```
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;
  }
  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:  Konda, Tirumaleswar Reddy.K
              ~~<mailto:TirumaleswarReddy_Konda&McAfee.com>~~
              <mailto:TirumaleswarReddy_Konda@McAfee.com>

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

     Author:  Prashanth Patil
              ~~<mailto:praspati&cisco.com>~~
              <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
       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 8782; see
       the RFC itself for full legal notices.";

    revision 2020-07-02 {
      description
        "Updated revision.";
      reference
        "RFC xxxx: xxx";
    }
    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 {
        key ~~"cuid mid";~~ "mid";
        description
          "The scope of the request.";
        leaf ~~cdid {~~
~~          type string;~~
~~          description~~
~~            "The cdid should be included by a server-domain~~
~~             DOTS gateway to propagate the client domain~~
~~             identification information from the~~
~~             gateway's client-facing side to the gateway's~~
~~             server-facing side, and from the gateway's~~
~~             server-facing side to the DOTS server.~~
~~           It may be used by the final DOTS server~~
~~           for policy enforcement purposes.";~~
~~      }~~
~~      leaf cuid {~~
~~        type string;~~
~~        description~~
~~          "A unique identifier that is~~
~~           generated by a DOTS client to prevent~~
~~           request collisions.  It is expected that the~~
~~           cuid will remain consistent throughout the~~
~~           lifetime of the DOTS client.";~~
~~      }~~
~~      leaf~~ mid {
        type uint32;
        description
          "Mitigation request identifier.

           This identifier must be unique for each mitigation
           request bound to the DOTS client.";
      }
      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**
          **"xxx.";**
        **case server-to-client {**
          **description**
            **"xx.";**
```

```
  leaf mitigation-start {
    type uint64;
~~config false;~~
    description
      "Mitigation start time is represented in seconds
       relative to 1970-01-01T00:00:00Z in UTC time.";
  }
  leaf status {
    type iana-signal:status;
~~config false;~~
    description
      "Indicates the status of a mitigation request.
       It must be included in responses only.";
  }
  container conflict-information {
~~config false;~~
    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 leafref {
          path "../../../mid";
        }
        description
          "Reference to the conflicting 'mid' bound to
           the same DOTS client.";
      }
    }
  }
  leaf bytes-dropped {
    type yang:zero-based-counter64;
    units "bytes";
~~config false;~~
    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;
~~config false;~~
    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;
        ~~config false;~~
            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;
        ~~config false;~~
            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.";
      leaf ~~max-value~~ **current-value** {
        type uint16;
        units "seconds";
        ~~config false;~~
        **default "30";**
        description
          ~~"Maximum acceptable~~
          **"Current** heartbeat-interval ~~value.";~~ **value.**
```

```
           '0' means that heartbeat mechanism is deactivated.";
      }
      choice direction {
        description
          "xxx.";
        case server-to-client {
          description
            "xx.";
          leaf ~~min-value~~ max-value {
            type uint16;
            units "seconds";
        ~~config false;~~
            description
          ~~"Minimum~~
              "Maximum acceptable heartbeat-interval value.";
          }
          leaf ~~current-value~~ min-value {
            type uint16;
            units "seconds";
        ~~default "30";~~
            description
          ~~"Current~~
              "Minimum acceptable heartbeat-interval ~~value.~~

           ~~'0' means that heartbeat mechanism is deactivated.";~~ value.";
          }
        }
      }
    }
    container missing-hb-allowed {
      description
        "Maximum number of missing heartbeats allowed.";
      leaf ~~max-value~~ current-value {
        type uint16;
        ~~config false;~~
        default "15";
        description
          ~~"Maximum acceptable~~
          "Current missing-hb-allowed value.";
      }
      choice direction {
        description
          "xxx.";
        case server-to-client {
          description
            "xx.";
          leaf ~~min-value~~ max-value {
            type uint16;
        ~~config false;~~
            description
          ~~"Minimum~~
              "Maximum acceptable missing-hb-allowed value.";
          }
        leaf ~~current-value~~ min-value {
         type uint16;
       ~~default "15";~~
         description
        ~~"Current~~
          "Minimum acceptable missing-hb-allowed value.";
        }
       }
      }
     }
     container probing-rate {
      description
       "The limit for sending Non-confirmable messages with
        no response.";
      leaf ~~max-value~~ current-value {
       type uint16;
       units "byte/second";
       ~~config false;~~
       default "5";
       description
        ~~"Maximum acceptable~~
        "Current probing-rate value.";
      }
      choice direction {
       description
        "xxx.";
       case server-to-client {
        description
         "xx.";
        leaf ~~min-value~~ max-value {
         type uint16;
         units "byte/second";
       ~~config false;~~
         description
        ~~"Minimum~~
          "Maximum acceptable probing-rate value.";
        }
        leaf ~~current-value~~ min-value {
         type uint16;
         units "byte/second";
       ~~default "5";~~
         description
        ~~"Current~~
          "Minimum acceptable probing-rate value.";
        }
       }
      }
     }
     container max-retransmit {
      description
       "Maximum number of retransmissions of a Confirmable
        message.";
      leaf ~~max-value~~ current-value {
```

```
    type uint16;
    ~~config false;~~
    default "3";
    description
      ~~"Maximum acceptable~~
      "Current max-retransmit value.";
  }
  choice direction {
    description
      "xxx.";
    case server-to-client {
      description
        "xx.";
      leaf ~~min-value~~ max-value {
        type uint16;
    ~~config false;~~
        description
      ~~"Minimum~~
          "Maximum acceptable max-retransmit value.";
      }
      leaf ~~current-value~~ min-value {
        type uint16;
    ~~default "3";~~
        description
      ~~"Current~~
          "Minimum acceptable max-retransmit value.";
      }
    }
  }
}
container ack-timeout {
  description
    "Initial retransmission timeout value.";
  leaf ~~max-value-decimal~~ current-value-decimal {
    type decimal64 {
      fraction-digits 2;
    }
    units "seconds";
    ~~config false;~~
    default "2";
    description
      ~~"Maximum~~
      "Current ack-timeout value.";
  }
  choice direction {
    description
      "xxx.";
    case server-to-client {
      description
        "xx.";
      leaf ~~min-value-decimal~~ max-value-decimal {
        type decimal64 {
          fraction-digits 2;
```

```
            }
            units "seconds";
        ~~config false;~~
            description
          ~~"Minimum~~
              **"Maximum** ack-timeout value.";
          }
          leaf ~~current-value-decimal~~ **min-value-decimal** {
            type decimal64 {
              fraction-digits 2;
            }
            units "seconds";
        ~~default "2";~~
            description
          ~~"Current~~
              **"Minimum** ack-timeout value.";
          }
        }
      **}**
    **}**
    container ack-random-factor {
      description
        "Random factor used to influence the timing of
         retransmissions.";
      leaf ~~max-value-decimal~~ **current-value-decimal** {
        type decimal64 {
          fraction-digits 2;
        }
        ~~config false;~~
        **default "1.5";**
        description
          ~~"Maximum acceptable~~
          **"Current** ack-random-factor value.";
      }
      **choice direction {**
        **description**
          **"xxx.";**
        **case server-to-client {**
          **description**
            **"xx.";**
          leaf ~~min-value-decimal~~ **max-value-decimal** {
            type decimal64 {
              fraction-digits 2;
            }
        ~~config false;~~
            description
          ~~"Minimum~~
              **"Maximum** acceptable ack-random-factor value.";
          }
          leaf ~~current-value-decimal~~ **min-value-decimal** {
            type decimal64 {
              fraction-digits 2;
            }
        ~~default "1.5";~~
            description
          ~~"Current~~
              **"Minimum acceptable** ack-random-factor value.";
          }
        }
      }
    **}**
  **}**

  grouping signal-config {
    description
      "DOTS signal channel session configuration.";
    leaf sid {
      type uint32;
      mandatory true;
      description
        "An identifier for the DOTS signal channel
         session configuration data.";
    }
    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**
        **"xxx.";**
      **case server-to-client {**
        **description**
          **"xx.";**
        leaf alt-server {
          type string;
      ~~config false;~~
          mandatory true;
          description
            "FQDN of an alternate server.";
        }
        leaf-list alt-server-record {
          type inet:ip-address;
      ~~config false;~~
```

```
          description
            "List of records for the alternate server.";
        }
      }
    }
  }

  /*
   * Main Container for DOTS Signal Channel
   */

  ~~container dots-signal~~
  sx:structure "dots-signal" {
    description
      "Main container 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;
      }
      case signal-config {
        description
          "Configuration message.";
        uses signal-config;
      }
      case redirected-signal {
        description
          "Redirected signaling.";
        uses redirected-signal;
      }
      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.";
        }
      }
    }
  }
}
```