description
    "Grouping for the redirected signaling.";
  choice direction {
    description
      "Indicates the communication direction in which the
       nodes can be included.";
    case server-to-client-only {
      description
        "These nodes appear only in a mitigation message
         sent from the server to the client.";
      leaf alt-server {
        type string;
        mandatory true;
        description
          "FQDN of an alternate server.";
      }
      leaf-list alt-server-record {
        type inet:ip-address;
        description
          "List of records for the alternate server.";
      }
    }
  }
}

/*
 * DOTS Signal Channel Structure
 */

sx:structure dots-signal {
  description
    "Main structure for DOTS signal message.
```

```

    A DOTS signal message can be a mitigation, a configuration,
    or a redirected signal message.";
choice message-type {
  description
    "Can be a mitigation, a configuration, or a redirect
    message.";
  case mitigation-scope {

```

```

    description
      "Mitigation scope of a mitigation message.";
    uses mitigation-scope;
    reference
      "Section 4.4 of RFC 8782";
  }
  case signal-config {
    description
      "Configuration message.";
    uses signal-config;
    reference
      "Section 4.5 of RFC 8782";
  }
  case redirected-signal {
    description
      "Redirected signaling.";
    uses redirected-signal;
    reference
      "Section 4.6 of RFC 8782";
  }
  case heartbeat {
    description
      "DOTS heartbeats.";
    leaf peer-hb-status {
      type boolean;
      mandatory true;
      description
        "Indicates whether a DOTS agent receives heartbeats
        from its peer. The value is set to 'true' if the
        DOTS agent is receiving heartbeat messages
        from its peer.";
    }
    reference

```

```
        "Section 4.7 of RFC 8782";
    }
}
}
}
<CODE ENDS>
```

5. Security Considerations

This document defines YANG data structures that are meant to be used as an abstract representation of DOTS signal channel messages. As such, this document does not introduce any new vulnerabilities beyond those specified [Section 10 of \[RFC8782\]](#).

Boucadair & Shallow

Expires January 7, 2021

[Page 18]

Internet-Draft

DOTS Signal Channel YANG Module

July 2020

6. IANA Considerations

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

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

This document requests IANA to register the following YANG modules in the "YANG Module Names" subregistry [\[RFC6020\]](#) within the "YANG Parameters" registry.

Name: ietf-dots-signal-channel
Namespace: urn:ietf:params:xml:ns:yang:ietf-dots-signal-channel
Maintained by IANA: N
Prefix: signal
Reference: RFC XXXX

7. Acknowledgements

Many thanks to Martin Bjoerklund for the suggestion to use [RFC8791](#).

The initial version of the DOTS signal channel YANG model was specified in [\[RFC8782\]](#) authored by Tirumaleswar Reddy.K, Mohamed Boucadair, Prashanth Patil, Andrew Mortensen, and Nik Teague.

[8.](#) References

[8.1.](#) Normative References

- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), DOI 10.17487/RFC3688, January 2004, <<https://www.rfc-editor.org/info/rfc3688>>.
- [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>>.
- [RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", [RFC 6991](#), DOI 10.17487/RFC6991, July 2013, <<https://www.rfc-editor.org/info/rfc6991>>.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", [RFC 7252](#), DOI 10.17487/RFC7252, June 2014, <<https://www.rfc-editor.org/info/rfc7252>>.

- [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>>.
- [RFC8782] Reddy.K, T., Ed., Boucadair, M., Ed., Patil, P., Mortensen, A., and N. Teague, "Distributed Denial-of-Service Open Threat Signaling (DOTS) Signal Channel Specification", [RFC 8782](#), DOI 10.17487/RFC8782, May 2020, <<https://www.rfc-editor.org/info/rfc8782>>.
- [RFC8783] Boucadair, M., Ed. and T. Reddy.K, Ed., "Distributed Denial-of-Service Open Threat Signaling (DOTS) Data Channel Specification", [RFC 8783](#), DOI 10.17487/RFC8783, May 2020, <<https://www.rfc-editor.org/info/rfc8783>>.
- [RFC8791] Bierman, A., Bjorklund, M., and K. Watsen, "YANG Data Structure Extensions", [RFC 8791](#), DOI 10.17487/RFC8791, June 2020, <<https://www.rfc-editor.org/info/rfc8791>>.

8.2. Informative References

[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

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Jon Shallow
United Kingdom

Email: supjps-ietf@jpshallow.com