|  |
| --- |
| نگارش : محمد حسن شهبازی ، احمد مصیبی ، نسیم نوایی ، پروین کرمی |
| پایگاه اطلاعات مدیریتی برای مدیرت شبکه های اینرتنت مبتنی بر tcp/ip  MIB II |
| **وضعیت این یادداشت** |
| این یادداشت دومین نسخه از پایگاه اطلاعات مدیریتی(MIB II) برای استفاده با پروتکل های مدیریت شبکه در شبکه های مبتنی بر TCP/IPرا تعریف می کند. این RFC یک پروتکل ردیابی استانداردهای IAB را برای جامعه اینترنت معرفی می کند و پذیرای پیشنهادات و مباحثات در جهت بهبود RFC می باشد. جهت دسترسی به حالت استانداردسازی و وضعیت این پروتکل به نسخه فعلی «استانداردهای رسمی پروتکل IAB» مراجعه کنید. انتشار این یادداشت بدون محدودیت و آزاد است. |
| **فهرست مندرجات**  **چکیده**  **مقدمه**  **تغیرات نسبت RFC 1156**  **موارد منسوخ شده**  **آدرس های فیزیکی**  **گروه سیستم ها**  **گروه رابط ها**  **گروه ترجمه آدرس**  **گروه IP** |
| **چکیده**  در این یادداشت دومین نسخه پایگاه اطلاعات مدیریتی(MIB II) برای استفاده با پروتکل های مدیریت شبکه در شبکه های مبتنی بر TCP/IP تعریف می شود. به طور ویژه در کنار یادداشت های مرتبط با این نسخه که به معرفی SMI(RFC 1155) می پردازد و هم چنین به موازات آن در پروتکل مدیریت شبکه(SNMP-RFC1157) برای شبکه های مبتنی بر TCP/IP، این اسناد یک معماری ساده و قابل اجرا و سیستمی برای مدیریت شبکه های مبتنی بر TCP/IP و به طور خاص برای جامعه اینترنت ارائه می دهند. |
| **مقدمه**  همان طور که در RFC 1052 گفته شد، پیشنهادات IAB برای توسعه استانداردهای مدیریت شبکه های اینترنتی به صورت یک راهبرد دوبخشی برای مدیریت شبکه های مبتنی بر TCP/IP ارائه گردید. بخش اول) راهبرد کوتاه مدت: SNMP بصورت کوتاه مدت و به منظور مدیریت گره ها در بستر اینترنت در نظر گرفته شود. بخش دوم) راهبرد بلند مدت: کارکرد مدل OSI در مدیریت شبکه بصورت بلند مدت مورد بررسی قرار بگیرد. این دو مستند با هدف تعریف اطلاعات مدیریت ایجاد شدند؛ بدین صورت که در RFC 1065 ساختار اطلاعات مدیریتی(SMI) و در RFC 1066 پایگاه اطلاعات مدیریتی(MIB) تعریف شده است. هر دو این مستندات با این قید که با SNMP و چارچوب مدیریت شبکه OSI سازگار باشند، طراحی شدند.  راهبردی که در بخش قبلی ذکر شد در بعد کوتاه مدت کاملا موفق بود: فناوری مدیریت شبکه های مبتنی بر اینترنت، طی چند ماه توسط هر دو جامعه پژوهشی و تجاری به کار گرفته شد. در نتیجه این اتفاق، در یک بازه زمانی بخش قابل توجهی از جامعه اینترنت به قابلیت مدیریت پذیری مجهز شدند.  همان طور که در RFC 1109 با عنوان «گزارش دومین گروه ارزیابی مدیریت شبکه های موردی(Ad hoc)» گفته شد، نیازمندی های SNMP و چارچوب های مدیریت شبکه OSI تفاوت های بسیار زیادی با آن چه که انتظار می رفت، داشتند. بدین ترتیب، نیاز سازگاری بین SMI/MIB و دو چارچوب SNMP و OSI به حالت تعلیق در آمد. این اتفاق، به چارچوب عملی مدیریت شبکه که SNMP بود، این مجوز را داد تا با تولید این مستند به نیازهای عملی جدید در جامعه اینترنت پاسخ دهد.  بنابراین، چارچوب فعلی مدیریت شبکه برای شبکه های اینترنتی مبتنی بر TCP/IP شامل این موارد است:  RFC 1155 با عنوان ساختار و مشخصات اطلاعات مدیریتی شبکه های اینترنتی مبتنی بر TCP/IP که به توصیف چگونگی تعریف مولفه های مدیرت شده در MIB می پردازد؛  پایگاه اطلاعات مدیریتی برای مدیریت شبکه های اینترنتی مبتنی بر TCP/IP: MIB II (همین سندی که در حال مطالعه آن هستید) که مولفه های موجود در MIB را توصیف می کند؛ (and supercedes RFC 1156)  و RFC 1098 با عنوان پروتکل ساده مدیریت شبکه(SNMP) که پروتکل برای مدیریت این مولفه ها را تعریف می کند. |
| **تغییرات نسبت به RFC 1156:**  خصوصیات این MIB شامل این موارد است:   * موارد اضافی با هدف بازتاب نیازمندی های جدید عملیاتی * سازگاری رو به افزایش با SMI/MIB و SNMP * پشتیبانی بهبودیافته برای موجودیت های چندپروتکلی * اصلاحات متنی در MIB با هدف بهبود وضوح و خوانایی   مولفه هایی که در MIB II تعریف شده اند با پیشوند OBJECT IDENTIFIER مشخص می شوند:  **mib-2 OBJECT IDENTIFIER ::= { mgmt 1 }**  که با پیشوندی که در MIB I به کار می رفت یکسان است. |
| **مولفه های منسوخ شده**  برای اینکه اجرا کنندگان برای تغییرات بعدی MIB بهتر آماده شوند، اصلاح جدید «منسوخ شده» ممکن است در هنگام تعریف یک مولفه به کار گرفته شود. یک شیء منسوخ شده در MIB چیزی است که باید پشتیبانی شود، ولی به احتمال زیاد در نسخه بعدی MIB حذف خواهد شد.(در اینجا مثلا MIB III)  MIB II یک شیء را به عنوان منسوخ شده نشان می دهد:  **atTable**  با منسوخ کردن شی atTable، تمام گروه ترجمه آدرس نیز منسوخ می شود.  شایان ذکر است که با منسوخ شدن اشیاءی که در ادامه می آید، هیچ کارکردی از بین نخواهد رفت: اشیاء جدیدی که یک شیء هم ارز یا با کارکردی فراتر از آن در MIB II تعریف شده است. |
| **Display Strings**  در گذشته، از MIB در شرایطی که یک رشته از Octetها باید شامل کاراکترهای قابل چاپ به این معنی که توسط انسان ها قابل درک باشد، سوء برداشت هایی می شد. به عنوان یک قرارداد متنی در MIB، نوع داده زیر معرفی می شود:  **DisplayString ::=**  **OCTET STRING**  یک DisplayString همان طور که در صفحات 10 و 11 همین RFC تعریف شده، به مجموعه کاراکترهای NVT ASCII محدود شده است. اشیاء زیر با همین روش تعریف شده اند:  **sysDescr** و **ifDescr**  این نکته هم باید ذکر شود که این تغییر تاثیری بر روی نحو و معنای این اشیاء ندارد. کاربرد نماد DisplayString صرفا یک شیوه توضیحی ساختگی است که در MIB II و MIB های بعدی استفاده می شود.  به علاوه، این مورد هم باید ذکر شود که هر شیئی که با OCTET STRING تعریف می شود می تواند شامل اطلاعات باینری دلخواه باشد که در آن مقادیر ممکن هر OCTET از 0 تا 255 است. |
| **آدرس های فیزیکی**  یک نمونه دیگر از قواعد متنی در MIB نوع داده زیر است:  **PhysAddress ::=**  **OCTET STRING**  که برای نشان دادن رسانه ها یا آدرس های سطح فیزیکی معرفی شده است.  اشیاء زیر با استفاده از PhysAddress تعریف شده اند:  **ifPhysAddress**  **atPhysAddress**  **ipNetToMediaPhysAddress**  این نکته هم باید ذکر شود که این تغییرات تاثیری بر روی نحو و معنای این اشیاء ندارد. کاربرد نماد PhysAddress صرفا یک شیوه توضیحی ساختگی است که در MIB II و MIB های بعدی استفاده می شود. |
| گروه سیستم  4 شیء جدید به این گروه اضافه شده است:  **sysContact**  **sysName**  **sysLocation**  **sysServices**  این ها با توجه به گره های مدیریت شده، اطلاعات تماس، اجرایی، موقعیت و خدمات را تامین می کنند. |
| گروه واسط ها  تعریف شیء ifNumber به این خاطر که برای پشتیبانی از IP به همه واسط ها نیاز داشت، اشتباه بود.(برای مثال، ادوات بدون IP مانند پل های لایه مک در صورت اجرای جدی این تعریف امکان مدیریت نداشتند) به همین دلیل تعریف ifNumber تغیر کرده است.  شیء ifTable به اشتباه به عنوان read-write به حساب می آمده که به not-accessible تغییر یافته و اصلاح شده است. به علاوه، چند مقدار جدید به ستون ifType در شیء ifTable اضافه شده است:  **ppp(23)**  **softwareLoopback(24)**  **eon(25)**  **ethernet-3Mbit(26)**  **nsip(27)**  **slip(28)**  **ultra(29)**  **ds3(30)**  **sip(31)**  **frame-relay(32)**  و در نهایت، یک ستون جدید به شیء ifTable اضافه شده است:  **ifSpecific**  که اطلاعات مرتبط با اطلاعات مختص رسانه ی استفاده شده برای تشخیص واسط را تهیه می کند. |
| **گروه ترجمه آدرس**  این گروه در MIB I شامل جدولی بود که اجازه می داد نگاشتی بین آدرس های شبکه ( مثلا آدرس IP ) به آدرس های فیزیکی(مثلا آدرس های مک) انجام پذیرد. تجربه نشان داده است که پیاده سازی و اجرای بهینه این جدول دو فرض دارد: 1) یک محیط منفرد پروتکل شبکه؛ 2) نگاشت فقط از آدرس شبکه به آدرس فیزیکی انجام می شود.  نیاز پشتیبانی از گره های چندپروتکلی (مثلا گره هایی که IP و CLNP آن ها فعال است) و نیاز به پشتیبانی نگاشت معکوس، هر دو فرض قبلی را باطل کرده اند. هم چنین شیء atTable هم منسوخ شده است.  برای این که هر دو نیاز چندپروتکلی و نگاشت معکوس را تامین کنیم، MIB II و نسخه های جانشین آن تا 2 جدول ترجمه ادرس در هر گروه پروتکل شبکه اختصاص خواهند داد. این که گروه IP شامل یک جدول ترجمه آدرس خواهد بود، برای رفتن از آدرس های IP به آدرس های فیزیکی است. به همین ترتیب، وقتی که سندی، اشیاء MIB را برای CLNP تولید می کند، شامل 2 جدول برای نگاشت در هر دو جهت خواهد بود که برای عملکرد همه جانبه لازم است.  این نکته باید ذکر شود که انتخاب دو جدول(برای هر جهت نگاشت یک عدد) سادگی پیاده سازی را در موارد متعددی به ارمغان می آورد و بار غیرضروری روی پیاده سازی ها که مفهوم ترجمه آدرس را از طریق یک جدول داخلی منفرد را می فهمد، ایجاد نمی کند. |
| **گروه IP**  خصیصه دسترسی ipForwarding متغیر، از read-only به read-write تغییر کرده است.  به علاوه ستون جدید زیر به شیء apAddrTable اضافه شده است:  apAdEntReasmMaxSize که بزرگترین دیتاگرام IP که می تواند روی یک واسط مشخص بازسازی شود را ردیابی می کند.  نام معرف شیء ipRoutingTable با هدف یکسان سازی با سایر اشیاء مسیریابی IP، به ipRouteTable تغییر یافته است. همچنین سه ستون جدید دیگر به شیء ipRouteTable اضافه شده اند:  **ipRouteMask**  **ipRouteMetric5**  **ipRouteInfo**  اولی برای مسیریابی زیرسیستم هایی که subnet mask های دلخواه را پشتیبانی می کنند به کار می رود و دو مورد بعدی پروتکل های خاص منظوره مسیریابی IP هستند.دو شیء جدید به گروه IP اضافه شده اند:  **ipNetToMediaTable**  **ipRoutingDiscards**  اولی جدول ترجمه آدرس برای گروه IP (تامین عملکرد یکسان برای atTable منسوخ شده در گروه ترجمه آدرس) و دومی اطلاعات را در شرایطی که مسیرها به دلیل فقدان فضای بافر از دست می روند، تامین می کند. |
| **گروه ICMP**  تغییراتی در این گروه وجود ندارد.  **گروه TCP**  دو متغیر جدید اضافه شده است:  **tcpInErrs**  **tcpOutRsts**  که این دو متغیر مسیرتعدادِ سگمنت های TCP ورودی با خطا و ریست های تولیدشده توسطTCP را نگهداری میکنند.  **گروه UDP**  جدول جدید udpTable اضافه شده است.  **گروه EGP**  تجربه نشان داده است که object های دیگری است که در نظارت (مانیتورینگ) EGP مفید واقع شوند مورد نیاز هستند.. علاوه بر ایجاد چندین مورد اضافی در مورد egpNeighborTable :  egpNeighAs  egpNeighInMsgs  egpNeighInErrs  egpNeighOutMsgs  egpNeighOutErrs  egpNeighInErrMsgs  egpNeighOutErrMsgs  egpNeighStateUps  egpNeighStateDowns  egpNeighIntervalHello  egpNeighIntervalPoll  egpNeighMode  egpNeighEventTrigger  یک متغیر جدید egpAs اضافه شده که یک سیستم خودمختار مرتبط با این موجودیت EGP را ارائه می دهد.  **گروه Transmission**  MIB-I این کمبود را داشت که تفاوتی میان انواع مختلف رسانه های انتقال قائل نمی شد. یک گروه جدید به نام گروه انتقال برای این هدف اختصاص یافته است:  **transmission OBJECT IDENTIFIER ::= { mib-2 10 }**  هنگامی که تعاریف استاندارد اینترنت برای مدیریت رسانه انتقال تعریف می شوند، گروه انتقال برای فراهم آوری پیشوند برای نام های آن اشیا مورد استفاده قرار می گیرد.  به طور معمولی چنین تعاریفی در بخش تجربی MIB قرار می گیرند تا زمانی که " به اثبات برسند"، سپس به عنوان بخشی از فرآیند استاندارد سازی اینترنت باز هم مورد ارزیابی قرار می گیرند و یک شناسه شی جدید تحت گروه انتقال تعریف می شود. طبق قرارداد، نام اختصاص یافته برابر است با:  **type OBJECT IDENTIFIER ::= { transmission number }**  که در آن "نوع" مقدار نمادین مورد استفاده برای رسانه در ستون ifType شی ifTable است و "عدد" مقدار صحیح (اینتجر) واقعی مربوط به نماد است.  **گروه SNMP**  به کارگروه های مبتنی برکاربرد IETF وظیفه تعریف متغیرهای MIB مختص به کاربرد های مربوطه واگذار شده است.  برای SNMP داشتن اطلاعات آماری سودمند است. گروه جدیدی به نام گروه SNMP به این منظور اختصاص یافته است.  **snmp OBJECT IDENTIFIER ::= { mib-2 11 }**  **تغییرات نسبت به RFC 1158 :**  ویژگی های این MIB شاملِ:  1. اشیای مدیریت شده در این مستند با استفاده از قرارداد های تعریف شده در SMI اینترنت استاندارد مطابق با اصلاحات انجام گرفته توسط افزونه های مخصوص در [14] تعریف شده است. بایستی تاکید کرد که تعاریف ایجاد شده توسط این افزونه ها از نظر مفهومی مشابه با موارد موجود در RFC 1158 است.  2. قرارداد متنی PhysAddress برای نمایش آدرس های رسانه معرفی شده است.  3. بخشِACCESS از sysLocation در حال حاضر خواندنی- نوشتنی است.  4. تعریف sysServices تصریح شده است.  5. مقادیر جدید ifType (29-32) تعریف شده است. علاوه بر این توصیف کننده متنی برای انواع رابط های DS1 و E1 تصحیح شده است.  6. تعریف ipForwarding تصریح شده است.  7. تعریف ipRouteType تصریح شده است.  8. اشیای ipRouteMetric5 و ipRouteInfo تعریف شده اند.  9. بخشِ ACCESS از tcpConnState در حال حاضر خواندنی- نوشتنی است تا از حذف TCB مرتبط با اتصال TCP پشتیبانی کند. تعریف این شی برای توضیح این کاربرد تصریح شده است.  10. تعریف egpNeighEventTrigger تصریح شده است.  11.تعریف متغیر های متعدد در گروه جدید snmp تصریح شده است. علاوه بر این اشیای snmpInBadTypes و snmpOutReadOnlys در حال حاضر وجود ندارند(با این حال شناسه های تخصیص یافته به این شی برای جلوگیری از استفاده ی آتی نگهداری میشوند)  12. تعریف snmpInReadOnlys تصریح شدده است.  13. توصیف کننده متنی snmpEnableAuthTraps به snmpEnableAuthenTraps تغییر پیدا کرده است و تصریح شده است.  14. شی ipRoutingDiscards اضافه شده بوده است.  15. استفاده انتخابی از عدد صحیح مثبت و کوچک و مستقل از پیاده سازی در زمان تعریف آدرس IP و جداول مسیریابی اجازه داده نشده است.  **اشیا**  اشیای مدیریت شده از طریق یک منبع اطلاعات مجازی که پایگاه اطلاعات مدیریت[[1]](#footnote-1) (MIB) نامیده می شود، قابل دسترسی هستند.اشیای موجود در MIB با استفاده از زیر مجموعه ی ASN.1 تعریف شده در SMI ، تعریف میشوند.به طور خاص هر شیء یک نام،یک ساختار و یک رمزگذاری دارد.یک نام شناسه شیء است، نامی که به صورت مدیریتی اختصاص یافته،که نوع شی را مشخص می کند. نوع شی همراه با یک نمونه شی به طور یکتا نمونه سازی مخصوص به یک شی را شناسایی می کند.برای راحتی انسان ،ما اغلب از یک رشته متنی که OBJECT DESCRIPTOR نام دارد،و همچنین به نوع شیءاشاره دارد استفاده میکنیم.  سینتکس یک نوع شی ساختار داده انتزاعی را متناسب با آن نوع شی تعریف می کند. زبان ANS.1 به این منظور مورد استفاده قرار می گیرد. اگر چه SMI [12] به صورت هدفمند ساختار های ANS.1 را که ممکن استفاده شوند محدود می کند. این محدودیت ها صریحا برای سادگی ایجاد می شوند.  رمزگذاری(Encoding) نوع شیء صرفاً نحوه نمایش آن نوع شی با استفاده از ساختار نوع شی است.نحوه نمایش نوع شی هنگامی که در شبکه انتقال داده می شود به طور ضمنی با مفهوم ساختار نوع شی و رمزگشایی گره خورده است.  SMI استفاده از قوانین پایه ای انکدینگ مربوط به ANS.1 را مشخص می کند، مشروط بر نیازمندی های اضافی وضع شده توسطSNMP.  **فرمت تعاریف**  بخش 6 شامل مشخصات تمامی انواع شیء موجود در این ماژول MIB است. انواع شیء با استفاده از قرارداد های تعریف شده در SMI ،مطابق با اصلاحات مشخص در تعریف می شوند.  **بررسی اجمالی**  مطابق با بخشنامه IAB برای تولید سیستم های ساده و کارا در مدت زمان کوتاه، فهرستی از اشیای تحت مدیریت که در اینجا تعریف شده است، تنها از عناصر ضروری گرفته شده است.  این روش دربرگیری اشیای ضروری محدود کننده نیست چرا که SMI تعریف شده سه مکانیزم توسعه ای را فراهم می کند:اول افزودن اشیای استاندارد جدید از طریق تعریف نسخه های جدید MIB ، دوم تعریف به طور گسترده در دسترس اما غیر استاندارد اشیا از طریق زیردرخت تجربی و سوم افزودن اشیای خصوصی از طریق زیردرخت ها.  چنین اشیای اضافی نه تنها برای عناصر مخصوص به سازنده مورد استفاده قرار می گیرد، بلکه همچنین برای آزمایش به عنوان نیازمندی برای دانش بیشتر با سایر اشیا ضروری است.  طراحی MIB-II به شدت تحت تاثیر اولین مکانیزم توسعه پذیری قرار گرفته است . چندین متغیر جدید بر مبنای آزمایش عملیاتی و نیاز افزوده شده است. بر این مبنا معیار دربرگیری یک شی در MIB-II به طور قابل توجهی با معیار MIB-I شباهت دارد:  1. یک شی برای ضروری بودن به عنوان مدیریت خطا یا پیکربندی مورد نیاز است.  2. فقط اشیای کنترلی ضعیف اجازه داده شده است (منظور از ضعیف این است که دستکاری آن ها تنها صدمات کمی را به بار می آورد). این معیار این حقیقت را منعکس می کند که پروتکل های فعلی مدیریت به اندازه کافی امن نیستند تا بتوانند اقدامات کنترلی بسیار قدرتمندی را انجام دهند.  3. شواهد استفاده فعلی و کاربرد مورد نیاز بوده است.  4. در MIB-I تلاشی برای محدود سازی تعداد اشیا به حدود 100 عدد اجرا شد تا سازندگان ساده تر بتوانند نرم افزار های خودشان را به طور کامل مورد استفاده قرار بدهند. در MIB-II این محدودیت افزایش یافت.  5. برای اجتناب از متغیر های اضافی نیاز بود که هر شی ای که می تواند از سایر اشیا در MIB گرفته شود، دربر گرفته نشود.  6. پیاده سازی اشیای ویژه (به عنوان مثال برای BSD UNIX) مستثنی شده است.  7. موافقت شده بود تا از پیاده سازی شدید ؟ بخش های کد جلوگیری شود. راهنمایی عمومی یک شمارنده به ازای هر بخش حیاتی از لایه بود.  MIB-II همانند نسخه قبلی یعنی MIB اینترنت-استاندارد، فقط شامل عناصر ضروری است. نیازی نیست که اشیای منحصر به فرد اختیاری باشند. در مقابل، این اشیا به گروه های ذیل سازمان دهی می شود:   * سیستم * رابط ها * تبدیل آدرس (منسوخ شده) * IP * ICMP * TCP * UDP * EGP * انتقال * SNMP   این گروه ها واحد اصلی انطباق است: این روش به این صورت است که: اگر مفاهیم یک گروه به یک پیاده سازی قابل اعمال باشد، آنگاه می بایست تمامی اشیا در این گروه را پیاده سازی نماید. به عنوان مثال یک پیاده سازی می بایست گروه EGP را پیاده سازی کند اگر و تنها اگر EGP را پیاده سازی می کند.  دو دلیل برای تعریف این گروه ها وجود دارد: فراهم کردن وسیله ای برای تخصیص شناسه های شی و ارائه روشی برای پیاده سازی کارگزار های مدیریتی برای اطلاع از این موضوع که کدام شی بایستی پیاده سازی شود  **تعاریف**  **RFC1213-MIB DEFINITIONS ::= BEGIN**  **IMPORTS**  **mgmt, NetworkAddress, IpAddress, Counter, Gauge,**  **TimeTicks**  **FROM RFC1155-SMI OBJECT-TYPE**  **FROM RFC-1212;**  -- این ماژول MIB از ماکرو OBJECT-TYPE تعمیم یافته استفاده می کند:  -- MIB-II  **mib-2 OBJECT IDENTIFIER ::= { mgmt 1 }**  **--** قراردادهای متنی  **DisplayString ::=**  **OCTET STRING**  **--** این نوع داده جهت مدل کردن اطلاعات متنی گرفته شده از مجموعه کاراکترهای NVT ASCII به کار رفته است. طبق قرارداد اشیا دارای چنین syntax ای دارای سایز ( SIZE (0 ..255)) هستند.  گروه های MIB-II :  **system OBJECT IDENTIFIER ::= { mib-2 1 }**  **interfaces OBJECT IDENTIFIER ::= { mib-2 2 }**  **at OBJECT IDENTIFIER ::= { mib-2 3 }**  **ip OBJECT IDENTIFIER ::= { mib-2 4 }**  **icmp OBJECT IDENTIFIER ::= { mib-2 5 }**  **tcp OBJECT IDENTIFIER ::= { mib-2 6 }**  **udp OBJECT IDENTIFIER ::= { mib-2 7 }**  **egp OBJECT IDENTIFIER ::= { mib-2 8 }**  **-- cmot OBJECT IDENTIFIER ::= { mib-2 9 }**  **transmission OBJECT IDENTIFIER ::= { mib-2 10 }**  **snmp OBJECT IDENTIFIER ::= { mib-2 11 }**  **گروه سیستم :**  -- پیاده سازی این گروه برای تمام سیستم ها اجباری است. اگر در پیکره بندی این عامل[[2]](#footnote-2) هیچیک از این متغییر ها مقدار دهی نشده باشند ، یک رشته با طول 0 برگردانده می شود.    **sysDescr OBJECT-TYPE**  **SYNTAX DisplayString (SIZE (0..255))**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  توصیف متنی موجودیت. این مقدار باید شامل نام کامل و شناسه نسخه نوع سخت افزار سیستم، سیستم عامل نرم افزاری و نرم افزار شبکه باشد. این مقدار فقط باید شامل کاراکترهای قابل چاپ ASCII باشد.  **::= { system 1 }**    **sysObjectID OBJECT-TYPE**  **SYNTAX OBJECT IDENTIFIER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  شناسه معتبر فروشنده متعلق به زیر سیستم مدیریت شبکه در موجودیت قرار دارد. این مقدار در زیردرخت سرمایه گذاری های SMI(1.3.6.1.4.1) قرار گرفته و یک راه ساده و غیرمبهم برای تعیین این که چه نوع جعبه ای در حال مدیریت است، ارائه می دهد. به طور مثال اگر شرکت حقوقی Flintstone توسط زیردرخت 1.3.6.1.4.1.4242.1.1 مشخص شود، می تواند شناسه 1.3.6.1.4.1.4242.1.1 را به روتر فِرِد خود اختصاص دهد.  **::= { system 2 }**    **sysUpTime OBJECT-TYPE**  **SYNTAX TimeTicks**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مدت زمان(یک صدم ثانیه) از آخرین باری که وقتی که قطعه مدیریت شبکه سیستم دوباره مقداردهی شده است.  **::= { system 3 }**    **sysContact OBJECT-TYPE**  **SYNTAX DisplayString (SIZE (0..255))**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  شناسه متنی شخص مخاطب برای این گره مدیریت شده به همراه اطلاعات نحوه ارتباط با این شخص.  **::= { system 4 }**    **sysName OBJECT-TYPE**  **SYNTAX DisplayString (SIZE (0..255))**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  نام اختصاص یافته اداری این گره مدیریت شد. طبق قرارداد، این عبارت، نام کاملا واجد شرایط دامنه است.  **::= { system 5 }**    **sysLocation OBJECT-TYPE**  **SYNTAX DisplayString (SIZE (0..255))**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  موقعیت فیزیکی این گره(به طور مثال: طبقه سوم، صندوق تلفن).  **::= { system 6 }**    **sysServices OBJECT-TYPE**  **SYNTAX INTEGER (0..127)**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مقداری نشان دهنده مجموعه سرویس هایی این موجودیت در درجه اول پیشنهاد می‌دهد.  مقدار ذکر شده یک جمع است. این جمع در ابتدا مقدار 0 میگیرد. سپس برای هر لایه L در محدوده 1 تا 7 که این گره برایش تراکنش انجام می دهد، 2 تا L-1 به مقدارش اضافه می شود. به طور مثال مقدار گره ای که عملکردهای ابتدایی مسیریابی را انجام می دهد، 4 خواهد بود. 2^)3-1( در مقابل، گره ای که خدمات کاربرد پیشنهاد میزبان است مقدار 72 را خواهد داشت. (2^(4-1) + 2^(7-1)  به این نکته باید توجه داشت که در حوزه مجموعه پروتکل های اینترنت، مقادیر باید بر اساس موارد زیر محاسبه شوند:  عملکرد لایه:  1 فیزیکی (مثال: تکرارکننده ها)  2 پیوند داده/ریز شبکه (مثال: پل ها)  3 اینترنت (مثلا دروازه های IP)  4 پایان به پایان (مثال: میزبانی های IP)  7 کاربردها (مثال: رله های ایمیل)  برای سیستم هایی که شامل پروتکل های OSI هستند، ممکن است لایه های 5 و 6 هم به حساب بیاید.  **::= { system 7 }**    **گروه واسط ها :**  -- پیاده سازی این گروه برای تمام سیستم ها اجباری است.  **ifNumber OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد واسط های شبکه(فارغ از وضعیت کنونی) در این سیستم حاضر می شوند.  **::= { interfaces 1 }**    -- جدول واسط ها  -- جدول واسط ها حاوی اطلاعاتی در خصوص واسط های موجودیت است. فرض شده هریک از واسط ها به subnetwork متصل هستند. بایستی توجه گردد این عبارت با subnet مطرح شده در بحث آدرس دهی در پروتکل IP متفاوت است.  **ifTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF IfEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  یک لیست از ورودی های واسط ها. تعداد ورودی ها با مقدار ifNumber داده می شود.  **::= { interfaces 2 }**    **ifEntry OBJECT-TYPE**  **SYNTAX IfEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  یک ورودی واسط، شامل اشیاء لایه زیرشبکه و پایین تر از آن، برای یک واسط مشخص.  **INDEX { ifIndex }**  **::= { ifTable 1 }**    **IfEntry ::=**  **SEQUENCE {**  **ifIndex**  **INTEGER,**  **ifDescr**  **DisplayString,**  **ifType**  **INTEGER,**  **ifMtu**  **INTEGER,**  **ifSpeed**  **Gauge,**  **ifPhysAddress**  **PhysAddress,**  **ifAdminStatus**  **INTEGER,**  **ifOperStatus**  **INTEGER,**  **ifLastChange**  **TimeTicks,**  **ifInOctets**  **Counter,**  **ifInUcastPkts**  **Counter,**  **ifInNUcastPkts**  **Counter,**  **ifInDiscards**  **Counter,**  **ifInErrors**  **Counter,**  **ifInUnknownProtos**  **Counter,**  **ifOutOctets**  **Counter,**  **ifOutUcastPkts**  **Counter,**  **ifOutNUcastPkts**  **Counter,**  **ifOutDiscards**  **Counter,**  **ifOutErrors**  **Counter,**  **ifOutQLen**  **Gauge,**  **ifSpecific**  **OBJECT IDENTIFIER**  **}**  **ifIndex OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مقداری یکتا برای هر واسط که با عددی بین 1 و ifNumber مقداردهی می شود. مقدار هر واسط باید حداقل بین یک بازمقداردهی اولیه تا بازمقداردهی اولیه بعدی ثابت بماند.  **::= { ifEntry 1 }**  **ifDescr OBJECT-TYPE**  **SYNTAX DisplayString (SIZE (0..255))**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  یک رشته متنی شامل اطلاعاتی درباره واسط. این رشته باید شامل نام تولیدکننده، نام محصول و نسخه واسط سخت افزار باشد.  **::= { ifEntry 2 }**  **ifType OBJECT-TYPE**  **SYNTAX INTEGER {**  **other(1),** --هیچ بک از موارد زیر  **regular1822(2),**  **hdh1822(3),**  **ddn-x25(4),**  **rfc877-x25(5),**  **ethernet-csmacd(6),**  **iso88023-csmacd(7),**  **iso88024-tokenBus(8),**  **iso88025-tokenRing(9),**  **iso88026-man(10),**  **starLan(11),**  **proteon-10Mbit(12),**  **proteon-80Mbit(13),**  **hyperchannel(14),**  **fddi(15),**  **lapb(16),**  **sdlc(17),**  **ds1(18), -- T-1**  **e1(19), -- european equiv. of T-1**  **basicISDN(20),**  **primaryISDN(21),** -- سریال اختصاصی  **propPointToPointSerial(22),**  **ppp(23),**  **softwareLoopback(24),**  **eon(25), -- CLNP over IP [11]**  **ethernet-3Mbit(26),**  **nsip(27), -- XNS over IP**  **slip(28), -- generic SLIP**  **ultra(29), -- ULTRA technologies**  **ds3(30), -- T-3**  **sip(31), -- SMDS**  **frame-relay(32)**  **}**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  نوع واسط، مطابق پروتکل(های) فیزیکی/پیوند در پشته پروتکلی بلافاصله زیر لایه شبکه قرار می گیرد و از این طریق متمایز می گردد.  **::= { ifEntry 3 }**  **ifMtu OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  اندازه بزرگترین دیتاگرام ارسالی/دریافتی روی واسط که با اکتت(هشت تایی) مشخص می شود. برای واسط هایی که به جهت انتقال دیتاگرام های شبکه استفاده می شوند، این مقدار نشان دهنده بزرگترین دیتاگرامی است که می توان بر روی واسط ارسال کرد.  **::= { ifEntry 4 }**  **ifSpeed OBJECT-TYPE**  **SYNTAX Gauge**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تخمینی از پهنای باند فعلی واسط در واحد بیت بر ثانیه. برای واسط هایی که پهنای باند ثابت دارند و یا آن هایی که نمی توان تخمین دقیقی به دست آورد از مقدار اسمی استفاده خواهیم کرد.  **::= { ifEntry 5 }**  **ifPhysAddress OBJECT-TYPE**  **SYNTAX PhysAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس واسط در پشته پروتکلی که بلافاصله در زیر لایه شبکه قرار دارد. برای واسط هایی که آدرسی ندارند(مثلا یک خط سریال) این شیء باید یک رشته هشت تایی با طول 0 بگیرد.  **::= { ifEntry 6 }**  **ifAdminStatus OBJECT-TYPE**  **SYNTAX INTEGER {**  **up(1),**  **down(2),**  **testing(3)**  **}**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  وضعیت مورد نظر واسط. وضعیت testing(3) نشان دهنده این است که هیچ بسته عملیاتی نمی تواند عبور کند.  **::= { ifEntry 7 }**  **ifOperStatus OBJECT-TYPE**  **SYNTAX INTEGER {**  **up(1), -- ready to pass packets**  **down(2),**  **testing(3) -- in some test mode**  **}**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  وضعیت مورد نظر واسط. وضعیت testing(3) نشان دهنده این است که هیچ بسته عملیاتی نمی تواند عبور کند.  **::= { ifEntry 8 }**  **ifLastChange OBJECT-TYPE**  **SYNTAX TimeTicks**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مقدار sysUpTime در زمانی که واسط وارد وضعیت عملیاتی حال حاضرش شده است. اکر وضعیت فعلی قبل از آخرین باز مقداردهی اولیه زیر سیستم محلی مدیریت شبکه وارد شده باشد، این شیء باید مقدار 0 بگیرد.  **::= { ifEntry 9 }**    **ifInOctets OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  همه هشت تایی هایی که از واسط گرفته شده، از جمله کاراکترهای کادربندی.  **::= { ifEntry 10 }**  **ifInUcastPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های subnetwork-unicastی که به یک لایه بالاتر تحویل داده شده است.  **::= { ifEntry 11 }**  **ifInNUcastPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های غیر unicast (یعنی subnetwork-broadcast یا subnetwork-multicast) که به یک لایه بالاتر تحویل داده شده است..  **::= { ifEntry 12 }**  **ifInDiscards OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های ورودی شامل خطاهایی که مانع رسیدن آن ها به پروتکلی از لایه بالاتر می شوند.  **::= { ifEntry 13 }**  **ifInErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های رسیده از طریق واسط که به خاطر یک پروتکل ناشناخته یا خارج از پشتیبانی دور انداخته شده اند.  **::= { ifEntry 14 }**  **ifInUnknownProtos OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های رسیده از طریق واسط که به خاطر یک پروتکل ناشناخته یا خارج از پشتیبانی دور انداخته شده اند.  **::= { ifEntry 15 }**  **ifOutOctets OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه هشت تایی ها با احتساب کاراکترهای کادربندی که از واسط خارج شده اند.  **::= { ifEntry 16 }**  **ifOutUcastPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه بسته ها\_از جمله آن هایی که دور انداخته شدند یا ارسال نشدند\_که توسط لایه های بالاتر برای انتقال به آدرس subnetwork-unicast درخواست شدند.  **::= { ifEntry 17 }**  **ifOutNUcastPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه بسته ها\_از جمله آن هایی که دور انداخته شدند یا ارسال نشدند\_که توسط لایه های بالاتر برای انتقال به یک آدرس غیر unicast(یعنی یک subnetwork-broadcast یا subnetwork-multicast) درخواست شدند.  **::= { ifEntry 18 }**  **ifOutDiscards OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های خروجی که برای دور انداختن انتخاب شده اند، حتی اگر هیچ خطایی برای جلوگیری از ارسال آن ها شناسایی نشده باشد. یک دلیل ممکن برای دور انداختن بسته می تواند خالی کردن فضای بافر باشد.  **::= { ifEntry 19 }**  **ifOutErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد بسته های خروجی که به علت خطاها نمی توانند ارسال شوند.  **::= { ifEntry 20 }**  **ifOutQLen OBJECT-TYPE**  **SYNTAX Gauge**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  طول صف خروجی بسته(در بسته ها)  **::= { ifEntry 21 }**  **ifSpecific OBJECT-TYPE**  **SYNTAX OBJECT IDENTIFIER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  ارجاعی به تعاریف MIB به طور خاص درباره رسانه های مشخصی که در حال استفاده برای درک واسط هستند. به طور مثال اگر واسط توسط یک اترنت فهمیده شده باشد، آن گاه مقدار این شیء به سندی که به طور خاص برای اترنت اشیاء را تعریف می کند، ارجاع می دهد. اگر این اطلاعات موجود نیست، مقدارش باید OBJECT IDENTIFIER {0 0} تنظیم شود که از نظر نحوی یک object identifier صحیح است و هر پیاده سازی مطابق ASN.1 و BER باید بتواند این مقدار را تولید کند و بشناسد.  **::= { ifEntry 22 }**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **گروه ترجمه آدرس ها :**  **atTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF AtEntry**  **ACCESS not-accessible**  **STATUS deprecated**  **DESCRIPTION**  جداول ترجمه آدرس، شامل معادل های آدرس شبکه به آدرس فیزیکی است. بعضی واسط ها از جداول ترجمه برای تعیین معادل های آدرس استفاده نمی کنند(مثلا DDN-X.25 یک روش الگوریتمی دارد)؛ اگر همه واسط ها از این نوع باشند، آن گاه جدول ترجمه آدرس خالی است، یعنی هیچ مدخلی ندارد.  **::= { at 1 }**  **atEntry OBJECT-TYPE**  **SYNTAX AtEntry**  **ACCESS not-accessible**  **STATUS deprecated**  **DESCRIPTION**  هر ورودی شامل یک هم ارزی آدرس شبکه به آدرس فیزیکی است.  **INDEX { atIfIndex,**  **atNetAddress }**  **::= { atTable 1 }**  **AtEntry ::=**  **SEQUENCE {**  **atIfIndex**  **INTEGER,**  **atPhysAddress**  **PhysAddress,**  **atNetAddress**  **NetworkAddress**  **}**  **atIfIndex OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS deprecated**  **DESCRIPTION**  واسطی که در آن هم ارز این مدخل موثر است. واسطی که از طریق مقدار مشخص این اندیس شناخته شده با واسطی که از طریق همان مقدار ifIndex شناخته شده است، یکسان است.  **::= { atEntry 1 }**  **atPhysAddress OBJECT-TYPE**  **SYNTAX PhysAddress**  **ACCESS read-write**  **STATUS deprecated**  **DESCRIPTION**  آدرس فیزیکی وابسته به رسانه. تنظیم این شیء به یک رشته خالی(با طول صفر) تاثیری همچون باطل کردن مدخل متناظر در شیء atTable دارد. به این معنی که به طور موثر واسطی که با مدخل قید شده، مشخص شده است را از نگاشت مشخص شده با مدخل اشاره شده تفکیک می کند. این یک موضوع اجرای خاص که جایی که مامور یک مدخل باطل شده را از جدول پاک می کند. بدین ترتیب ایستگاه های مدیریت باید برای دریافت اطلاعات جدولی، از عامل هایی که مطابق با مدخل های آزاد هستند آماده شوند. تفسیر مناسب چنین ورودی هایی نیازمند امتحان کردن شیء atPhysAddress مرتبط است.  **::= { atEntry 2 }**  **atNetAddress OBJECT-TYPE**  **SYNTAX NetworkAddress**  **ACCESS read-write**  **STATUS deprecated**  **DESCRIPTION**  آدرس شبکه (مثلا آدرس IP) مطابق با آدرس فیزیکی وابسته به رسانه.  **::= { atEntry 3 }**  **گروه IP :**  **ipForwarding OBJECT-TYPE**  **SYNTAX INTEGER {**  **forwarding(1), -- acting as a gateway**  **not-forwarding(2) -- NOT acting as a gateway**  **}**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  نشانه این که این مدخل به عنوان یک دروازه IP در رابطه با هدایت دیتاگرام هایی که این مدخل دریافت کرده و به آن اشاره نشده رفتار می کند. دروازه های IP دیتاگرام ها را هدایت می کنند اما میزبان ها به جز آن هایی که از طریق میزبان مسیرشان از مبدأ مشخص است، این طور نیستند.  توجه داشته باشید که برای بعضی از گره های مدیریت شده، این شیء ممکن است فقط بخشی از مقادیر ممکن را بگیرد. بنابراین برای یک عامل خوب است که اگر یک ایستگاه مدیریت اقدام به تغییر این شیء به مقداری نامناسب کرد، یک پیغام badValue بازگرداند.  **::= { ip 1 }**  **ipDefaultTTL OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  مقدار پیش فرض درج شده در بخش Time-To-Live سرآیند IP دیتاگرام هایی که مبدأشان این مدخل است؛ در شرایطی که مقدار TTL توسط پروتکل لایه انتقال ایجاد نشده است.  **::= { ip 2 }**  **ipInReceives OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  همه مقادیر دیتاگرام های ورودی دریافت شده از واسط ها، از جمله آن هایی که با خطا دریافت شده اند.  **::= { ip 3 }**  **ipInHdrErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های ورودی که به خاطر خطاهای سرآیندهای IPشان از جمله جمع کنترلی خراب، عدم تطابق شماره نسخه، سایر خطاهای مربوط به قالب، تجاوز از مقدار TTL، خطاهایی که در پردازش گزینه های IP کشف می شود و ... دورانداخته شده است.  **::= { ip 4 }**  **ipInAddrErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های ورودی که بخاطر اینکه آدرس IP مقصد در سرآیند IPشان، مقدار صحیحی که توسط این مدخل دریافت شود، نبوده، دور انداخته شده است. این شماره شامل آدرس های نامعتبر (مثلا 0.0.0.0) و آدرس های کلاس های پشتیبانی نشده(مثل کلاس E) است. برای مدخل هایی که دروازه IP نیستند و در نتیجه آن دیتاگرام ها را هدایت نمی کنند، این شمارنده به این دلیل که آدرس مقصد یک آدرس محلی نبوده است، شامل دیتاگرام های دورانداخته شده است.  **::= { ip 5 }**  **ipForwDatagrams OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های ورودی برای موردی که این موجودیت آخرین مقصد IPاش نبوده، به این خاطر که یک تلاش با هدف پیدا کردن مسیری برای هدایت آن ها به مقصد نهایی انجام شده است. در موجودیت هایی که به عنوان دروازه های IP عمل نمی کنند، این شمارنده فقط شامل بسته هایی خواهد بود که به واسطه این موجودیت Source-Routed بودند و پردازش امکان Source-Route موفق بوده است.  **::= { ip 6 }**  **ipInUnknownProtos OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های که به صورت محلی آدرس دهی شده اند و به صورت موفق دریافت شده اند ولی به دلیل پروتکل ناشناخته یا پشتیبانی نشده دور انداخته شده اند.  **::= { ip 7 }**  **ipInDiscards OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های IP ورودی که هیچ مشکلی که مانع پردازش مستمر آن ها شود رخ نداده است اما دور انداخته شده اند. (مثلا به دلیل فقدان فضای بافر)  توجه داشته باشید که این شمارنده شامل دیتاگرام هایی که منتظر re-assembly بودند و دور انداخته شدند نیست.  **::= { ip 8 }**  **ipInDelivers OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه دیتاگرام های ورودی که به صورت موفقیت آمیز به پروتکل های کاربری IP تحویل داده شده اند.(شامل ICMP)  **::= { ip 9 }**  **ipOutRequests OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه دیتاگرام های IP که پروتکل های کاربری IP محلی (شامل ICMP) به خاطر انتقال به IP ارائه می شود.  **::= { ip 10 }**  **ipOutDiscards OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه دیتاگرام های IP خروجی که هیچ مشکلی که مانع انتقال‌شان به مقصد شود رخ نداد اما دور انداخته شده اند (مثلا به خاطر نبود فضای بافر). توجه داشته باشید که این شمارنده در بردارنده دیتاگرام هایی خواهد بود که در ipForwDatagrams شمرده شده؛ اگر چنین بسته هایی شامل این ملاک دور ریختن شوند.  **::= { ip 11 }**  **ipOutNoRoutes OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه دیتاگرام های IP که به دلیل پیدا نشدن هیچ مسیری برای انتقالشان به مقصد دور انداخته شده اند. توجه داشته باشید که این شمارنده شامل هر بسته ای که در ipForwDatagrams وجود دارد و شامل no-route می شود است. ذکر این نکته لازم است که این مورد شامل هر دیتاگرامی که یک میزبان به دلیل از دسترس خارج بودن همه دروازه های پیشفرض آن، نمی تواند به آن مسیری داشته باشد است.  **::= { ip 12 }**  **ipReasmTimeout OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حداکثر ثانیه هایی که قطعه های دریافت شده برای سر هم بندی مجدد در این موجودیت منتظر هستند، نگه داشته می شوند.  **::= { ip 13 }**  **ipReasmReqds OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد قطعه های IP دریافت شده که در این موجودیت باید دوباره سر هم بندی شوند.  **::= { ip 14 }**  **ipReasmOKs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های IP که به طور موفقیت آمیز مجدداً سر هم بندی شدند.  **::= { ip 15 }**  **ipReasmFails OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد خرابی هایی که توسط الگوریتم سر هم بندی مجدد تشخیص داده است (به هر دلیلی از جمله اتمام فرصت، خطاها و ...). توجه داشته باشید که تعداد قطعه های IP دور ریخته شده لازم نیست، چون بعضی الگوریتم ها (به خصوص الگوریتم RFC 815) ممکن است روند شمارش قطعه ها را به علت قاطی شدن با این عنوان که دریافت شده اند، از دست بدهد.  **::= { ip 16 }**  **ipFragOKs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های IP که در این موجودیت به صورت موفقیت آمیز قطعه بندی شده اند.  **::= { ip 17 }**  **ipFragFails OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های IP که به دلیل اینکه می خواستند در این موجودیت، تکه تکه شوند ولی نتوانستند (مثلا به این دلیل پرچم «قطعه قطعه نکن» آن ها 1 نشده بود)، دور ریخته شده اند.  **::= { ip 18 }**  **ipFragCreates OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد قطعه های دیتاگرام ها IP که در نتیجه قطعه بندی شدن در این موجودیت تولید شده اند.  **::= { ip 19 }**  -- جدول آدرس IP  **ipAddrTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF IpAddrEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدول اطلاعات آدرس دهی مربط با آدرس های IP این موجودیت.  **::= { ip 20 }**  **ipAddrEntry OBJECT-TYPE**  **SYNTAX IpAddrEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  اطلاعات آدرس دهی یکی از آدرس های IP این موجودیت.  **INDEX { ipAdEntAddr }**  **::= { ipAddrTable 1 }**  **IpAddrEntry ::=**  **SEQUENCE {**  **ipAdEntAddr**  **IpAddress,**  **ipAdEntIfIndex**  **INTEGER,**  **ipAdEntNetMask**  **IpAddress,**  **ipAdEntBcastAddr**  **INTEGER,**  **ipAdEntReasmMaxSize**  **INTEGER (0..65535)**  **}**  **ipAdEntAddr OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP که به اطلاعات آدرس دهی این موجودیت مربوط می شود.  **::= { ipAddrEntry 1 }**  **ipAdEntIfIndex OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مقدار اندیسی که به طور اختصاصی واسطی که به این مدخل قابل اطلاق است را شناسایی می کند. واسطی که به وسیله یک مقدار خاص این اندیس شناسایی شده است، همانند واسطی است که به وسیله مقدار یکسان ifIndex شناخته شده است.  **::= { ipAddrEntry 2 }**  **ipAdEntNetMask OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  ماسک زیرشبکه مرتبط با آدرس IP این موجودیت. مقدار ماسک، یک آدرس IP است که همه بیت های بخش شبکه برابر با 1 و همه بیت های بخش میزبان برابر با 0 است.  **::= { ipAddrEntry 3 }**  **ipAdEntBcastAddr OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مقدار کم اهمیت ترین بیت در آدرس همه پخشی IP که برای ارسال دیتاگرام ها بر روی واسط(منطقی) مرتبط با آدرس IP این مدخل استفاده شده است. به طور مثال وقتی که آدرس همه پخشی استاندارد اینترنت که همه بیت های آن 1 است استفاده شده، مقدار 1 خواهد بود. این مقدار به هر دو آدرس همه پخشی شبکه و زیرشبکه استفاده شده توسط موجودیت این واسط (منطقی) اعمال می شود.  **::= { ipAddrEntry 4 }**  **ipAdEntReasmMaxSize OBJECT-TYPE**  **SYNTAX INTEGER (0..65535)**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  اندازه بزرگترین دیتاگرام IP که این موجودیت می تواند از دیتاگرام های قطعه بندی شده ای که روی این واسط دریافت شده، دوباره سر هم بندی کند.  **::= { ipAddrEntry 5 }**  -- جدول مسیریابی IP  **ipRouteTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF IpRouteEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدول مسیریابی IP این موجودیت.  **::= { ip 21 }**  **ipRouteEntry** **OBJECT-TYPE**  **SYNTAX IpRouteEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  یک مسیر به مقصد مشخص.  **INDEX { ipRouteDest }**  **::= { ipRouteTable 1 }**  **IpRouteEntry ::=**  **SEQUENCE {**  **ipRouteDest**  **IpAddress,**  **ipRouteIfIndex**  **INTEGER,**  **ipRouteMetric1**  **INTEGER,**  **ipRouteMetric2**  **INTEGER,**  **ipRouteMetric3**  **INTEGER,**  **ipRouteMetric4**  **INTEGER,**  **ipRouteNextHop**  **IpAddress,**  **ipRouteType**  **INTEGER,**  **ipRouteProto**  **INTEGER,**  **ipRouteAge**  **INTEGER,**  **ipRouteMask**  **IpAddress,**  **ipRouteMetric5**  **INTEGER,**  **ipRouteInfo**  **OBJECT IDENTIFIER**  **}**  **ipRouteDest OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP مقصد این مسیر. یک مدخل با مقدار 0.0.0.0 یک مسیر پیشفرض در نظر گرفته شده است. مسیرهای چندگانه به یک مقصد می تواند در جدول ظاهر شود، اما دسترسی به چنین مدخل های چندگانه ای به مکانیزم های دسترسی به جدول که توسط پروتکل مدیریت شبکه در حال استفاده تعریف شده وابسته است.  **::= { ipRouteEntry 1 }**  **ipRouteIfIndex OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  مقدار اندیس که به طور یکتا واسط محلی را از طریق اینکه کدام هاپ انتخاب بعدی این مسیر خواهد بود، شناسایی می کند.  **::= { ipRouteEntry 2 }**  **ipRouteMetric1 OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  معیار اصلی مسیریابی برای این مسیر. معناشناسی این میار از طریق پروتکل مسیریابی مشخص شده در مقدار ipRouteProto مسیر معین شده است. اگر این معیار استفاده نشده، این مقدار باید -1 قرار بگیرد.  **::= { ipRouteEntry 3 }**  **ipRouteMetric2 OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  معیار جایگزین مسیریابی برای این مسیر. معناشناسی این معیار از طریق پروتکل مسیریابی مشخص شده در مقدار ipRouteProto مسیر معین شده است. اگر این معیار استفاده نشده، این مقدار باید -1 قرار بگیرد  **::= { ipRouteEntry 4 }**  **ipRouteMetric3 OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  معناشناسی این معیار از طریق پروتکل مسیریابی مشخص شده در مقدار ipRouteProto مسیر معین شده است. اگر این معیار استفاده نشده، این مقدار باید -1 قرار بگیرد.  **::= { ipRouteEntry 5 }**  **ipRouteMetric4 OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  معناشناسی این معیار از طریق پروتکل مسیریابی مشخص شده در مقدار ipRouteProto مسیر معین شده است. اگر این معیار استفاده نشده، این مقدار باید -1 قرار بگیرد.  **::= { ipRouteEntry 6 }**  **ipRouteNextHop OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP هاپ بعدی این مسیر. (در شرایطی که یک مسیر که به یکاسطی که از طریق رسانه همه پخشی درک شده محدود شده است، مقدار این بخش آدرس IP عامل روی آن واسط است)  **::= { ipRouteEntry 7 }**  **ipRouteType OBJECT-TYPE**  **SYNTAX INTEGER {**  **other(1), -- none of the following**  **invalid(2), -- an invalidated route**  **-- route to directly**  **direct(3), -- connected (sub-)network**  **-- route to a non-local**  **indirect(4) -- host/network/sub-network**  **}**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  نوع مسیر. توجه داشته باشید مقادیر مستقیم(3) و غیر مستقیم(4) به مفهوم مسیریابی مستقیم و غیر مستقیم در معماری IP ارجاع می دهند.  تنظیم این شیء به مقدار نادرست(2) نتیجه باطل شدن مدخل مرتبط در شیء ipRouteTable را در پی دارد. به عبارتی به صورت موثر مقصد شناسایی شده با مدخل گفته شده را از مسیری که با مدخل گفته شده شناسایی شده است جدا می کند. این یک موضوع خاص اجرا است که عامل یک مدخل باطل شده را از جدول حذف می کند. بدین ترتیب، ایستگاه های مدیریت باید آماده دریافت اطلاعات جدولی، از عامل های مرتبط به مدخل هایی که اکنون در حال استفاده نیستند، باشند. تفسیر مناسب چنین مدخل هایی نیازمند بررسی شیء ipRouteType مربوطه می باشد.  **::= { ipRouteEntry 8 }**  **ipRouteType OBJECT-TYPE**  **SYNTAX INTEGER {**  **other(1),**  **invalid(2**  **netmgmt(3),**  **icmp(4),**  **egp(5),**  **ggp(6),**  **hello(7),**  **rip(8),**  **is-is(9),**  **es-is(10),**  **ciscoIgrp(11),**  **bbnSpfIgp(12),**  **ospf(13),**  **bgp(14)**  **}**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مکانیزم مسیریابی از طریق آن که این مسیر آموخته شده است. دربرداشتن مقادیر جهت پروتکل های مسیریابی دروازه برای رساندن اینکه آن میزبان ها باید آن پروتکل ها را پشتیبانی کنند، مورد نظر نیست.  **::= { ipRouteEntry 9 }**  **ipRouteAge OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  ثانیه ها از زمانی که این مسیر آخرین بار به روز شده یا در غیر این صورت مشخص شده که صحیح است. توجه کنید که هیچ معنا و مفهومی از «خیلی قدیمی» جز از طریق دانستن پروتکل مسیریابی که به وسیله آن مسیر یادگرفته شده، نمی تواند برداشت شود.  **::= { ipRouteEntry 10 }**  **ipRouteMask OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  ماسک را به صورت AND منطقی قبل از اینکه با مقدار بخش ipRouteDest مقایسه شود به صورت «و» منطقی با آدرس مقصد نشان می دهد. برای سیستم هایی که ماسک های زیرشبکه دلخواه را پشتیبانی نمی کنند، یک عامل، مقدار ipRouteMask را به وسیله مشخص کردن اینکه مقدار بخش ipRouteDest به کلاس A، B یا C شبکه تعلق دارد می سازد و سپس یکی از موارد زیر را استفاده می کند:  **mask network**  **255.0.0.0 class-A**  **255.255.0.0 class-B**  **255.255.255.0 class-C**  اگر مقدار ipRouteDest 0.0.0.0 باشد (مسیر پیش فرض)، آن گاه مقدار ماسک هم 0.0.0.0 است. این نکته باید ذکر شود که همه زیرسیستم های مسیریابی IP به صورت ضمنی از این مکانیزم استفاده می کنند.  **::= { ipRouteEntry 11 }**  **ipRouteMetric5 OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  یک معیار مسیریابی جایگزین برای این مسیر. معنا و مفهوم این معیار توسط پروتکل مسیریابی مشخص شده در مقدار ipRouteProte مسیر مشخص شده است. اگر این معیار استفاده نشده، مقدارش باید -1 قرار بگیرد.  **::= { ipRouteEntry 12 }**  **ipRouteInfo OBJECT-TYPE**  **SYNTAX OBJECT IDENTIFIER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  یک ارجاع به تعاریف MIB مخصوص پروتکل مسیریابی خاص که مسئول این مسیر است، آنگونه که توسط ارزش مشخص شده در مقدار ipRouteProto مسیر مشخص شده است. اگر این اطلاعات موجود نیست، این مقدار باید OBJECT IDENTIFIER { 0 0 } قرار بگیرد که از نظر لغوی یک object identifier صحیح است و هر پیاده سازی مطابق ASN.1 و BER باید قادر به تولید و شناسایی این مقدار باشند.  **::= { ipRouteEntry 13 }**  -- جدول ترجمه آدرس IP  **ipNetToMediaTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF IpNetToMediaEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدول ترجمه آدرس IP استفاده شده برای نگاشت از آدرس های IP به آدرس های فیزیکی.  **::= { ip 22 }**  **ipNetToMediaEntry OBJECT-TYPE**  **SYNTAX IpNetToMediaEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  هر مدخل شامل یک آدرس IP به آدرس «فیزیکی» معادل است.  **INDEX { ipNetToMediaIfIndex,**  **ipNetToMediaNetAddress }**  **::= { ipNetToMediaTable 1 }**  **IpNetToMediaEntry ::=**  **SEQUENCE {**  **ipNetToMediaIfIndex**  **INTEGER,**  **ipNetToMediaPhysAddress**  **PhysAddress,**  **ipNetToMediaNetAddress**  **IpAddress,**  **ipNetToMediaType**  **INTEGER**  **}**  **ipNetToMediaIfIndex OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  واسطی که در آن معادل مدخل موثر است. واسط شناخته شده به وسیله مقدار مشخص از این اندیس، همسان با واسطی است که با مقدار مشابه ifIndex شناسایی شده است.  **::= { ipNetToMediaEntry 1 }**  **ipNetToMediaPhysAddress OBJECT-TYPE**  **SYNTAX PhysAddress**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  آدرس «فیزیکی» وابسته به رسانه.  **::= { ipNetToMediaEntry 2 }**  **ipNetToMediaNetAddress OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP مرتبط به آدرس «فیزیکی» وابسته به رسانه.  **::= { ipNetToMediaEntry 3 }**  **ipNetToMediaType OBJECT-TYPE**  **SYNTAX INTEGER {**  **other(1), -- none of the following**  **invalid(2), -- an invalidated mapping**  **dynamic(3),**  **static(4)**  **}**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  نوع نگاشت. تنظیم این شیء به مقدار نادرست(2) باعث باطل شدن مدخل مرتبط در ipNetToMediaTable می شود. به عبارتی به صورت موثر واسط شناسایی شده با مدخل گفته شده را از نگاشتی که با مدخل گفته شده شناسایی شده است جدا می کند. این یک موضوع خاص اجرا است که عامل یک مدخل باطل شده را از جدول حذف می کند. بدین ترتیب، ایستگاه های مدیریت باید آماده دریافت اطلاعات جدولی، از عامل های مرتبط به مدخل هایی که اکنون در حال استفاده نیستند، باشند. تفسیر مناسب چنین مدخل هایی نیازمند بررسی شیء ipNetToMediaType مربوطه می باشد.  **::= { ipNetToMediaEntry 4 }**  **-- additional IP objects**  **ipRoutingDiscards OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد مدخل های مسیریابی که برای دور انداخته شدن انتخاب شده اند، هر چند معتبر باشند. یک دلیل ممکن برای دور ریختن چنین مدخلی می تواند آزاد کردن فضای بافر برای سایر مدخل های مسیریابی باشد.  **::= { ip 23 }**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*** |
| **گروه ICMP :**  ---اجرای این گروه برای همه سیستم ها اجباری است  **icmpInMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه پیام های ICMP که موجودیت دریافت کرده است. توجه داشته باشید که این شمارنده شامل همه آن هایی است که به وسیله icmpInErrors شمرده شده اند.  **::= { icmp 1 }**  **icmpInErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد همه پیام های ICMP که موجودیت دریافت کرده است، اما به عنوان پیام های دارای خطاهای مخصوص ICMP معین شده است (جمع کنترلی ICMP خراب، طول خراب و ...)  **::= { icmp 2 }**  **icmpInDestUnreachs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Destination Unreachable .  **::= { icmp 3 }**  **icmpInTimeExcds OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Time Exceeded .  **::= { icmp 4 }**  i**cmpInParmProbs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Parameter Problem .  **::= { icmp 5 }**    **icmpInSrcQuenchs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**   1. تعداد پیغام های ICMP دریافت شده از نوع Source Quench .   **::= { icmp 6 }**    **icmpInRedirects OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع ICMP Redirection .  **::= { icmp 7 }**  **icmpInEchos OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع ICMP Echo (request) .  **::= { icmp 8 }**    **icmpInEchoReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Echo Reply .  **::= { icmp 9 }**  **icmpInTimestamps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع . Timestamp (request)  **::= { icmp 10 }**  **icmpInTimestampReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Timestamp Reply.  **::= { icmp 11 }**  **icmpInAddrMasks OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Address Mask Request.  **::= { icmp 12 }**  **icmpInAddrMaskReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP دریافت شده از نوع Address Mask Reply .  **::= { icmp 13 }**  **icmpOutMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های ICMP که این موجودیت تلاش کرده تا آن ها را ارسال نماید. توجه شود که در این شمارش تمامی پیغام های ICMP در نظر گرفته شده در icmpOutErrors ( یا همان icmp 15 ) هم در نظر گرفته می شوند.  **::= { icmp 14 }**  **icmpOutErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال نشده توسط این موجودیت بدلیل مشکلاتی چون کمبود بافر . در محاسبه این تعداد، نبایستی خطاهای غیر از نوع ICMP در نظر گرفته شود بطور مثال خطای ناشی از ناتوانی لایه IP در انجام عملیات مسیریابی دیتاگرام ( حاصل از بسته بندی پیغام ICMP در بسته IP) در این شمارش در نظر گرفته نمی شود. ممکن است در برخی پیاده سازی ها هیچ نوع خطایی در این شمارش در نظر گرفته نشوند.  **::= { icmp 15 }**  **icmpOutDestUnreachs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Destination Unreachable .  **::= { icmp 16 }**  **icmpOutTimeExcds OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Time Exceeded.  **::= { icmp 17 }**    **icmpOutParmProbs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Parameter Problem .  **::= { icmp 18 }**    **icmpOutSrcQuenchs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Source Quench .  **::= { icmp 19 }**  **icmpOutRedirects OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Redirection ، بدلیل آن که یک host هرگز چنین پیغمی ارسال نمی کند، مقدار این شی برای یک host همواره صفر است.  **::= { icmp 20 }**  **icmpOutEchos OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Echo (request) .  **::= { icmp 21 }**    **icmpOutEchoReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Echo Reply .  **::= { icmp 22 }**  **icmpOutTimestamps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Timestamp (Request).  **::= { icmp 23 }**  **icmpOutTimestampReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Timestamp Reply.  **::= { icmp 24 }**  **icmpOutAddrMasks OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Address Mask Request .  **::= { icmp 25 }**  **icmpOutAddrMaskReps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های ICMP ارسال شده از نوع Address Mask Reply .  **::= { icmp 26 }**  **گروه TCP :**  **tcpRtoAlgorithm OBJECT-TYPE**  **SYNTAX INTEGER {**  **other(1), -- none of the following**  **constant(2), -- a constant rto**  **rsre(3), -- MIL-STD-1778, Appendix B**  **vanj(4) -- Van Jacobson’s algorithm [10]**  **}**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  الگوریتم مورد استفاده جهت تعیین مقدار timeout مورد نیاز برای بازارسال بایت های ACK نشده.  **::= { tcp 1 }**  **tcpRtoMin OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حداقل مقدار مجاز تعیین شده در پیاده سازی TCP برای timeout بازارسال ( در واحد میلی ثانیه) . معناشناسی (semantic) دقیق تر اشیاء با این نوع ، به الگوریتم استفاده شده جهت تعیین زمان timeout بستگی دارد. به عنوان یک مثال خاص ، اگر الگوریتم timeout یاد شده ، rsre(3) باشد ، شیء تعریف شده از این نوع دارای معناشناسی (semantic) LBOUND ( که در RFC 793 توضیح داده شده است. )  **::= { tcp 2 }**    **tcpRtoMax OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حداکثر مقدار مجاز تعیین شده در پیاده سازی TCP برای timeout بازارسال ( در واحد میلی ثانیه) . معناشناسی (semantic) دقیق تر اشیاء با این نوع ، به الگوریتم استفاده شده جهت تعیین زمان timeout بستگی دارد. به عنوان یک مثال خاص ، اگر الگوریتم timeout یاد شده ، rsre(3) باشد ، شیء تعریف شده از این نوع دارای معناشناسی (semantic) UBOUND ( که در RFC 793 توضیح داده شده است. )  **::= { tcp 3 }**  **tcpMaxConn OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حد تعداد کل ارتباط قابل پشتیبانی توسط یک موجودیت. مقدار این شیء در موجودیت هایی که تعداد حداکثر ارتباط در آن ها متغییر است، باید -1 باشد.  **::= { tcp 4 }**    **tcpActiveOpens OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دفعاتی که ارتباطات TCP به صورت مستقیم از حالت CLOSED به حالت SYN-SENT گذار انجام داده اند.  **::= { tcp 5 }**  **tcpPassiveOpens OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دفعاتی که ارتباطات TCP به صورت مستقیم از حالت LISTEN به حالت SYN-SENT گذار انجام داده اند.  **::= { tcp 6 }**  **tcpAttemptFails OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مجموع تعداد دفعاتی که ارتباطات TCP به صورت مستقیم از حالت SYN-SENT یا حالت SYN-RCVD به حالت CLOSED گذار انجام داده اند و تعداد دفعاتی که ارتباطات TCP به صورت مستقیم از حالت SYN-RCVD به حالت LISTEN گذار انجام داده اند.  **::= { tcp 7 }**  **tcpEstabResets OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دفعاتی که ارتباطات TCP به صورت مستقیم از حالت ESTABLISHED یا حالت LOSE-WAIT به حالت CLOSED گذار انجام داده اند.  **::= { tcp 8 }**  **tcpCurrEstab OBJECT-TYPE**  **SYNTAX Gauge**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد ارتباطات TCP که حالت کنونی آنها ESTABLISHED یا CLOSE یا WAIT باشد.  **::= { tcp 9 }**  **tcpInSegs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل segment های دریافتی را نشان می دهد. در این شمارش، segment های دریافت شده در ارتباطات کنونی در نظر گرفته می شوند. همچنین segment های دریافت شده با خطا هم در نظر گرفته می شوند.  **::= { tcp 10 }**  **tcpOutSegs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل segment هایی که ارسال شده اند را نشان می دهد. در این شمارش، segment های ارسال شده در ارتباط کنونی در نظر گرفته می شوند اما segment هایی که تماما شامل بایت های بازارسال شده باشند در این شمارش نظر گرفته نمی شوند.  **::= { tcp 11 }**  **tcpRetransSegs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل segment هایی که بازارسال شده اند را نشان می دهد. یعنی تعداد segment های TCP که تعداد یک بایت یا بیشتر از آن ها بازارسال شده باشند.  **::= { tcp 12 }**  **tcpConnTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF TcpConnEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدولی حاوی اطلاعات یک ارتباط به خصوص است.  **::= { tcp 13 }**  **tcpConnEntry OBJECT-TYPE**  **SYNTAX TcpConnEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  اطلاعاتی در خصوص ارتباط TCP خاص کنونی است. شیء ای از این نوع بصورت گذرا است ، یعنی وقتی ارتباط به حالت CLOSED می رود ، قطع می گردد ( یا مدت کوتاهی پس از این گذار قطع می گردد.)  **INDEX { tcpConnLocalAddress,**  **tcpConnLocalPort,**  **tcpConnRemAddress,**  **tcpConnRemPort }**  **::={ tcpConnTable 1 }**  **TcpConnEntry ::=**  **SEQUENCE {**  **tcpConnState**  **INTEGER,**  **tcpConnLocalAddress**  **IpAddress,**  **tcpConnLocalPort**  **INTEGER (0..65535),**  **tcpConnRemAddress**  **IpAddress,**  **tcpConnRemPort**  **INTEGER (0..65535)**  **}**  **tcpConnState OBJECT-TYPE**  **SYNTAX INTEGER {**  **closed(1),**  **listen(2),**  **synSent(3),**  **synReceived(4),**  **established(5),**  **finWait1(6),**  **finWait2(7),**  **closeWait(8),**  **lastAck(9),**  **closing(10),**  **timeWait(11),**  **deleteTCB(12)**  **}**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  حالت این ارتباط TCP را نشان می دهد.  تنها مقداری که در این شی توسط ایستگاه مدیریتی می تواند قرار داده شود deleteTCB(12) است. لذا اگر یک ایستگاه مدیریتی بخواهد مقداری غیر از مقدار یاد شده را در این شیء قرار دهد ، نماینده می تواند یک پاسخ ‘badValue’ برگرداند.  قراردادن مقدار deleteTCB(12) در این شی به منزله پاک کردن TCB ( یاد شده در RFC 793 ) مربوط به ارتباط نظیر در گره مدیریت خواهد بود که این امر منجر به قطع بلافاصله ارتباط خواهد شد.  **::= { tcpConnEntry 1 }**    **tcpConnLocalAddress OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP محلی در این ارتباط TCP . در صورت برقراری ارتباط در حالت "گوش کن" ، که خواهان دریافت دیتاگرام به ازای هریک از رابط های مرتبط با این گره است، مقدار این شیء برابر 0.0.0.0 خواهد بود.  **::= { tcpConnEntry 2 }**  **tcpConnLocalPort OBJECT-TYPE**  **SYNTAX INTEGER (0..65535)**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  شماره پورت محلی در این ارتباط TCP .  **::= { tcpConnEntry 3 }**    **tcpConnRemAddress OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP راه دور در این ارتباط TCP .  **::= { tcpConnEntry 4 }**  **tcpConnRemPort OBJECT-TYPE**  **SYNTAX INTEGER (0..65535)**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  شماره پورت راه دور در این ارتباط TCP .  **::={ tcpConnEntry 5 }**    **tcpInErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل segment های TCP که با خطا دریافت شده اند. (مثلا بدیل نادرستی Checksum )  **::= { tcp 14 }**    **tcpOutRsts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد segment های TCP ارسال شده که حاوی RST flag بوده اند.  **::= { tcp 15 }**    **گروه UDP :**  **udpInDatagrams OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل دیتاگرام های UDP تحویل داده شده به کاربران UDP .  **::= { udp 1 }**    **udpNoPorts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل دیتاگرام های UDP دریافت شده که در پورت مقصد برایشان اپلیکیشنی وجود نداشته است.  **::= { udp 2 }**  **udpInErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد دیتاگرام های UDP دریافت شده که به دلایلی غیر از عدم وجود اپلیکیشن در پورت مقصد برای آن ها ، تحویل داده نشده اند.  **::= { udp 3 }**  **udpOutDatagrams OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل دیتاگرام های UDP ارسال شده از سوی این موجودیت.  **::= { udp 4 }**    -- جدول شنونده UDP  -- جدول شنونده UDP اطلاعاتی در خصوص end-point های این موجودیت را در بر می گیرد که بر روی این end-pointها یک اپلیکیشن محلی در حال حاضر دیتاگرام ها را می پذیرد.  **udpTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF UdpEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدولی است که اطلاعات شنونده UDP را در بر دارد.  **::= { udp 5 }**      **udpEntry OBJECT-TYPE**  **SYNTAX UdpEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  ستون های جدول udpTable ( یا همان udp 5 ) که اطلاعاتی در خصوص شنونده UDP کنونی هستند.  **INDEX { udpLocalAddress, udpLocalPort }**  **::= { udpTable 1 }**  **UdpEntry ::=**  **SEQUENCE {**  **udpLocalAddress**  **IpAddress,**  **udpLocalPort**  **INTEGER (0..65535)**  **}**  **udpLocalAddress OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP محلی این شنونده UDP است. در حالتی که شنونده UDP خواهان دریافت دیتاگرام به ازای هریک از رابط های مرتبط با این گره است، مقدار این شیء برابر 0.0.0.0 خواهد بود.  **::= { udpEntry 1 }**  **udpLocalPort OBJECT-TYPE**  **SYNTAX INTEGER (0..65535)**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  شماره پورت محلی این شنونده UDP را نشان می دهد  **::= { udpEntry 2 }**  **گروه EGP :**  **egpInMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های EGP دریافت شده بدون خطا را نشان می دهد.  **::= { egp 1 }**  **egpInErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های EGP دریافت شده و همراه با خطا را نشان می دهد.  **::= { egp 2 }**    **egpOutMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های EGP ایجاد شده را نشان می دهد.  **::= { egp 3 }**    **egpOutErrors OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های EGP ایجاد شده بصورت محلی که بدلیل محدودیت منابع داخل یک موجودیت از نوع EGP ارسال نشده اند.  **::= { egp 4 }**  **egpNeighTable OBJECT-TYPE**  **SYNTAX SEQUENCE OF EgpNeighEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  جدول همسایه در پروتکل EGP  **::= { egp 5 }**    **egpNeighEntry OBJECT-TYPE**  **SYNTAX EgpNeighEntry**  **ACCESS not-accessible**  **STATUS mandatory**  **DESCRIPTION**  اطلاعاتی در خصوص رابطه این موجودیت با یک همسایه تحت پروتکل EGP می دهد.  **INDEX { egpNeighAddr }**  **::={ egpNeighTable 1 }**  **EgpNeighEntry ::=**  **SEQUENCE {**  **egpNeighState**  **INTEGER,**  **egpNeighAddr**  **IpAddress,**  **egpNeighAs**  **INTEGER,**  **egpNeighInMsgs**  **Counter,**  **egpNeighInErrs**  **Counter,**  **egpNeighOutMsgs**  **Counter,**  **egpNeighOutErrs**  **Counter,**  **egpNeighInErrMsgs**  **Counter,**  **egpNeighOutErrMsgs**  **Counter,**  **egpNeighStateUps**  **Counter,**  **egpNeighStateDowns**  **Counter,**  **egpNeighIntervalHello**  **INTEGER,**  **egpNeighIntervalPoll**  **INTEGER,**  **egpNeighMode**  **INTEGER,**  **egpNeighEventTrigger**  **INTEGER**  **}**  **egpNeighState OBJECT-TYPE**  **SYNTAX INTEGER {**  **idle(1),**  **acquisition(2),**  **down(3),**  **up(4),**  **cease(5)**  **}**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حالت EGP سیستم محلی نسبت به همسایه این موجودیت تحت پروتکل EGP را نشان می دهد. هر حالت EGP با یک مقداری که یکی از مقدار عددی مربوط به حالت یاد شده در RFC 904 بیشتر است نمایش داده می شود.  **::= { egpNeighEntry 1 }**  **egpNeighAddr OBJECT-TYPE**  **SYNTAX IpAddress**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  آدرس IP همسایه این موجودیت تحت پروتکل EGP را نشان می دهد.  **::= { egpNeighEntry 2 }**  **egpNeighAs OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  سیستم خودمختار این همتای تحت پروتکل EGP را نشان می دهد. اگر شماره همسایه این سیستم خودمختار نامشخص باشد بایستی مقدار این عضو جدول را صفر قرار داد.  **::= { egpNeighEntry 3 }**  **egpNeighInMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیام های ارسالی از سوی این همتای تحت پروتکل EGP که بدون خطا دریافت شده اند.  **::= { egpNeighEntry 4 }**  **egpNeighInErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  عداد پیام های ارسالی از سوی این همتای تحت پروتکل EGP که همراه با خطا همراه بوده اند. ( مثلا checksum نادرست EGP)  **::= { egpNeighEntry 5 }**  **egpNeighOutMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های EGP ایجاد شده بصورت محلی که به این همتای از نوع EGP ارسال شده اند.  **::= { egpNeighEntry 6 }**  **egpNeighOutErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های EGP ایجاد شده بصورت محلی که بدلیل محدودیت منابع داخل یک موجودیت از نوع EGP به این همتای از نوع EGP ارسال نشده اند.  **::= { egpNeighEntry 7 }**  **egpNeighInErrMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های خطای تعریف شده تحت پروتکل EGP که توسط این همتای از نوع EGP دریافت شده اند.  **::= { egpNeighEntry 8 }**  **egpNeighOutErrMsgs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد پیغام های خطای تعریف شده تحت پروتکل EGP که به این همتای از نوع EGP ارسال شده اند.  **::= { egpNeighEntry 9 }**  **egpNeighStateUps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد گذارهای به حالت UP ، با این همتای از نوع EGP .  **::= { egpNeighEntry 10 }**  **egpNeighStateDowns OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد گذارهای از حالت UP به سایر حالات ، با این همتای از نوع EGP .  **::= { egpNeighEntry 11 }**  **egpNeighIntervalHello OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حد فاصل بین ارسال مجدد فرمان های EGP Hello (در حد صد ثانیه ) . این شیء نمایانگر تایمر t1 تعریف شده در RFC 904 است.  **::= { egpNeighEntry 12 }**  **egpNeighIntervalPoll OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  حد فاصل بین ارسال مجدد فرمان های نمونه برداری (رای گیری) (در حد صد ثانیه ) . این شیء نمایانگر تایمر t3 تعریف شده در RFC 904 است.  **::= { egpNeighEntry 13 }**  **egpNeighMode OBJECT-TYPE**  **SYNTAX INTEGER { active(1), passive(2) }**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  مُد نمونه برداری(رای گیری) این موجودیت EGP را نشان می دهد که می تواند passive یا active باشد.  **::= { egpNeighEntry 14 }**  **egpNeighEventTrigger OBJECT-TYPE**  **SYNTAX INTEGER { start(1), stop(2) }**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  یک متغییر کنترلی است که جهت ایجاد رخ دادهای شروع و توقف از سوی اپراتور، مورد استفاده قرار می گیرد. وقتی این متغییر در حالت read باشد همواره آخرین مقدار موجود در egpNeighEventTrigger را برمی گرداند. اگر از زمان آخرین مقداردهی زیر-سیستم مدیریت شبکه موجود در گره مقداردهی نشده باشد، مقدار ‘stop’ را بر می گرداند.  **::= { egpNeighEntry 15 }**  **egpAs OBJECT-TYPE**  **SYNTAX INTEGER**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  شماره سیستم خودمختار مربوط به این موجودیت EGP را نشان می دهد.  **::= { egp 6 }**  **گروهSNMP :**  **snmpInPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های SNMP که توسط سرویسی در لایه انتقال به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند را نشان می دهد.  **::= { snmp 1 }**  **snmpOutPkts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های SNMP که توسط یک موجودیت تحت مدیریت پروتکل SNMP به سرویسی در لایه انتقال تحویل داده شده بوده اند ، را نشان می دهد.  **::= { snmp 2 }**  **snmpInBadVersions OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های SNMP که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مربوط به یک نسخه SNMP پشتیبانی نشده بوده اند ، را نشان می دهد.  **::= { snmp 3 }**  **snmpInBadCommunityNames OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های SNMP که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و در آن ها از نام یک انجمن SNMP که برای آن موجودیت ناشناخته بوده استفاده شده است، را نشان می دهد.  **::= { snmp 4 }**  **snmpInBadCommunityUses OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل پیغام های SNMP که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و نشان دهنده ی یک عملیات SNMP بوده که توسط یک انجمن SNMP یاد شده در آن پیغام غیر مجاز اعلام شده اند، را نشان می دهد.  **::= { snmp 5 }**  **snmpInASNParseErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل خطاهای نحوی در زبان ASN.1 یا خطاهای رمزگذاری به روش BER را که یک موجودیت تحت مدیریت پروتکل SNMP هنگام رمزگشایی پیغام های SNMP (رسیده به خودش) با آن ها رو به رو شده است را نشان می دهد.  **::= { snmp 6 }**  **::= { snmp 7 }**مورد استفاده قرار نگرفته است  **snmpInTooBigs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های معتبری که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مقدار فیلد error-status آن ها برابر ‘tooBig’بوده است را نشان می دهد.  **::= { snmp 8 }**  **snmpInNoSuchNames OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های معتبری که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مقدار فیلد error-status آن ها برابر ‘noSuchName’بوده است را نشان می دهد.  **::= { snmp 9 }**  **snmpInBadValues OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های معتبری که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مقدار فیلد error-status آن ها برابر ‘badValue’بوده است را نشان می دهد.  **::= { snmp 10 }**  **snmpInReadOnlys OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های معتبری که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مقدار فیلد error-status آن ها برابر ‘readOnly’ بوده است را نشان می دهد. توجه گردد این شیء یک خطای پروتکل و هدف از وجود آن ایجاد ایجاد PDU هایی با مقدار ‘readOnly’ در فیلد error-status آن هاست تا بوسیله این پروتکل پیاده سازی های اشتباه پروتکل SNMP مورد شناسایی قرار گیرند.  **::= { snmp 11 }**  **snmpInGenErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU هایی که به یک موجودیت تحت مدیریت پروتکل SNMP تحویل داده شده اند و مقدار فیلد error-status آن ها برابر ‘genErr’بوده است را نشان می دهد.  **::= { snmp 12 }**  **snmpInTotalReqVars OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل اشیاء موجود در MIB که بدلیل دریافت PDU های معتبر از نوع Get-Request و Get-Next بصورت موفقیت آمیز توسط یک موجودیت تحت مدیریت پروتکل SNMP بازیابی شده اند را نشان می دهد.  **::= { snmp 13 }**  **snmpInTotalSetVars OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل اشیاء موجود در MIB که بدلیل دریافت PDU های معتبر از نوع Set-Request بصورت موفقیت آمیز توسط یک موجودیت تحت مدیریت پروتکل SNMP تغییر پیدا کرده اند را نشان می دهد.  **::={ snmp 14 }**  **snmpInGetRequests OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Requestکه توسط پروتکل SNMP پذیرفته شده و مورد پردازش قرار گرفته اند را نشان می دهد.  **::= { snmp 15 }**  **snmpInGetNexts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Next که توسط پروتکل SNMP پذیرفته شده و مورد پردازش قرار گرفته اند را نشان می دهد.  **::= { snmp 16 }**  **snmpInSetRequests OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Set-Request که توسط پروتکل SNMP پذیرفته شده و مورد پردازش قرار گرفته اند را نشان می دهد.  **::={ snmp 17}**  **snmpInGetResponses OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Response که توسط پروتکل SNMP پذیرفته شده و مورد پردازش قرار گرفته اند را نشان می دهد.  **::= { snmp 18 }**  **snmpInTraps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل تله های PDU ای که توسط پروتکل SNMP پذیرفته شده و مورد پردازش قرار گرفته اند را نشان می دهد.  **::= { snmp 19 }**  **snmpOutTooBigs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU هایی که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند و مقدار فیلد error-status آن ها برابر ‘tooBig’ بوده است را نشان می دهد.  **::= { snmp 20 }**  **snmpOutNoSuchNames OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU هایی که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند و مقدار فیلد error-status آن ها برابر ‘noSuchName’ بوده است را نشان می دهد.  **::={snmp 21 }**  **snmpOutBadValues OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU هایی که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند و مقدار فیلد error-status آن ها برابر ‘badValue’بوده است را نشان می دهد.  **::={ snmp22}**  **snmpOutGenErrs OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU هایی که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند و مقدار فیلد error-status آن ها برابر ‘genErr’بوده است را نشان می دهد.  **::={ snmp24}**  **snmpOutGetRequests OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Request که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند را نشان می دهد.  **::= { snmp 25 }**  **snmpOutGetNexts OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Next که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند را نشان می دهد.  **::= { snmp 26 }**  **snmpOutSetRequests OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Set-Request که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند را نشان می دهد.  **::= { snmp 27 }**  **snmpOutGetResponses OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل PDU های از نوع Get-Response که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند را نشان می دهد.  **::= { snmp 28 }**  **snmpOutTraps OBJECT-TYPE**  **SYNTAX Counter**  **ACCESS read-only**  **STATUS mandatory**  **DESCRIPTION**  تعداد کل تله های PDU که توسط یک موجودیت تحت مدیریت پروتکل SNMP ایجاد شده اند را نشان می دهد.  **::= { snmp 29 }**  **snmpEnableAuthenTraps OBJECT-TYPE**  **SYNTAX INTEGER { enabled(1), disabled(2) }**  **ACCESS read-write**  **STATUS mandatory**  **DESCRIPTION**  تعیین می کند آیا برنامه نماینده در پروتکل SNMP اجازه ایجاد تله های احراز هویت را دارد یا خیر. مقدار این شیء تمام اطلاعات پیکره بندی را نادیده می گیرد و به این ترتیب این امکان را فراهم می آورد تا تله های یاد شده غیر فعال شوند.  بایستی توجه گردد قویا توصیه می گردد این شی در حافظه غیر فعال ذخیره گردد تا در صورت مقداردهی اولیه مجدد سیستم مدیریت شبکه دچار دگرگونی و تغییر نگردد.  **::= { snmp 30 }**  **END** |

1. .management information base [↑](#footnote-ref-1)
2. Agent [↑](#footnote-ref-2)