

Network Working Group
Request for Comments: 2940
Category: Standards Track

A. Smith
Consultant
D. Partain
Ericsson
J. Seligson
Nortel Networks
October 2000

Definitions of Managed Objects for Common Open Policy Service (COPS)
Protocol Clients

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 Internet Society (2000). All Rights Reserved.

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP based internets. In particular it defines objects for managing a client of the Common Open Policy Service (COPS) protocol.

This memo includes a MIB module in a manner that is compliant to the SMIV2 [V2SMI].

1. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- o An overall architecture, described in an Architecture for Describing SNMP Management Frameworks [ARCH].
- o Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIV1 and described in STD 16, RFC 1155 [V1SMI], STD 16, RFC 1212 [V1CONCISE] and RFC 1215 [V1TRAPS]. The second version, called SMIV2, is described in STD 58, RFC 2578 [V2SMI], STD 58, RFC 2579 [V2TC] and STD 58, RFC 2580 [V2CONFORM].
- o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [V1PROTO]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [V2COMMUNITY] and RFC 1906 [V2TRANS]. The third version of the message protocol is called SNMPv3 and described in RFC1906 [V2TRANS], Message Processing and Dispatching [V3MPC] and User-based Security Model [V3USM].
- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [V1PROTO]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [V2PROTO].
- o A set of fundamental applications described in SNMPv3 Applications [V3APPS] and the view-based access control mechanism described in View-based Access Control Model [V3VACM].

A more detailed introduction to the current SNMP Management Framework can be found in RFC 2570 [V3INTRO].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIV2. A MIB conforming to the SMIV1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no

translation is possible (use of Counter64). Some machine readable information in SMIV2 will be converted into textual descriptions in SMIV1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB.

2. Overview

The COPS protocol [COPS] is a client-server protocol intended for the communication of policy requests and decisions between a Policy Enforcement Point (PEP) and a Policy Decision Point (PDP). The PEP acts as a COPS client in this scenario. The model for policy outsourcing, of which the COPS protocol provides one part, is described in [FRAMEWORK].

2.1. Scope

This MIB is intended to provide management of the important features of a COPS protocol client module. It does not provide management for a COPS server - this is outside the scope of the current memo. It provides for monitoring of status and protocol statistics, as well as for configuration of the client, in particular for telling it where to locate its servers. Other mechanisms for achieving this function without SNMP configuration might include use of the Service Location Protocol [SRVLOC] although this is outside the scope of this memo and are not specified by the COPS protocol itself.

This MIB also does not provide management of specific COPS client-types e.g., for use with the RSVP protocol [RSVP][COPSRSPV].

3. Structure of COPS Client MIB

Objects in this MIB are arranged into groups. Each group is organized as a set of related objects. The overall structure is described below.

3.1. copsClientCapabilitiesGroup

This group contains objects that represent COPS protocol capabilities implemented by this COPS client.

3.2. copsClientStatusGroup

This group contains objects that indicate the current status of connection(s) to COPS servers, including per-server protocol statistics. It maintains last-known statistics for all of the servers with which the client has ever been connected since agent restart.

3.3. copsConfigGroup

This group contains objects that allow for configuration of COPS server addresses and the order to which connections should be attempted. It contains a table of per-server objects as well as scalars for configuration of the retry algorithm to be used by a client to obtain a connection to an appropriate server.

3.4. Textual Conventions

The datatypes CopsClientState, CopsServerEntryType, CopsErrorCode, CopsTcpPort and CopsAuthType are used as textual conventions in this document. These textual conventions have NO effect on either the syntax nor the semantics of any managed object. Objects defined using these conventions are always encoded by means of the rules that define their primitive type. Hence, no changes to the SMI or the SNMP are necessary to accommodate these textual conventions which are adopted merely for the convenience of readers.

3.5. Relationship to Other MIBs

3.5.1. Relationship to the 'system' group

This MIB contains definitions for a single COPS protocol client represented by a single SNMP agent and instance of the MIB-2 system group [MIB2]. It does not address the case of multiple co-located COPS protocol clients.

4. Definitions for COPS Client MIB

COPS-CLIENT-MIB DEFINITIONS ::= BEGIN

-- -----
 -- -----

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, Counter32, Integer32,
 Unsigned32, mib-2
 FROM SNMPv2-SMI
 TimeStamp, TimeInterval, RowStatus, TEXTUAL-CONVENTION
 FROM SNMPv2-TC
 MODULE-COMPLIANCE, OBJECT-GROUP
 FROM SNMPv2-CONF
 InetAddressType, InetAddress
 FROM INET-ADDRESS-MIB;

-- REFERENCE

-- "The COPS (Common Open Policy Service) Protocol RFC 2748

copsClientMIB MODULE-IDENTITY

LAST-UPDATED "200009280000Z"

ORGANIZATION "IETF RSVP Admission Policy Working Group"

CONTACT-INFO

" Andrew Smith (WG co-chair)

Phone: +1 408 579 2821

Email: ah_smith@pacbell.net

Mark Stevens (WG co-chair)

Phone: +1 978 287 9102

Email: markstevens@lucent.com

Editor: Andrew Smith

Phone: +1 408 579 2821

Email: ah_smith@pacbell.net

Editor: David Partain

Phone: +46 13 28 41 44

Email: David.Partain@ericsson.com

Editor: John Seligson

Phone: +1 408 495 2992

Email: jseligso@nortelnetworks.com"

DESCRIPTION

"The COPS Client MIB module"

REVISION "200009280000Z"

DESCRIPTION "This version published as RFC 2940"

::= { mib-2 89 }

copsClientMIBObjects OBJECT IDENTIFIER ::= { copsClientMIB 1 }

-- -----
 -- Textual Conventions
 -- -----

CopsClientState ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A value indicating the state of a COPS client."

SYNTAX INTEGER {

copsClientInvalid(1), -- default state.

copsClientTcpconnected(2), -- TCP connection up but COPS

-- not yet open.

```

    copsClientAuthenticating(3), -- TCP connection up but still
                                -- authenticating.
    copsClientSecAccepted(4),   -- connection authenticated.
    copsClientAccepted(5),      -- COPS server accepted client.
    copsClientTimeout(6)        -- Keepalive timer has expired,
                                -- client is in process of
                                -- tearing down connection.
}

```

```
CopsServerEntryType ::= TEXTUAL-CONVENTION
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A value indicating how a COPS server entry came into existence."
```

```
SYNTAX INTEGER {
```

```
    copsServerStatic(1),          -- configured by manager
    copsServerRedirect(2)         -- notified by COPS server
}
```

```
CopsErrorCode ::= TEXTUAL-CONVENTION
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A value describing a COPS protocol error. Codes are identical
to those used by the COPS protocol itself."
```

```
SYNTAX INTEGER {
```

```
    errorOther(0),                -- none of the below
    errorBadHandle(1),
    errorInvalidHandleReference(2),
    errorBadMessageFormat(3),
    errorUnableToProcess(4),
    errorMandatoryClientSiMissing(5),
    errorUnsupportedClientType(6),
    errorMandatoryCopsObjectMissing(7),
    errorClientFailure(8),
    errorCommunicationFailure(9),
    errorUnspecified(10),         -- client-type specific subcode
    errorShuttingDown(11),
    errorRedirectToPreferredServer(12),
    errorUnknownCopsObject(13),
    errorAuthenticationFailure(14),
    errorAuthenticationMissing(15)
}
```

```
-- REFERENCE
```

```
-- "RFC 2748 section 2.2.8"
```

```
CopsTcpPort ::= TEXTUAL-CONVENTION
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A value indicating a TCP protocol port number."
```

SYNTAX INTEGER (0..65535)

CopsAuthType ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A value indicating a type of security authentication mechanism."

SYNTAX INTEGER {
 authNone(0),
 authOther(1),
 authIpSecAh(2),
 authIpSecEsp(3),
 authTls(4),
 authCopsIntegrity(5)
 }

 copsClientCapabilitiesGroup OBJECT IDENTIFIER
 ::= { copsClientMIBObjects 1 }

 --
 -- Capabilities of the COPS client to connect to a COPS server:
 --

copsClientCapabilities OBJECT-TYPE

SYNTAX BITS {
 copsClientVersion1(0), -- supports version1 of COPS protocol
 copsClientAuthIpSecAh(1) , -- supports IP-SEC Authentication
 copsClientAuthIpSecEsp(2), -- supports IP-SEC Encryption
 copsClientAuthTls(3), -- supports Transport-Layer Security
 copsClientAuthInteg(4) -- supports COPS Integrity
 }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A list of the optional capabilities that this COPS client supports."

::= { copsClientCapabilitiesGroup 1 }

 copsClientStatusGroup OBJECT IDENTIFIER ::= { copsClientMIBObjects 2 }

 --
 -- Current status of COPS server connections, all read-only.
 --

copsClientServerCurrentTable OBJECT-TYPE

SYNTAX SEQUENCE OF CopsClientServerCurrentEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION

"A table of information regarding COPS servers as seen from the point of view of a COPS client. This table contains entries for both statically-configured and dynamically-learned servers (from a PDP Redirect operation). One entry exists in this table for each COPS Client-Type served by the COPS server. In addition, an entry will exist with copsClientServerClientType 0 (zero) representing information about the underlying connection itself: this is consistent with the COPS specification which reserves this value for this purpose."

::= { copsClientStatusGroup 1 }

copsClientServerCurrentEntry OBJECT-TYPE

SYNTAX CopsClientServerCurrentEntry
 MAX-ACCESS not-accessible
 STATUS current
 DESCRIPTION

"A set of information regarding a single COPS server serving a single COPS Client-Type from the point of view of a COPS client."

INDEX { copsClientServerAddressType, copsClientServerAddress,
 copsClientServerClientType }

::= { copsClientServerCurrentTable 1 }

CopsClientServerCurrentEntry ::=

SEQUENCE {
 copsClientServerAddressType InetAddressType,
 copsClientServerAddress InetAddress,
 copsClientServerClientType INTEGER,
 copsClientServerTcpPort CopsTcpPort,
 copsClientServerType CopsServerEntryType,
 copsClientServerAuthType CopsAuthType,
 copsClientServerLastConnAttempt TimeStamp,
 copsClientState CopsClientState,
 copsClientServerKeepaliveTime TimeInterval,
 copsClientServerAccountingTime TimeInterval,
 copsClientInPkts Counter32,
 copsClientOutPkts Counter32,
 copsClientInErrs Counter32,
 copsClientLastError CopsErrorCode,
 copsClientTcpConnectAttempts Counter32,
 copsClientTcpConnectFailures Counter32,
 copsClientOpenAttempts Counter32,

```

copsClientOpenFailures           Counter32,
copsClientErrUnsupportClienttype Counter32,
copsClientErrUnsupportedVersion  Counter32,
copsClientErrLengthMismatch     Counter32,
copsClientErrUnknownOpcode      Counter32,
copsClientErrUnknownCnum        Counter32,
copsClientErrBadCtype           Counter32,
copsClientErrBadSends           Counter32,
copsClientErrWrongObjects       Counter32,
copsClientErrWrongOpcode        Counter32,
copsClientKaTimeoutClients      Counter32,
copsClientErrAuthFailures       Counter32,
copsClientErrAuthMissing        Counter32
}

```

copsClientServerAddressType OBJECT-TYPE

```

SYNTAX      InetAddressType
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The type of address in copsClientServerAddress."
 ::= { copsClientServerCurrentEntry 1 }

```

copsClientServerAddress OBJECT-TYPE

```

SYNTAX      InetAddress
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The IPv4, IPv6 or DNS address of a COPS Server. Note that,
     since this is an index to the table, the DNS name must be
     short enough to fit into the maximum length of indices allowed
     by the management protocol in use."
REFERENCE
    "RFC 2748 section 2.3"
 ::= { copsClientServerCurrentEntry 2 }

```

copsClientServerClientType OBJECT-TYPE

```

SYNTAX      INTEGER (0..65535)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The COPS protocol Client-Type for which this entry
     applies. Multiple Client-Types can be served by a single
     COPS server. The value 0 (zero) indicates that this
     entry contains information about the underlying connection
     itself."
REFERENCE
    "RFC 2748 section 6, IANA"

```

```
::= { copsClientServerCurrentEntry 3 }
```

```
copsClientServerTcpPort OBJECT-TYPE
```

```
SYNTAX      CopsTcpPort
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The TCP port number on the COPS server to which the
client should connect/is connected."
```

```
::= { copsClientServerCurrentEntry 4 }
```

```
copsClientServerType OBJECT-TYPE
```

```
SYNTAX      CopsServerEntryType
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Indicator of the source of this COPS server information.
COPS servers may be configured by network management
into copsClientServerConfigTable and appear in this entry
with type copsServerStatic(1). Alternatively, the may be
notified from another COPS server by means of the COPS
PDP-Redirect mechanism and appear as copsServerRedirect(2)."
```

```
::= { copsClientServerCurrentEntry 5 }
```

```
copsClientServerAuthType OBJECT-TYPE
```

```
SYNTAX      CopsAuthType
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Indicator of the current security mode in use between
client and this COPS server."
```

```
::= { copsClientServerCurrentEntry 6 }
```

```
copsClientServerLastConnAttempt OBJECT-TYPE
```

```
SYNTAX      TimeStamp
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Timestamp of the last time that this client attempted to
connect to this COPS server."
```

```
::= { copsClientServerCurrentEntry 7 }
```

```
copsClientState OBJECT-TYPE
```

```
SYNTAX      CopsClientState
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The state of the connection and COPS protocol with respect
```

to this COPS server."
 ::= { copsClientServerCurrentEntry 8 }

copsClientServerKeepaliveTime OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of the COPS protocol Keepalive timeout, in centiseconds, currently in use by this client, as specified by this COPS server in the Client-Accept operation. A value of zero indicates no keepalive activity is expected."

REFERENCE

"RFC 2748 section 3.7, 4.4"

::= { copsClientServerCurrentEntry 9 }

copsClientServerAccountingTime OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of the COPS protocol Accounting timeout, in centiseconds, currently in use by this client, as specified by the COPS server in the Client-Accept operation. A value of zero indicates no accounting activity is to be performed."

REFERENCE

"RFC 2748 section 3.7"

::= { copsClientServerCurrentEntry 10 }

copsClientInPkts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from this COPS server marked for this Client-Type. This value is cumulative since agent restart and is not zeroed on new connections."

::= { copsClientServerCurrentEntry 11 }

copsClientOutPkts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has sent to this COPS server marked for this Client-Type. This value is cumulative since agent restart and is not zeroed on new

```
connections."  
 ::= { copsClientServerCurrentEntry 12 }
```

copsClientInErrs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from this COPS server marked for this Client-Type that contained errors in syntax. This value is cumulative since agent restart and is not zeroed on new connections."

```
 ::= { copsClientServerCurrentEntry 13 }
```

copsClientLastError OBJECT-TYPE

SYNTAX CopsErrorCode

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The code contained in the last COPS protocol Error Object received by this client from this COPS server marked for this Client-Type. This value is not zeroed on COPS Client-Open operations."

REFERENCE

"RFC 2748 section 2.2.8"

```
 ::= { copsClientServerCurrentEntry 14 }
```

copsClientTcpConnectAttempts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the number of times that this COPS client has tried (successfully or otherwise) to open an TCP connection to a COPS server. This value is cumulative since agent restart and is not zeroed on new connections. This value is not incremented for entries representing a non-zero Client-Type."

```
 ::= { copsClientServerCurrentEntry 15 }
```

copsClientTcpConnectFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the number of times that this COPS client has failed to open an TCP connection to a COPS server. This value is cumulative since agent restart and is not zeroed on new connections. This value is not incremented for

entries representing a non-zero Client-Type."
 ::= { copsClientServerCurrentEntry 16 }

copsClientOpenAttempts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the number of times that this COPS client has tried to perform a COPS Client-Open to a COPS server for this Client-Type. This value is cumulative since agent restart and is not zeroed on new connections."

::= { copsClientServerCurrentEntry 17 }

copsClientOpenFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the number of times that this COPS client has failed to perform a COPS Client-Open to a COPS server for this Client-Type. This value is cumulative since agent restart and is not zeroed on new connections."

::= { copsClientServerCurrentEntry 18 }

copsClientErrUnsupportClienttype OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers that referred to Client-Types that are unsupported by this client. This value is cumulative since agent restart and is not zeroed on new connections. This value is not incremented for entries representing a non-zero Client-Type."

::= { copsClientServerCurrentEntry 19 }

copsClientErrUnsupportedVersion OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that had a COPS protocol Version number that is unsupported by this client. This value is cumulative since agent restart and is not zeroed on new connections."

```
::= { copsClientServerCurrentEntry 20 }
```

copsClientErrLengthMismatch OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that had a COPS protocol Message Length that did not match the actual received message. This value is cumulative since agent restart and is not zeroed on new connections."

```
::= { copsClientServerCurrentEntry 21 }
```

copsClientErrUnknownOpcode OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that had a COPS protocol Op Code that was unrecognised by this client. This value is cumulative since agent restart and is not zeroed on new connections."

```
::= { copsClientServerCurrentEntry 22 }
```

copsClientErrUnknownCnum OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that contained a COPS protocol object C-Num that was unrecognised by this client. This value is cumulative since agent restart and is not zeroed on new connections."

```
::= { copsClientServerCurrentEntry 23 }
```

copsClientErrBadCtype OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that contained a COPS protocol object C-Type that was not defined for the C-Nums known by this client. This value is cumulative since agent restart and is not zeroed on new connections."

```
::= { copsClientServerCurrentEntry 24 }
```

```
copsClientErrBadSends OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A count of the total number of COPS messages that this client attempted to send to COPS servers marked for this Client-Type that resulted in a transmit error. This value is cumulative since agent restart and is not zeroed on new connections."
```

```
::= { copsClientServerCurrentEntry 25 }
```

```
copsClientErrWrongObjects OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that did not contain a permitted set of COPS protocol objects. This value is cumulative since agent restart and is not zeroed on new connections."
```

```
::= { copsClientServerCurrentEntry 26 }
```

```
copsClientErrWrongOpcode OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A count of the total number of COPS messages that this client has received from COPS servers marked for this Client-Type that had a COPS protocol Op Code that should not have been sent to a COPS client e.g. Open-Requests. This value is cumulative since agent restart and is not zeroed on new connections."
```

```
::= { copsClientServerCurrentEntry 27 }
```

```
copsClientKaTimedoutClients OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"A count of the total number of times that this client has been shut down for this Client-Type by COPS servers that had detected a COPS protocol Keepalive timeout. This value is cumulative since agent restart and is not zeroed on new connections."
```

```
::= { copsClientServerCurrentEntry 28 }
```

copsClientErrAuthFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of times that this client has received a COPS message marked for this Client-Type which could not be authenticated using the authentication mechanism used by this client."

::= { copsClientServerCurrentEntry 29 }

copsClientErrAuthMissing OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A count of the total number of times that this client has received a COPS message marked for this Client-Type which did not contain authentication information."

::= { copsClientServerCurrentEntry 30 }

copsClientConfigGroup OBJECT IDENTIFIER ::= { copsClientMIBObjects 3 }

copsClientServerConfigTable OBJECT-TYPE

SYNTAX SEQUENCE OF CopsClientServerConfigEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Table of possible COPS servers to try to connect to in order of copsClientServerConfigPriority. There may be multiple entries in this table for the same server and client-type which specify different security mechanisms: these mechanisms will be attempted by the client in the priority order given. Note that a server learned by means of PDPRedirect always takes priority over any of these configured entries."

::= { copsClientConfigGroup 1 }

copsClientServerConfigEntry OBJECT-TYPE

SYNTAX CopsClientServerConfigEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A set of configuration information regarding a single

COPS server from the point of view of a COPS client."

```
INDEX { copsClientServerConfigAddrType,
        copsClientServerConfigAddress,
        copsClientServerConfigClientType,
        copsClientServerConfigAuthType }
 ::= { copsClientServerConfigTable 1 }
```

CopsClientServerConfigEntry ::=

```
SEQUENCE {
    copsClientServerConfigAddrType      InetAddressType,
    copsClientServerConfigAddress        InetAddress,
    copsClientServerConfigClientType    INTEGER,
    copsClientServerConfigAuthType      CopsAuthType,
    copsClientServerConfigTcpPort       CopsTcpPort,
    copsClientServerConfigPriority       Integer32,
    copsClientServerConfigRowStatus     RowStatus
}
```

copsClientServerConfigAddrType OBJECT-TYPE

```
SYNTAX      InetAddressType
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The type of address in copsClientServerConfigAddress."
 ::= { copsClientServerConfigEntry 1 }
```

copsClientServerConfigAddress OBJECT-TYPE

```
SYNTAX      InetAddress
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The IPv4, IPv6 or DNS address of a COPS Server. Note that,
     since this is an index to the table, the DNS name must be
     short enough to fit into the maximum length of indices allowed
     by the management protocol in use."
REFERENCE
    "RFC 2748 section 2.3"
 ::= { copsClientServerConfigEntry 2 }
```

copsClientServerConfigClientType OBJECT-TYPE

```
SYNTAX      INTEGER (0..65535)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The COPS protocol Client-Type for which this entry
     applies and for which this COPS server is capable
     of serving. Multiple Client-Types can be served by a
     single COPS server."
```

REFERENCE

"RFC 2748 section 6, IANA"

::= { copsClientServerConfigEntry 3 }

copsClientServerConfigAuthType OBJECT-TYPE

SYNTAX CopsAuthType

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The type of authentication mechanism for this COPS client to request when negotiating security at the start of a connection to a COPS server."

REFERENCE

"RFC 2748 section 4."

::= { copsClientServerConfigEntry 4 }

copsClientServerConfigTcpPort OBJECT-TYPE

SYNTAX CopsTcpPort

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The TCP port number on the COPS server to which the client should connect."

::= { copsClientServerConfigEntry 5 }

copsClientServerConfigPriority OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The priority of this entry relative to other entries. COPS client will attempt to contact COPS servers for the appropriate Client-Type. Higher numbers are tried first. The order to be used amongst server entries with the same priority is undefined. COPS servers that are notified to the client using the COPS protocol PDP-Redirect mechanism are always used in preference to any entries in this table."

::= { copsClientServerConfigEntry 6 }

copsClientServerConfigRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"State of this entry in the table."

::= { copsClientServerConfigEntry 7 }

copsClientServerConfigRetryAlgrm OBJECT-TYPE

```
SYNTAX      INTEGER {
                other(1),
                sequential(2),
                roundRobin(3)
            }
```

```
MAX-ACCESS  read-write
```

```
STATUS      current
```

DESCRIPTION

"The algorithm by which the client should retry when it fails to connect to a COPS server."

```
DEFVAL { sequential }
```

```
::= { copsClientConfigGroup 2 }
```

copsClientServerConfigRetryCount OBJECT-TYPE

```
SYNTAX      Unsigned32
```

```
MAX-ACCESS  read-write
```

```
STATUS      current
```

DESCRIPTION

"A retry count for use by the retry algorithm. Each retry algorithm needs to specify how it uses this value.

For the 'sequential(2)' algorithm, this value is the number of times the client should retry to connect to one COPS server before moving on to another.

For the 'roundRobin(3)' algorithm, this value is not used."

```
DEFVAL { 1 }
```

```
::= { copsClientConfigGroup 3 }
```

copsClientServerConfigRetryIntvl OBJECT-TYPE

```
SYNTAX      TimeInterval
```

```
UNITS       "centi-seconds"
```

```
MAX-ACCESS  read-write
```

```
STATUS      current
```

DESCRIPTION

"A retry interval for use by the retry algorithm. Each retry algorithm needs to specify how it uses this value.

For the 'sequential(2)' algorithm, this value is the time to wait between retries of a connection to the same COPS server.

For the 'roundRobin(3)' algorithm, the client always attempts to connect to each Server in turn, until one succeeds or they all fail; if they all fail, then the client waits for the value of this interval before restarting the algorithm."

```
DEFVAL { 1000 }
```

```
::= { copsClientConfigGroup 4 }
```

```

-----
-- Conformance Information
-----

copsClientConformance OBJECT IDENTIFIER ::= { copsClientMIB 2 }

copsClientGroups OBJECT IDENTIFIER ::= { copsClientConformance 1 }
copsClientCompliances OBJECT IDENTIFIER ::= { copsClientConformance 2 }

-----
-- units of conformance
-----

copsDeviceStatusGroup OBJECT-GROUP
  OBJECTS {
    copsClientCapabilities,
    copsClientServerTcpPort, copsClientServerType,
    copsClientServerAuthType, copsClientServerLastConnAttempt,
    copsClientState, copsClientServerKeepaliveTime,
    copsClientServerAccountingTime, copsClientInPkts,
    copsClientOutPkts, copsClientInErrs, copsClientLastError,
    copsClientTcpConnectAttempts, copsClientTcpConnectFailures,
    copsClientOpenAttempts, copsClientOpenFailures,
    copsClientErrUnsupportedClienttype,
    copsClientErrUnsupportedVersion, copsClientErrLengthMismatch,
    copsClientErrUnknownOpcode, copsClientErrUnknownCnum,
    copsClientErrBadCtype, copsClientErrBadSends,
    copsClientErrWrongObjects, copsClientErrWrongOpcode,
    copsClientKaTimedoutClients, copsClientErrAuthFailures,
    copsClientErrAuthMissing
  }
  STATUS      current
  DESCRIPTION
    "A collection of objects for monitoring the status of
    connections to COPS servers and statistics for a COPS client."
  ::= { copsClientGroups 1 }

copsDeviceConfigGroup OBJECT-GROUP
  OBJECTS {
    copsClientServerConfigTcpPort, copsClientServerConfigPriority,
    copsClientServerConfigRowStatus,
    copsClientServerConfigRetryAlgrm,
    copsClientServerConfigRetryCount,
    copsClientServerConfigRetryIntvl
  }
  STATUS      current
  DESCRIPTION
    "A collection of objects for configuring COPS server

```

```
information."  
 ::= { copsClientGroups 2 }
```

```
-----  
-- compliance statements  
-----
```

copsClientCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION

"The compliance statement for device support of management of the COPS client."

MODULE

MANDATORY-GROUPS {
 copsDeviceStatusGroup, copsDeviceConfigGroup
}

OBJECT copsClientServerConfigTcpPort

MIN-ACCESS read-only

DESCRIPTION

"Write access is required only if the device supports the configuration of COPS server information."

OBJECT copsClientServerConfigPriority

MIN-ACCESS read-only

DESCRIPTION

"Write access is required only if the device supports the configuration of COPS server information."

OBJECT copsClientServerConfigRowStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is required only if the device supports the configuration of COPS server information."

OBJECT copsClientServerConfigRetryAlgrm

MIN-ACCESS read-only

DESCRIPTION

"Write access is required only if the device supports the configuration of COPS server information."

OBJECT copsClientServerConfigRetryCount

MIN-ACCESS read-only

DESCRIPTION

"Write access is required only if the device supports the configuration of COPS server information."

```
OBJECT      copsClientServerConfigRetryIntvl
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is required only if the device supports the
    configuration of COPS server information."
```

```
::= { copsClientCompliances 1 }
```

END

5. Acknowledgments

This document describes instrumentation for the client side of the COPS protocol which was defined by the RSVP Admission Policy (rap) Working Group, now known as the Resource Allocation Protocol (rap) Working Group.

6. Security Considerations

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of 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.

SNMPv1 by itself is not a secure environment. 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.

It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [USM] and the View-based Access Control Model [VACM] is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, 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.

7. References

- [ARCH] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.
- [V1PROTO] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.
- [V1SMI] Rose, M. and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.
- [V1CONCISE] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.
- [V1TRAPS] Rose, M., "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.
- [V2SMI] 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.
- [V2TC] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [V2CONFORM] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [V2COMMUNITY] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.
- [V2TRANS] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.
- [V2PROTO] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

- [V3INTRO] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.
- [V3MPC] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.
- [V3USM] Blumenthal, U. and B. Wijnen, "The User-Based Security Model (USM) for Version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.
- [V3APPS] Levi, D., Meyer, P. and B. Stewart, "SNMP Applications", RFC 2573, April 1999.
- [V3VACM] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.
- [MIB2] McCloghrie K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets", STD 17, RFC 1213, March 1991.
- [FRAMEWORK] Yavatkar, R., Pendarakis, D. and Guerin, R., "A Framework for Policy-based Admission Control", RFC 2753, January 2000.
- [COPS] Boyle, J., Cohen, R., Durham, D., Herzog, S., Rajan, R. and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.
- [RSVP] Braden, R. ed., Zhang, L., Berson, S., Herzog, S. and S. Jamin, "Resource ReSerVation Protocol (RSVP) Version 1 - Functional Specification", RFC 2205, September 1997.
- [COPSRSPV] Boyle, J., Cohen, R., Durham, D., Herzog, S., Rajan, R. and A. Sastry, "COPS Usage for RSVP", RFC 2749, January 2000.
- [SRVLOC] Guttman, E., Perkins, C., Veizades, J. and M. Day, "Service Location Protocol, Version 2", RFC 2608, June 1999.
- [ADDRESSMIB] Daniele, M., Haberman, B., Routhier, S. and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 2851, June 2000.

[PROCESS] Bradner, S., "The Internet Standards Process --
Revision 3", BCP 9, RFC 2026, October 1996.

8. Authors' Addresses

Andrew Smith

Fax: +1 415 345 1827
Email: ah_smith@pacbell.net

David Partain
Ericsson Radio Systems
Research and Innovation
P.O. Box 1248
SE-581 12 Linkoping
Sweden

Phone: +46 13 28 41 44
EMail: David.Partain@ericsson.com

John Seligson
Nortel Networks, Inc.
4401 Great America Parkway
Santa Clara, CA 95054
USA

Phone: +1 408 495 2992
EMail: jseligso@nortelnetworks.com

9. Notices

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication 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 implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

10. Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS 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.

Acknowledgement

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

