**1- وضعیت این یادداشت**

این RFC باز­نشری از RFC 1065 می­باشد، این سند فقط «وضعیت یادداشت» تغییر یافته و همچنین کمی اصلاحات چاپی نسبت به قبلی دارد.

محتوای فنی این سند نسبت به RFC1065 تغییری نکرده است.

این یادداشت ،تعاریف رایج برای ساختار و شناسایی اطلاعات مدیریت شبکه‌های TCP/IP را آماده کرده است.

به طور خاص، این یادداشت با یادداشت‌های همراهش اطلاعات مدیریت را بر اساس پروتکل‌های مدیریت شبکه بر پایه TCP/IP توصیف میکنند، این اسناد، معماری و سیستم عملی و ساده‌ای را برای مدیریت شبکه‌های TCP/IP و به طور ویژه اینترنت، ارائه می­دهند.

این یادداشت پروتکل استانداردی را برای جامعه‌ی اینترنتی تصریح می‌کند. وضعیت این یادداشت «توصیه شده» است. انتظار می‌رود پیاده سازی‌های TCP/IP در اینترنت که قابل مدیریت شبکه‌ای هستند، این معیارها را اتخاذ و پیاده‌سازی نمایند.

هیئت فعالیت‌های اینترنتی (The Internet Activites Board) توصیه میکند که همه‌ی پیاده سازی‌های IP و TCP قابل مدیریت شبکه‌ای باشند. این موضوع بر پیاده‌سازی Internet MIB (RFC-1156) و حداقل یکی از دو پروتکل توصیه شده‌ی مدیریت – SNMP (RFC-1157) یا CMOT (RFC-1095) – دلالت دارد. باید متذکر شد که در حال حاضر، SNMP یک استاندارد کامل اینترنت است و CMOT یک استاندارد پیش نویس است. برای کسب اطلاعات بیشتر در مورد کاربرد و قابلیت اجرای این استاندارد RFCهای Host and Gateway Requierments را مطالعه کنید.

**-2 مقدمه**

این یادداشت ساختارها و طرح‌های شناسایی رایج برای توصیف اطلاعات مدیریت استفاده شده در شبکه‌های TCP/IP را توصیف میکند.

به علاوه، توصیفات یک Object مدل اطلاعات نیز برای مدیریت شبکه همراه با یک مجموعه از type های کلی استفاده شده برای توصیف اطلاعات مدیریت، اضافه شده است.

توصیفات رسمی این ساختار با استفاده از Abstract Syntax Notation One (ASN.1) داده شده است.

این یادداشت به صورت گسترده به دغدغه‌های سازمانی و سیاست‌های اداری اهمیت میدهد. و نه objectهایی که مدیریت شده‌اند و نه پروتکل‌هایی که برای مدیریت آن objectها استفاده شده‌اند را مشخص نمی­کند،‌.

این مهم‌ها به وسیله دو یادداشت همراه تشریح شده‌اند: یکی Management Information Base (MIB) را ، و دیگری Simple Network Management Protocol (SNMP) را توصیف میکند.

این یادداشت براساس کارهای انجام شده توسط کارگروه مهندسی اینترنت (Internet engineering Task Force)،  به طور خاص یادداشت کاری با عنوان «ساختار و شناسایی اطلاعات مدیریت برای اینترنت» است. این یادداشت از یک ساختار مشتق شده از یادداشت ذکر شده استفاده میکند،‌ اما در یک مورد بسیار قابل توجه متفاوت است: آن یادداشت به طور کامل روی استفاده از مدیریت شبکه‌ی OSI-style تمرکز کرده است. همینطور برای استفاده به همراه SNMP مناسب نیست.

این یادداشت قصد رسیدن به دو هدف را دارد: سادگی و توسعه پذیری. هردو از یک نگرانی رایج الهام گرفته شده اند: اگرچه مدیریت شبکه‌های TCP/IP برای مدتی موضوع مطالعه بودند، مولفان گمان نمیکنند گستره و عمق چنین مفهومی کامل شده باشد. به صورت بی پرده تر، ما حس میکنیم تجربه‌های گذشته، با اینکه به جامعه بینش میدهند اما کاملا قطعی نیستند. با توسعه یک SMI ساده، کمترین تعداد محدودیت‌ها روی روش‌های بالقوه‌ی آینده اعمال میشوند. بعلاوه، با توسعه یک SMI توسعه پذیر بیشترین تعداد روش‌های بالقوه برای آزمایش کردن در دسترس خواهد بود.

باور بر این است که این یادداشت و دو یادداشت همراهش مطابق با راهنما‌های تنظیم شده در RFCهای زیر هستند:

RFC 1052 - “IAB Recommendations for Development of Internet Network Management Standard”

RFC 1109 - “Report Of the Second Ad Hoc Network Management Review Group”

به طور خاص ما حس میکنیم که این یادداشت به همراه یادداشتی که پایه‌ی اطلاعات مدیریت را توصیف میکند، بنیادی مستحکم برای مدیریت شبکه‌ی اینترنت ارائه میدهند.

-**3 ساختار و شناسایی اطلاعات مدیریت**

objectهای مدیریت شده به وسیله‌ی یک منبع ذخیره‌ی مجازی اطلاعات، تحت عنوان Management Information Base یا MIB در دسترس هستند. Objectها در MIB با استفاده از ASN.1 توصیف شده اند.

هر نوع از object (تحت عنوان object type) یک نام،‌ گرامر و کدگذاری دارد. نام آن به صورت منحصر به فرد تحت عنوان OBJECT IDENTIFIER نمایش داده میشود. یک OBJECT IDENTIFIER در‌واقع یک نام اختصاص داده شده به صورت مدیریتی است. در مورد سیاست‌های مدیریتی استفاده شده برای اختصاص دادن نام‌ها بعداً در این یادداشت بحث خواهد شد.

گرامر یک object type ساختار داده‌ای انتزاعی متناظر با همان object type توصیف میکند. برای مثال ساختار یک object type داده شده ممکن است INTEGER یا OCTET STRING باشد. بنابراین به طور کلی، ما باید اجازه دهیم هر ساختار ASN.1 برای استفاده در توصیف گرامر یک object type در دسترس باشد، این یادداشت عمداً ساختارهای ASN.1 که ممکن است استفاده شوند را محدود کرده است. این محدودیت‌ها منحصرا برای دستیابی به سادگی اعمال شده اند.

کدگذاری یک object type به طور ساده این است که چگونه نمونه‌های آن object type با استفاده از گرامر نوع آن object نمایش داده شده‌اند. به طور ضمنی ، زوج گره خورده تعریف گرامر یک object و کدگذاری مشخص می­کند چگونه object در زمان انتقال در شبکه نمایش داده می‌شود. این یادداشت استفاده از قوانین پایه‌ی کدگذاری ASN.1 را مشخص میکند.

تعریف استفاده از MIB برای مدیریت شبکه یا پروتکل مدیریت شبکه در حوزه‌ی این یادداشت نیست. همانطور که قبلاً اشاره شد این وظایف به یادداشت‌های همراه سپرده شده است. این یادداشت قصد دارد تا محدودیت‌های قرار داده شده بر یادداشت‌های همراه خود را کاهش دهد تا عمومیت افزایش پیدا کند. هرچند در برخی موارد محدودیت‌هایی (‌برای مثال گرامری که ممکن است در زمان توصیف object type در MIB استفاده شود) به منظور تقویت یک سبک مدیریت خاص ایجاد شده است. ویرایش‌های آینده‌ی این یادداشت ممکن است این محدودیت‌ها را حذف کنند.

**-3-1 نام‌ها**

نام‌ها برای شناسایی object های مدیریت شده استفاده میشوند. این یادداشت نام‌هایی را مشخص میکند که به طور طبیعی سلسله مراتبی هستند. مفهوم OBJECT IDENTIFIER برای مدل کردن این ایده استفاده شده است. یک OBJECT IDENTIFIER میتواند برای اهدافی به غیر از نامگذاری object typeهای مدیریت شده استفاده شود. برای مثال، هر استاندارد بین‌المللی به منظور شناسایی یک OBJECT IDENTIFIER اختصاصی دارد. به طور خلاصه،‌ OBJECT IDENTIFIERها برای شناسایی برخی object ها هستند، بدون در نظر گرفتن معانی لغوی که به همراه آن object هستند. (برای مثال یک object شبکه، یک سند استاندارد و غیره …).

یک OBJECT IDENTIFIER ترتیبی است از اعداد صحیح که یک درخت global را پیمایش میکنند. درخت شامل یک ریشه است که به تعدادی گره برچسب گذاری شده به وسیله یال‌ها متصل است. هر گره به نوبه‌ی خود ممکن است دارای فرزندانی باشد که خود برچسب گذاری شده‌اند. در این مورد ممکن است به آن گره یک زیر درخت بگوییم. این فرآیند ممکن است تا یک عمق قردادی ادامه پیدا کند.نکته اصلی در مفهوم OBJECT IDENTIFIER درک این موضوع است که کنترل‌ های مدیریتی اختصاص داده شده به گره ممکن است هنگام پیمایش درخت واگذار شوند. یک برچسب، جفتی از یک توصیف متنی مختصر و یک عدد صحیح می­باشد.

خود ریشه برچسب گذاری نشده است، اما حداقل ۳ فرزند به طور مستقیم در زیر خودش دارد: یک گره توسط سازمان استاندارد سازی بین‌المللی مدیریت شده است، که برچسب iso(1) دارد، دیگری توسط کمیته‌ی مشورتی تلفن و تلگراف بین‌المللی مدیریت شده است، که برچسب ccitt(0) دارد و سومی مشترکا توسط ISO و CCITT مدیریت شده است و برچسب joint-iso-ccitt(2) دارد.

سازمان ISO در زیر گره iso(1) زیر درختی را برای استفاده توسط سایر سازمان‌ها تعیین کرده است که برچست org(3) دارد. دو گره از گره‌های فرزند این والد، به مؤسسه ملی فناوری و استانداردهای ایالات متحده اختصاص داده شده است. یکی از این زیر درخت‌ها توسط NIST به وزارت دفاع ایالات متحده انتقال داده شده است که برچسب dod(6) دارد.

تا زمان تألیف این یادداشت، وزارت دفاع مشخص نکرده است که چگونه از زیر درخت OBJECT IDENTIFIER خود استفاده خواهد کرد. این یادداشت فرض میکند که وزارت دفاع گره‌ای را به جامعه‌ی اینترنت اختصاص خواهد داد، تا به صورت زیر توسط هیئت فعالیت‌های اینترنتی (IAB)مدیریت شود:

internet OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }

این زیر درخت OBJECT IDENTIFIER اینترنت است که با این پیشوند آغاز می­شود:

1.3.6.1.

این یادداشت به عنوان یک استاندارد تأیید شده توسط IAB،اکنون سیاستی که این زیردرخت OBJECT IDENTIFIER بر مبنای آن مدیریت شده است را مشخص میکند. در ابتدا،‌ ۴ گره حضور دارند:

directory OBJECT IDENTIFIER ::= { internet 1 }

mgmt OBJECT IDENTIFIER ::= { internet 2 }

experimental OBJECT IDENTIFIER ::= { internet 3 }

private OBJECT IDENTIFIER ::= { internet 4 }

**-3-1-1 directory**

زیر درخت directory(1) برای استفاده با یک یادداشت آتی، که توضیح میدهد چگونه OSI Directory ممکن است در اینترنت استفاده شود، رزرو شده است.

**3-1-2- mgmt**

زیر درخت mgmt(2) برای شناسایی object هایی استفاده می‌شود که در اسناد تأیید شده‌ی IAB تعریف شده اند. مدیریت زیردرخت mgmt(2) توسط IAB به مرجع شماره‌های تخصیص یافته‌ی اینترنت واگذار شده است. وقتی RFC هایی که نسخه‌های جدیدی از استاندارد اینترنت MIB را تعریف میکنند،‌تایید شوند، یک OBJECT IDENTIFIER توسط مرجع شماره‌های تخصیص یافته‌ی اینترنت برای شناسایی object هایی که توسط آن یادداشت تعریف شده‌اند،‌ به آن‌ها اختصاص داده میشود.

برای مثال RFCای که استاندارد ابتدایی اینترنت MIB را تعریف میکند ممکن است شماره‌ی سند مدیریت ۱ به آن اختصاص داده شده باشد. این RFC ممکن است از OBJECT IDENTIFIER زیر استفاده کرده باشد:

{mgmt 1}

یا

1.3.6.1.2.1

در تعریف استاندارد اینترنت MIB.

تولید نسخه‌های جدید استاندرد اینترنت MIB، فرآیندی سخت است. بخش ۵ این یادداشت، قوانینی در زمان تعریف نسخه‌ جدید استفاده می شود را توصیف میکند.

**3-1-3 – آزمایشی(experimental)**

زیر درخت experimental(3) برای شناسایی object های مورد استفاده در آزمایش‌های اینترنتی استفاده می­شود. مدیریت زیردرخت experimental(3) توسط IAB به مرجع شماره‌های تخصیص یافته‌ی اینترنت واگذار شده است.

برای مثال،‌یک آزمایش کننده ممکن است عدد ۱۷ را دریافت کرده باشد و OBJECT IDENTIFIER

{experimental 17}

یا

1.3.6.1.3.17

را برای استفاده در دسترس داشته باشد.

به عنوان بخشی از فرآیند تخصیص، مرجع شماره‌های تخصیص یافته ی اینترنت ممکن است نیازمندی‌هایی ایجاد کند برای این موضوع که زیر درخت مورد نظر چگونه مورد استفاده قرار گیرد.

**3-1-4 – محرمانه(private)**

زیر درخت private(4) برای شناسایی objectهایی که به صورت یک‌طرفه تعریف شده، استفاده می­شود. مدیریت زیر درخت private(4) توسط IAB به مرجع شماره‌های تخصیص یافته‌ی اینترنت واگذار شده است.در ابتدا این زیر درخت حداقل یک فرزند دارد:

enterprises OBJECT IDENTIFIER ::= { private 1 }

زیر درخت enterprises(1)،‌ علاوه بر موارد دیگر،‌ برای اجازه دادن به اشخاص در زمینه‌ی تهیه‌ی زیر سیستم‌های شبکه‌ای خودشان برای ثبت مدل‌های محصولاتشان، مورد استفاده قرار می­گیرد.

درکنار دریافت یک زیر درخت، آن شرکت ممکن است، برای مثال در این زیر درخت MIB Object جدیدی تعریف کند. به علاوه، به شدت توصیه شده است که شرکت به منظور تهیه‌ی یک مکانیزم شناسایی غیر مبهم برای استفاده در پروتکل های مدیریت، زیر سیستم‌های شبکه‌ای خود را زیر این زیرسیستم ثبت کند. برای مثال اگر شرکت Flintstones Inc. زیرسیستم‌های شبکه‌ای تولید کند، آنگاه آن‌ها میتوانند یک گره در زیر زیردرخت enterprise از مرجع شماره‌های تخصیص یافته‌ی اینترنت درخواست کنند. چنین گره‌ای ممکن است شماره گذاری شده باشد:

1.3.6.1.4.1.42

آنگاه شرکت Flintstones Inc ممکن است Fred Router خود را تحت نام زیر ثبت کند:

1.3.6.1.4.1.42.1.1

**3-2– گرامر(Syntax)**

گرامر(Syntax) برای تعریف ساختار مربوط به object type استفاده میشود. قالب‌های ASN.1 برای تعریف این ساختار استفاده شده اند، هرچند عمومیت کلی ASN.1 مجاز نیست.

ObjectSyntax از نوع ASN.1 گرامر متفاوتی را تعریف میکند که ممکن است خود در تعریف یک نوع object استفاده شود.

۳.۲.۱ - type های ابتدایی(Primitive Types)

فقط type های ابتدایی ASN.1 (ASN.1 Primitive Types) INTEGER، OCTET STRING، OBJECT IDENTIFIER و NULL مجاز میباشند. گاهی اوقات به آن‌ها typeهای غیرتجمیعی(non-aggregate) گفته میشود.

**3-2-1-1 – دستورالعمل‌ها برای INTEGER های شمارش شده**

اگر یک INTEGER شمارش شده به عنوان یک object type لیست شده باشد، آنگاه یک عدد نام گذاری شده که مقدار 0 دارد نباید در لیست شمارش‌ها حضور داشته باشد. استفاده از این مقدار ممنوع شده است.

**3-2-2 - Constructor Types**

type سازنده‌ی ASN.1 یعنی SEQUENCE مجاز است، و برای تولید هر دو لیست یا جداول از آن استفاده میشود.

برای لیست‌ها، گرامر شکل زیر را به خود میگیرد:

SEQUENCE {<type1>, ..., <typeN>}

که هر <type> به یکی از typeهای ابتدایی که در بالا گفته شد تبدیل میشود. به علاوه، این typeهای ASN.1 همیشه حضور دارند ( شروط DEFAULT و OPTIONAL در تعریف SEQUENCE ظاهر نمیشوند).

برای جداول، گرامر شکل زیر را به خود میگیرد:

SEQUENCE of <entry>

که <entry> به یک سازنده‌ی لیست تجزیه می­شود.

گاهی اوقات به لیست‌ها و جداول typeهای انبوه یا مجتع گفته می­شود.

**3-2-3– typeهای تعریف شده**

به علاوه، typeهای گستره‌ی برنامه جدید نیز ممکن است تعریف شوند، مادامی که به یک type ابتدایی مانند لیست، جداول یا سایر typeهای در گستره‌ی برنامه تبدیل شوند که به صورت ضمنی در ASN.1 تعریف شده اند. در ابتدا، تعداد کمی typeهای در گستره‌ی برنامه تعریف شده اند. یادداشت‌های آینده بدون شک typeهای دیگر را به محض اینکه یک توافق و اجماع ایجاد شود ، تعریف خواهند کرد.

**3-2-3-1- آدرس‌ شبکه(NetworkAddress)**

این انتخاب آدرسی را از یکی از چندین خانواده‌ی پروتکل‌ها به نمایش می­گذارد. در حال حاضر، تنها یک خانواده‌ی پروتکل، خانواده‌ی اینترنت، در این انتخاب موجود است.

**3-2-3-2- آدرس‌های ip(IpAddress)**

این type در گستره‌ی برنامه ،یک آدرس اینترنتی ۳۲ بیتی را به نمایش میگذارد. این آدرس به عنوان یک OCTET STRING با طول ۴، در ترتیب بایت شبکه نمایش داده می­شود.

وقتی این نوع ASN.1 با استفاده از قوانین پایه‌ی کدگذاری ASN.1 کدگذاری شود، فقط شکل کدگذاری ابتدایی (اساسی) باید استفاده شود.

**3-2-3-3- شمارنده(Counter)**

این type در گستره‌ی برنامه، یک عدد صحیح غیر منفی نمایش می­دهد که به صورت یکی یکی تا زمان رسیدن به یک مقدار بیشینه افزایش می­یابد و مجددا از صفر شروع می­کند این یادداشت یک مقدار بیشینه‌ی 2^31-1(4294967295 دهدهی) را برای شمارنده‌ها مشخص می­کند.

**3-2-3-4- معیار(Gauge)**

این type در گستره‌ی برنامه، یک عدد صحیح غیر منفی را نمایش می­دهد که ممکن است افزایش یا کاهش پیدا کند، اما در یک مقدار بیشینه نگه‌داشته می­شود. این یادداشت یک مقدار بیشینه‌ی 2^32-1 (4294967295 دهدهی) را برای معیارها مشخص می­کند.

**3-2-3-5- نشانه‌های زمان(TimeTicks)**

این type در گستره‌ی برنامه، یک عدد صحیح غیر منفی را نمایش میدهد که زمان را در صدم ثانیه از یک مبدأ خاص می­شمارد. وقتی که object typeها در MIB که از این type ASN.1 استفاده می­کند، تعریف شوند،‌ توصیف object type مرجع این مبدأ زمانی را شناسایی میکند.

**3-2-3-2 – مبهم(Opaque)**

این type در گستره‌ی برنامه از قابلیت گذراندن گرامر قراردادی ASN.1 پشتیبانی میکند. یک مقدار با استفاده از قوانین پایه‌ی ASN.1 در قالب رشته‌ای از octetها کدگذاری میشود. این به نوبه‌ی خود،‌ به عنوان یک OCTET STRING و تحت تأثیر آن مقدار ASN.1 اصلی ”double-wrapping”، کدگذاری میشود.

توجه شود که یک پیاده‌سازی تأیید کننده لازم است فقط قابلیت قبول و شناسایی داده‌های کدگذاری شده به صورت مبهم را داشته باشد. لازم نیست داده را باز و محتوای آن را تفسیر کند.

تذکر دیگر این است که با استفاده از type خارجی (EXTERNAL) در ASN.1 کدگذاری‌های غیر از ASN.1 نیز ممکن است در داده‌های کدگذاری شده به صورت مبهم استفاده شود.

**3-3 – کدگذاری ها(Encodings)**

به محض اینکه یک نمونه از یک object type شناسایی شد، مقدارش ممکن است با اعمال قوانین پایه‌ی کدگذاری ASN.1 روی گرامر آن object type انتقال داده شود.

**4- objectهای مدیریت شده(Managed Objects)**

اگرچه هدف این یادداشت تعریف object های MIB نیست، اما این یادداشت قالبی را برای استفاده توسط سایر یادداشت‌هایی که این objectها را تعریف میکنند، مشخص میکند.

تعریف یک object type شامل ۵ زمینه است:

OBJECT: یک نام متنی،‌ تحت عنوان OBJECT DESCRITOR برای object typeها به همراه OBJECT IDENTIFIER مربوط به خودش.

گرامر(Syntax:): گرامر انتزاعی برای object type. این گرامر باید به یک نوع ObjectSyntax از ASN.1 تبدیل شود ( که در زیر تعریف شده است).

تعریف(Definition:): یک توصیف متنی از معانی object type. از آنجایی که این MIB قرار است در محیط های چند طرفه (چند شخصی) مورد استفاده قرار گیرد، پیاده سازی‌ها باید اطمینان حاصل کنند که نمونه‌هایشان از یک object این تعریف را برآورده کند. مثلاً حیاتی است که object ها در همه‌ی ماشین‌ها معنی ثابت داشته باشند.

دسترسی(Access:): یکی از موارد: فقط خواندنی، قابل خواندن و نوشتن، فقط نوشتنی، یا غیرقابل دسترسی.

وضعیت(Status): یکی از موارد: اجباری، اختیاری، یا منسوخ

یادداشت‌های آتی همچنین ممکن است زمینه‌های دیگری را برای objectهایی که تعریف میکنند مشخص کنند.

**4-1– دستورالعمل‌های نام‌های object**

هیچ object type ی در MIB استاندارد اینترنت نباید یک زیر شناسه 0 در نامش داشته باشد. این مقدار برای استفاده به همراه الحاقیه‌های آینده رزرو شده است.

هر OBJECT DESCRIPTOR مربوط به یک object type در MIB استاندار اینترنت باید یک رشته‌ی منحصر به فرد اما قابل بخاطر سپاری و چاپ باشد. این موضوع یک زبان رایج برای انسان‌ها را به منظور استفاده در زمان بحث در مورد MIB توسعه میدهد، و همچنین طرح ریزی جداول را برای رابط‌های کاربری آسانتر میکند.

**4-2 – object typeها و نمونه‌ها**

یک object type تعریفی از یک نوع object مدیریت شده است، که ذاتا به صورت اعلانی است.

در مقابل، یک نمونه از object، تعریفی از object است که به یک مقدار مشخص محدود و مقید شده است. برای مثال مفهوم یک ورودی در یک جدول مسیریابی ممکن است در MIB تعریف شده باشد. چنین مفهومی به یک object type مربوط است، ورودی‌های انفرادی در یک جدول مسیریابی خاص که در زمانی مشخص وجود دارد نمونه‌های object از آن object type هستند.

مجموعه‌ای از object typeها در MIB تعریف شده­اند. هرکدام از این چنین object typeها به صورت منحصر به فرد توسط OBJECT IDENTIFIER خودش نام گذاری شده است و همچنین یک نام متنی دارد، که OBJECT DESCRIPTOR همان object type است. مفادی که توسط آن‌ها به نمونه‌های object ارجاع داده می‌شود در MIB تعریف نشده اند. ارجاع به نمونه‌های object به وسیله‌ی یک مکانیزم مختص به پروتکل قابل دستیابی است. موافق بودن با SMI برای تعریف این مکانیزم مسئولیت و وظیفه‌ی هر پروتکل مدیریتی است.

یک object type ممکن است طوری در MIB تعریف شده باشد که نمونه‌ای از آن object type ، انبوهی از اطلاعاتی که توسط نمونه‌های تعدادی از object typeهای «تابعه» به نمایش گذاشته شده‌اند، را نمایش دهد. برای مثلاً فرض کنیم object typeهای زیر در MIB تعریف شده‌اند:

OBJECT:

-------

atIndex {atEntry 1}

Syntax:

INTEGER

Definition:

The interface number for the physical address.

Access:

read-write.

Status:

mandatory.

OBJECT:

-------

atPhysAddress {atEntry 2}

Syntax:

OCTET STRING

Definition:

The media-dependent physical address.

Access:

read-write.

Status:

mandatory.

OBJECT:

-------

atNetAddress {atEntry 3}

Syntax:

NetworkAddress

Definition:

The network address corresponding to the media-dependent physical

address.

Access:

read-write.

Status:

mandatory.

سپس، Object چهارم همچنین ممکنه در MIB به صورت زیر تعریف شود:

OBJECT:

-------

atEntry {atTable 1}

Syntax:

AtEntry ::= SEQUENCE {

atIndex

INTEGER,

atPhysAddress

OCTET STRING,

atNetAddress

NetworkAddress

}

Definition:

An entry in the address translation table.

Access:

read-write.

Status:

mandatory.

هر نمونه از این object type دربردارنده‌ی اطلاعاتی هستند که توسط نمونه‌های سه object type قبلی به نمایش گذاشته شده­اند. به Object typeای که به این روش تعریف شده است یک لیست می­گویند.

به طور مشابه، جداول می­توانند توسط انبوهی از typeهای لیست شکل یابند. برای مثال یک object type پنجم ممکن است در MIB تعریف شده باشد:

OBJECT:

------

atTable { at 1 }

Syntax:

SEQUENCE OF AtEntry

Definition:

The address translation table.

Access:

read-write.

Status:

mandatory.

به طوریکه هر نمونه از atTable object اطلاعاتی را دربرداشته باشد که توسط مجموعه‌ای از atEntry object type ها که مجموعاً نمونه‌ی مشخص از atTable Object را تشکیل می­دهند، نمایش داده شده است، این یک جدول تفسیر آدرس مشخص است.

در نظر بگیرید چگونه شخصی ممکن است یه یک object ساده در یک جدول اشاره کند (ارجاع دهد). با ادامه دادن مثال قبلی، شخصی ممکن است object type را به این صورت نام گذاری کند:

{atPhysAddress}

و نمونه‌ی object را با استفاده از مکانیز مختص به پروتکل مشخص کند.

{atNetAddress} = {internet "10.0.0.52”}

این جفت سازی object type و نمونه‌ی object، میتواند به همه‌ی نمونه‌های atPhysAddress اشاره کند که بخشی از هر ورودی در برخی جداول تفسیر آدرس هستند که مقدار atNetAddress مربوط به آن‌ها {internet “10.0.0.52”} است.

برای ادامه با این مثال، در نظر بگیرید چگونه شخصی ممکن است به object انبوهی (لیست) در یک جدول اشاره کند.

با نامگذاری object type به صورت

{internet "10.0.0.52”}

و مشخص کردن نمونه‌ی object با استفاده از یک مکانیزم مختص به پروتکل

{atNetAddress} = {internet "10.0.0.52”}

به همه‌ی نمونه‌های یک ورودی در جدولی اشاره میکند که مقدار atNetAddress مربوط به آن {internet “10.0.0.52”} است.

هر پروتکل مدیریت باید مکانیزمی را برای دسترسی به object typeهای ساده (غیر انبوه) تهیه کند. هر پروتکل مدیریت مشخص می­کند که آیا دسترسی به object typeهای انبوه را پشتیبانی می­کند یا خیر. به علاوه،‌ پروتکل باید مشخص کند کدام نمونه‌ها در زمانی که یک جفت نمونه/نوع object به بیشتر از یک نمونه از یک type اشاره می­کند، «بازگشت داده شده» است.

برای برآمدن از عهده‌ی پشتیبانی پروتکل‌های گوناگون مدیریت، همه اطلاعاتی که توسط آن‌ها نمونه‌های یک object type مشخص ممکن است از یکدیگر متمایز شوند، توسط نمونه‌های object type های تعریف شده در MIB نمایش داده شده اند.

**4-3- ماکروها(Macros) برای objectهای مدیریت شده(Managed Objects)**

به منظور آسان سازی استفاده از ابزارهای پردازش تعریف در MIB،‌ ماکروی (کلان) OBJECT-TYPE ممکن است مورد استفاده قرار گیرد. این ماکرو اجازه میدهد جنبه‌های کلیدی یک object type به روشی رسمی نمایش داده شود.

OBJECT-TYPE MACRO ::=

BEGIN

TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)

"ACCESS" Access

"STATUS" Status

VALUE NOTATION ::= value (VALUE ObjectName)

Access ::= "read-only"

| "read-write"

| "write-only"

| "not-accessible"

Status ::= "mandatory"

| "optional"

| "obsolete"

END

با در نظر گرفتن object typeهایی که قبلاً تعریف شدند،‌ ممکن است متصور شویم که تعریف زیر در MIB حضور داشته باشد:

atIndex OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

::= { atEntry 1 }

atPhysAddress OBJECT-TYPE

SYNTAX OCTET STRING

ACCESS read-write

STATUS mandatory

::= { atEntry 2 }

atNetAddress OBJECT-TYPE

SYNTAX NetworkAddress

ACCESS read-write

STATUS mandatory

::= { atEntry 3 }

atEntry OBJECT-TYPE

SYNTAX AtEntry

ACCESS read-write

STATUS mandatory

::= { atTable 1 }

atTable OBJECT-TYPE

SYNTAX SEQUENCE OF AtEntry

ACCESS read-write

STATUS mandatory

::= { at 1 }

AtEntry ::= SEQUENCE {

atIndex

INTEGER,

atPhysAddress

OCTET STRING,

atNetAddress

NetworkAddress

}

پنج تعریف اول object typeها را توصیف می­کنند، و برای مثلاً OBJECT DESCRIPTOR atIndex را به OBJECT IDENTIFIER {atEntry 1} ربط می­دهند. به علاوه گرامر این Object به صورت (INTEGER) و به همراه اجازه‌ی دسترسی (خواندن و نوشتن) و وضعیت (اجباری) تعریف شده است. تعریف ششم یک ASN.1 type که AtEntry نامیده می‌شود را توصیف می­کند.

**۵ – الحاقات به MIB**

هر سند استاندارد اینترنت MIB تمام اسناد قبل از خود را منسوخ می­کند. قسمت نام، تحت عنوان عقبه با توجه به OBJECT IDENTIFIER زیر:

{mgmt version-number}

مورد استفاده برای نام گذاری object ها ، باید بین نسخه‌ها بدون تغییر باقی بماند. نسخه‌های جدید ممکن است:

۱ – object type های قدیمی را منسوخ اعلام کنند (در صورت لزوم)، اما نام‌هایشان را حذف نمی­کنند

۲- افزودن به تعریف یک object type مربوط به یک لیست با استفاده از object typeهای غیر انبوه، به object typeهای موجود در لیست. یا،

۳- تعریف یک object type کاملاً جدید

نسخه‌های جدید ممکن است شامل موارد زیر نباشند:

۱- تغییر معانی هر object تعریف شده‌ی سابق بدون تغییر نام آن object

این قوانین مهم هستند، بدلیل اینکه باعث پشتیبانی آسان‌تری برای نسخه‌های چندگانه‌ی استاندارد اینترنت MIB می­شوند. بطور خاص، معانی مربوط به عقبه‌ی یک نام، در میان نسخه‌های مختلف MIB ثابت می­ماند. بدلیل اینکه نسخه‌های چندگانه‌ی MIB ممکن است در «فضای-عقبه» با هم مشترک باشند، پیاده‌سازی­هایی که نسخه‌های چندگانه از MIB را پشتیبانی می­کنند ،‌ می­توانند به طور گسترده‌ای ساده سازی شوند.

هرچند، به عنوان یک پیامد، یک عامل مدیریت ممکن است نمونه‌ای مربوط به والد (در سطح بالاتر) یک object type مورد نظر را بازگرداند. با پیروی از اصل قدرتمندی،‌ در این مورد استثنایی، یک مدیر باید هر اطلاعاتی فرای تعریف object type مورد نظر را نادیده بگیرد. هرچند،‌ اصل قدرتمندی نیازمند این است که یک عمل به رعایت فعالیت­های کنترلی اهمیت دهد: اگر یک نمونه، همان گرامر مشابه object type مورد انتظار خود را ندارد آنگاه آن فعالیت‌های کنترلی باید با شکست مواجه شوند. در هر دو مورد نظارت و کنترل، نام object ای که با یک عملیات بازگردانده می‌شود باید مشابه نامی باشد که توسط یک عملیات درخواست داده می­شود.

**6. تعاریف**

RFC1155-SMI DEFINITIONS ::= BEGIN

EXPORTS -- EVERYTHING

internet, directory, mgmt,

experimental, private, enterprises,

OBJECT-TYPE, ObjectName, ObjectSyntax, SimpleSyntax,

ApplicationSyntax, NetworkAddress, IpAddress,

Counter, Gauge, TimeTicks, Opaque;

-- the path to the root

internet OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }

directory OBJECT IDENTIFIER ::= { internet 1 }

mgmt OBJECT IDENTIFIER ::= { internet 2 }

experimental OBJECT IDENTIFIER ::= { internet 3 }

private OBJECT IDENTIFIER ::= { internet 4 }

enterprises OBJECT IDENTIFIER ::= { private 1 }

-- definition of object types

OBJECT-TYPE MACRO ::=

BEGIN

TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)

"ACCESS" Access

"STATUS" Status

VALUE NOTATION ::= value (VALUE ObjectName)

Access ::= "read-only"

| "read-write"

| "write-only"

| "not-accessible"

Status ::= "mandatory"

| "optional"

| "obsolete"

END

-- names of objects in the MIB

ObjectName ::=

OBJECT IDENTIFIER

-- syntax of objects in the MIB

ObjectSyntax ::=

CHOICE {

simple

SimpleSyntax,

-- note that simple SEQUENCEs are not directly

-- mentioned here to keep things simple (i.e.,

-- prevent mis-use). However, application-wide

-- types which are IMPLICITly encoded simple

-- SEQUENCEs may appear in the following CHOICE

application-wide

ApplicationSyntax

}

SimpleSyntax ::=

CHOICE {

number

INTEGER,

string

OCTET STRING,

object

OBJECT IDENTIFIER,

empty

NULL

}

ApplicationSyntax ::=

CHOICE {

address

NetworkAddress,

counter

Counter,

gauge

Gauge,

ticks

TimeTicks,

arbitrary

Opaque

-- other application-wide types, as they are

-- defined, will be added here

}

-- application-wide types

NetworkAddress ::=

CHOICE {

internet

IpAddress

}

IpAddress ::=

[APPLICATION 0] -- in network-byte order

IMPLICIT OCTET STRING (SIZE (4))

Counter ::=

[APPLICATION 1]

IMPLICIT INTEGER (0..4294967295)

Gauge ::=

[APPLICATION 2]

IMPLICIT INTEGER (0..4294967295)

TimeTicks ::=

[APPLICATION 3]

IMPLICIT INTEGER (0..4294967295)

Opaque ::=

[APPLICATION 4] -- arbitrary ASN.1 value,

IMPLICIT OCTET STRING -- "double-wrapped"

END