Stream: Internet Engineering Task Force (IETF)
RFC: 9348
Category: Standards Track
Published: January 2023
ISSN: 2070-1721
Authors: D. Fedyk
LabN Consulting, L.L.C.
C. Hopps
LabN Consulting, L.L.C.

RFC 9348
A YANG Data Model for IP Traffic Flow Security

Abstract

This document describes a YANG module for the management of IP Traffic Flow Security (IP-TFS) additions to Internet Key Exchange Protocol version 2 (IKEv2) and IPsec.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9348.

Table of Contents

1. Introduction
2. Overview
3. YANG Management
   3.1. YANG Tree
   3.2. YANG Module
4. IANA Considerations
   4.1. Updates to the IETF XML Registry
   4.2. Updates to the YANG Module Names Registry
5. Security Considerations
6. References
   6.1. Normative References
   6.2. Informative References

Appendix A. Examples
   A.1. Example XML Configuration
   A.2. Example XML Operational Data
   A.3. Example JSON Configuration
   A.4. Example JSON Operational Data
   A.5. Example JSON Operational Statistics

Acknowledgements

Authors' Addresses

1. Introduction

This document defines a YANG module [RFC7950] for the management of the IP Traffic Flow Security (IP-TFS) extensions defined in [RFC9347]. IP-TFS provides enhancements to an IPsec tunnel Security Association (SA) to provide improved traffic confidentiality. Traffic confidentiality reduces the ability of traffic analysis to determine identity and correlate observable traffic patterns. IP-TFS offers efficiency when aggregating traffic in fixed-size IPsec tunnel packets.

The YANG data model in this document conforms to the Network Management Datastore Architecture (NMDA) defined in [RFC8342].

The published YANG modules for IPsec are defined in [RFC9061]. This document uses these models as a general IPsec model that is augmented for IP-TFS. The models in [RFC9061] provide for both an IKE and an IKE-less model.

2. Overview

This document defines configuration and operational parameters of IP Traffic Flow Security (IP-TFS). IP-TFS, defined in [RFC9347], defines a security association for tunnel mode IPsec with characteristics that improve traffic confidentiality and reduce bandwidth efficiency loss. These documents assume familiarity with the IPsec concepts described in [RFC4301].

IP-TFS uses tunnel mode to improve confidentiality by hiding inner packet identifiable information, packet size, and packet timing. IP-TFS provides a general capability allowing aggregation of multiple packets in uniform-size outer tunnel IPsec packets. It maintains the outer packet size by utilizing combinations of aggregating, padding, and fragmenting inner packets to fill out the IPsec outer tunnel packet. Padding is used to fill the packet when no data is available to send.

This document specifies an extensible configuration model for IP-TFS. This version utilizes the capabilities of IP-TFS to configure fixed-size IP-TFS packets that are transmitted at a constant rate. This model is structured to allow for different types of operation through future augmentation.

The IP-TFS YANG module augments the IPsec YANG module from [RFC9061]. IP-TFS makes use of IPsec tunnel mode and adds a small number of configuration items to IPsec tunnel mode. As defined in [RFC9347], any SA configured to use IP-TFS supports only IP-TFS packets, i.e., no mixed IPsec modes.

The behavior for IP-TFS is controlled by the source. The self-describing format of an IP-TFS packet allows a sending side to adjust the packet size and timing independently from any receiver. Both directions are also independent, e.g., IP-TFS may be run only in one direction. This means that counters, which are created here for both directions, may be 0 or not updated in the case of an SA that uses IP-TFS only in on direction.

Cases where IP-TFS statistics are active for one direction:

- SA one direction - IP-TFS enabled
- SA both directions - IP-TFS only enabled in one direction

Case where IP-TFS statistics are active for both directions:

- SA both directions - IP-TFS enable for both directions

The IP-TFS model supports IP-TFS configuration and operational data.

This YANG module supports configuration of fixed-size and fixed-rate packets, as well as elements that may be augmented to support future configuration. The protocol specification [RFC9347] goes beyond this simple, fixed mode of operation by defining a general format for any type of scheme. In this document, the outer IPsec packets can be sent with fixed or variable size (without padding). The configuration allows the fixed packet size to be determined by the path MTU. The fixed packet size can also be configured if a value lower than the path MTU is desired.

Other configuration items include:

Congestion Control:
: A congestion control setting to allow IP-TFS to reduce the packet rate when congestion is detected.

Fixed-Rate Configuration:
: The IP-TFS tunnel rate can be configured by taking into account either layer 2 overhead or layer 3 overhead. Layer 3 overhead is the IP data rate, and layer 2 overhead is the rate of bits on the link. The combination of packet size and rate determines the nominal maximum bandwidth and the transmission interval when fixed-size packets are used.

User Packet Fragmentation Control:
: While fragmentation is recommended for improved efficiency, a configuration is provided if users wish to observe the effect of no fragmentation on their data flows.

The YANG operational data allows the readout of the configured parameters, as well as the per-SA statistics and error counters for IP-TFS. Per-SA IPsec packet statistics are provided as a feature, and per-SA IP-TFS-specific statistics are provided as another feature. Both sets of statistics augment the IPsec YANG modules with counters that allow observation of IP-TFS packet efficiency.

IPsec YANG management objects are set in [RFC9061]. IP-TFS YANG augments the IKE and the IKE-less models. In these models, the Security Policy database entry and Security Association entry for an IPsec tunnel can be augmented with IP-TFS. In addition, this model uses YANG types defined in [RFC6991].

3. YANG Management

3.1. YANG Tree

The following is the YANG tree diagram [RFC8340] for the IP-TFS extensions.

```
module: ietf-ipsec-iptfs
  augment /nsfike:ipsec-ike/nsfike:conn-entry/nsfike:spd
            /nsfike:spd-entry/nsfike:ipsec-policy-config
            /nsfike:processing-info/nsfike:ipsec-sa-cfg:
    +--rw traffic-flow-security
       +--rw congestion-control?           boolean
       +--rw packet-size
       |  +--rw use-path-mtu-discovery?   boolean
```

```
      |  +--rw outer-packet-size?        uint16
      +--rw (tunnel-rate)?
      |  +--:(l2-fixed-rate)
      |  |  +--rw l2-fixed-rate?          yang:gauge64
      |  +--:(l3-fixed-rate)
      |     +--rw l3-fixed-rate?          yang:gauge64
      +--rw dont-fragment?                boolean
      +--rw max-aggregation-time?         decimal64
      +--rw window-size?                  uint16
      +--rw send-immediately?             boolean
      +--rw lost-packet-timer-interval?   decimal64
augment /nsfike:ipsec-ike/nsfike:conn-entry/nsfike:child-sa-info:
  +--ro traffic-flow-security
      +--ro congestion-control?           boolean
      +--ro packet-size
      |  +--ro use-path-mtu-discovery?   boolean
      |  +--ro outer-packet-size?        uint16
      +--ro (tunnel-rate)?
      |  +--:(l2-fixed-rate)
      |  |  +--ro l2-fixed-rate?          yang:gauge64
      |  +--:(l3-fixed-rate)
      |     +--ro l3-fixed-rate?          yang:gauge64
      +--ro dont-fragment?                boolean
      +--ro max-aggregation-time?         decimal64
      +--ro window-size?                  uint16
      +--ro send-immediately?             boolean
      +--ro lost-packet-timer-interval?   decimal64
augment /nsfikels:ipsec-ikeless/nsfikels:spd/nsfikels:spd-entry
          /nsfikels:ipsec-policy-config/nsfikels:processing-info
          /nsfikels:ipsec-sa-cfg:
  +--rw traffic-flow-security
      +--rw congestion-control?           boolean
      +--rw packet-size
      |  +--rw use-path-mtu-discovery?   boolean
      |  +--rw outer-packet-size?        uint16
      +--rw (tunnel-rate)?
      |  +--:(l2-fixed-rate)
      |  |  +--rw l2-fixed-rate?          yang:gauge64
      |  +--:(l3-fixed-rate)
      |     +--rw l3-fixed-rate?          yang:gauge64
      +--rw dont-fragment?                boolean
      +--rw max-aggregation-time?         decimal64
      +--rw window-size?                  uint16
      +--rw send-immediately?             boolean
      +--rw lost-packet-timer-interval?   decimal64
augment /nsfikels:ipsec-ikeless/nsfikels:sad/nsfikels:sad-entry:
  +--ro traffic-flow-security
      +--ro congestion-control?           boolean
      +--ro packet-size
      |  +--ro use-path-mtu-discovery?   boolean
      |  +--ro outer-packet-size?        uint16
      +--ro (tunnel-rate)?
      |  +--:(l2-fixed-rate)
      |  |  +--ro l2-fixed-rate?          yang:gauge64
      |  +--:(l3-fixed-rate)
      |     +--ro l3-fixed-rate?          yang:gauge64
      +--ro dont-fragment?                boolean
      +--ro max-aggregation-time?         decimal64
```

```
      +--ro window-size?                 uint16
      +--ro send-immediately?            boolean
      +--ro lost-packet-timer-interval?  decimal64
augment /nsfike:ipsec-ike/nsfike:conn-entry/nsfike:child-sa-info:
  +--ro ipsec-stats {ipsec-stats}?
  |  +--ro tx-pkts?        yang:counter64
  |  +--ro tx-octets?      yang:counter64
  |  +--ro tx-drop-pkts?   yang:counter64
  |  +--ro rx-pkts?        yang:counter64
  |  +--ro rx-octets?      yang:counter64
  |  +--ro rx-drop-pkts?   yang:counter64
  +--ro iptfs-inner-pkt-stats {iptfs-stats}?
  |  +--ro tx-pkts?              yang:counter64
  |  +--ro tx-octets?            yang:counter64
  |  +--ro rx-pkts?              yang:counter64
  |  +--ro rx-octets?            yang:counter64
  |  +--ro rx-incomplete-pkts?   yang:counter64
  +--ro iptfs-outer-pkt-stats {iptfs-stats}?
     +--ro tx-all-pad-pkts?       yang:counter64
     +--ro tx-all-pad-octets?     yang:counter64
     +--ro tx-extra-pad-pkts?     yang:counter64
     +--ro tx-extra-pad-octets?   yang:counter64
     +--ro rx-all-pad-pkts?       yang:counter64
     +--ro rx-all-pad-octets?     yang:counter64
     +--ro rx-extra-pad-pkts?     yang:counter64
     +--ro rx-extra-pad-octets?   yang:counter64
     +--ro rx-errored-pkts?       yang:counter64
     +--ro rx-missed-pkts?        yang:counter64
augment /nsfikels:ipsec-ikeless/nsfikels:sad/nsfikels:sad-entry:
  +--ro ipsec-stats {ipsec-stats}?
  |  +--ro tx-pkts?        yang:counter64
  |  +--ro tx-octets?      yang:counter64
  |  +--ro tx-drop-pkts?   yang:counter64
  |  +--ro rx-pkts?        yang:counter64
  |  +--ro rx-octets?      yang:counter64
  |  +--ro rx-drop-pkts?   yang:counter64
  +--ro iptfs-inner-pkt-stats {iptfs-stats}?
  |  +--ro tx-pkts?              yang:counter64
  |  +--ro tx-octets?            yang:counter64
  |  +--ro rx-pkts?              yang:counter64
  |  +--ro rx-octets?            yang:counter64
  |  +--ro rx-incomplete-pkts?   yang:counter64
  +--ro iptfs-outer-pkt-stats {iptfs-stats}?
     +--ro tx-all-pad-pkts?       yang:counter64
     +--ro tx-all-pad-octets?     yang:counter64
     +--ro tx-extra-pad-pkts?     yang:counter64
     +--ro tx-extra-pad-octets?   yang:counter64
     +--ro rx-all-pad-pkts?       yang:counter64
     +--ro rx-all-pad-octets?     yang:counter64
     +--ro rx-extra-pad-pkts?     yang:counter64
     +--ro rx-extra-pad-octets?   yang:counter64
     +--ro rx-errored-pkts?       yang:counter64
     +--ro rx-missed-pkts?        yang:counter64
```

3.2. YANG Module

The following is the YANG module for managing the IP-TFS extensions. The model contains references to [RFC9347] and [RFC5348].

```
<CODE BEGINS> file "ietf-ipsec-iptfs@2023-01-31.yang"

module ietf-ipsec-iptfs {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-ipsec-iptfs";
  prefix iptfs;

  import ietf-i2nsf-ike {
    prefix nsfike;
    reference
      "RFC 9061: A YANG Data Model for IPsec Flow Protection Based on
       Software-Defined Networking (SDN), Section 5.2";
  }
  import ietf-i2nsf-ikeless {
    prefix nsfikels;
    reference
      "RFC 9061: A YANG Data Model for IPsec Flow Protection Based on
       Software-Defined Networking (SDN), Section 5.3";
  }
  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }

  organization
    "IETF IPSECME Working Group (IPSECME)";
  contact
    "WG Web:  <https://datatracker.ietf.org/wg/ipsecme/>
     WG List: <mailto:ipsecme@ietf.org>

     Author: Don Fedyk
             <mailto:dfedyk@labn.net>

     Author: Christian Hopps
             <mailto:chopps@chopps.org>";

  description
    "This module defines the configuration and operational state for
     managing the IP Traffic Flow Security functionality (RFC 9348).

     Copyright (c) 2023 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 Revised BSD License
     set forth in Section 4.c of the IETF Trust's Legal Provisions
     Relating to IETF Documents
     (https://trustee.ietf.org/license-info).
```

```
     This version of this YANG module is part of RFC 9348; see
     the RFC itself for full legal notices.";

  revision 2023-01-31 {
    description
      "Initial revision";
    reference
      "RFC 9348: A YANG Data Model for IP Traffic Flow Security";
  }

  feature ipsec-stats {
    description
      "This feature indicates the device supports
       per-SA IPsec statistics.";
  }

  feature iptfs-stats {
    description
      "This feature indicates the device supports
       per-SA IP Traffic Flow Security statistics.";
  }

  /*--------------------*/
  /*   groupings        */
  /*--------------------*/

  grouping ipsec-tx-stat-grouping {
    description
      "IPsec outbound statistics";
    leaf tx-pkts {
      type yang:counter64;
      config false;
      description
        "Outbound Packet count";
    }
    leaf tx-octets {
      type yang:counter64;
      config false;
      description
        "Outbound Packet bytes";
    }
    leaf tx-drop-pkts {
      type yang:counter64;
      config false;
      description
        "Outbound dropped packets count";
    }
  }

  grouping ipsec-rx-stat-grouping {
    description
      "IPsec inbound statistics";
    leaf rx-pkts {
      type yang:counter64;
      config false;
      description
        "Inbound Packet count";
```

```
    }
    leaf rx-octets {
      type yang:counter64;
      config false;
      description
        "Inbound Packet bytes";
    }
    leaf rx-drop-pkts {
      type yang:counter64;
      config false;
      description
        "Inbound dropped packets count";
    }
  }

  grouping iptfs-inner-tx-stat-grouping {
    description
      "IP-TFS outbound inner packet statistics";
    leaf tx-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS inner packets sent.  This
         count is whole packets only.  A fragmented packet
         counts as one packet.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS)";
    }
    leaf tx-octets {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS inner octets sent.  This is
         inner packet octets only. It does not count padding.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS)";
    }
  }

  grouping iptfs-outer-tx-stat-grouping {
    description
      "IP-TFS outbound inner packet statistics";
    leaf tx-all-pad-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of transmitted IP-TFS packets that
         were all padding with no inner packet data.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3";
    }
    leaf tx-all-pad-octets {
```

```
      type yang:counter64;
      config false;
      description
        "Total number transmitted octets of padding added to
         IP-TFS packets with no inner packet data.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3";
    }
    leaf tx-extra-pad-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of transmitted outer IP-TFS packets
         that included some padding.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3.1";
    }
    leaf tx-extra-pad-octets {
      type yang:counter64;
      config false;
      description
        "Total number of transmitted octets of padding added
         to outer IP-TFS packets with data.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3.1";
    }
  }

  grouping iptfs-inner-rx-stat-grouping {
    description
      "IP-TFS inner packet inbound statistics";
    leaf rx-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS inner packets received.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2";
    }
    leaf rx-octets {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS inner octets received.  It does
         not include padding or overhead.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2";
    }
```

```
    leaf rx-incomplete-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS inner packets that were
         incomplete.  Usually this is due to fragments that are
         not received.  Also, this may be due to misordering or
         errors in received outer packets.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS)";
    }
  }

  grouping iptfs-outer-rx-stat-grouping {
    description
      "IP-TFS outer packet inbound statistics";
    leaf rx-all-pad-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of received IP-TFS packets that were
         all padding with no inner packet data.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3";
    }
    leaf rx-all-pad-octets {
      type yang:counter64;
      config false;
      description
        "Total number of received octets of padding added to
         IP-TFS packets with no inner packet data.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3";
    }
    leaf rx-extra-pad-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of received outer IP-TFS packets that
         included some padding.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3.1";
    }
    leaf rx-extra-pad-octets {
      type yang:counter64;
      config false;
      description
        "Total number of received octets of padding added to
         outer IP-TFS packets with data.";
      reference
```

```
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS), Section 2.2.3.1";
    }
    leaf rx-errored-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS outer packets dropped due to
         errors.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS)";
    }
    leaf rx-missed-pkts {
      type yang:counter64;
      config false;
      description
        "Total number of IP-TFS outer packets missing,
         indicated by a missing sequence number.";
      reference
        "RFC 9347: Aggregation and Fragmentation Mode for
         Encapsulating Security Payload (ESP) and Its Use for
         IP Traffic Flow Security (IP-TFS)";
    }
  }

  grouping iptfs-config {
    description
      "This is the grouping for IP-TFS configuration.";
    container traffic-flow-security {
      description
        "Configure the IPsec TFS in the Security
         Association Database (SAD).";
      leaf congestion-control {
        type boolean;
        default "true";
        description
          "When set to true, the default, this enables the
           congestion control on-the-wire exchange of data that is
           required by congestion control algorithms, as defined by
           RFC 5348.  When set to false, IP-TFS sends fixed-size
           packets over an IP-TFS tunnel at a constant rate.";
        reference
          "RFC 9347: Aggregation and Fragmentation Mode for
           Encapsulating Security Payload (ESP) and Its Use for
           IP Traffic Flow Security (IP-TFS), Section 2.4.2;
           RFC 5348: TCP Friendly Rate Control (TFRC): Protocol
           Specification";
      }
      container packet-size {
        description
          "Packet size is either auto-discovered or manually
           configured.";
        leaf use-path-mtu-discovery {
          type boolean;
          default "true";
```

```
          description
            "Utilize path MTU discovery to determine maximum
             IP-TFS packet size.  If the packet size is explicitly
             configured, then it will only be adjusted downward if
             use-path-mtu-discovery is set.";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 4.2";
        }
        leaf outer-packet-size {
          type uint16;
          units "bytes";
          description
            "On transmission, the size of the outer encapsulating
             tunnel packet (i.e., the IP packet containing
             Encapsulating Security Payload (ESP)).";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 4.2";
        }
      }
      choice tunnel-rate {
        description
          "The TFS bit rate may be specified at layer 2 wire
           rate or layer 3 packet rate.";
        leaf l2-fixed-rate {
          type yang:gauge64;
          units "bits/second";
          description
            "On transmission, target bandwidth/bit rate in
             bits/second for IP-TFS tunnel.  This fixed rate is the
             nominal timing for the fixed-size packet.  If
             congestion control is enabled, the rate may be
             adjusted down (or up if unset).";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 4.1";
        }
        leaf l3-fixed-rate {
          type yang:gauge64;
          units "bits/second";
          description
            "On transmission, target bandwidth/bit rate in
             bits/second for IP-TFS tunnel.  This fixed rate is the
             nominal timing for the fixed-size packet.  If
             congestion control is enabled, the rate may be
             adjusted down (or up if unset).";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 4.1";
        }
      }
      leaf dont-fragment {
        type boolean;
```

```
          default "false";
          description
            "On transmission, disable packet fragmentation across
             consecutive IP-TFS tunnel packets; inner packets larger
             than what can be transmitted in outer packets will be
             dropped.";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 2.2.4 and
             6.1.4";
        }
        leaf max-aggregation-time {
          type decimal64 {
            fraction-digits 6;
          }
          units "milliseconds";
          description
            "On transmission, maximum aggregation time is the
             maximum length of time a received inner packet can be
             held prior to transmission in the IP-TFS tunnel.  Inner
             packets that would be held longer than this time, based
             on the current tunnel configuration, will be dropped
             rather than be queued for transmission.  Maximum
             aggregation time is configurable in milliseconds or
             fractional milliseconds down to 1 nanosecond.";
        }
        leaf window-size {
          type uint16 {
            range "0..65535";
          }
          description
            "On reception, the maximum number of out-of-order
             packets that will be reordered by an IP-TFS receiver
             while performing the reordering operation.  The value 0
             disables any reordering.";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 2.2.3";
        }
        leaf send-immediately {
          type boolean;
          default "false";
          description
            "On reception, send inner packets as soon as possible; do
             not wait for lost or misordered outer packets.
             Selecting this option reduces the inner (user) packet
             delay but can amplify out-of-order delivery of the
             inner packet stream in the presence of packet
             aggregation and any reordering.";
          reference
            "RFC 9347: Aggregation and Fragmentation Mode for
             Encapsulating Security Payload (ESP) and Its Use for
             IP Traffic Flow Security (IP-TFS), Section 2.5";
        }
        leaf lost-packet-timer-interval {
          type decimal64 {
```

```
          fraction-digits 6;
        }
        units "milliseconds";
        description
          "On reception, this interval defines the length of time
           an IP-TFS receiver will wait for a missing packet before
           considering it lost.  If not using send-immediately,
           then each lost packet will delay inner (user) packets
           until this timer expires.  Setting this value too low
           can impact reordering and reassembly.  The value is
           configurable in milliseconds or fractional milliseconds
           down to 1 nanosecond.";
        reference
          "RFC 9347: Aggregation and Fragmentation Mode for
           Encapsulating Security Payload (ESP) and Its Use for
           IP Traffic Flow Security (IP-TFS), Section 2.2.3";
      }
    }
  }

  /*
   * IP-TFS ike configuration
   */

  augment "/nsfike:ipsec-ike/nsfike:conn-entry/nsfike:spd/"
        + "nsfike:spd-entry/"
        + "nsfike:ipsec-policy-config/"
        + "nsfike:processing-info/"
        + "nsfike:ipsec-sa-cfg" {
    description
      "IP-TFS configuration for this policy.";
    uses iptfs-config;
  }

  augment "/nsfike:ipsec-ike/nsfike:conn-entry/"
        + "nsfike:child-sa-info" {
    description
      "IP-TFS configured on this SA.";
    uses iptfs-config {
      refine "traffic-flow-security" {
        config false;
      }
    }
  }

  /*
   * IP-TFS ikeless configuration
   */

  augment "/nsfikels:ipsec-ikeless/nsfikels:spd/"
        + "nsfikels:spd-entry/"
        + "nsfikels:ipsec-policy-config/"
        + "nsfikels:processing-info/"
        + "nsfikels:ipsec-sa-cfg" {
    description
      "IP-TFS configuration for this policy.";
    uses iptfs-config;
  }
```

```
  augment "/nsfikels:ipsec-ikeless/nsfikels:sad/"
        + "nsfikels:sad-entry" {
    description
      "IP-TFS configured on this SA.";
    uses iptfs-config {
      refine "traffic-flow-security" {
        config false;
      }
    }
  }

  /*
   * packet counters
   */

  augment "/nsfike:ipsec-ike/nsfike:conn-entry/"
        + "nsfike:child-sa-info" {
    description
      "Per-SA counters";
    container ipsec-stats {
      if-feature "ipsec-stats";
      config false;
      description
        "IPsec per-SA packet counters.
         tx = outbound, rx = inbound";
      uses ipsec-tx-stat-grouping;
      uses ipsec-rx-stat-grouping;
    }
    container iptfs-inner-pkt-stats {
      if-feature "iptfs-stats";
      config false;
      description
        "IP-TFS per-SA inner packet counters.
         tx = outbound, rx = inbound";
      uses iptfs-inner-tx-stat-grouping;
      uses iptfs-inner-rx-stat-grouping;
    }
    container iptfs-outer-pkt-stats {
      if-feature "iptfs-stats";
      config false;
      description
        "IP-TFS per-SA outer packets counters.
         tx = outbound, rx = inbound";
      uses iptfs-outer-tx-stat-grouping;
      uses iptfs-outer-rx-stat-grouping;
    }
  }

  /*
   * packet counters
   */

  augment "/nsfikels:ipsec-ikeless/nsfikels:sad/"
        + "nsfikels:sad-entry" {
    description
      "Per-SA counters";
    container ipsec-stats {
```

```
        if-feature "ipsec-stats";
        config false;
        description
          "IPsec per-SA packet counters.
           tx = outbound, rx = inbound";
        uses ipsec-tx-stat-grouping;
        uses ipsec-rx-stat-grouping;
      }
      container iptfs-inner-pkt-stats {
        if-feature "iptfs-stats";
        config false;
        description
          "IP-TFS per-SA inner packet counters.
           tx = outbound, rx = inbound";
        uses iptfs-inner-tx-stat-grouping;
        uses iptfs-inner-rx-stat-grouping;
      }
      container iptfs-outer-pkt-stats {
        if-feature "iptfs-stats";
        config false;
        description
          "IP-TFS per-SA outer packets counters.
           tx = outbound, rx = inbound";
        uses iptfs-outer-tx-stat-grouping;
        uses iptfs-outer-rx-stat-grouping;
      }
    }
  }

<CODE ENDS>
```

4. IANA Considerations

4.1. Updates to the IETF XML Registry

Per this document, IANA has registered a URI in the "IETF XML Registry" [RFC3688] as follows.

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

4.2. Updates to the YANG Module Names Registry

Per this document, IANA has registered one YANG module in the "YANG Module Names" registry [RFC6020] as follows.

Name: ietf-ipsec-iptfs
Namespace: urn:ietf:params:xml:ns:yang:ietf-ipsec-iptfs
Prefix: iptfs
Reference: RFC 9348

5. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocols such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) to these data nodes without proper protection can have a negative effect on network operations. These are the subtrees and data nodes and their sensitivity/vulnerability:

../traffic-flow-security: Enabling IP-TFS is controlled by setting the entries under traffic-flow-security in IKE or IKE-less models. IP-TFS is set either to be congestion sensitive or a fixed rate by setting parameters in this subtree.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

../iptfs-inner-pkt-stats and ../iptfs-outer-pkt-stats: Access to IP-TFS statistics can provide information that IP-TFS obscures, such as the true activity of the flows using IP-TFS.

6. References

6.1. Normative References

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[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>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <https://www.rfc-editor.org/info/rfc6242>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[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>.

[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>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9061] Marin-Lopez, R., Lopez-Millan, G., and F. Pereniguez-Garcia, "A YANG Data Model for IPsec Flow Protection Based on Software-Defined Networking (SDN)", RFC 9061, DOI 10.17487/RFC9061, July 2021, <https://www.rfc-editor.org/info/rfc9061>.

[RFC9347] Hopps, C., "Aggregation and Fragmentation Mode for Encapsulating Security Payload (ESP) and Its Use for IP Traffic Flow Security (IP-TFS)", RFC 9347, DOI 10.17487/RFC9347, January 2023, <https://www.rfc-editor.org/info/rfc9347>.

6.2. Informative 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>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <https://www.rfc-editor.org/info/rfc5348>.

[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>.

Appendix A. Examples

The following examples show configuration and operational data for the IKE-less and IKE cases using XML and JSON. Also, the operational statistics for the IKE-less case is illustrated.

A.1. Example XML Configuration

This example illustrates configuration for IP-TFS in the IKE-less case. Note that, since this augments the IPsec IKE-less schema, only a minimal IKE-less configuration to satisfy the schema has been populated.

```
<i:ipsec-ikeless
  xmlns:i="urn:ietf:params:xml:ns:yang:ietf-i2nsf-ikeless"
  xmlns:tfs="urn:ietf:params:xml:ns:yang:ietf-ipsec-iptfs">
  <i:spd>
    <i:spd-entry>
      <i:name>protect-policy-1</i:name>
      <i:direction>outbound</i:direction>
      <i:ipsec-policy-config>
        <i:traffic-selector>
          <i:local-prefix>192.0.2.0/16</i:local-prefix>
          <i:remote-prefix>198.51.100.0/16</i:remote-prefix>
        </i:traffic-selector>
        <i:processing-info>
          <i:action>protect</i:action>
          <i:ipsec-sa-cfg>
            <tfs:traffic-flow-security>
             <tfs:congestion-control>true</tfs:congestion-control>
              <tfs:packet-size>
                <tfs:use-path-mtu-discovery
                   >true</tfs:use-path-mtu-discovery>
              </tfs:packet-size>
              <tfs:l2-fixed-rate>1000000000</tfs:l2-fixed-rate>
              <tfs:max-aggregation-time
                 >0.1</tfs:max-aggregation-time>
              <tfs:window-size>5</tfs:window-size>
              <tfs:send-immediately>false</tfs:send-immediately>
              <tfs:lost-packet-timer-interval
                 >0.2</tfs:lost-packet-timer-interval>
            </tfs:traffic-flow-security>
          </i:ipsec-sa-cfg>
        </i:processing-info>
      </i:ipsec-policy-config>
    </i:spd-entry>
  </i:spd>
</i:ipsec-ikeless>
```

Figure 1: Example IP-TFS XML Configuration

A.2. Example XML Operational Data

This example illustrates operational data for IP-TFS in the IKE-less case. Note that, since this augments the IPsec IKE-less schema only, a minimal IKE-less configuration to satisfy the schema has been populated.

```
<i:ipsec-ikeless
  xmlns:i="urn:ietf:params:xml:ns:yang:ietf-i2nsf-ikeless"
  xmlns:tfs="urn:ietf:params:xml:ns:yang:ietf-ipsec-iptfs">
  <i:sad>
    <i:sad-entry>
      <i:name>sad-1</i:name>
      <i:ipsec-sa-config>
        <i:spi>1</i:spi>
        <i:traffic-selector>
          <i:local-prefix>2001:db8:1::/48</i:local-prefix>
          <i:remote-prefix>2001:db8:2::/48</i:remote-prefix>
        </i:traffic-selector>
      </i:ipsec-sa-config>
      <tfs:traffic-flow-security>
        <tfs:congestion-control>true</tfs:congestion-control>
        <tfs:packet-size>
          <tfs:use-path-mtu-discovery
            >true</tfs:use-path-mtu-discovery>
        </tfs:packet-size>
        <tfs:l2-fixed-rate>1000000000</tfs:l2-fixed-rate>
        <tfs:max-aggregation-time>0.100</tfs:max-aggregation-time>
        <tfs:window-size>0</tfs:window-size>
        <tfs:send-immediately>true</tfs:send-immediately>
        <tfs:lost-packet-timer-interval
          >0.200</tfs:lost-packet-timer-interval>
      </tfs:traffic-flow-security>
    </i:sad-entry>
  </i:sad>
</i:ipsec-ikeless>
```

Figure 2: Example IP-TFS XML Operational Data

A.3. Example JSON Configuration

This example illustrates configuration data for IP-TFS in the IKE case. Note that, since this augments the IPsec IKE schema, only a minimal IKE configuration to satisfy the schema has been populated.

```
{
  "ietf-i2nsf-ike:ipsec-ike": {
    "ietf-i2nsf-ike:conn-entry": [
      {
        "name": "my-peer-connection",
        "ike-sa-encr-alg": [
          {
            "id": 1,
            "algorithm-type": 12,
            "key-length": 128
          }
          ],
          "local": {
            "local-pad-entry-name": "local-1"
          },
          "remote": {
            "remote-pad-entry-name": "remote-1"
          },
          "ietf-i2nsf-ike:spd": {
          "spd-entry": [
            {
              "name": "protect-policy-1",
              "ipsec-policy-config": {
                "traffic-selector": {
                  "local-prefix": "192.0.2.0/16",
                  "remote-prefix": "198.51.100.0/16"
                },
                "processing-info": {
                  "action": "protect",
                  "ipsec-sa-cfg": {
                    "ietf-ipsec-iptfs:traffic-flow-security": {
                      "congestion-control": true,
                      "l2-fixed-rate": "1000000000",
                      "packet-size": {
                        "use-path-mtu-discovery": true
                      },
                      "max-aggregation-time": "0.1",
                      "window-size": 1,
                      "send-immediately": false,
                      "lost-packet-timer-interval": "0.2"
                    }
                  }
                }
              }
            }
          ]
        }
      }
    ]
  }
}
```

Figure 3: Example IP-TFS JSON Configuration

A.4. Example JSON Operational Data

This example illustrates operational data for IP-TFS in the IKE case. Note that, since this augments the IPsec IKE tree, only a minimal IKE configuration to satisfy the schema has been populated.

```
{
  "ietf-i2nsf-ike:ipsec-ike": {
    "ietf-i2nsf-ike:conn-entry": [
      {
        "name": "my-peer-connection",
        "ike-sa-encr-alg": [
        {
          "id": 1,
          "algorithm-type": 12,
          "key-length": 128
        }
        ],
        "local": {
          "local-pad-entry-name": "local-1"
        },
        "remote": {
          "remote-pad-entry-name": "remote-1"
        },
        "ietf-i2nsf-ike:child-sa-info": {
          "ietf-ipsec-iptfs:traffic-flow-security": {
            "congestion-control": true,
            "l2-fixed-rate": "1000000000",
            "packet-size": {
              "use-path-mtu-discovery": true
            },
            "max-aggregation-time": "0.1",
            "window-size": 5,
            "send-immediately": false,
            "lost-packet-timer-interval": "0.2"
          }
        }
      }
    ]
  }
}
```

Figure 4: Example IP-TFS JSON Operational Data

A.5. Example JSON Operational Statistics

This example shows the JSON formatted statistics for IP-TFS. Note a unidirectional IP-TFS transmit side is illustrated, with arbitrary numbers for transmit.

```
{
  "ietf-i2nsf-ikeless:ipsec-ikeless": {
    "sad": {
      "sad-entry": [
        {
          "name": "sad-1",
          "ipsec-sa-config": {
            "spi": 1,
            "traffic-selector": {
              "local-prefix": "192.0.2.1/16",
              "remote-prefix": "198.51.100.0/16"
            }
          },
          "ietf-ipsec-iptfs:traffic-flow-security": {
            "window-size": 5,
            "send-immediately": false,
            "lost-packet-timer-interval": "0.2"
          },
          "ietf-ipsec-iptfs:ipsec-stats": {
            "tx-pkts": "300",
            "tx-octets": "80000",
            "tx-drop-pkts": "2",
            "rx-pkts": "0",
            "rx-octets": "0",
            "rx-drop-pkts": "0"
          },
          "ietf-ipsec-iptfs:iptfs-inner-pkt-stats": {
            "tx-pkts": "250",
            "tx-octets": "75000",
            "rx-pkts": "0",
            "rx-octets": "0",
            "rx-incomplete-pkts": "0"
          },
          "ietf-ipsec-iptfs:iptfs-outer-pkt-stats": {
            "tx-all-pad-pkts": "40",
            "tx-all-pad-octets": "40000",
            "tx-extra-pad-pkts": "200",
            "tx-extra-pad-octets": "30000",
            "rx-all-pad-pkts": "0",
            "rx-all-pad-octets": "0",
            "rx-extra-pad-pkts": "0",
            "rx-extra-pad-octets": "0",
            "rx-errored-pkts": "0",
            "rx-missed-pkts": "0"
          },
          "ipsec-sa-state": {
            "sa-lifetime-current": {
              "time": 80000,
              "bytes": "400606",
              "packets": 1000,
              "idle": 5
            }
          }
        }
      ]
    }
```

```
    }
  }
```

Figure 5: Example IP-TFS JSON Statistics

Acknowledgements

The authors would like to thank Eric Kinzie, Jürgen Schönwälder, Lou Berger, and Tero Kivinen for their feedback and review on the YANG module.

Authors' Addresses

Don Fedyk
LabN Consulting, L.L.C.
Email: dfedyk@labn.net

Christian Hopps
LabN Consulting, L.L.C.
Email: chopps@chopps.org