

## TCP Extended Statistics MIB

### Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

### Copyright Notice

Copyright (C) The IETF Trust (2007).

### Abstract

This document describes extended performance statistics for TCP. They are designed to use TCP's ideal vantage point to diagnose performance problems in both the network and the application. If a network-based application is performing poorly, TCP can determine if the bottleneck is in the sender, the receiver, or the network itself. If the bottleneck is in the network, TCP can provide specific information about its nature.

### Table of Contents

1. Introduction .....	2
2. The Internet-Standard Management Framework .....	2
3. Overview .....	2
3.1. MIB Initialization and Persistence .....	4
3.2. Relationship to TCP Standards .....	4
3.3. Diagnosing SYN-Flood Denial-of-Service Attacks .....	6
4. TCP Extended Statistics MIB .....	7
5. Security Considerations .....	69
6. IANA Considerations .....	70
7. Normative References .....	70
8. Informative References .....	72
9. Contributors .....	73
10. Acknowledgments .....	73

## 1. Introduction

This document describes extended performance statistics for TCP. They are designed to use TCP's ideal vantage point to diagnose performance problems in both the network and the application. If a network-based application is performing poorly, TCP can determine if the bottleneck is in the sender, the receiver, or the network itself. If the bottleneck is in the network, TCP can provide specific information about its nature.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

The Simple Network Management Protocol (SNMP) objects defined in this document extend TCP MIB, as specified in RFC 4022 [RFC4022]. In addition to several new scalars and other objects, it augments two tables and makes one clarification to RFC 4022. Existing management stations for the TCP MIB are expected to be fully compatible with these clarifications.

## 2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

## 3. Overview

The TCP-ESTATS-MIB defined in this memo consists of two groups of scalars, seven tables, and two notifications:

- \* The first group of scalars contain statistics of the TCP protocol engine not covered in RFC 4022. This group consists of the single scalar tcpEStatsListenerTableLastChange, which provides management stations with an easier mechanism to validate their listener caches.

- \* The second group of scalars consist of knobs to enable and disable information collection by the tables containing connection-related statistics/information. For example, the tcpEStatsControlPath object controls the activation of the tcpEStatsPathTable. The tcpEStatsConnTableLatency object determines how long connection table rows are retained after a TCP connection transitions into the closed state.
- \* The tcpEStatsListenerTable augments tcpListenerTable in TCP-MIB [RFC4022] to provide additional information on the active TCP listeners on a device. It supports objects to monitor and diagnose SYN-flood denial-of-service attacks as described below.
- \* The tcpEStatsConnectIdTable augments the tcpConnectionTable in TCP-MIB [RFC4022] to provide a mapping between connection 4-tuples (which index tcpConnectionTable) and an integer connection index, tcpEStatsConnectIndex. The connection index is used to index into the five remaining tables in this MIB module, and is designed to facilitate rapid polling of multiple objects associated with one TCP connection.
- \* The tcpEStatsPerfTable contains objects that are useful for measuring TCP performance and first check problem diagnosis.
- \* The tcpEStatsPathTable contains objects that can be used to infer detailed behavior of the Internet path, such as the extent that there are segment losses or reordering, etc.
- \* The tcpEStatsStackTable contains objects that are most useful for determining how well the TCP control algorithms are coping with this particular path.
- \* The tcpEStatsAppTable provides objects that are useful for determining if the application using TCP is limiting TCP performance.
- \* The tcpEStatsTuneTable provides per-connection controls that can be used to work around a number of common problems that plague TCP over some paths.
- \* The two notifications defined in this MIB module are tcpEStatsEstablishNotification, indicating that a new connection has been accepted (or established, see below), and tcpEStatsCloseNotification, indicating that an existing connection has recently closed.

### 3.1. MIB Initialization and Persistence

The TCP protocol itself is specifically designed not to preserve any state whatsoever across system reboots, and enforces this by requiring randomized Initial Sequence numbers and ephemeral ports under any conditions where segments from old connections might corrupt new connections following a reboot.

All of the objects in the MIB MUST have the same persistence properties as the underlying TCP implementation. On a reboot, all zero-based counters MUST be cleared, all dynamically created table rows MUST be deleted, and all read-write objects MUST be restored to their default values. It is assumed that all TCP implementation have some initialization code (if nothing else, to set IP addresses) that has the opportunity to adjust `tcpEStatsConnTableLatency` and other read-write scalars controlling the creation of the various tables, before establishing the first TCP connection. Implementations MAY also choose to make these control scalars persist across reboots.

The `ZeroBasedCounter32` and `ZeroBasedCounter64` objects in the listener and connection tables are initialized to zero when the table row is created.

The `tcpEStatsConnTableLatency` object determines how long connection table rows are retained after a TCP connection transitions into the closed state, to permit reading final connection completion statistics. In RFC 4022 (TCP-MIB), the discussion of `tcpConnectionTable` row latency (page 9) the words "soon after" are understood to mean after `tcpEStatsConnTableLatency`, such that all rows of all tables associated with one connection are retained at least `tcpEStatsConnTableLatency` after connection close. This clarification to RFC 4022 only applies when TCP-ESTATS-MIB is implemented. If TCP-ESTATS-MIB is not implemented, RFC 4022 permits an unspecified delay between connection close and row deletion.

### 3.2. Relationship to TCP Standards

There are more than 70 RFCs and other documents that specify various aspects of the Transmission Control Protocol (TCP) [RFC4614]. While most protocols are completely specified in one or two documents, this has not proven to be feasible for TCP. TCP implements a reliable end-to-end data transport service over a very weakly constrained IP datagram service. The essential problem that TCP has to solve is balancing the applications need for fast and reliable data transport against the need to make fair, efficient, and equitable use of network resources, with only sparse information about the state of the network or its capabilities.

TCP maintains this balance through the use of many estimators and heuristics that regulate various aspects of the protocol. For example, RFC 2988 describes how to calculate the retransmission timer (RTO) from the average and variance of the network round-trip-time (RTT), as estimated from the round-trip time sampled on some data segments. Although these algorithms are standardized, they are a compromise which is optimal for only common Internet environments. Other estimators might yield better results (higher performance or more efficient use of the network) in some environments, particularly under uncommon conditions.

It is the consensus of the community that nearly all of the estimators and heuristics used in TCP might be improved through further research and development. For this reason, nearly all TCP documents leave some latitude for future improvements, for example, by the use of "SHOULD" instead of "MUST" [RFC2119]. Even standard algorithms that are required because they critically effect fairness or the dynamic stability of Internet congestion control, include some latitude for evolution. As a consequence, there is considerable diversity in the details of the TCP implementations actually in use today.

The fact that the underlying algorithms are not uniform makes it difficult to tightly specify a MIB. We could have chosen the point of view that the MIB should publish precisely defined metrics of the network path, even if they are different from the estimators in use by TCP. This would make the MIB more useful as a measurement tool, but less useful for understanding how any specific TCP implementation is interacting with the network path and upper protocol layers. We chose instead to have the MIB expose the estimators and important states variables of the algorithms in use, without constraining the TCP implementation.

As a consequence, the MIB objects are defined in terms of fairly abstract descriptions (e.g., round-trip time), but are intended to expose the actual estimators or other state variables as they are used in TCP implementations, possibly transformed (e.g., scaled or otherwise adjusted) to match the spirit of the object descriptions in this document.

This may mean that MIB objects may not be exactly comparable between two different TCP implementations. A general management station can only assume the abstract descriptions, which are useful for a general assessment of how TCP is functioning. To a TCP implementer with detailed knowledge about the TCP implementation on a specific host, this MIB might be useful for debugging or evaluating the algorithms in their implementation.

Under no conditions is this MIB intended to constrain TCP to use (or exclude) any particular estimator, heuristic, algorithm, or implementation.

### 3.3. Diagnosing SYN-Flood Denial-of-Service Attacks

The `tcpEStatsListenerTable` is specifically designed to provide information that is useful for diagnosing SYN-flood Denial-of-Service attacks, where a server is overwhelmed by forged or otherwise malicious connection attempts. There are several different techniques that can be used to defend against SYN-flooding but none are standardized [Edd06]. These different techniques all have the same basic characteristics that are instrumentable with a common set of objects, even though the techniques differ greatly in the details.

All SYN-flood defenses avoid allocating significant resources (memory or CPU) to incoming (passive open) connections until the connections meet some liveness criteria (to defend against forged IP source addresses) and the server has sufficient resources to process the incoming request. Note that allocating resources is an implementation-specific event that may not correspond to an observable protocol event (e.g., segments on the wire). There are two general concepts that can be applied to all known SYN-flood defenses. There is generally a well-defined event when a connection is allocated full resources, and a "backlog" -- a queue of embryonic connections that have been allocated only partial resources.

In many implementations, incoming TCP connections are allocated resources as a side effect of the POSIX [POSIX] `accept()` call. For this reason we use the terminology "accepting a connection" to refer to this event: committing sufficient network resources to process the incoming request. Accepting a connection typically entails allocating memory for the protocol control block [RFC793], the per-connection table rows described in this MIB and CPU resources, such as process table entries or threads.

Note that it is not useful to accept connections before they are ESTABLISHED, because this would create an easy opportunity for Denial-of-Service attacks, using forged source IP addresses.

The backlog consists of connections that are in SYN-RCVD or ESTABLISHED states, that have not been accepted. For purposes of this MIB, we assume that these connections have been allocated some resources (e.g., an embryonic protocol control block), but not full resources (e.g., do not yet have MIB table rows).

Note that some SYN-Flood defenses dispense with explicit SYN-RCVD state by cryptographically encoding the state in the ISS (initial sequence number sent) of the SYN-ACK (sometimes called a syn-cookie), and then using the sequence number of the first ACK to reconstruct the SYN-RCVD state before transitioning to the ESTABLISHED state. For these implementations there is no explicit representation of the SYN-RCVD state, and the backlog only consists of connections that are ESTABLISHED and are waiting to be ACCEPTED.

Furthermore, most SYN-flood defenses have some mechanism to throttle connections that might otherwise overwhelm this endpoint. They generally use some combination of discarding incoming SYNs and discarding connections already in the backlog. This does not cause all connections from legitimate clients to fail, as long as the clients retransmit the SYN or first ACK as specified in RFC 793. Most diversity in SYN flood defenses arise from variations in these algorithms to limit load, and therefore cannot be instrumented with a common standard MIB.

The Listen Table instruments all passively opened TCP connections in terms of observable protocol events (e.g., sent and received segments) and resource allocation events (entering the backlog and being accepted). This approach eases generalization to SYN-flood mechanisms that use alternate TCP state transition diagrams and implicit mechanisms to encode some states.

#### 4. TCP Extended Statistics MIB

This MIB module IMPORTS definitions from [RFC2578], [RFC2579], [RFC2580], [RFC2856], [RFC4022], and [RFC4502]. It uses REFERENCE clauses to refer to [RFC791], [RFC793], [RFC1122], [RFC1191], [RFC1323], [RFC2018], [RFC2581], [RFC2861], [RFC2883], [RFC2988], [RFC3168], [RFC3260], [RFC3517], [RFC3522], and [RFC3742].

TCP-ESTATS-MIB DEFINITIONS ::= BEGIN

IMPORTS

```

    MODULE-IDENTITY, Counter32, Integer32, Unsigned32,
    Gauge32, OBJECT-TYPE, mib-2,
    NOTIFICATION-TYPE
        FROM SNMPv2-SMI                                -- [RFC2578]
    MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
        FROM SNMPv2-CONF                                -- [RFC2580]
    ZeroBasedCounter32
        FROM RMON2-MIB                                  -- [RFC4502]
    ZeroBasedCounter64
        FROM HCNUM-TC                                    -- [RFC2856]
    TEXTUAL-CONVENTION,
    DateAndTime, TruthValue, TimeStamp

```

```
FROM SNMPv2-TC -- [RFC2579]
tcpListenerEntry, tcpConnectionEntry
FROM TCP-MIB; -- [RFC4022]
```

tcpEStatsMIB MODULE-IDENTITY

LAST-UPDATED "200705180000Z" -- 18 May 2007

ORGANIZATION "IETF TSV Working Group"

CONTACT-INFO

"Matt Mathis  
John Heffner  
Web100 Project  
Pittsburgh Supercomputing Center  
300 S. Craig St.  
Pittsburgh, PA 15213  
Email: mathis@psc.edu, jheffner@psc.edu

Rajiv Raghunarayan  
Cisco Systems Inc.  
San Jose, CA 95134  
Phone: 408 853 9612  
Email: raraghun@cisco.com

Jon Saperia  
84 Kettell Plain Road  
Stow, MA 01775  
Phone: 617-201-2655  
Email: saperia@jdscons.com "

DESCRIPTION

"Documentation of TCP Extended Performance Instrumentation  
variables from the Web100 project. [Web100]

All of the objects in this MIB MUST have the same  
persistence properties as the underlying TCP implementation.  
On a reboot, all zero-based counters MUST be cleared, all  
dynamically created table rows MUST be deleted, and all  
read-write objects MUST be restored to their default values.

It is assumed that all TCP implementation have some  
initialization code (if nothing else to set IP addresses)  
that has the opportunity to adjust tcpEStatsConnTableLatency  
and other read-write scalars controlling the creation of the  
various tables, before establishing the first TCP  
connection. Implementations MAY also choose to make these  
control scalars persist across reboots.

Copyright (C) The IETF Trust (2007). This version  
of this MIB module is a part of RFC 4898; see the RFC  
itself for full legal notices."



REVISION "200705180000Z" -- 18 May 2007

DESCRIPTION

"Initial version, published as RFC 4898."

::= { mib-2 156 }

```
tcpEStatsNotifications OBJECT IDENTIFIER ::= { tcpEStatsMIB 0 }
tcpEStatsMIBObjects     OBJECT IDENTIFIER ::= { tcpEStatsMIB 1 }
tcpEStatsConformance   OBJECT IDENTIFIER ::= { tcpEStatsMIB 2 }
tcpEStats               OBJECT IDENTIFIER ::= { tcpEStatsMIBObjects 1 }
tcpEStatsControl        OBJECT IDENTIFIER ::= { tcpEStatsMIBObjects 2 }
tcpEStatsScalar         OBJECT IDENTIFIER ::= { tcpEStatsMIBObjects 3 }
```

--

-- Textual Conventions

--

TcpEStatsNegotiated ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Indicates if some optional TCP feature was negotiated.

Enabled(1) indicates that the feature was successfully negotiated on, which generally requires both hosts to agree to use the feature.

selfDisabled(2) indicates that the local host refused the feature because it is not implemented, configured off, or refused for some other reason, such as the lack of resources.

peerDisabled(3) indicates that the local host was willing to negotiate the feature, but the remote host did not do so."

```
SYNTAX INTEGER {
    enabled(1),
    selfDisabled(2),
    peerDisabled(3)
}
```

--

-- TCP Extended statistics scalars

--

tcpEStatsListenerTableLastChange OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```

        "The value of sysUpTime at the time of the last
        creation or deletion of an entry in the tcpListenerTable.
        If the number of entries has been unchanged since the
        last re-initialization of the local network management
        subsystem, then this object contains a zero value."
 ::= { tcpEStatsScalar 3 }

-- =====
--
-- The tcpEStatsControl Group
--
-- The scalar objects in this group are used to control the
-- activation and deactivation of the TCP Extended Statistics
-- tables and notifications in this module.
--

tcpEStatsControlPath OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS       read-write
    STATUS           current
    DESCRIPTION
        "Controls the activation of the TCP Path Statistics
        table.

        A value 'true' indicates that the TCP Path Statistics
        table is active, while 'false' indicates that the
        table is inactive."
    DEFVAL           { false }
    ::= { tcpEStatsControl 1 }

tcpEStatsControlStack OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS       read-write
    STATUS           current
    DESCRIPTION
        "Controls the activation of the TCP Stack Statistics
        table.

        A value 'true' indicates that the TCP Stack Statistics
        table is active, while 'false' indicates that the
        table is inactive."
    DEFVAL           { false }
    ::= { tcpEStatsControl 2 }

tcpEStatsControlApp OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS       read-write

```

STATUS current

DESCRIPTION

"Controls the activation of the TCP Application Statistics table.

A value 'true' indicates that the TCP Application Statistics table is active, while 'false' indicates that the table is inactive."

DEFVAL { false }

::= { tcpEStatsControl 3 }

tcpEStatsControlTune OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Controls the activation of the TCP Tuning table.

A value 'true' indicates that the TCP Tuning table is active, while 'false' indicates that the table is inactive."

DEFVAL { false }

::= { tcpEStatsControl 4 }

tcpEStatsControlNotify OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Controls the generation of all notifications defined in this MIB.

A value 'true' indicates that the notifications are active, while 'false' indicates that the notifications are inactive."

DEFVAL { false }

::= { tcpEStatsControl 5 }

tcpEStatsConnTableLatency OBJECT-TYPE

SYNTAX Unsigned32

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Specifies the number of seconds that the entity will retain entries in the TCP connection tables, after the connection first enters the closed state. The entity SHOULD provide a configuration option to enable

customization of this value. A value of 0 results in entries being removed from the tables as soon as the connection enters the closed state. The value of this object pertains to the following tables:

```
tcpEStatsConnectIdTable
tcpEStatsPerfTable
tcpEStatsPathTable
tcpEStatsStackTable
tcpEStatsAppTable
tcpEStatsTuneTable"
```

```
DEFVAL { 0 }
```

```
::= { tcpEStatsControl 6 }
```

```
-- =====
--
-- Listener Table
--
```

```
tcpEStatsListenerTable OBJECT-TYPE
```

```
SYNTAX      SEQUENCE OF TcpEStatsListenerEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"This table contains information about TCP Listeners,
in addition to the information maintained by the
tcpListenerTable RFC 4022."
```

```
::= { tcpEStats 1 }
```

```
tcpEStatsListenerEntry OBJECT-TYPE
```

```
SYNTAX      TcpEStatsListenerEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Each entry in the table contains information about
a specific TCP Listener."
```

```
AUGMENTS { tcpListenerEntry }
```

```
::= { tcpEStatsListenerTable 1 }
```

```
TcpEStatsListenerEntry ::= SEQUENCE {
```

tcpEStatsListenerStartTime	TimeStamp,
tcpEStatsListenerSynRcvd	ZeroBasedCounter32,
tcpEStatsListenerInitial	ZeroBasedCounter32,
tcpEStatsListenerEstablished	ZeroBasedCounter32,
tcpEStatsListenerAccepted	ZeroBasedCounter32,
tcpEStatsListenerExceedBacklog	ZeroBasedCounter32,
tcpEStatsListenerHCSynRcvd	ZeroBasedCounter64,
tcpEStatsListenerHCInitial	ZeroBasedCounter64,
tcpEStatsListenerHCEstablished	ZeroBasedCounter64,

```

        tcpEStatsListenerHCAccepted      ZeroBasedCounter64,
        tcpEStatsListenerHCExceedBacklog ZeroBasedCounter64,
        tcpEStatsListenerCurConns       Gauge32,
        tcpEStatsListenerMaxBacklog      Unsigned32,
        tcpEStatsListenerCurBacklog     Gauge32,
        tcpEStatsListenerCurEstabBacklog Gauge32
    }

tcpEStatsListenerStartTime OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value of sysUpTime at the time this listener was
        established. If the current state was entered prior to
        the last re-initialization of the local network management
        subsystem, then this object contains a zero value."
    ::= { tcpEStatsListenerEntry 1 }

tcpEStatsListenerSynRcvd OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of SYNs which have been received for this
        listener. The total number of failed connections for
        all reasons can be estimated to be tcpEStatsListenerSynRcvd
        minus tcpEStatsListenerAccepted and
        tcpEStatsListenerCurBacklog."
    ::= { tcpEStatsListenerEntry 2 }

tcpEStatsListenerInitial OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total number of connections for which the Listener
        has allocated initial state and placed the
        connection in the backlog. This may happen in the
        SYN-RCVD or ESTABLISHED states, depending on the
        implementation."
    ::= { tcpEStatsListenerEntry 3 }

tcpEStatsListenerEstablished OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION

```

"The number of connections that have been established to this endpoint (e.g., the number of first ACKs that have been received for this listener)."

::= { tcpEStatsListenerEntry 4 }

tcpEStatsListenerAccepted OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of connections for which the Listener has successfully issued an accept, removing the connection from the backlog."

::= { tcpEStatsListenerEntry 5 }

tcpEStatsListenerExceedBacklog OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of connections dropped from the backlog by this listener due to all reasons. This includes all connections that are allocated initial resources, but are not accepted for some reason."

::= { tcpEStatsListenerEntry 6 }

tcpEStatsListenerHCSynRcvd OBJECT-TYPE

SYNTAX ZeroBasedCounter64

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of SYNs that have been received for this listener on systems that can process (or reject) more than 1 million connections per second. See tcpEStatsListenerSynRcvd."

::= { tcpEStatsListenerEntry 7 }

tcpEStatsListenerHCInitial OBJECT-TYPE

SYNTAX ZeroBasedCounter64

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of connections for which the Listener has allocated initial state and placed the connection in the backlog on systems that can process (or reject) more than 1 million connections per second. See tcpEStatsListenerInitial."

::= { tcpEStatsListenerEntry 8 }

tcpEStatsListenerHCEstablished OBJECT-TYPE  
SYNTAX ZeroBasedCounter64  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The number of connections that have been established to  
    this endpoint on systems that can process (or reject) more  
    than 1 million connections per second. See  
    tcpEStatsListenerEstablished."  
 ::= { tcpEStatsListenerEntry 9 }

tcpEStatsListenerHCAccepted OBJECT-TYPE  
SYNTAX ZeroBasedCounter64  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The total number of connections for which the Listener  
    has successfully issued an accept, removing the connection  
    from the backlog on systems that can process (or reject)  
    more than 1 million connections per second. See  
    tcpEStatsListenerAccepted."  
 ::= { tcpEStatsListenerEntry 10 }

tcpEStatsListenerHCExceedBacklog OBJECT-TYPE  
SYNTAX ZeroBasedCounter64  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The total number of connections dropped from the  
    backlog by this listener due to all reasons on  
    systems that can process (or reject) more than  
    1 million connections per second. See  
    tcpEStatsListenerExceedBacklog."  
 ::= { tcpEStatsListenerEntry 11 }

tcpEStatsListenerCurConns OBJECT-TYPE  
SYNTAX Gauge32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The current number of connections in the ESTABLISHED  
    state, which have also been accepted. It excludes  
    connections that have been established but not accepted  
    because they are still subject to being discarded to  
    shed load without explicit action by either endpoint."  
 ::= { tcpEStatsListenerEntry 12 }

tcpEStatsListenerMaxBacklog OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum number of connections allowed in the backlog at one time."

::= { tcpEStatsListenerEntry 13 }

tcpEStatsListenerCurBacklog OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current number of connections that are in the backlog. This gauge includes connections in ESTABLISHED or SYN-RECEIVED states for which the Listener has not yet issued an accept."

If this listener is using some technique to implicitly represent the SYN-RECEIVED states (e.g., by cryptographically encoding the state information in the initial sequence number, ISS), it MAY elect to exclude connections in the SYN-RECEIVED state from the backlog."

::= { tcpEStatsListenerEntry 14 }

tcpEStatsListenerCurEstabBacklog OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current number of connections in the backlog that are in the ESTABLISHED state, but for which the Listener has not yet issued an accept."

::= { tcpEStatsListenerEntry 15 }

```
-- =====
--
-- TCP Connection ID Table
--
```

tcpEStatsConnectIdTable OBJECT-TYPE

SYNTAX SEQUENCE OF TcpEStatsConnectIdEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table maps information that uniquely identifies each active TCP connection to the connection ID used by



other tables in this MIB Module. It is an extension of tcpConnectionTable in RFC 4022.

Entries are retained in this table for the number of seconds indicated by the tcpEStatsConnTableLatency object, after the TCP connection first enters the closed state."

```
::= { tcpEStats 2 }
```

tcpEStatsConnectIdEntry OBJECT-TYPE

SYNTAX TcpEStatsConnectIdEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Each entry in this table maps a TCP connection 4-tuple to a connection index."

AUGMENTS { tcpConnectionEntry }

```
::= { tcpEStatsConnectIdTable 1 }
```

TcpEStatsConnectIdEntry ::= SEQUENCE {

tcpEStatsConnectIndex

Unsigned32

}

tcpEStatsConnectIndex OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A unique integer value assigned to each TCP Connection entry.

The RECOMMENDED algorithm is to begin at 1 and increase to some implementation-specific maximum value and then start again at 1 skipping values already in use."

```
::= { tcpEStatsConnectIdEntry 1 }
```

```
-- =====
--
-- Basic TCP Performance Statistics
--
```

tcpEStatsPerfTable OBJECT-TYPE

SYNTAX SEQUENCE OF TcpEStatsPerfEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains objects that are useful for

measuring TCP performance and first line problem diagnosis. Most objects in this table directly expose some TCP state variable or are easily implemented as simple functions (e.g., the maximum value) of TCP state variables.

Entries are retained in this table for the number of seconds indicated by the tcpEStatsConnTableLatency object, after the TCP connection first enters the closed state."

```
::= { tcpEStats 3 }
```

tcpEStatsPerfEntry OBJECT-TYPE

SYNTAX TcpeStatsPerfEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Each entry in this table has information about the characteristics of each active and recently closed TCP connection."

INDEX { tcpEStatsConnectIndex }

```
::= { tcpEStatsPerfTable 1 }
```

TcpEStatsPerfEntry ::= SEQUENCE {

tcpEStatsPerfSegsOut	ZeroBasedCounter32,
tcpEStatsPerfDataSegsOut	ZeroBasedCounter32,
tcpEStatsPerfDataOctetsOut	ZeroBasedCounter32,
tcpEStatsPerfHCDataOctetsOut	ZeroBasedCounter64,
tcpEStatsPerfSegsRetrans	ZeroBasedCounter32,
tcpEStatsPerfOctetsRetrans	ZeroBasedCounter32,
tcpEStatsPerfSegsIn	ZeroBasedCounter32,
tcpEStatsPerfDataSegsIn	ZeroBasedCounter32,
tcpEStatsPerfDataOctetsIn	ZeroBasedCounter32,
tcpEStatsPerfHCDataOctetsIn	ZeroBasedCounter64,
tcpEStatsPerfElapsedSecs	ZeroBasedCounter32,
tcpEStatsPerfElapsedMicroSecs	ZeroBasedCounter32,
tcpEStatsPerfStartTimeStamp	DateAndTime,
tcpEStatsPerfCurMSS	Gauge32,
tcpEStatsPerfPipeSize	Gauge32,
tcpEStatsPerfMaxPipeSize	Gauge32,
tcpEStatsPerfSmoothedRTT	Gauge32,
tcpEStatsPerfCurRTO	Gauge32,
tcpEStatsPerfCongSignals	ZeroBasedCounter32,
tcpEStatsPerfCurCwnd	Gauge32,
tcpEStatsPerfCurSsthresh	Gauge32,
tcpEStatsPerfTimeouts	ZeroBasedCounter32,
tcpEStatsPerfCurRwinSent	Gauge32,

```

        tcpEStatsPerfMaxRwinSent      Gauge32,
        tcpEStatsPerfZeroRwinSent     ZeroBasedCounter32,
        tcpEStatsPerfCurRwinRcvd     Gauge32,
        tcpEStatsPerfMaxRwinRcvd      Gauge32,
        tcpEStatsPerfZeroRwinRcvd     ZeroBasedCounter32,
        tcpEStatsPerfSndLimTransRwin  ZeroBasedCounter32,
        tcpEStatsPerfSndLimTransCwnd  ZeroBasedCounter32,
        tcpEStatsPerfSndLimTransSnd   ZeroBasedCounter32,
        tcpEStatsPerfSndLimTimeRwin   ZeroBasedCounter32,
        tcpEStatsPerfSndLimTimeCwnd   ZeroBasedCounter32,
        tcpEStatsPerfSndLimTimeSnd    ZeroBasedCounter32
    }

--
-- The following objects provide statistics on aggregate
-- segments and data sent on a connection. These provide a
-- direct measure of the Internet capacity consumed by a
-- connection.
--

tcpEStatsPerfSegsOut OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The total number of segments sent."
        ::= { tcpEStatsPerfEntry 1 }

tcpEStatsPerfDataSegsOut OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The number of segments sent containing a positive length
        data segment."
        ::= { tcpEStatsPerfEntry 2 }

tcpEStatsPerfDataOctetsOut OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS        "octets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The number of octets of data contained in transmitted
        segments, including retransmitted data. Note that this does
        not include TCP headers."
        ::= { tcpEStatsPerfEntry 3 }

```

tcpEStatsPerfHCDataOctetsOut OBJECT-TYPE  
SYNTAX ZeroBasedCounter64  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The number of octets of data contained in transmitted  
    segments, including retransmitted data, on systems that can  
    transmit more than 10 million bits per second. Note that  
    this does not include TCP headers."  
 ::= { tcpEStatsPerfEntry 4 }

tcpEStatsPerfSegsRetrans OBJECT-TYPE  
SYNTAX ZeroBasedCounter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The number of segments transmitted containing at least some  
    retransmitted data."  
REFERENCE  
    "RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsPerfEntry 5 }

tcpEStatsPerfOctetsRetrans OBJECT-TYPE  
SYNTAX ZeroBasedCounter32  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The number of octets retransmitted."  
REFERENCE  
    "RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsPerfEntry 6 }

tcpEStatsPerfSegsIn OBJECT-TYPE  
SYNTAX ZeroBasedCounter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The total number of segments received."  
 ::= { tcpEStatsPerfEntry 7 }

tcpEStatsPerfDataSegsIn OBJECT-TYPE  
SYNTAX ZeroBasedCounter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The number of segments received containing a positive

```
        length data segment."
 ::= { tcpEStatsPerfEntry 8 }

tcpEStatsPerfDataOctetsIn  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "octets"
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The number of octets contained in received data segments,
         including retransmitted data.  Note that this does not
         include TCP headers."
 ::= { tcpEStatsPerfEntry 9 }

tcpEStatsPerfHCDataOctetsIn  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter64
    UNITS       "octets"
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The number of octets contained in received data segments,
         including retransmitted data, on systems that can receive
         more than 10 million bits per second.  Note that this does
         not include TCP headers."
 ::= { tcpEStatsPerfEntry 10 }

tcpEStatsPerfElapsedSecs  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "seconds"
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The seconds part of the time elapsed between
         tcpEStatsPerfStartTimeStamp and the most recent protocol
         event (segment sent or received)."
 ::= { tcpEStatsPerfEntry 11 }

tcpEStatsPerfElapsedMicroSecs  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "microseconds"
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The micro-second part of time elapsed between
         tcpEStatsPerfStartTimeStamp to the most recent protocol
         event (segment sent or received).  This may be updated in
         whatever time granularity is the system supports."
 ::= { tcpEStatsPerfEntry 12 }
```

```

tcpEStatsPerfStartTimeStamp  OBJECT-TYPE
    SYNTAX          DateAndTime
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Time at which this row was created and all
        ZeroBasedCounters in the row were initialized to zero."
    ::= { tcpEStatsPerfEntry 13 }

--
-- The following objects can be used to fit minimal
-- performance models to the TCP data rate.
--

tcpEStatsPerfCurMSS  OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "octets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The current maximum segment size (MSS), in octets."
    REFERENCE
        "RFC 1122, Requirements for Internet Hosts - Communication
        Layers"
    ::= { tcpEStatsPerfEntry 14 }

tcpEStatsPerfPipeSize  OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           "octets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The TCP senders current estimate of the number of
        unacknowledged data octets in the network.

        While not in recovery (e.g., while the receiver is not
        reporting missing data to the sender), this is precisely the
        same as 'Flight size' as defined in RFC 2581, which can be
        computed as SND.NXT minus SND.UNA. [RFC793]

        During recovery, the TCP sender has incomplete information
        about the state of the network (e.g., which segments are
        lost vs reordered, especially if the return path is also
        dropping TCP acknowledgments).  Current TCP standards do not
        mandate any specific algorithm for estimating the number of
        unacknowledged data octets in the network.

        RFC 3517 describes a conservative algorithm to use SACK

```

information to estimate the number of unacknowledged data octets in the network. tcpEStatsPerfPipeSize object SHOULD be the same as 'pipe' as defined in RFC 3517 if it is implemented. (Note that while not in recovery the pipe algorithm yields the same values as flight size).

If RFC 3517 is not implemented, the data octets in flight SHOULD be estimated as SND.NXT minus SND.UNA adjusted by some measure of the data that has left the network and retransmitted data. For example, with Reno or NewReno style TCP, the number of duplicate acknowledgment is used to count the number of segments that have left the network. That is,

PipeSize=SND.NXT-SND.UNA+(retransmits-dupacks)\*CurMSS"

#### REFERENCE

"RFC 793, RFC 2581, RFC 3517"

::= { tcpEStatsPerfEntry 15 }

#### tcpEStatsPerfMaxPipeSize OBJECT-TYPE

SYNTAX Gauge32  
 UNITS "octets"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The maximum value of tcpEStatsPerfPipeSize, for this connection."

#### REFERENCE

"RFC 793, RFC 2581, RFC 3517"

::= { tcpEStatsPerfEntry 16 }

#### tcpEStatsPerfSmoothedRTT OBJECT-TYPE

SYNTAX Gauge32  
 UNITS "milliseconds"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The smoothed round trip time used in calculation of the RTO. See SRTT in [RFC2988]."

#### REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"

::= { tcpEStatsPerfEntry 17 }

#### tcpEStatsPerfCurRTO OBJECT-TYPE

SYNTAX Gauge32  
 UNITS "milliseconds"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The current value of the retransmit timer RTO."

#### REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"

::= { tcpEStatsPerfEntry 18 }

tcpEStatsPerfCongSignals OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of multiplicative downward congestion window adjustments due to all forms of congestion signals, including Fast Retransmit, Explicit Congestion Notification (ECN), and timeouts. This object summarizes all events that invoke the MD portion of Additive Increase Multiplicative Decrease (AIMD) congestion control, and as such is the best indicator of how a cwnd is being affected by congestion.

Note that retransmission timeouts multiplicatively reduce the window implicitly by setting ssthresh, and SHOULD be included in tcpEStatsPerfCongSignals. In order to minimize spurious congestion indications due to out-of-order segments, tcpEStatsPerfCongSignals SHOULD be incremented in association with the Fast Retransmit algorithm."

#### REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPerfEntry 19 }

tcpEStatsPerfCurCwnd OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The current congestion window, in octets."

#### REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPerfEntry 20 }

tcpEStatsPerfCurSsthresh OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The current slow start threshold in octets."

#### REFERENCE

"RFC 2581, TCP Congestion Control"



```

 ::= { tcpEStatsPerfEntry 21 }

tcpEStatsPerfTimeouts OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The number of times the retransmit timeout has expired when
         the RTO backoff multiplier is equal to one."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
 ::= { tcpEStatsPerfEntry 22 }

--
-- The following objects instrument receiver window updates
-- sent by the local receiver to the remote sender. These can
-- be used to determine if the local receiver is exerting flow
-- control back pressure on the remote sender.
--

tcpEStatsPerfCurRwinSent OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "octets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The most recent window advertisement sent, in octets."
    REFERENCE
        "RFC 793, Transmission Control Protocol"
 ::= { tcpEStatsPerfEntry 23 }

tcpEStatsPerfMaxRwinSent OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "octets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The maximum window advertisement sent, in octets."
    REFERENCE
        "RFC 793, Transmission Control Protocol"
 ::= { tcpEStatsPerfEntry 24 }

tcpEStatsPerfZeroRwinSent OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The number of acknowledgments sent announcing a zero

```

receive window, when the previously announced window was not zero."

## REFERENCE

"RFC 793, Transmission Control Protocol"  
::= { tcpEStatsPerfEntry 25 }

--

-- The following objects instrument receiver window updates  
-- from the far end-system to determine if the remote receiver  
-- has sufficient buffer space or is exerting flow-control  
-- back pressure on the local sender.  
--

## tcpEStatsPerfCurRwinRcvd OBJECT-TYPE

SYNTAX Gauge32  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"The most recent window advertisement received, in octets."

## REFERENCE

"RFC 793, Transmission Control Protocol"  
::= { tcpEStatsPerfEntry 26 }

## tcpEStatsPerfMaxRwinRcvd OBJECT-TYPE

SYNTAX Gauge32  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"The maximum window advertisement received, in octets."

## REFERENCE

"RFC 793, Transmission Control Protocol"  
::= { tcpEStatsPerfEntry 27 }

## tcpEStatsPerfZeroRwinRcvd OBJECT-TYPE

SYNTAX ZeroBasedCounter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"The number of acknowledgments received announcing a zero receive window, when the previously announced window was not zero."

## REFERENCE

"RFC 793, Transmission Control Protocol"  
::= { tcpEStatsPerfEntry 28 }

--

```
-- The following optional objects can be used to quickly
-- identify which subsystems are limiting TCP performance.
-- There are three parallel pairs of instruments that measure
-- the extent to which TCP performance is limited by the
-- announced receiver window (indicating a receiver
-- bottleneck), the current congestion window or
-- retransmission timeout (indicating a path bottleneck) and
-- all others events (indicating a sender bottleneck).
--
-- These instruments SHOULD be updated every time the TCP
-- output routine stops sending data. The elapsed time since
-- the previous stop is accumulated into the appropriate
-- object as determined by the previous stop reason (e.g.,
-- stop state). The current stop reason determines which timer
-- will be updated the next time TCP output stops.
--
-- Since there is no explicit stop at the beginning of a
-- timeout, it is necessary to retroactively reclassify the
-- previous stop as 'Congestion Limited'.
```

```
tcpEStatsPerfSndLimTransRwin OBJECT-TYPE
    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of transitions into the 'Receiver Limited' state
        from either the 'Congestion Limited' or 'Sender Limited'
        states. This state is entered whenever TCP transmission
        stops because the sender has filled the announced receiver
        window, i.e., when SND.NXT has advanced to SND.UNA +
        SND.WND - 1 as described in RFC 793."
    REFERENCE
        "RFC 793, Transmission Control Protocol"
        ::= { tcpEStatsPerfEntry 31 }
```

```
tcpEStatsPerfSndLimTransCwnd OBJECT-TYPE
    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of transitions into the 'Congestion Limited'
        state from either the 'Receiver Limited' or 'Sender
        Limited' states. This state is entered whenever TCP
        transmission stops because the sender has reached some
        limit defined by congestion control (e.g., cwnd) or other
        algorithms (retransmission timeouts) designed to control
        network traffic. See the definition of 'CONGESTION WINDOW'
```

in RFC 2581."

#### REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPerfEntry 32 }

tcpEStatsPerfSndLimTransSnd OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of transitions into the 'Sender Limited' state from either the 'Receiver Limited' or 'Congestion Limited' states. This state is entered whenever TCP transmission stops due to some sender limit such as running out of application data or other resources and the Karn algorithm. When TCP stops sending data for any reason, which cannot be classified as Receiver Limited or Congestion Limited, it MUST be treated as Sender Limited."

::= { tcpEStatsPerfEntry 33 }

tcpEStatsPerfSndLimTimeRwin OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "milliseconds"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The cumulative time spent in the 'Receiver Limited' state. See tcpEStatsPerfSndLimTransRwin."

::= { tcpEStatsPerfEntry 34 }

tcpEStatsPerfSndLimTimeCwnd OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "milliseconds"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The cumulative time spent in the 'Congestion Limited' state. See tcpEStatsPerfSndLimTransCwnd. When there is a retransmission timeout, it SHOULD be counted in tcpEStatsPerfSndLimTimeCwnd (and not the cumulative time for some other state.)"

::= { tcpEStatsPerfEntry 35 }

tcpEStatsPerfSndLimTimeSnd OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "milliseconds"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The cumulative time spent in the 'Sender Limited' state.  
See tcpEStatsPerfSndLimTransSnd."

::= { tcpEStatsPerfEntry 36 }

```
-- =====
--
-- Statistics for diagnosing path problems
--
```

tcpEStatsPathTable OBJECT-TYPE

SYNTAX SEQUENCE OF TcpEStatsPathEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This table contains objects that can be used to infer detailed behavior of the Internet path, such as the extent that there is reordering, ECN bits, and if RTT fluctuations are correlated to losses.

Entries are retained in this table for the number of seconds indicated by the tcpEStatsConnTableLatency object, after the TCP connection first enters the closed state."

::= { tcpEStats 4 }

tcpEStatsPathEntry OBJECT-TYPE

SYNTAX TcpEStatsPathEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"Each entry in this table has information about the characteristics of each active and recently closed TCP connection."

INDEX { tcpEStatsConnectIndex }

::= { tcpEStatsPathTable 1 }

TcpEStatsPathEntry ::= SEQUENCE {

tcpEStatsPathRetranThresh	Gauge32,
tcpEStatsPathNonRecovDAEpisodes	ZeroBasedCounter32,
tcpEStatsPathSumOctetsReordered	ZeroBasedCounter32,
tcpEStatsPathNonRecovDA	ZeroBasedCounter32,
tcpEStatsPathSamplerRTT	Gauge32,
tcpEStatsPathRTTVar	Gauge32,
tcpEStatsPathMaxRTT	Gauge32,
tcpEStatsPathMinRTT	Gauge32,
tcpEStatsPathSumRTT	ZeroBasedCounter32,

```

    tcpEStatsPathHCSumRTT          ZeroBasedCounter64,
    tcpEStatsPathCountRTT          ZeroBasedCounter32,
    tcpEStatsPathMaxRTO            Gauge32,
    tcpEStatsPathMinRTO            Gauge32,
    tcpEStatsPathIpTtl             Unsigned32,
    tcpEStatsPathIpTosIn           OCTET STRING,
    tcpEStatsPathIpTosOut          OCTET STRING,
    tcpEStatsPathPreCongSumCwnd     ZeroBasedCounter32,
    tcpEStatsPathPreCongSumRTT     ZeroBasedCounter32,
    tcpEStatsPathPostCongSumRTT    ZeroBasedCounter32,
    tcpEStatsPathPostCongCountRTT  ZeroBasedCounter32,
    tcpEStatsPathECNsignals         ZeroBasedCounter32,
    tcpEStatsPathDupAckEpisodes     ZeroBasedCounter32,
    tcpEStatsPathRcvRTT            Gauge32,
    tcpEStatsPathDupAcksOut         ZeroBasedCounter32,
    tcpEStatsPathCERcvd            ZeroBasedCounter32,
    tcpEStatsPathECESent           ZeroBasedCounter32
}

```

```

--
-- The following optional objects can be used to infer segment
-- reordering on the path from the local sender to the remote
-- receiver.
--

```

tcpEStatsPathRetranThresh OBJECT-TYPE

```

    SYNTAX          Gauge32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION

```

"The number of duplicate acknowledgments required to trigger Fast Retransmit. Note that although this is constant in traditional Reno TCP implementations, it is adaptive in many newer TCPs."

REFERENCE

"RFC 2581, TCP Congestion Control"

```

 ::= { tcpEStatsPathEntry 1 }

```

tcpEStatsPathNonRecovDAEpisodes OBJECT-TYPE

```

    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION

```

"The number of duplicate acknowledgment episodes that did not trigger a Fast Retransmit because ACK advanced prior to the number of duplicate acknowledgments reaching RetranThresh."

In many implementations this is the number of times the 'dupacks' counter is set to zero when it is non-zero but less than RetranThresh.

Note that the change in tcpEStatsPathNonRecovDAEpisodes divided by the change in tcpEStatsPerfDataSegsOut is an estimate of the frequency of data reordering on the forward path over some interval."

#### REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPathEntry 2 }

tcpEStatsPathSumOctetsReordered OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The sum of the amounts SND.UNA advances on the acknowledgment which ends a dup-ack episode without a retransmission.

Note the change in tcpEStatsPathSumOctetsReordered divided by the change in tcpEStatsPathNonRecovDAEpisodes is an estimate of the average reordering distance, over some interval."

::= { tcpEStatsPathEntry 3 }

tcpEStatsPathNonRecovDA OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"Duplicate acks (or SACKS) that did not trigger a Fast Retransmit because ACK advanced prior to the number of duplicate acknowledgments reaching RetranThresh.

In many implementations, this is the sum of the 'dupacks' counter, just before it is set to zero because ACK advanced without a Fast Retransmit.

Note that the change in tcpEStatsPathNonRecovDA divided by the change in tcpEStatsPathNonRecovDAEpisodes is an estimate of the average reordering distance in segments over some interval."

#### REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPathEntry 4 }

```
--
-- The following optional objects instrument the round trip
-- time estimator and the retransmission timeout timer.
--

tcpEStatsPathSampleRTT OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "milliseconds"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The most recent raw round trip time measurement used in
        calculation of the RTO."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
    ::= { tcpEStatsPathEntry 11 }

tcpEStatsPathRTTVar OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "milliseconds"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The round trip time variation used in calculation of the
        RTO. See RTTVAR in [RFC2988]."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
    ::= { tcpEStatsPathEntry 12 }

tcpEStatsPathMaxRTT OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "milliseconds"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The maximum sampled round trip time."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
    ::= { tcpEStatsPathEntry 13 }

tcpEStatsPathMinRTT OBJECT-TYPE
    SYNTAX      Gauge32
    UNITS        "milliseconds"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The minimum sampled round trip time."
    REFERENCE
```



"RFC 2988, Computing TCP's Retransmission Timer"  
 ::= { tcpEStatsPathEntry 14 }

tcpEStatsPathSumRTT OBJECT-TYPE  
 SYNTAX ZeroBasedCounter32  
 UNITS "milliseconds"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The sum of all sampled round trip times.

Note that the change in tcpEStatsPathSumRTT divided by the change in tcpEStatsPathCountRTT is the mean RTT, uniformly averaged over an enter interval."

#### REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"  
 ::= { tcpEStatsPathEntry 15 }

tcpEStatsPathHCSumRTT OBJECT-TYPE  
 SYNTAX ZeroBasedCounter64  
 UNITS "milliseconds"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The sum of all sampled round trip times, on all systems that implement multiple concurrent RTT measurements.

Note that the change in tcpEStatsPathHCSumRTT divided by the change in tcpEStatsPathCountRTT is the mean RTT, uniformly averaged over an enter interval."

#### REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"  
 ::= { tcpEStatsPathEntry 16 }

tcpEStatsPathCountRTT OBJECT-TYPE  
 SYNTAX ZeroBasedCounter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The number of round trip time samples included in tcpEStatsPathSumRTT and tcpEStatsPathHCSumRTT."

#### REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"  
 ::= { tcpEStatsPathEntry 17 }

tcpEStatsPathMaxRTO OBJECT-TYPE  
 SYNTAX Gauge32  
 UNITS "milliseconds"

```

MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "The maximum value of the retransmit timer RTO."
REFERENCE
    "RFC 2988, Computing TCP's Retransmission Timer"
 ::= { tcpEStatsPathEntry 18 }

```

```

tcpEStatsPathMinRTO  OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS            "milliseconds"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The minimum value of the retransmit timer RTO."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
    ::= { tcpEStatsPathEntry 19 }

```

```

--
-- The following optional objects provide information about
-- how TCP is using the IP layer.
--

```

```

tcpEStatsPathIpTtl  OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The value of the TTL field carried in the most recently
         received IP header. This is sometimes useful to detect
         changing or unstable routes."
    REFERENCE
        "RFC 791, Internet Protocol"
    ::= { tcpEStatsPathEntry 20 }

```

```

tcpEStatsPathIpTosIn  OBJECT-TYPE
    SYNTAX          OCTET STRING (SIZE(1))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The value of the IPv4 Type of Service octet, or the IPv6
         traffic class octet, carried in the most recently received
         IP header.

        This is useful to diagnose interactions between TCP and any
        IP layer packet scheduling and delivery policy, which might
        be in effect to implement Diffserv."

```

## REFERENCE

"RFC 3260, New Terminology and Clarifications for Diffserv"  
 ::= { tcpEStatsPathEntry 21 }

tcpEStatsPathIpTosOut OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1))

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The value of the IPv4 Type Of Service octet, or the IPv6 traffic class octet, carried in the most recently transmitted IP header.

This is useful to diagnose interactions between TCP and any IP layer packet scheduling and delivery policy, which might be in effect to implement Diffserv."

## REFERENCE

"RFC 3260, New Terminology and Clarifications for Diffserv"  
 ::= { tcpEStatsPathEntry 22 }

--

-- The following optional objects characterize the congestion  
 -- feedback signals by collecting statistics on how the  
 -- congestion events are correlated to losses, changes in RTT  
 -- and other protocol events.

--

tcpEStatsPathPreCongSumCwnd OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The sum of the values of the congestion window, in octets, captured each time a congestion signal is received. This MUST be updated each time tcpEStatsPerfCongSignals is incremented, such that the change in tcpEStatsPathPreCongSumCwnd divided by the change in tcpEStatsPerfCongSignals is the average window (over some interval) just prior to a congestion signal."

::= { tcpEStatsPathEntry 23 }

tcpEStatsPathPreCongSumRTT OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "milliseconds"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"Sum of the last sample of the RTT (tcpEStatsPathSampleRTT) prior to the received congestion signals. This MUST be updated each time tcpEStatsPerfCongSignals is incremented, such that the change in tcpEStatsPathPreCongSumRTT divided by the change in tcpEStatsPerfCongSignals is the average RTT (over some interval) just prior to a congestion signal."

```
 ::= { tcpEStatsPathEntry 24 }
```

tcpEStatsPathPostCongSumRTT OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Sum of the first sample of the RTT (tcpEStatsPathSampleRTT) following each congestion signal. Such that the change in tcpEStatsPathPostCongSumRTT divided by the change in tcpEStatsPathPostCongCountRTT is the average RTT (over some interval) just after a congestion signal."

```
 ::= { tcpEStatsPathEntry 25 }
```

tcpEStatsPathPostCongCountRTT OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "milliseconds"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of RTT samples included in tcpEStatsPathPostCongSumRTT such that the change in tcpEStatsPathPostCongSumRTT divided by the change in tcpEStatsPathPostCongCountRTT is the average RTT (over some interval) just after a congestion signal."

```
 ::= { tcpEStatsPathEntry 26 }
```

--

-- The following optional objects can be used to detect other  
 -- types of non-loss congestion signals such as source quench  
 -- or ECN.

--

tcpEStatsPathECNsignals OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of congestion signals delivered to the TCP sender via explicit congestion notification (ECN). This is typically the number of segments bearing Echo Congestion

Experienced (ECE) bits, but should also include segments failing the ECN nonce check or other explicit congestion signals."

## REFERENCE

"RFC 3168, The Addition of Explicit Congestion Notification (ECN) to IP"

::= { tcpEStatsPathEntry 27 }

```
--
-- The following optional objects are receiver side
-- instruments of the path from the sender to the receiver. In
-- general, the receiver has less information about the state
-- of the path because the receiver does not have a robust
-- mechanism to infer the sender's actions.
--
```

## tcpEStatsPathDupAckEpisodes OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of Duplicate Acks Sent when prior Ack was not duplicate. This is the number of times that a contiguous series of duplicate acknowledgments have been sent.

This is an indication of the number of data segments lost or reordered on the path from the remote TCP endpoint to the near TCP endpoint."

## REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPathEntry 28 }

## tcpEStatsPathRcvRTT OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The receiver's estimate of the Path RTT.

Adaptive receiver window algorithms depend on the receiver to having a good estimate of the path RTT."

::= { tcpEStatsPathEntry 29 }

## tcpEStatsPathDupAcksOut OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of duplicate ACKs sent. The ratio of the change in tcpEStatsPathDupAcksOut to the change in tcpEStatsPathDupAckEpisodes is an indication of reorder or recovery distance over some interval."

## REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsPathEntry 30 }

tcpEStatsPathCERcvd OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of segments received with IP headers bearing Congestion Experienced (CE) markings."

## REFERENCE

"RFC 3168, The Addition of Explicit Congestion Notification (ECN) to IP"

::= { tcpEStatsPathEntry 31 }

tcpEStatsPathECESent OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"Number of times the Echo Congestion Experienced (ECE) bit in the TCP header has been set (transitioned from 0 to 1), due to a Congestion Experienced (CE) marking on an IP header. Note that ECE can be set and reset only once per RTT, while CE can be set on many segments per RTT."

## REFERENCE

"RFC 3168, The Addition of Explicit Congestion Notification (ECN) to IP"

::= { tcpEStatsPathEntry 32 }

```
-- =====
--
-- Statistics for diagnosing stack algorithms
--
```

tcpEStatsStackTable OBJECT-TYPE

SYNTAX SEQUENCE OF TcpEStatsStackEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This table contains objects that are most useful for determining how well some of the TCP control algorithms are coping with this particular

path.

Entries are retained in this table for the number of seconds indicated by the tcpEStatsConnTableLatency object, after the TCP connection first enters the closed state."

```
::= { tcpEStats 5 }
```

tcpEStatsStackEntry OBJECT-TYPE

SYNTAX TcPEStatsStackEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Each entry in this table has information about the characteristics of each active and recently closed TCP connection."

INDEX { tcpEStatsConnectIndex }

```
::= { tcpEStatsStackTable 1 }
```

TcpEStatsStackEntry ::= SEQUENCE {

tcpEStatsStackActiveOpen	TruthValue,
tcpEStatsStackMSSSent	Unsigned32,
tcpEStatsStackMSSRcvd	Unsigned32,
tcpEStatsStackWinScaleSent	Integer32,
tcpEStatsStackWinScaleRcvd	Integer32,
tcpEStatsStackTimeStamps	TcpEStatsNegotiated,
tcpEStatsStackECN	TcpEStatsNegotiated,
tcpEStatsStackWillSendSACK	TcpEStatsNegotiated,
tcpEStatsStackWillUseSACK	TcpEStatsNegotiated,
tcpEStatsStackState	INTEGER,
tcpEStatsStackNagle	TruthValue,
tcpEStatsStackMaxSsCwnd	Gauge32,
tcpEStatsStackMaxCaCwnd	Gauge32,
tcpEStatsStackMaxSsthresh	Gauge32,
tcpEStatsStackMinSsthresh	Gauge32,
tcpEStatsStackInRecovery	INTEGER,
tcpEStatsStackDupAcksIn	ZeroBasedCounter32,
tcpEStatsStackSpuriousFrDetected	ZeroBasedCounter32,
tcpEStatsStackSpuriousRtoDetected	ZeroBasedCounter32,
tcpEStatsStackSoftErrors	ZeroBasedCounter32,
tcpEStatsStackSoftErrorReason	INTEGER,
tcpEStatsStackSlowStart	ZeroBasedCounter32,
tcpEStatsStackCongAvoid	ZeroBasedCounter32,
tcpEStatsStackOtherReductions	ZeroBasedCounter32,
tcpEStatsStackCongOverCount	ZeroBasedCounter32,
tcpEStatsStackFastRetran	ZeroBasedCounter32,
tcpEStatsStackSubsequentTimeouts	ZeroBasedCounter32,

tcpEStatsStackCurTimeoutCount	Gauge32,
tcpEStatsStackAbruptTimeouts	ZeroBasedCounter32,
tcpEStatsStackSACKsRcvd	ZeroBasedCounter32,
tcpEStatsStackSACKBlocksRcvd	ZeroBasedCounter32,
tcpEStatsStackSendStall	ZeroBasedCounter32,
tcpEStatsStackDSACKDups	ZeroBasedCounter32,
tcpEStatsStackMaxMSS	Gauge32,
tcpEStatsStackMinMSS	Gauge32,
tcpEStatsStackSndInitial	Unsigned32,
tcpEStatsStackRecInitial	Unsigned32,
tcpEStatsStackCurRetxQueue	Gauge32,
tcpEStatsStackMaxRetxQueue	Gauge32,
tcpEStatsStackCurReasmQueue	Gauge32,
tcpEStatsStackMaxReasmQueue	Gauge32

}

--  
 -- The following objects reflect TCP options carried on the  
 -- SYN or SYN-ACK. These options are used to provide  
 -- additional protocol parameters or to enable various  
 -- optional TCP features or algorithms.  
 --  
 -- Except as noted, the TCP protocol does not permit these  
 -- options to change after the SYN exchange.  
 --

tcpEStatsStackActiveOpen OBJECT-TYPE  
 SYNTAX TruthValue  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "True(1) if the local connection traversed the SYN-SENT  
 state, else false(2)."  
 REFERENCE  
 "RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsStackEntry 1 }

tcpEStatsStackMSSSent OBJECT-TYPE  
 SYNTAX Unsigned32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The value sent in an MSS option, or zero if none."  
 REFERENCE  
 "RFC 1122, Requirements for Internet Hosts - Communication  
 Layers"  
 ::= { tcpEStatsStackEntry 2 }



tcpEStatsStackMSSRcvd OBJECT-TYPE  
SYNTAX Unsigned32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The value received in an MSS option, or zero if none."  
REFERENCE  
"RFC 1122, Requirements for Internet Hosts - Communication Layers"  
::= { tcpEStatsStackEntry 3 }

tcpEStatsStackWinScaleSent OBJECT-TYPE  
SYNTAX Integer32 (-1..14)  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The value of the transmitted window scale option if one was sent; otherwise, a value of -1.  
  
Note that if both tcpEStatsStackWinScaleSent and tcpEStatsStackWinScaleRcvd are not -1, then Rcv.Wind.Scale will be the same as this value and used to scale receiver window announcements from the local host to the remote host."  
REFERENCE  
"RFC 1323, TCP Extensions for High Performance"  
::= { tcpEStatsStackEntry 4 }

tcpEStatsStackWinScaleRcvd OBJECT-TYPE  
SYNTAX Integer32 (-1..14)  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The value of the received window scale option if one was received; otherwise, a value of -1.  
  
Note that if both tcpEStatsStackWinScaleSent and tcpEStatsStackWinScaleRcvd are not -1, then Snd.Wind.Scale will be the same as this value and used to scale receiver window announcements from the remote host to the local host."  
REFERENCE  
"RFC 1323, TCP Extensions for High Performance"  
::= { tcpEStatsStackEntry 5 }

tcpEStatsStackTimeStamps OBJECT-TYPE  
SYNTAX TcpEStatsNegotiated  
MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Enabled(1) if TCP timestamps have been negotiated on, selfDisabled(2) if they are disabled or not implemented on the local host, or peerDisabled(3) if not negotiated by the remote hosts."

REFERENCE

"RFC 1323, TCP Extensions for High Performance"

::= { tcpEStatsStackEntry 6 }

tcpEStatsStackECN OBJECT-TYPE

SYNTAX TcpEStatsNegotiated

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Enabled(1) if Explicit Congestion Notification (ECN) has been negotiated on, selfDisabled(2) if it is disabled or not implemented on the local host, or peerDisabled(3) if not negotiated by the remote hosts."

REFERENCE

"RFC 3168, The Addition of Explicit Congestion Notification (ECN) to IP"

::= { tcpEStatsStackEntry 7 }

tcpEStatsStackWillSendSACK OBJECT-TYPE

SYNTAX TcpEStatsNegotiated

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Enabled(1) if the local host will send SACK options, selfDisabled(2) if SACK is disabled or not implemented on the local host, or peerDisabled(3) if the remote host did not send the SACK-permitted option."

Note that SACK negotiation is not symmetrical. SACK can be enabled on one side of the connection and not the other."

REFERENCE

"RFC 2018, TCP Selective Acknowledgement Options"

::= { tcpEStatsStackEntry 8 }

tcpEStatsStackWillUseSACK OBJECT-TYPE

SYNTAX TcpEStatsNegotiated

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Enabled(1) if the local host will process SACK options, selfDisabled(2) if SACK is disabled or not implemented on the local host, or peerDisabled(3) if the remote host sends

duplicate ACKs without SACK options, or the local host otherwise decides not to process received SACK options.

Unlike other TCP options, the remote data receiver cannot explicitly indicate if it is able to generate SACK options. When sending data, the local host has to deduce if the remote receiver is sending SACK options. This object can transition from Enabled(1) to peerDisabled(3) after the SYN exchange.

Note that SACK negotiation is not symmetrical. SACK can be enabled on one side of the connection and not the other."

#### REFERENCE

"RFC 2018, TCP Selective Acknowledgement Options"  
 ::= { tcpEStatsStackEntry 9 }

```
--
-- The following two objects reflect the current state of the
-- connection.
--
```

tcpEStatsStackState OBJECT-TYPE

```
SYNTAX          INTEGER {
    tcpESStateClosed(1),
    tcpESStateListen(2),
    tcpESStateSynSent(3),
    tcpESStateSynReceived(4),
    tcpESStateEstablished(5),
    tcpESStateFinWait1(6),
    tcpESStateFinWait2(7),
    tcpESStateCloseWait(8),
    tcpESStateLastAck(9),
    tcpESStateClosing(10),
    tcpESStateTimeWait(11),
    tcpESStateDeleteTcb(12)
}
```

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"An integer value representing the connection state from the TCP State Transition Diagram.

The value listen(2) is included only for parallelism to the old tcpConnTable, and SHOULD NOT be used because the listen state is managed by the tcpListenerTable.

The value DeleteTcb(12) is included only for parallelism to the tcpConnTable mechanism for terminating connections,

although this table does not permit writing."

REFERENCE

"RFC 793, Transmission Control Protocol"

::= { tcpEStatsStackEntry 10 }

tcpEStatsStackNagle OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"True(1) if the Nagle algorithm is being used, else false(2)."

REFERENCE

"RFC 1122, Requirements for Internet Hosts - Communication Layers"

::= { tcpEStatsStackEntry 11 }

--

-- The following objects instrument the overall operation of

-- TCP congestion control and data retransmissions. These

-- instruments are sufficient to fit the actual performance to

-- an updated macroscopic performance model [RFC2581] [Mat97]

-- [Pad98].

--

tcpEStatsStackMaxSsCwnd OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum congestion window used during Slow Start, in octets."

REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsStackEntry 12 }

tcpEStatsStackMaxCaCwnd OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum congestion window used during Congestion Avoidance, in octets."

REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsStackEntry 13 }

**tcpEStatsStackMaxSsthresh OBJECT-TYPE**

SYNTAX Gauge32  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"The maximum slow start threshold, excluding the initial value."

**REFERENCE**

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsStackEntry 14 }

**tcpEStatsStackMinSsthresh OBJECT-TYPE**

SYNTAX Gauge32  
UNITS "octets"  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"The minimum slow start threshold."

**REFERENCE**

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsStackEntry 15 }

**tcpEStatsStackInRecovery OBJECT-TYPE**

SYNTAX INTEGER {  
tcpESDataContiguous(1),  
tcpESDataUnordered(2),  
tcpESDataRecovery(3)  
}

MAX-ACCESS read-only  
STATUS current  
DESCRIPTION

"An integer value representing the state of the loss recovery for this connection."

tcpESDataContiguous(1) indicates that the remote receiver is reporting contiguous data (no duplicate acknowledgments or SACK options) and that there are no unacknowledged retransmissions.

tcpESDataUnordered(2) indicates that the remote receiver is reporting missing or out-of-order data (e.g., sending duplicate acknowledgments or SACK options) and that there are no unacknowledged retransmissions (because the missing data has not yet been retransmitted).

tcpESDataRecovery(3) indicates that the sender has outstanding retransmitted data that is still

```

        unacknowledged."
REFERENCE
    "RFC 2581, TCP Congestion Control"
    ::= { tcpEStatsStackEntry 16 }

tcpEStatsStackDupAcksIn  OBJECT-TYPE
    SYNTAX                ZeroBasedCounter32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
        "The number of duplicate ACKs received."
REFERENCE
    "RFC 2581, TCP Congestion Control"
    ::= { tcpEStatsStackEntry 17 }

tcpEStatsStackSpuriousFrDetected  OBJECT-TYPE
    SYNTAX                ZeroBasedCounter32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
        "The number of acknowledgments reporting out-of-order
        segments after the Fast Retransmit algorithm has already
        retransmitted the segments. (For example as detected by the
        Eifel algorithm).'"
REFERENCE
    "RFC 3522, The Eifel Detection Algorithm for TCP"
    ::= { tcpEStatsStackEntry 18 }

tcpEStatsStackSpuriousRtoDetected  OBJECT-TYPE
    SYNTAX                ZeroBasedCounter32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
        "The number of acknowledgments reporting segments that have
        already been retransmitted due to a Retransmission Timeout."
    ::= { tcpEStatsStackEntry 19 }

--
-- The following optional objects instrument unusual protocol
-- events that probably indicate implementation problems in
-- the protocol or path.
--

tcpEStatsStackSoftErrors  OBJECT-TYPE
    SYNTAX                ZeroBasedCounter32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION

```

"The number of segments that fail various consistency tests during TCP input processing. Soft errors might cause the segment to be discarded but some do not. Some of these soft errors cause the generation of a TCP acknowledgment, while others are silently discarded."

## REFERENCE

"RFC 793, Transmission Control Protocol"

::= { tcpEStatsStackEntry 21 }

tcpEStatsStackSoftErrorReason OBJECT-TYPE

```
SYNTAX          INTEGER {
    belowDataWindow(1),
    aboveDataWindow(2),
    belowAckWindow(3),
    aboveAckWindow(4),
    belowTSWindow(5),
    aboveTSWindow(6),
    dataChecksum(7),
    otherSoftError(8)
}
```

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object identifies which consistency test most recently failed during TCP input processing. This object SHOULD be set every time tcpEStatsStackSoftErrors is incremented. The codes are as follows:

belowDataWindow(1) - All data in the segment is below SND.UNA. (Normal for keep-alives and zero window probes).

aboveDataWindow(2) - Some data in the segment is above SND.WND. (Indicates an implementation bug or possible attack).

belowAckWindow(3) - ACK below SND.UNA. (Indicates that the return path is reordering ACKs)

aboveAckWindow(4) - An ACK for data that we have not sent. (Indicates an implementation bug or possible attack).

belowTSWindow(5) - TSecr on the segment is older than the current TS.Recent (Normal for the rare case where PAWS detects data reordered by the network).

aboveTSWindow(6) - TSecr on the segment is newer than the current TS.Recent. (Indicates an implementation bug or possible attack).

dataChecksum(7) - Incorrect checksum. Note that this value is intrinsically fragile, because the header fields used to identify the connection may have been corrupted.

otherSoftError(8) - All other soft errors not listed above."

#### REFERENCE

"RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsStackEntry 22 }

--  
 -- The following optional objects expose the detailed  
 -- operation of the congestion control algorithms.  
 --

#### tcpEStatsStackSlowStart OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of times the congestion window has been increased by the Slow Start algorithm."

#### REFERENCE

"RFC 2581, TCP Congestion Control"  
 ::= { tcpEStatsStackEntry 23 }

#### tcpEStatsStackCongAvoid OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of times the congestion window has been increased by the Congestion Avoidance algorithm."

#### REFERENCE

"RFC 2581, TCP Congestion Control"  
 ::= { tcpEStatsStackEntry 24 }

#### tcpEStatsStackOtherReductions OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of congestion window reductions made as a result of anything other than AIMD congestion control algorithms. Examples of non-multiplicative window reductions include Congestion Window Validation [RFC2861] and experimental algorithms such as Vegas [Bra94]."



All window reductions MUST be counted as either  
tcpEStatsPerfCongSignals or tcpEStatsStackOtherReductions."

## REFERENCE

"RFC 2861, TCP Congestion Window Validation"

::= { tcpEStatsStackEntry 25 }

tcpEStatsStackCongOverCount OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of congestion events that were 'backed out' of the congestion control state machine such that the congestion window was restored to a prior value. This can happen due to the Eifel algorithm [RFC3522] or other algorithms that can be used to detect and cancel spurious invocations of the Fast Retransmit Algorithm.

Although it may be feasible to undo the effects of spurious invocation of the Fast Retransmit congestion events cannot easily be backed out of tcpEStatsPerfCongSignals and tcpEStatsPathPreCongSumCwnd, etc."

## REFERENCE

"RFC 3522, The Eifel Detection Algorithm for TCP"

::= { tcpEStatsStackEntry 26 }

tcpEStatsStackFastRetran OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of invocations of the Fast Retransmit algorithm."

## REFERENCE

"RFC 2581, TCP Congestion Control"

::= { tcpEStatsStackEntry 27 }

tcpEStatsStackSubsequentTimeouts OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of times the retransmit timeout has expired after the RTO has been doubled. See Section 5.5 of RFC 2988."

## REFERENCE

"RFC 2988, Computing TCP's Retransmission Timer"

::= { tcpEStatsStackEntry 28 }

```

tcpEStatsStackCurTimeoutCount  OBJECT-TYPE
    SYNTAX          Gauge32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The current number of times the retransmit timeout has
        expired without receiving an acknowledgment for new data.
        tcpEStatsStackCurTimeoutCount is reset to zero when new
        data is acknowledged and incremented for each invocation of
        Section 5.5 of RFC 2988."
    REFERENCE
        "RFC 2988, Computing TCP's Retransmission Timer"
    ::= { tcpEStatsStackEntry 29 }

tcpEStatsStackAbruptTimeouts  OBJECT-TYPE
    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of timeouts that occurred without any
        immediately preceding duplicate acknowledgments or other
        indications of congestion. Abrupt Timeouts indicate that
        the path lost an entire window of data or acknowledgments.

        Timeouts that are preceded by duplicate acknowledgments or
        other congestion signals (e.g., ECN) are not counted as
        abrupt, and might have been avoided by a more sophisticated
        Fast Retransmit algorithm."
    REFERENCE
        "RFC 2581, TCP Congestion Control"
    ::= { tcpEStatsStackEntry 30 }

tcpEStatsStackSACKsRcvd  OBJECT-TYPE
    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of SACK options received."
    REFERENCE
        "RFC 2018, TCP Selective Acknowledgement Options"
    ::= { tcpEStatsStackEntry 31 }

tcpEStatsStackSACKBlocksRcvd  OBJECT-TYPE
    SYNTAX          ZeroBasedCounter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of SACK blocks received (within SACK options)."
```

## REFERENCE

"RFC 2018, TCP Selective Acknowledgement Options"  
::= { tcpEStatsStackEntry 32 }

tcpEStatsStackSendStall OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of interface stalls or other sender local  
resource limitations that are treated as congestion  
signals."

::= { tcpEStatsStackEntry 33 }

tcpEStatsStackDSACKDups OBJECT-TYPE

SYNTAX ZeroBasedCounter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of duplicate segments reported to the local host  
by D-SACK blocks."

REFERENCE

"RFC 2883, An Extension to the Selective Acknowledgement  
(SACK) Option for TCP"

::= { tcpEStatsStackEntry 34 }

--

-- The following optional objects instrument path MTU  
-- discovery.

--

tcpEStatsStackMaxMSS OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The maximum MSS, in octets."

REFERENCE

"RFC 1191, Path MTU discovery"

::= { tcpEStatsStackEntry 35 }

tcpEStatsStackMinMSS OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```

        "The minimum MSS, in octets."
REFERENCE
    "RFC 1191, Path MTU discovery"
    ::= { tcpEStatsStackEntry 36 }

--
-- The following optional initial value objects are useful for
-- conformance testing instruments on application progress and
-- consumed network resources.
--

tcpEStatsStackSndInitial  OBJECT-TYPE
    SYNTAX                Unsigned32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
        "Initial send sequence number.  Note that by definition
        tcpEStatsStackSndInitial never changes for a given
        connection."
    REFERENCE
        "RFC 793, Transmission Control Protocol"
    ::= { tcpEStatsStackEntry 37 }

tcpEStatsStackRecInitial  OBJECT-TYPE
    SYNTAX                Unsigned32
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
        "Initial receive sequence number.  Note that by definition
        tcpEStatsStackRecInitial never changes for a given
        connection."
    REFERENCE
        "RFC 793, Transmission Control Protocol"
    ::= { tcpEStatsStackEntry 38 }

--
-- The following optional objects instrument the senders
-- buffer usage, including any buffering in the application
-- interface to TCP and the retransmit queue.  All 'buffer
-- memory' instruments are assumed to include OS data
-- structure overhead.
--

tcpEStatsStackCurRetxQueue  OBJECT-TYPE
    SYNTAX                Gauge32
    UNITS                  "octets"
    MAX-ACCESS             read-only
    STATUS                 current

```

## DESCRIPTION

"The current number of octets of data occupying the retransmit queue."

::= { tcpEStatsStackEntry 39 }

tcpEStatsStackMaxRetxQueue OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The maximum number of octets of data occupying the retransmit queue."

::= { tcpEStatsStackEntry 40 }

tcpEStatsStackCurReasmQueue OBJECT-TYPE

SYNTAX Gauge32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The current number of octets of sequence space spanned by the reassembly queue. This is generally the difference between rcv.nxt and the sequence number of the right most edge of the reassembly queue."

::= { tcpEStatsStackEntry 41 }

tcpEStatsStackMaxReasmQueue OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The maximum value of tcpEStatsStackCurReasmQueue"

::= { tcpEStatsStackEntry 42 }

```
-- =====
--
-- Statistics for diagnosing interactions between
-- applications and TCP.
--
```

tcpEStatsAppTable OBJECT-TYPE

SYNTAX SEQUENCE OF TcpEStatsAppEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This table contains objects that are useful for determining if the application using TCP is

limiting TCP performance.

Entries are retained in this table for the number of seconds indicated by the tcpEStatsConnTableLatency object, after the TCP connection first enters the closed state."

```
::= { tcpEStats 6 }
```

```
tcpEStatsAppEntry  OBJECT-TYPE
    SYNTAX          TcpEStatsAppEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Each entry in this table has information about the
        characteristics of each active and recently closed TCP
        connection."
    INDEX { tcpEStatsConnectIndex }
    ::= { tcpEStatsAppTable 1 }
```

```
TcpEStatsAppEntry ::= SEQUENCE {

    tcpEStatsAppSndUna          Counter32,
    tcpEStatsAppSndNxt          Unsigned32,
    tcpEStatsAppSndMax          Counter32,
    tcpEStatsAppThruOctetsAcked ZeroBasedCounter32,
    tcpEStatsAppHCThruOctetsAcked ZeroBasedCounter64,
    tcpEStatsAppRcvNxt          Counter32,
    tcpEStatsAppThruOctetsReceived ZeroBasedCounter32,
    tcpEStatsAppHCThruOctetsReceived ZeroBasedCounter64,
    tcpEStatsAppCurAppWQueue   Gauge32,
    tcpEStatsAppMaxAppWQueue    Gauge32,
    tcpEStatsAppCurAppRQueue   Gauge32,
    tcpEStatsAppMaxAppRQueue    Gauge32
}
```

```
--
-- The following objects provide throughput statistics for the
-- connection including sequence numbers and elapsed
-- application data. These permit direct observation of the
-- applications progress, in terms of elapsed data delivery
-- and elapsed time.
--
```

```
tcpEStatsAppSndUna  OBJECT-TYPE
    SYNTAX          Counter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
```

"The value of SND.UNA, the oldest unacknowledged sequence number.

Note that SND.UNA is a TCP state variable that is congruent to Counter32 semantics."

#### REFERENCE

"RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsAppEntry 1 }

tcpEStatsAppSndNxt OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The value of SND.NXT, the next sequence number to be sent.

Note that tcpEStatsAppSndNxt is not monotonic (and thus not a counter) because TCP sometimes retransmits lost data by pulling tcpEStatsAppSndNxt back to the missing data."

#### REFERENCE

"RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsAppEntry 2 }

tcpEStatsAppSndMax OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The farthest forward (right most or largest) SND.NXT value.

Note that this will be equal to tcpEStatsAppSndNxt except when tcpEStatsAppSndNxt is pulled back during recovery."

#### REFERENCE

"RFC 793, Transmission Control Protocol"  
 ::= { tcpEStatsAppEntry 3 }

tcpEStatsAppThruOctetsAked OBJECT-TYPE

SYNTAX ZeroBasedCounter32

UNITS "octets"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of octets for which cumulative acknowledgments have been received. Note that this will be the sum of changes to tcpEStatsAppSndUna."

::= { tcpEStatsAppEntry 4 }

tcpEStatsAppHCThruOctetsAked OBJECT-TYPE

SYNTAX ZeroBasedCounter64

UNITS "octets"

MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The number of octets for which cumulative acknowledgments have been received, on systems that can receive more than 10 million bits per second. Note that this will be the sum of changes in tcpEStatsAppSndUna."

::= { tcpEStatsAppEntry 5 }

tcpEStatsAppRcvNxt OBJECT-TYPE

SYNTAX Counter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The value of RCV.NXT. The next sequence number expected on an incoming segment, and the left or lower edge of the receive window."

Note that RCV.NXT is a TCP state variable that is congruent to Counter32 semantics."

REFERENCE

"RFC 793, Transmission Control Protocol"

::= { tcpEStatsAppEntry 6 }

tcpEStatsAppThruOctetsReceived OBJECT-TYPE

SYNTAX ZeroBasedCounter32  
 UNITS "octets"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The number of octets for which cumulative acknowledgments have been sent. Note that this will be the sum of changes to tcpEStatsAppRcvNxt."

::= { tcpEStatsAppEntry 7 }

tcpEStatsAppHCThruOctetsReceived OBJECT-TYPE

SYNTAX ZeroBasedCounter64  
 UNITS "octets"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The number of octets for which cumulative acknowledgments have been sent, on systems that can transmit more than 10 million bits per second. Note that this will be the sum of changes in tcpEStatsAppRcvNxt."

::= { tcpEStatsAppEntry 8 }

tcpEStatsAppCurAppWQueue OBJECT-TYPE



```

SYNTAX          Gauge32
UNITS           "octets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "The current number of octets of application data buffered
    by TCP, pending first transmission, i.e., to the left of
    SND.NXT or SndMax. This data will generally be transmitted
    (and SND.NXT advanced to the left) as soon as there is an
    available congestion window (cwnd) or receiver window
    (rwin). This is the amount of data readily available for
    transmission, without scheduling the application. TCP
    performance may suffer if there is insufficient queued
    write data."

```

```
 ::= { tcpEStatsAppEntry 11 }
```

```
tcpEStatsAppMaxAppWQueue OBJECT-TYPE
```

```

SYNTAX          Gauge32
UNITS           "octets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "The maximum number of octets of application data buffered
    by TCP, pending first transmission. This is the maximum
    value of tcpEStatsAppCurAppWQueue. This pair of objects can
    be used to determine if insufficient queued data is steady
    state (suggesting insufficient queue space) or transient
    (suggesting insufficient application performance or
    excessive CPU load or scheduler latency)."

```

```
 ::= { tcpEStatsAppEntry 12 }
```

```
tcpEStatsAppCurAppRQueue OBJECT-TYPE
```

```

SYNTAX          Gauge32
UNITS           "octets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "The current number of octets of application data that has
    been acknowledged by TCP but not yet delivered to the
    application."

```

```
 ::= { tcpEStatsAppEntry 13 }
```

```
tcpEStatsAppMaxAppRQueue OBJECT-TYPE
```

```

SYNTAX          Gauge32
UNITS           "octets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION

```

```

        "The maximum number of octets of application data that has
        been acknowledged by TCP but not yet delivered to the
        application."
 ::= { tcpEStatsAppEntry 14 }

-- =====
--
-- Controls for Tuning TCP
--

tcpEStatsTuneTable      OBJECT-TYPE
    SYNTAX      SEQUENCE OF TcpEStatsTuneEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains per-connection controls that can
        be used to work around a number of common problems that
        plague TCP over some paths. All can be characterized as
        limiting the growth of the congestion window so as to
        prevent TCP from overwhelming some component in the
        path.

        Entries are retained in this table for the number of
        seconds indicated by the tcpEStatsConnTableLatency
        object, after the TCP connection first enters the closed
        state."
    ::= { tcpEStats 7 }

tcpEStatsTuneEntry      OBJECT-TYPE
    SYNTAX      TcpEStatsTuneEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "Each entry in this table is a control that can be used to
        place limits on each active TCP connection."
    INDEX { tcpEStatsConnectIndex }
    ::= { tcpEStatsTuneTable 1 }

TcpEStatsTuneEntry ::= SEQUENCE {

        tcpEStatsTuneLimCwnd      Unsigned32,
        tcpEStatsTuneLimSsthresh  Unsigned32,
        tcpEStatsTuneLimRwin      Unsigned32,
        tcpEStatsTuneLimMSS       Unsigned32
    }

tcpEStatsTuneLimCwnd      OBJECT-TYPE
    SYNTAX      Unsigned32

```

```

UNITS          "octets"
MAX-ACCESS     read-write
STATUS         current
DESCRIPTION    "A control to set the maximum congestion window that may be
                used, in octets."
REFERENCE      "RFC 2581, TCP Congestion Control"
::= { tcpEStatsTuneEntry 1 }

```

#### tcpEStatsTuneLimSsthresh OBJECT-TYPE

```

SYNTAX         Unsigned32
UNITS          "octets"
MAX-ACCESS     read-write
STATUS         current
DESCRIPTION    "A control to limit the maximum queue space (in octets) that
                this TCP connection is likely to occupy during slowstart.

                It can be implemented with the algorithm described in
                RFC 3742 by setting the max_ssthresh parameter to twice
                tcpEStatsTuneLimSsthresh.

                This algorithm can be used to overcome some TCP performance
                problems over network paths that do not have sufficient
                buffering to withstand the bursts normally present during
                slowstart."
REFERENCE      "RFC 3742, Limited Slow-Start for TCP with Large Congestion
                Windows"
::= { tcpEStatsTuneEntry 2 }

```

#### tcpEStatsTuneLimRwin OBJECT-TYPE

```

SYNTAX         Unsigned32
UNITS          "octets"
MAX-ACCESS     read-write
STATUS         current
DESCRIPTION    "A control to set the maximum window advertisement that may
                be sent, in octets."
REFERENCE      "RFC 793, Transmission Control Protocol"
::= { tcpEStatsTuneEntry 3 }

```

#### tcpEStatsTuneLimMSS OBJECT-TYPE

```

SYNTAX         Unsigned32
UNITS          "octets"
MAX-ACCESS     read-write

```

```

        STATUS          current
        DESCRIPTION
            "A control to limit the maximum segment size in octets, that
             this TCP connection can use."
        REFERENCE
            "RFC 1191, Path MTU discovery"
        ::= { tcpEStatsTuneEntry 4 }

-- =====
--
-- TCP Extended Statistics Notifications Group
--

tcpEStatsEstablishNotification NOTIFICATION-TYPE
    OBJECTS      {
                    tcpEStatsConnectIndex
                }
    STATUS       current
    DESCRIPTION
        "The indicated connection has been accepted
         (or alternatively entered the established state).".
    ::= { tcpEStatsNotifications 1 }

tcpEStatsCloseNotification NOTIFICATION-TYPE
    OBJECTS      {
                    tcpEStatsConnectIndex
                }
    STATUS       current
    DESCRIPTION
        "The indicated connection has left the
         established state"
    ::= { tcpEStatsNotifications 2 }

-- =====
--
-- Conformance Definitions
--

tcpEStatsCompliances    OBJECT IDENTIFIER
    ::= { tcpEStatsConformance 1 }
tcpEStatsGroups         OBJECT IDENTIFIER
    ::= { tcpEStatsConformance 2 }

--
-- Compliance Statements
--

tcpEStatsCompliance MODULE-COMPLIANCE

```

```
STATUS current
DESCRIPTION
    "Compliance statement for all systems that implement TCP
    extended statistics."
MODULE -- this module
    MANDATORY-GROUPS {
        tcpEStatsListenerGroup,
        tcpEStatsConnectIdGroup,
        tcpEStatsPerfGroup,
        tcpEStatsPathGroup,
        tcpEStatsStackGroup,
        tcpEStatsAppGroup
    }
GROUP tcpEStatsListenerHCGroup
DESCRIPTION
    "This group is mandatory for all systems that can
    wrap the values of the 32-bit counters in
    tcpEStatsListenerGroup in less than one hour."

GROUP tcpEStatsPerfOptionalGroup
DESCRIPTION
    "This group is optional for all systems."

GROUP tcpEStatsPerfHCGroup
DESCRIPTION
    "This group is mandatory for systems that can
    wrap the values of the 32-bit counters in
    tcpEStatsPerfGroup in less than one hour.

    Note that any system that can attain 10 Mb/s
    can potentially wrap 32-Bit Octet counters in
    under one hour."

GROUP tcpEStatsPathOptionalGroup
DESCRIPTION
    "This group is optional for all systems."

GROUP tcpEStatsPathHCGroup
DESCRIPTION
    "This group is mandatory for systems that can
    wrap the values of the 32-bit counters in
    tcpEStatsPathGroup in less than one hour.

    Note that any system that can attain 10 Mb/s
    can potentially wrap 32-Bit Octet counters in
    under one hour."

GROUP tcpEStatsStackOptionalGroup
```

## DESCRIPTION

"This group is optional for all systems."

## GROUP tcpEStatsAppHCGroup

## DESCRIPTION

"This group is mandatory for systems that can wrap the values of the 32-bit counters in tcpEStatsStackGroup in less than one hour."

Note that any system that can attain 10 Mb/s can potentially wrap 32-Bit Octet counters in under one hour."

## GROUP tcpEStatsAppOptionalGroup

## DESCRIPTION

"This group is optional for all systems."

## GROUP tcpEStatsTuneOptionalGroup

## DESCRIPTION

"This group is optional for all systems."

## GROUP tcpEStatsNotificationsGroup

## DESCRIPTION

"This group is optional for all systems."

## GROUP tcpEStatsNotificationsCtlGroup

## DESCRIPTION

"This group is mandatory for systems that include the tcpEStatsNotificationGroup."

::= { tcpEStatsCompliances 1 }

-- =====

--

-- Units of Conformance

--

tcpEStatsListenerGroup OBJECT-GROUP

OBJECTS {

tcpEStatsListenerTableLastChange,  
tcpEStatsListenerStartTime,  
tcpEStatsListenerSynRcvd,  
tcpEStatsListenerInitial,  
tcpEStatsListenerEstablished,  
tcpEStatsListenerAccepted,  
tcpEStatsListenerExceedBacklog,  
tcpEStatsListenerCurConns,  
tcpEStatsListenerMaxBacklog,  
tcpEStatsListenerCurBacklog,

```

        tcpEStatsListenerCurEstabBacklog
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsListener group includes objects that
        provide valuable statistics and debugging
        information for TCP Listeners."
    ::= { tcpEStatsGroups 1 }

tcpEStatsListenerHCGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsListenerHCSynRcvd,
        tcpEStatsListenerHCInitial,
        tcpEStatsListenerHCEstablished,
        tcpEStatsListenerHCAccepted,
        tcpEStatsListenerHCExceedBacklog
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsListenerHC group includes 64-bit
        counters in tcpEStatsListenerTable."
    ::= { tcpEStatsGroups 2 }

tcpEStatsConnectIdGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsConnTableLatency,
        tcpEStatsConnectIndex
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsConnectId group includes objects that
        identify TCP connections and control how long TCP
        connection entries are retained in the tables."
    ::= { tcpEStatsGroups 3 }

tcpEStatsPerfGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsPerfSegsOut, tcpEStatsPerfDataSegsOut,
        tcpEStatsPerfDataOctetsOut,
        tcpEStatsPerfSegsRetrans,
        tcpEStatsPerfOctetsRetrans, tcpEStatsPerfSegsIn,
        tcpEStatsPerfDataSegsIn,
        tcpEStatsPerfDataOctetsIn,
        tcpEStatsPerfElapsedSecs,
        tcpEStatsPerfElapsedMicroSecs,
        tcpEStatsPerfStartTimeStamp, tcpEStatsPerfCurMSS,
        tcpEStatsPerfPipeSize, tcpEStatsPerfMaxPipeSize,
        tcpEStatsPerfSmoothedRTT, tcpEStatsPerfCurRTO,

```

```

        tcpEStatsPerfCongSignals, tcpEStatsPerfCurCwnd,
        tcpEStatsPerfCurSsthresh, tcpEStatsPerfTimeouts,
        tcpEStatsPerfCurRwinSent,
        tcpEStatsPerfMaxRwinSent,
        tcpEStatsPerfZeroRwinSent,
        tcpEStatsPerfCurRwinRcvd,
        tcpEStatsPerfMaxRwinRcvd,
        tcpEStatsPerfZeroRwinRcvd
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPerf group includes those objects that
        provide basic performance data for a TCP connection."
    ::= { tcpEStatsGroups 4 }

tcpEStatsPerfOptionalGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsPerfSndLimTransRwin,
        tcpEStatsPerfSndLimTransCwnd,
        tcpEStatsPerfSndLimTransSnd,
        tcpEStatsPerfSndLimTimeRwin,
        tcpEStatsPerfSndLimTimeCwnd,
        tcpEStatsPerfSndLimTimeSnd
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPerf group includes those objects that
        provide basic performance data for a TCP connection."
    ::= { tcpEStatsGroups 5 }

tcpEStatsPerfHCGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsPerfHCDataOctetsOut,
        tcpEStatsPerfHCDataOctetsIn
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPerfHC group includes 64-bit
        counters in the tcpEStatsPerfTable."
    ::= { tcpEStatsGroups 6 }

tcpEStatsPathGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsControlPath,
        tcpEStatsPathRetranThresh,
        tcpEStatsPathNonRecovDAEpisodes,
        tcpEStatsPathSumOctetsReordered,

```



```

        tcpEStatsPathNonRecovDA
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPath group includes objects that
        control the creation of the tcpEStatsPathTable,
        and provide information about the path
        for each TCP connection."
    ::= { tcpEStatsGroups 7 }

tcpEStatsPathOptionalGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsPathSamplerRTT, tcpEStatsPathRTTVar,
        tcpEStatsPathMaxRTT, tcpEStatsPathMinRTT,
        tcpEStatsPathSumRTT, tcpEStatsPathCountRTT,
        tcpEStatsPathMaxRTO, tcpEStatsPathMinRTO,
        tcpEStatsPathIpTtl, tcpEStatsPathIpTosIn,
        tcpEStatsPathIpTosOut,
        tcpEStatsPathPreCongSumCwnd,
        tcpEStatsPathPreCongSumRTT,
        tcpEStatsPathPostCongSumRTT,
        tcpEStatsPathPostCongCountRTT,
        tcpEStatsPathECNsignals,
        tcpEStatsPathDupAckEpisodes, tcpEStatsPathRcvRTT,
        tcpEStatsPathDupAcksOut, tcpEStatsPathCERcvd,
        tcpEStatsPathECESent
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPath group includes objects that
        provide additional information about the path
        for each TCP connection."
    ::= { tcpEStatsGroups 8 }

tcpEStatsPathHCGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsPathHCSumRTT
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsPathHC group includes 64-bit
        counters in the tcpEStatsPathTable."
    ::= { tcpEStatsGroups 9 }

tcpEStatsStackGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsControlStack,
        tcpEStatsStackActiveOpen, tcpEStatsStackMSSSent,

```

```

    tcpEStatsStackMSSRcvd, tcpEStatsStackWinScaleSent,
    tcpEStatsStackWinScaleRcvd,
    tcpEStatsStackTimeStamps, tcpEStatsStackECN,
    tcpEStatsStackWillSendSACK,
    tcpEStatsStackWillUseSACK, tcpEStatsStackState,
    tcpEStatsStackNagle, tcpEStatsStackMaxSsCwnd,
    tcpEStatsStackMaxCaCwnd,
    tcpEStatsStackMaxSsthresh,
    tcpEStatsStackMinSsthresh,
    tcpEStatsStackInRecovery, tcpEStatsStackDupAcksIn,
    tcpEStatsStackSpuriousFrDetected,
    tcpEStatsStackSpuriousRtoDetected
}
STATUS current
DESCRIPTION
    "The tcpEStatsConnState group includes objects that
    control the creation of the tcpEStatsStackTable,
    and provide information about the operation of
    algorithms used within TCP."
 ::= { tcpEStatsGroups 10 }

tcpEStatsStackOptionalGroup OBJECT-GROUP
    OBJECTS {
        tcpEStatsStackSoftErrors,
        tcpEStatsStackSoftErrorReason,
        tcpEStatsStackSlowStart, tcpEStatsStackCongAvoid,
        tcpEStatsStackOtherReductions,
        tcpEStatsStackCongOverCount,
        tcpEStatsStackFastRetran,
        tcpEStatsStackSubsequentTimeouts,
        tcpEStatsStackCurTimeoutCount,
        tcpEStatsStackAbruptTimeouts,
        tcpEStatsStackSACKsRcvd,
        tcpEStatsStackSACKBlocksRcvd,
        tcpEStatsStackSendStall, tcpEStatsStackDSACKDups,
        tcpEStatsStackMaxMSS, tcpEStatsStackMinMSS,
        tcpEStatsStackSndInitial,
        tcpEStatsStackRecInitial,
        tcpEStatsStackCurRetxQueue,
        tcpEStatsStackMaxRetxQueue,
        tcpEStatsStackCurReasmQueue,
        tcpEStatsStackMaxReasmQueue
    }
STATUS current
DESCRIPTION
    "The tcpEStatsConnState group includes objects that
    provide additional information about the operation of
    algorithms used within TCP."

```

```
::= { tcpEStatsGroups 11 }

tcpEStatsAppGroup OBJECT-GROUP
  OBJECTS {
    tcpEStatsControlApp,
    tcpEStatsAppSndUna, tcpEStatsAppSndNxt,
    tcpEStatsAppSndMax, tcpEStatsAppThruOctetsAcked,
    tcpEStatsAppRcvNxt,
    tcpEStatsAppThruOctetsReceived
  }
  STATUS current
  DESCRIPTION
    "The tcpEStatsConnState group includes objects that
    control the creation of the tcpEStatsAppTable,
    and provide information about the operation of
    algorithms used within TCP."
  ::= { tcpEStatsGroups 12 }

tcpEStatsAppHCGroup OBJECT-GROUP
  OBJECTS {
    tcpEStatsAppHCThruOctetsAcked,
    tcpEStatsAppHCThruOctetsReceived
  }
  STATUS current
  DESCRIPTION
    "The tcpEStatsStackHC group includes 64-bit
    counters in the tcpEStatsStackTable."
  ::= { tcpEStatsGroups 13 }

tcpEStatsAppOptionalGroup OBJECT-GROUP
  OBJECTS {
    tcpEStatsAppCurAppWQueue,
    tcpEStatsAppMaxAppWQueue,
    tcpEStatsAppCurAppRQueue,
    tcpEStatsAppMaxAppRQueue
  }
  STATUS current
  DESCRIPTION
    "The tcpEStatsConnState group includes objects that
    provide additional information about how applications
    are interacting with each TCP connection."
  ::= { tcpEStatsGroups 14 }

tcpEStatsTuneOptionalGroup OBJECT-GROUP
  OBJECTS {
    tcpEStatsControlTune,
    tcpEStatsTuneLimCwnd, tcpEStatsTuneLimSsthresh,
    tcpEStatsTuneLimRwin, tcpEStatsTuneLimMSS
```

```
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsConnState group includes objects that
        control the creation of the tcpEStatsConnectionTable,
        which can be used to set tuning parameters
        for each TCP connection."
    ::= { tcpEStatsGroups 15 }

tcpEStatsNotificationsGroup      NOTIFICATION-GROUP
    NOTIFICATIONS {
        tcpEStatsEstablishNotification,
        tcpEStatsCloseNotification
    }
    STATUS current
    DESCRIPTION
        "Notifications sent by a TCP extended statistics agent."
    ::= { tcpEStatsGroups 16 }

tcpEStatsNotificationsCtlGroup  OBJECT-GROUP
    OBJECTS {
        tcpEStatsControlNotify
    }
    STATUS current
    DESCRIPTION
        "The tcpEStatsNotificationsCtl group includes the
        object that controls the creation of the events
        in the tcpEStatsNotificationsGroup."
    ::= { tcpEStatsGroups 17 }

END
```

## 5. Security Considerations

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- \* Changing tcpEStatsConnTableLatency or any of the control objects in the tcpEStatsControl group (tcpEStatsControlPath, tcpEStatsControlStack, tcpEStatsControlApp, tcpEStatsControlTune) may affect the correctness of other management applications accessing this MIB. Generally, local policy should only permit limited write access to these controls (e.g., only by one management station or only during system configuration).
- \* The objects in the tcpEStatsControlTune group (tcpEStatsTuneLimCwnd, tcpEStatsTuneLimSsthresh, tcpEStatsTuneLimRwin) can be used to limit resources consumed by TCP connections or to limit TCP throughput. An attacker might manipulate these objects to reduce performance to levels below the minimum acceptable for a particular application.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- \* All objects which expose TCP sequence numbers (tcpEStatsAppSndUna, tcpEStatsAppSndNxt, tcpEStatsAppSndMax, tcpEStatsStackSndInitial, tcpEStatsAppRcvNxt, and tcpEStatsStackRecInitial) might make it easier for an attacker to forge in sequence TCP segments to disrupt TCP connections.
- \* Nearly all objects in this (or any other) MIB may be used to estimate traffic volumes, which may reveal unanticipated information about an organization to the outside world.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPsec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

## 6. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
-----	-----
tcpEStatsMIB	{ mib-2 156 }

## 7. Normative References

- [RFC791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.
- [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.
- [RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, May 1992.
- [RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", RFC 2579, STD 58, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", RFC 2580, STD 58, April 1999.
- [RFC2581] Allman, M., Paxson, V., and W. Stevens, "TCP Congestion Control", RFC 2581, April 1999.
- [RFC2856] Bierman, A., McCloghrie, K., and R. Presuhn, "Textual Conventions for Additional High Capacity Data Types", RFC 2856, June 2000.
- [RFC2883] Floyd, S., Mahdavi, J., Mathis, M., and M. Podolsky, "An Extension to the Selective Acknowledgement (SACK) Option for TCP", RFC 2883, July 2000.
- [RFC2988] Paxson, V. and M. Allman, "Computing TCP's Retransmission Timer", RFC 2988, November 2000.
- [RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.
- [RFC3517] Blanton, E., Allman, M., Fall, K., and L. Wang, "A Conservative Selective Acknowledgment (SACK)-based Loss Recovery Algorithm for TCP", RFC 3517, April 2003.
- [RFC4022] Raghunarayan, R., Ed., "Management Information Base for the Transmission Control Protocol (TCP)", RFC 4022, March 2005.
- [RFC4502] Waldbusser, S., "Remote Network Monitoring Management Information Base Version 2", RFC 4502, May 2006.

## 8. Informative References

- [Mat97] M. Mathis, J. Semke, J. Mahdavi, T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communication Review, volume 27, number 3, July 1997.
- [Bra94] Brakmo, L., O'Malley, S., "TCP Vegas, New Techniques for Congestion Detection and Avoidance", SIGCOMM'94, London, pp 24-35, October 1994.
- [Edd06] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", Work in Progress, May 2007.
- [POSIX] Portable Operating System Interface, IEEE Std 1003.1
- [Pad98] Padhye, J., Firoiu, V., Towsley, D., Kurose, J., "Modeling TCP Throughput: A Simple Model and its Empirical Validation", SIGCOMM'98.
- [Web100] Mathis, M., J. Heffner, R. Reddy, "Web100: Extended TCP Instrumentation for Research, Education and Diagnosis", ACM Computer Communications Review, Vol 33, Num 3, July 2003.
- [RFC2861] Handley, M., Padhye, J., and S. Floyd, "TCP Congestion Window Validation", RFC 2861, June 2000.
- [RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC 3260, April 2002.
- [RFC3410] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3522] Ludwig, R. and M. Meyer, "The Eifel Detection Algorithm for TCP", RFC 3522, April 2003.
- [RFC3742] Floyd, S., "Limited Slow-Start for TCP with Large Congestion Windows", RFC 3742, March 2004.
- [RFC4614] Duke M., Braden, R., Eddy, W., Blanton, E. "A Roadmap for Transmission Control Protocol (TCP) Specification Documents", RFC 4614, September 2006.



## 9. Contributors

The following people contributed text that was incorporated into this document:

Jon Saperia <saperia@jdscons.com> converted Web100 internal documentation into a true MIB.

Some of the objects in this document were moved from an early version of the TCP-MIB by Bill Fenner, et al.

Some of the object descriptions are based on an earlier unpublished document by Jeff Semke.

## 10. Acknowledgments

This document is a product of the Web100 project ([www.web100.org](http://www.web100.org)), a joint effort of Pittsburgh Supercomputing Center ([www.psc.edu](http://www.psc.edu)), National Center for Atmospheric Research ([www.ncar.ucar.edu](http://www.ncar.ucar.edu)), and National Center for Supercomputer Applications ([www.ncsa.edu](http://www.ncsa.edu)).

It would not have been possible without all of the hard work by the entire Web100 team, especially Peter O'Neal, who read and reread the entire document several times; Janet Brown and Marla Meehl, who patiently managed the unmanageable. The Web100 project would not have been successful without all of the early adopters who suffered our bugs to provide many good suggestions and insights into their needs for TCP instrumentation.

Web100 was supported by the National Science Foundation under Grant No. 0083285 and a research grant from Cisco Systems.

We would also like to thank all of the people who built experimental implementations of this MIB from early versions and provided us with constructive feedback: Glenn Turner at AARnet, Kristine Adamson at IBM, and Xinyan Zan at Microsoft.

And last, but not least, we would like to thank Dan Romascanu, our "MIB Doctor" and Bert Wijnen, the Operations Area Director, for patiently steering us through the MIB review process.

## Authors' Addresses

Matt Mathis  
Pittsburgh Supercomputing Center  
300 S. Craig St.  
Pittsburgh, PA 15213  
Phone: 412-268-4960  
EMail: mathis@psc.edu

John Heffner  
Pittsburgh Supercomputing Center  
300 S. Craig St.  
Pittsburgh, PA 15213  
Phone: 412-268-4960  
EMail: jheffner@psc.edu

Rajiv Raghunarayan  
Cisco Systems Inc.  
San Jose, CA 95134  
Phone: 408 853 9612  
EMail: raraghun@cisco.com

## Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

## Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

