

网络管理

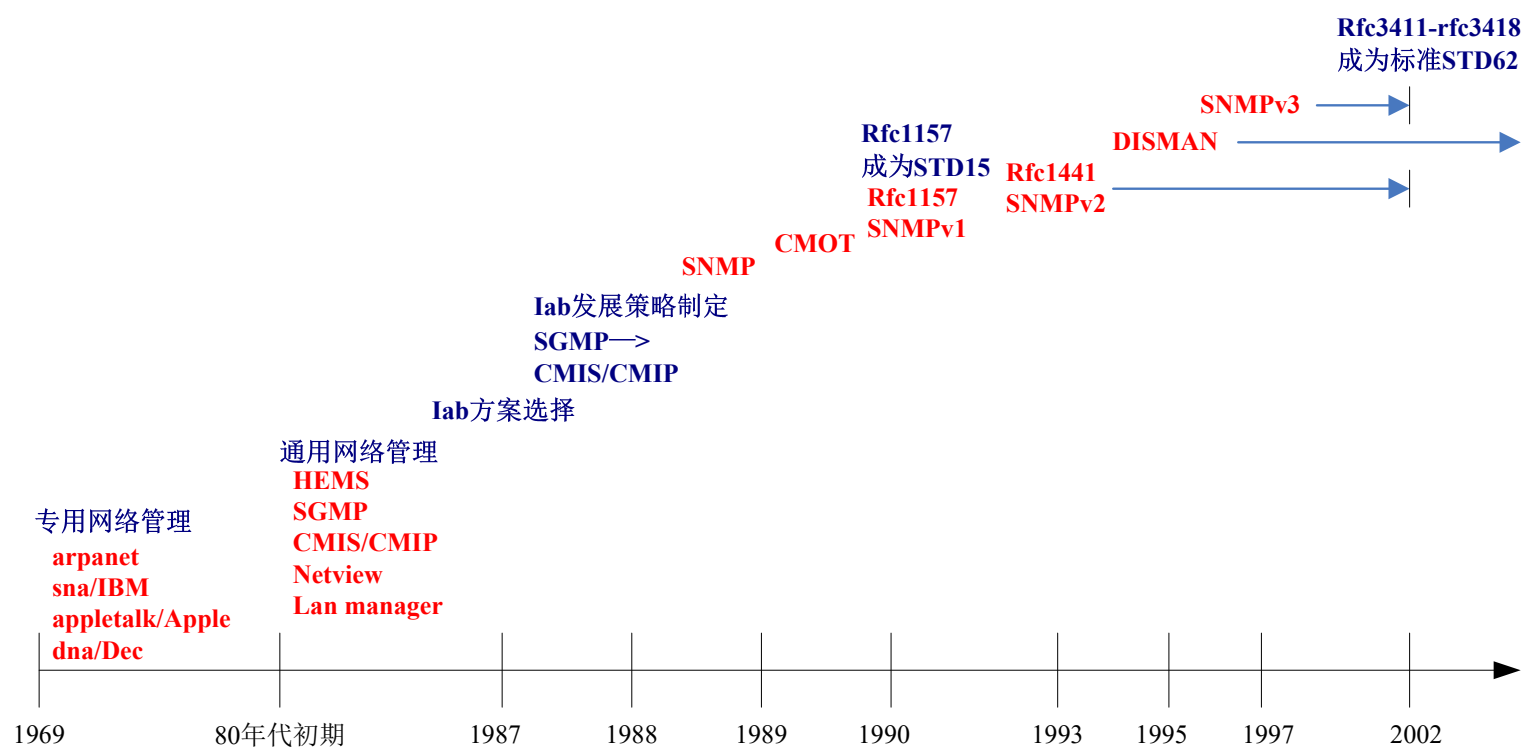
主要内容

- [网络管理背景](#)
- [网络管理模型](#)
 - 网络管理信息模型 (SMI)
 - 网络管理通信模型 (SNMP协议)
- [网络管理工具](#)

网络管理的重要性

- 网络的重要
 - 人们生活已经离不开网络
- 网络管理的重要
 - 网络自身
 - 规模和复杂性越来越大，厂商和设备繁多，需要管理
 - 用户角度
 - 网络服务越来越多，用户需要高性能网络
 - 理论角度
 - 网络是一个系统、一种组织
 - 任何一个系统都需要管理

网络管理的发展



网络管理发展ing

- 每两年一次NOMS
 - 网络营运与管理专题讨论会 (network operation and management symposium)
 - 主办方: IEEE通信学会下属CNOM
- 每两年一次综合网络管理专题讨论会
 - 主办方: 国际信息处理联合会(IFIP)
- 网络管理解决方案
 - 网络管理论坛 (NFM) 的OMNIPoint
 - 开放软件基金会(OSF)的DME
 - OMG的公共对象请求代理体系结构 (CORBA)
 - TINA-C的电信信息网络体系结构 (TINA)
 - ITU-T的电信管理网 (TMN) ...
- 网络管理系统
 - hp的openview
 - ibm的netview系列
 - sunsoft的sunnetmanager
 - CiscoWorks 2000
 - Raisecom的NView NNM...

网络管理定义

- ISO在iso/iec7498-4中定义
 - 开放系统互连管理(OSI management)是指这样一些功能，它们控制、协调、监视OSI 环境下的一些资源，这些资源保证OSI环境下的通信。
- **通俗定义**
 - 通过某种方式维护网络的正常运行，当网络出现故障时能及时报告和处理，并协调、保持网络系统的高效运行等。

[返回](#)

OSI网络管理模型

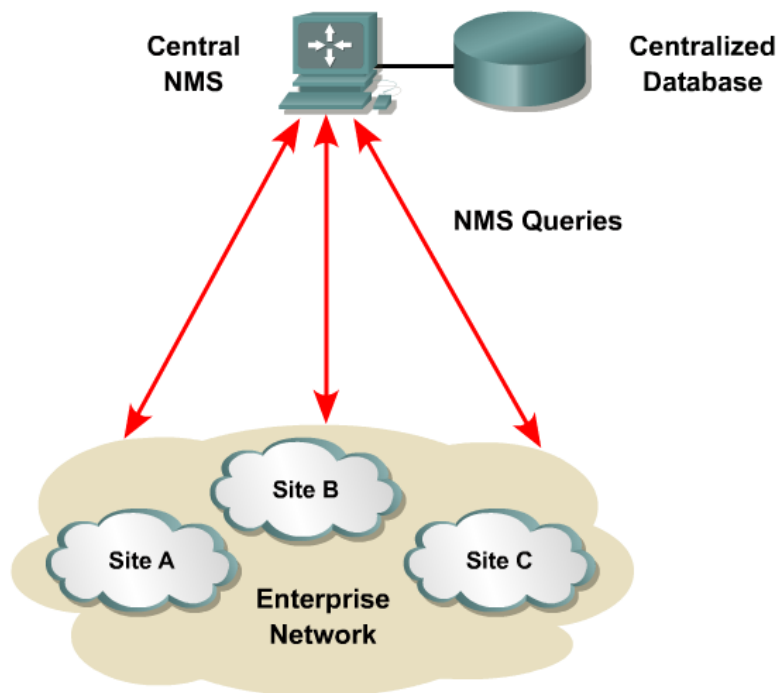
- 组织模型
 - 描述网络管理组件以及它们之间的关系
- 信息模型
 - 网络对象里管理信息的结构和存储
- 通信模型
 - 管理者和被管理者之间的通信，SNMP
- 功能模型
 - 着重点在网络管理系统上的网络管理程序

网络管理组件

- 管理工作站(NMS)
 - 必须有管理应用程序，用于数据分析、故障修复
 - 给网络管理员提供接口
- 管理代理(Agent)
 - 能够响应管理工作站的信息或者操作请求
 - 向管理工作站提供重要但未经请求的信息(trap)
- 管理信息库
 - 网络环境下资源的表示
 - MIB (management information base) 被管对象的集合
- 网络管理协议
 - 管理工作站和管理代理之间的通信机制

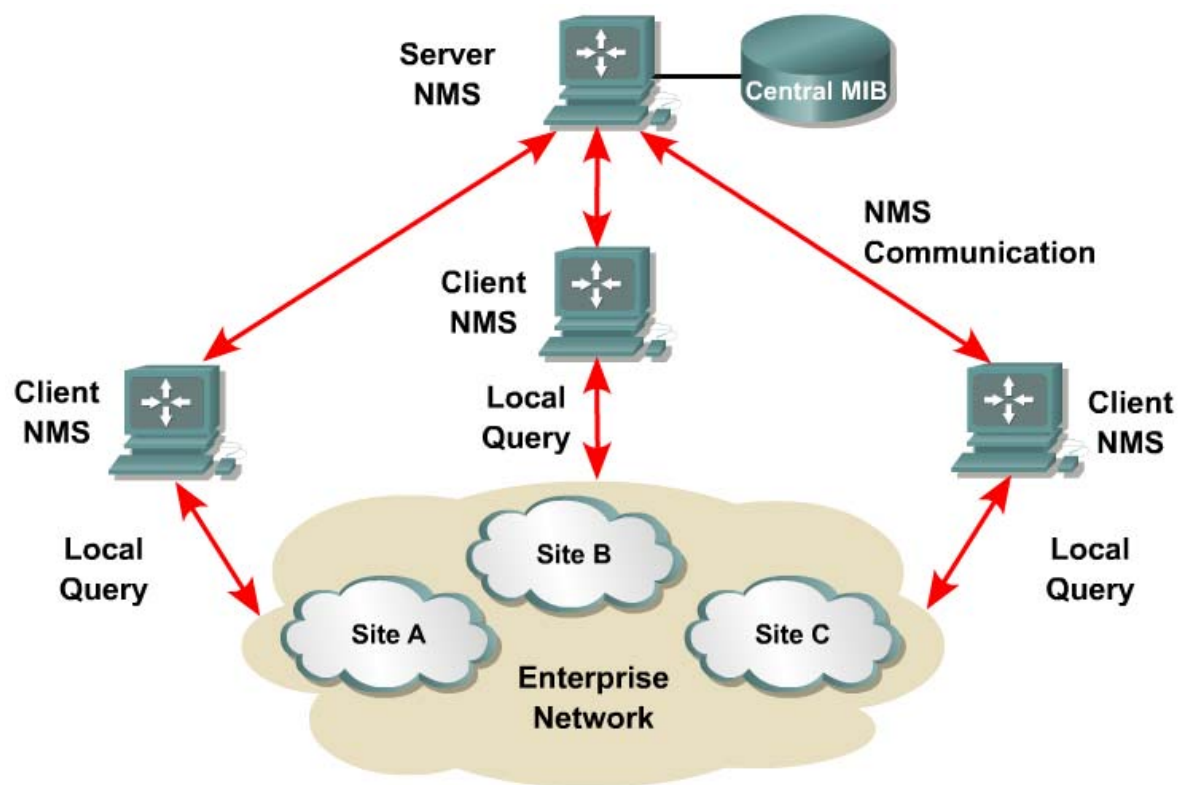
网络管理体系结构（一）

- 集中式网络管理结构



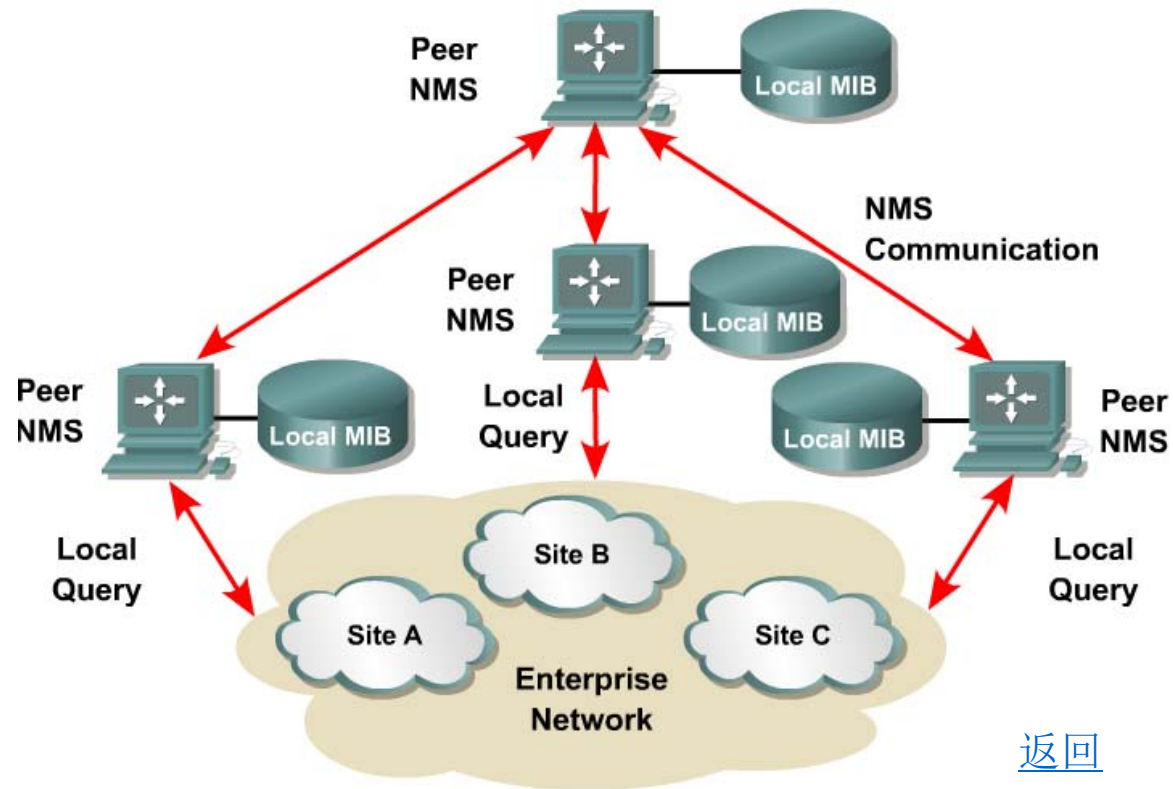
网络管理体系结构（二）

- 分层的网络管理结构



网络管理体系结构 (三)

- 分布式网络管理结构



网络管理功能模型

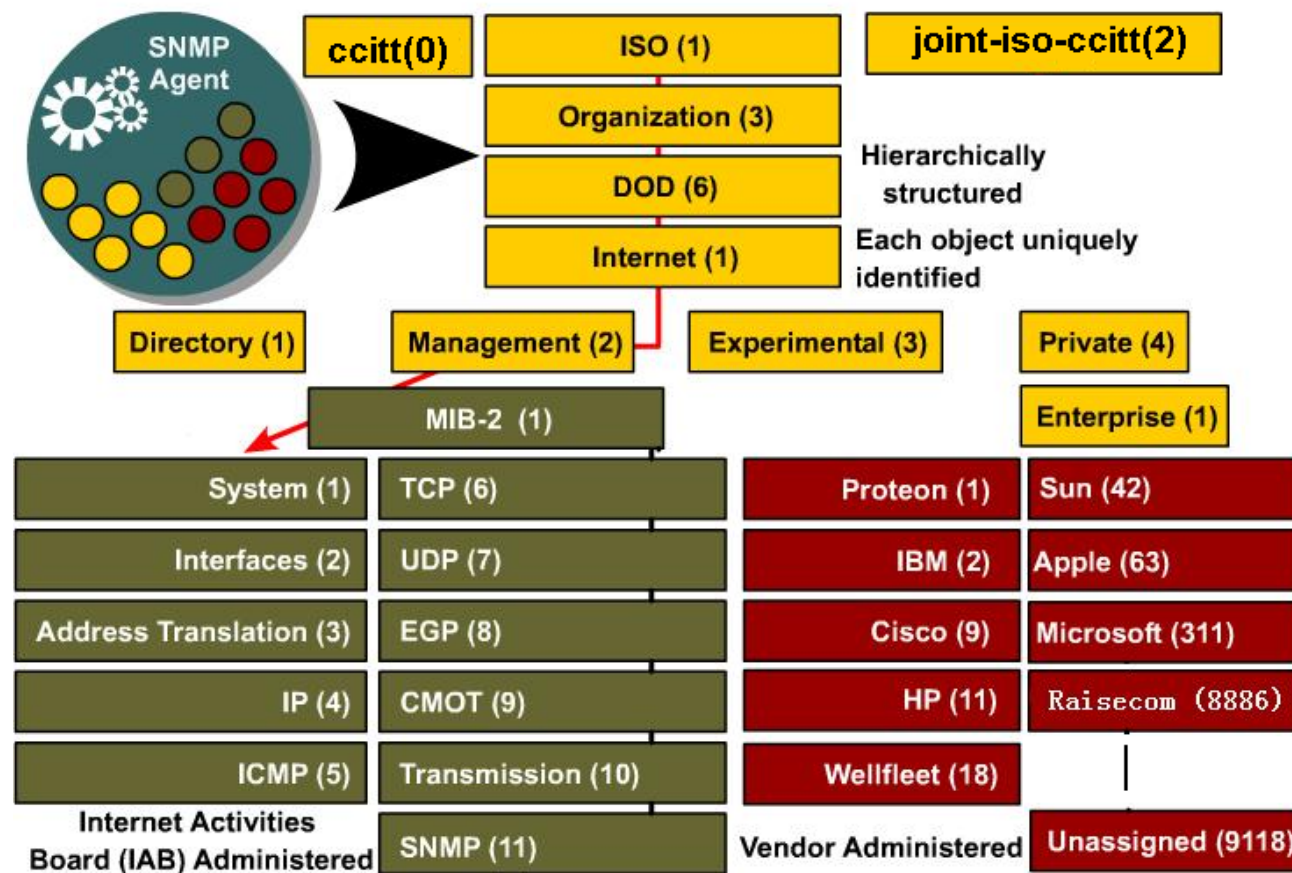
- 故障管理(fault management)
 - 探测、隔离和修正OSI环境下的不正常操作
- 计费管理(accounting management)
 - 记录网络资源的使用，控制和监测网络操作的费用和代价
- 配置管理(configuration management)
 - 初始化并配置网络；控制、识别被管理对象使其实现特定的功能或者使网络达到最优；
- 性能管理(performance management)
 - 对被管理对象的行为、系统资源的运行状况和通信活动的效率进行评价
- 安全管理(security management)
 - 管理对象信息保护和访问控制

[返回](#)

SNMP网络管理信息模型

- MIB
 - 网络管理信息被存储在一个数据库中，即MIB
 - 存储网络元素及他们属性的结构化信息
 - 分层或者树状结构
- SMI
 - Structure of Management Information, ASN.1的子集
 - 规定MIB中对象的命名方式、数据类型、编码和传输方式，是定义MIB必须遵循的标准
 - SMIv1-rfc1155、SMIv2-rfc2578

MIB树



公司私有mib树: 1.3.6.1.4.1.8886

MIB树介绍

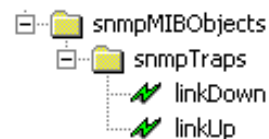
- OID (Object Identifier)
 - 表示一个被管理对象，点十进制，例如1.3.6.1
 - 对应一个可读的名字，例如internet, raisecom
 - 树的叶子节点才是真正能够被管理的对象
- MIB树
 - Internet在iso.organization.DoD下
 - Internet分为四个子树
 - directory:为将来OSI目录保留
 - mgmt: 在IAB批准文档中定义的对象 (标准MIB)
 - experimental: 在internet实验中使用的对象
 - private:厂商私有mib

MIB树举例

- System(.1.3.6.1.2.1.1)

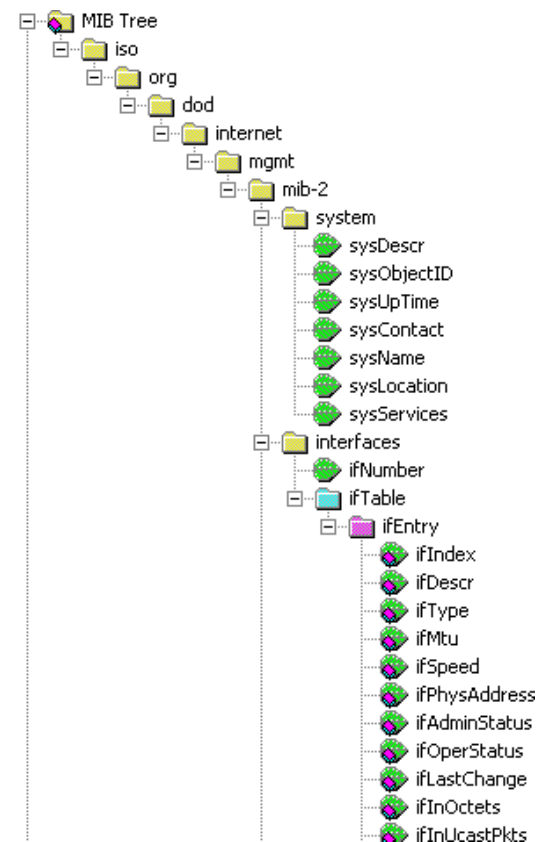
- 标量
- 表（二维标量）
 - 索引
 - 列

- trap



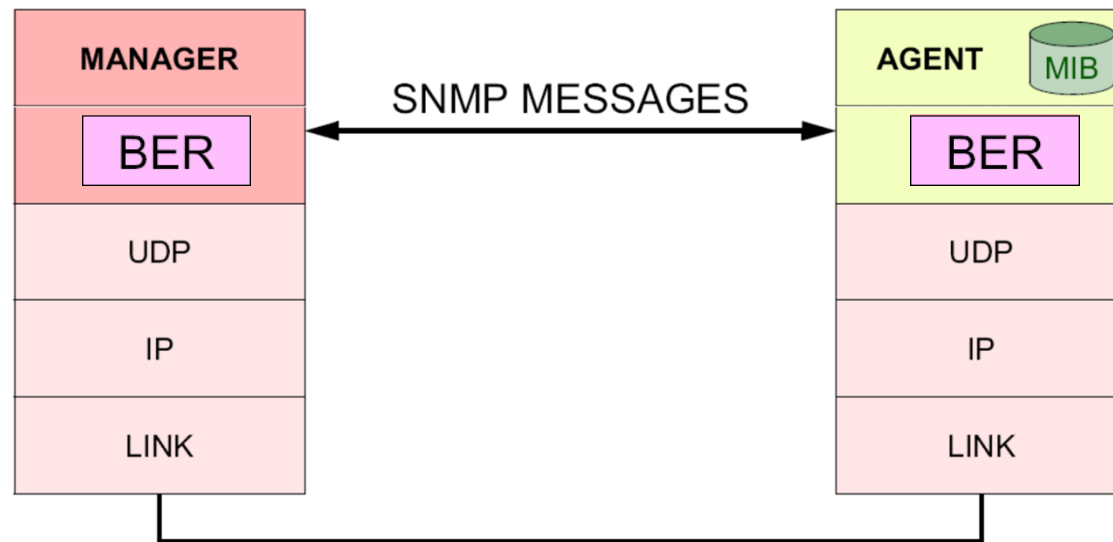
- object与instance

- 对象
- 实例



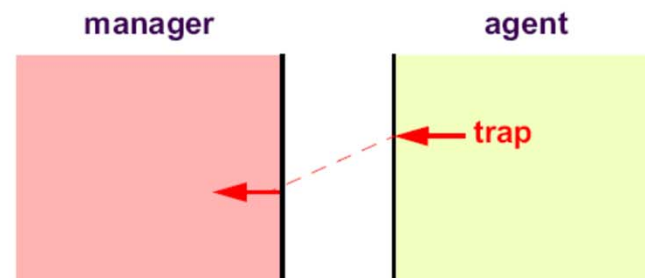
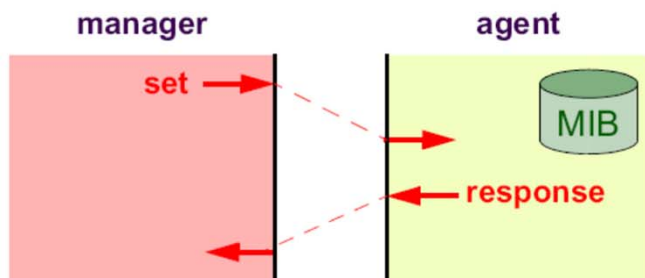
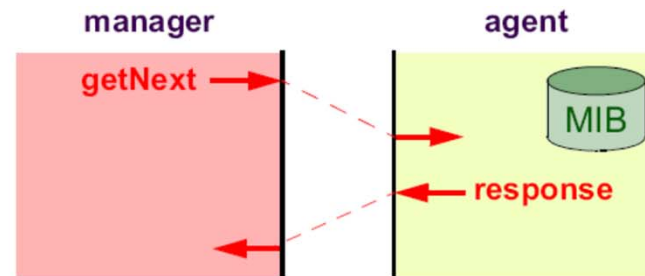
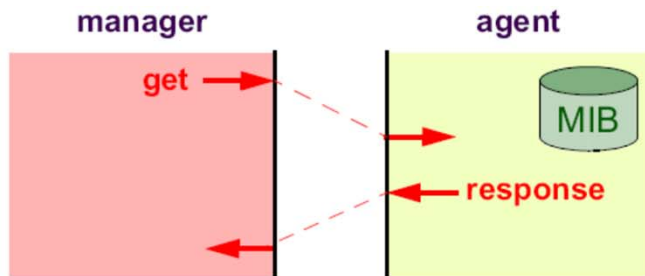
SNMPv1协议

- SNMPv1操作
- SNMPv1报文格式
- SNMPv1 trap
- SNMPv1缺陷



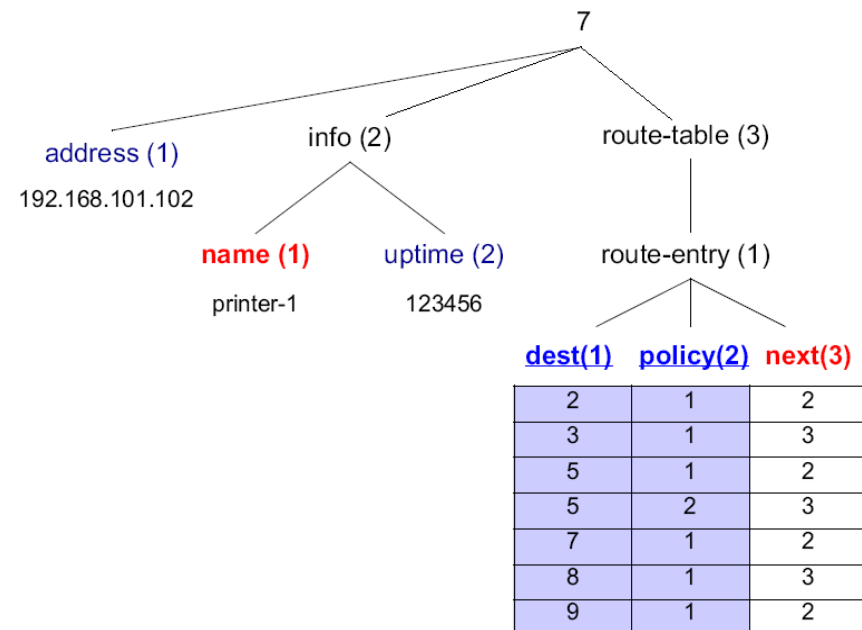
SNMPv1的操作

- Get、getNext、set、trap



SNMPv1 GET

- GET
 - 获取一个或者多个MIB变量
 - 错误码
 - noSuchName
 - tooBig
 - genErr
- 举例
 - `get(7.1.0);`
`response(7.1.0=>192.168.101.102)`
 - `get(7.1);` `response(noSuchName)`
 - `get(7.3.1.3.5.1);`
`response(7.3.1.3.5.1=>2)`
 - `get(7.3.1.1.5.1, 7.3.1.2.5.1,`
`7.3.1.3.5.1); response(7.3.1.1.5.1=>5,`
`7.3.1.2.5.1=>?, 7.3.1.3.5.1=>?)`



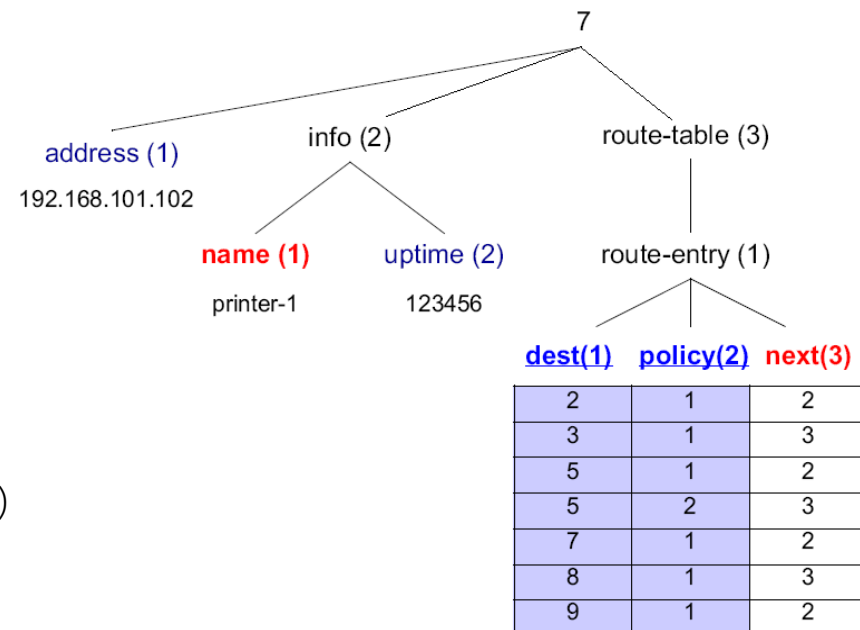
SNMPv1 SET

- SET

- 给一个对象实例赋值、创建表行
- 错误码
 - noSuchName
 - tooBig
 - badValue
 - genErr

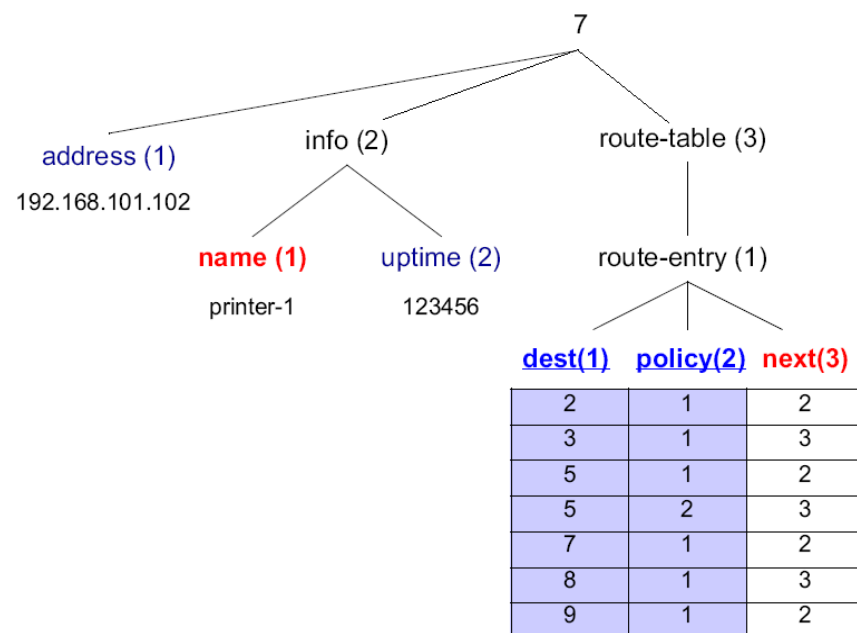
- 举例

- Set(7.2.1.0=>myPrinter);
response(7.2.1.0=>myPrinter)
- Set(7.2.1.0=>myPrinter,7.2.2.0=>0);
response(error-status=noSuchName; error-index=2)



SNMPv1 GET-NEXT

- GET-NEXT
 - 获取下一个mib变量oid和值
 - 错误码，跟GET一样
- 举例
 - getNext(7.1.0);
response(7.2.1.0=>printer-1)
 - getNext(7.2.1.0);
response(7.2.2.0=>123456)
 - getNext(1);
response(7.1.0=>192.168.101.102)
 - getNext(7.3.1.3.5.1);
response(7.3.1.3.5.2=>3)



SNMPv1报文格式

variable bindings:

NAME 1	VALUE 1	NAME 2	VALUE 2	NAME n	VALUE n
--------	---------	--------	---------	-----	-----	--------	---------

SNMP PDU:

PDU TYPE*	REQUEST ID	ERROR STATUS	ERROR INDEX	VARIABLE BINDINGS
-----------	------------	--------------	-------------	-------------------

SNMP message:

VERSION	COMMUNITY	SNMP PDU
---------	-----------	----------

- 差错状态 (error status)

- 0: noError
- 1: tooBig
- 2: noSuchName
- 3: badValue
- 4: 这个值不能修改
- 5: genErr

- 差错索引 (error index)

- 差错索引是一个偏移，它告诉管理器哪个变量引起这个差错。

```
SNMP: ----- Simple Network Management Protocol (Version 1) -----
SNMP:
SNMP: SNMP Version = 1
SNMP: Community = public
SNMP: Command = Get next request
SNMP: Request ID = 17
SNMP: Error status = 0 (No error)
SNMP: Error index = 0
SNMP:
SNMP: Object = {1.3.6.1.2.1.1.3} (sysUpTime)
SNMP: Value = NULL
SNMP:

00000000: 00 0e 5e 00 c3 66 00 19 5b 86 95 89 08 00 45 00 ..^昵..[谡?. E.
00000010: 00 43 b4 b7 00 00 80 11 5b d0 10 01 01 10 10 09 .C捷...[?...
00000020: 09 09 09 90 00 a1 00 2f 9a f1 30 25 02 01 00 04 ...??/泐 0%...
00000030: 06 70 75 62 6c 69 63 a1 18 02 01 11 02 01 00 02 .public?.....
00000040: 01 00 30 0d 30 0b 06 07 2b 06 01 02 01 01 03 05 ..0.0...+.....
00000050: 00
```

SNMPv1 TRAP

- 被管理站向管理站主动发送重要事件的通告
- SNMPv1 trap
 - 冷启动
 - 热启动
 - 端口up/down
 - 厂商自定义trap
- Trap报文


ENTERPRISE
AGENT-ADDRESS
GENERIC-TRAP
SPECIFIC-TRAP
TIME-STAMP
VARIABLE-BINDINGS

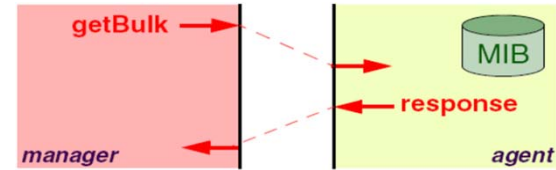
```
SNMP: ----- Simple Network Management Protocol (Version 1) -----
SNMP:
SNMP: SNMP Version = 1
SNMP: Community = public
SNMP: Command = Trap
SNMP: Enterprise = {1.3.6.1.6.3.1.1.5}
SNMP: Network address = [0.0.0.10]
SNMP: Generic trap = 3 (Link up)
SNMP: Specific trap = 0
SNMP: Time ticks = 139691700
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.1.1} (ifIndex.1)
SNMP: Value = 1
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.7.1} (ifAdminStatus.1)
SNMP: Value = 1 (up)
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.8.1} (ifOperStatus.1)
SNMP: Value = 1 (up)
SNMP:
00000000: 00 19 5b 86 95 89 00 0e 5e 00 c3 66 08 00 45 00 ..[悒?^.脛.. E.
00000010: 00 7c 0b 4a 00 00 40 11 45 05 10 09 09 09 10 01 ...|.J..@.E.....
00000020: 01 10 10 d7 00 a2 00 68 cd c0 30 5e 02 01 00 04 ....??h脛..0^....
00000030: 06 70 75 62 6c 69 63 a4 51 06 08 2b 06 01 06 03 ..public...+....
00000040: 01 01 05 40 04 00 00 00 0a 02 01 03 02 01 00 43 ...@.....C.....
00000050: 04 08 53 86 b4 30 33 30 0f 06 0a 2b 06 01 02 01 ...S悒030...+....
00000060: 02 02 01 01 01 02 01 01 30 0f 06 0a 2b 06 01 02 .....0...+....
00000070: 01 02 02 01 07 01 02 01 01 30 0f 06 0a 2b 06 01 .....0...+....
00000080: 02 01 02 02 01 08 01 02 01 01
```

SNMPv1缺陷与SNMPv2

- 没有提供成批存取机制，对大块数据进行存取效率很低；
 - 没有提供足够的安全机制，安全性很差；
 - 有限的告警
 - 有限的错误码
 - 缺乏层次结构
 - 传输层依赖：只在UDP协议上运行，不支持别的网络协议；
 - 没有提供管理者与管理者之间通信的机制，只适合集中式管理，而不利于进行分布式管理；
 - 只适于监测网络设备，不适于监测网络本身。
- v2提供GET-BULK操作，大大提高效率
 - v2在安全性方面没有改进
 - v2在告警方面通过NOTIFICATION-TYPE宏定义
 - 增强了错误码
 - 通过inform trap增强管理站到管理站的功能
 - 支持UDP、CLNS、DDP、IPX
 - v2开始分布式管理的研究，这部分研究被分离出来，转到DISMAN组研究
 - RMON mib增强了对网络本身的监测
 - SMIv2比SMIv1改进了很多

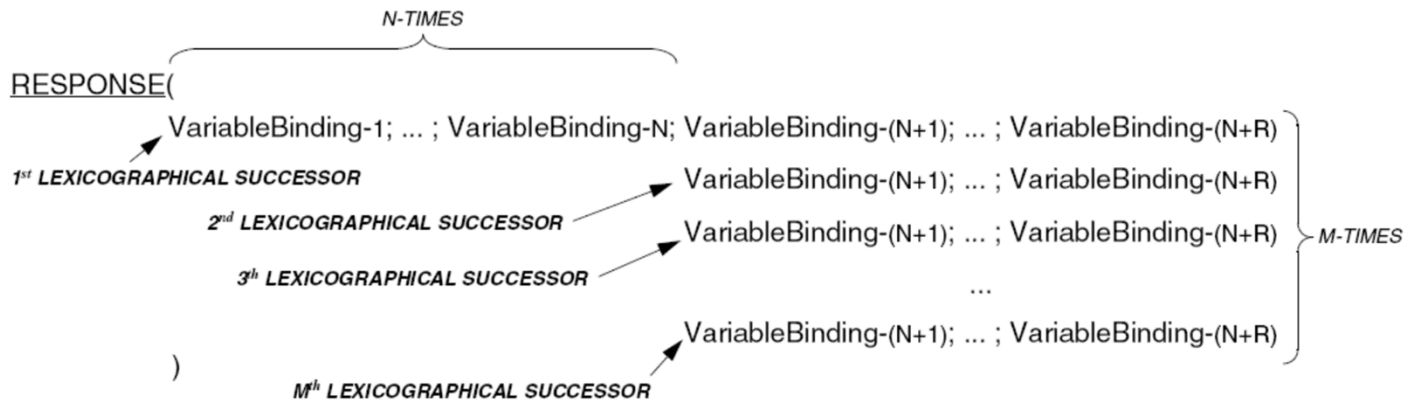
SNMPv2-Get Bulk

- Get Bulk的两个重要变量
 - non-repeaters(N)、max-repetitions(M)
 - 假设GetBulk绑定中有N+R个变量，则前N个按照普通GetNext操作返回，后R个返回每个后面的连续M个变量。
- 



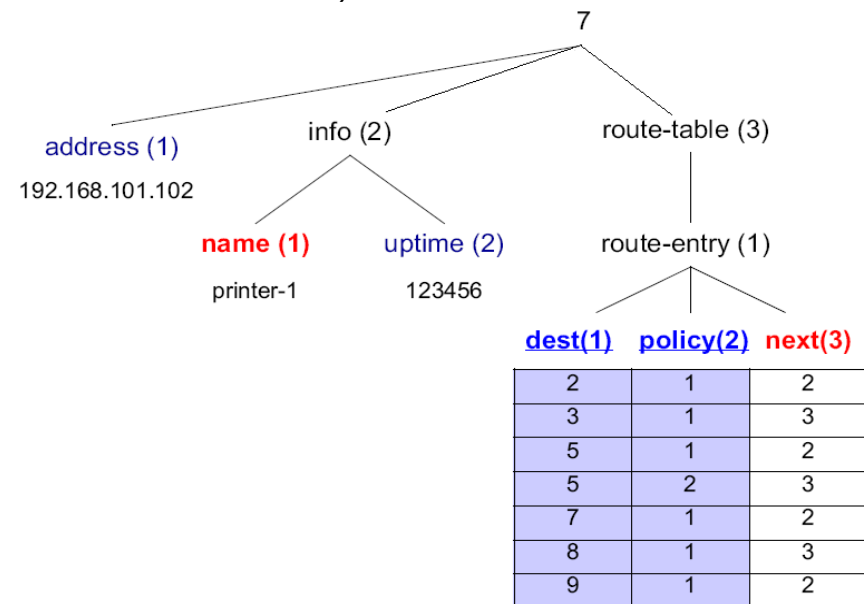
- 示意

```
REQUEST(non-repeaters = N; max-repetitions = M;
        VariableBinding-1; ... ; VariableBinding-N; VariableBinding-(N+1); ... ; VariableBinding-(N+R);
)
```



SNMPv2-Get Bulk举例

- getBulk(non-repeaters=1,max-repetitions=3; 7.2.1.0; 7.3.1.1; 7.3.1.2; 7.3.1.3);
- response(7.2.2.0=>123456;
- 7.3.1.1.2.1=>2; 7.3.1.1.3.1=>3; 7.3.1.1.5.1=>5;
7.3.1.2.2.1=>1; 7.3.1.2.3.1=>1; 7.3.1.2.5.1=>1;
7.3.1.3.2.1=>2; 7.3.1.3.3.1=>3; 7.3.1.3.5.1=>2;)



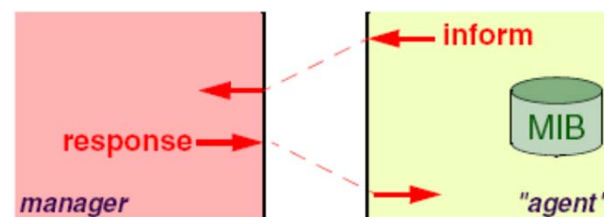
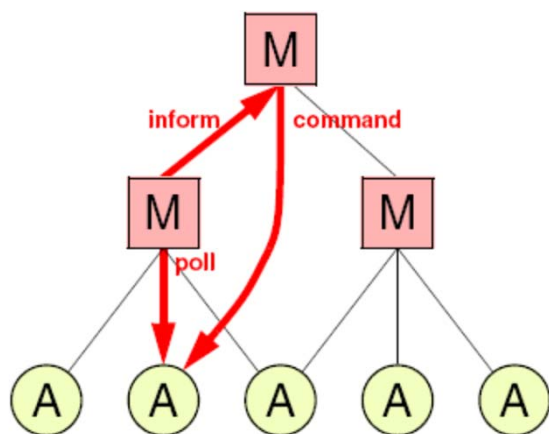
SNMPv2错误码

- 错误码被移到每个变量里面
- GET
 - noSuchObject、noSuchInstance
- GET-NEXT
 - noNextInstance、endOfView
- SET
 - 见右图
- GET-BULK
 - 跟getNext一样

SNMPv1	SNMPv2
badValue	wrongValue
badValue	wrongEncoding
badValue	wrongType
badValue	wrongLength
badValue	inconsistentValue
noSuchName	noAccess
noSuchName	notWritable
noSuchName	noCreation
noSuchName	inconsistentName
genErr	resourceUnavailable
genErr	genErr
genErr	CommitFailed
genErr	undoFailed

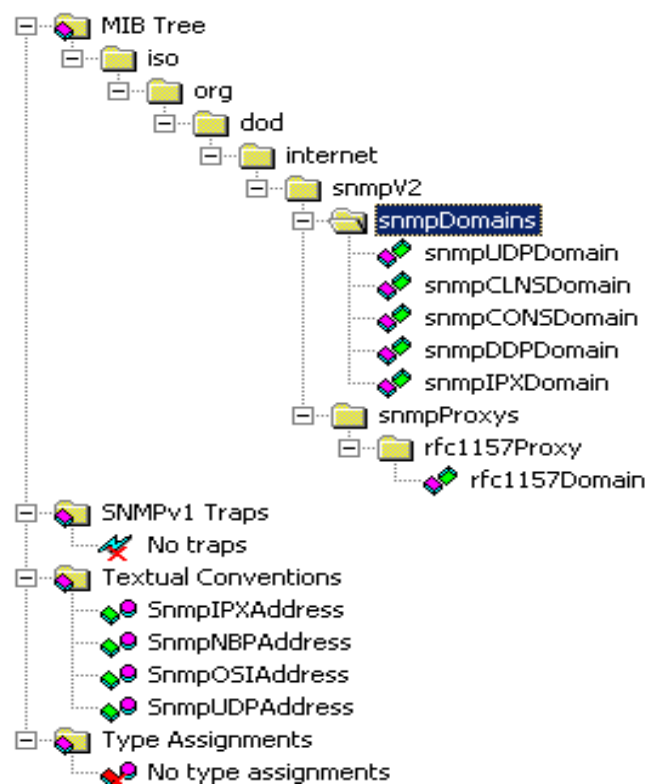
SNMPv2-inform

- 设计初衷：管理站与管理站之间通信
- 实际上report PDU没有统一定义，所以很少使用



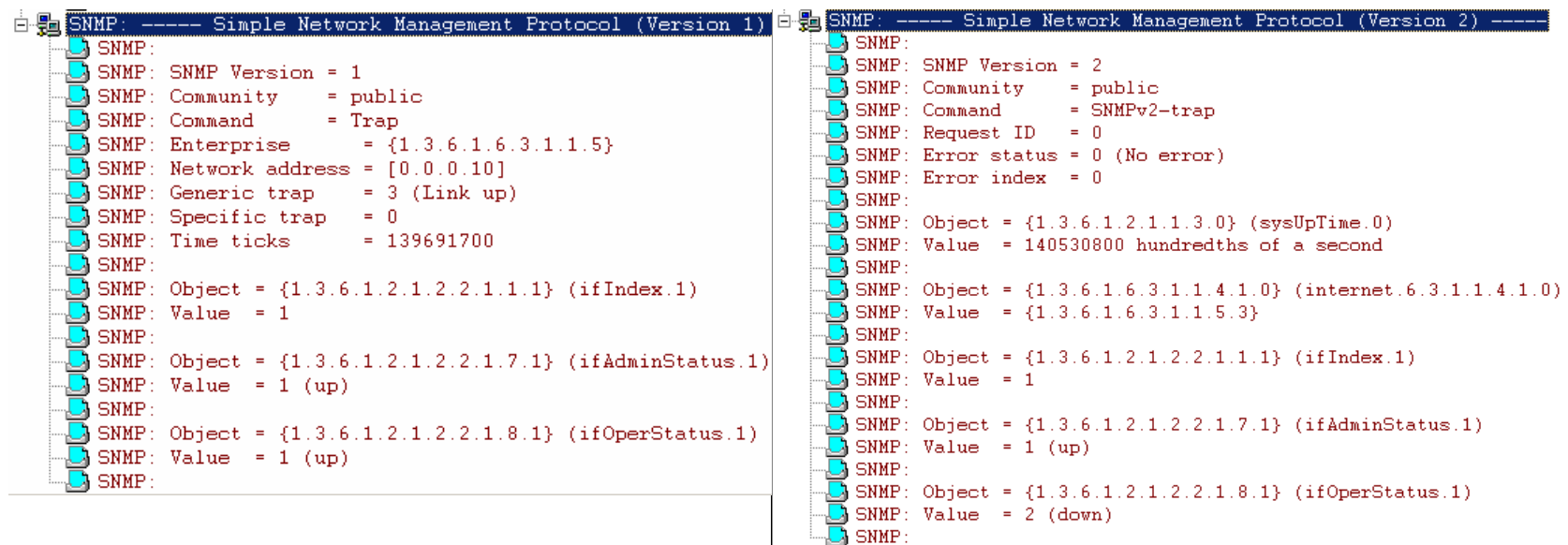
SNMPv2传输层

- 在SNMPv2-TM中定义了多个传输域



SNMPv2报文格式

- TRAP-PDU格式改成与其他pdu一致了，其他未变



The image displays two side-by-side network protocol analyzers showing SNMP trap messages. The left window shows an SNMPv1 message, and the right window shows an SNMPv2 message. Both messages are of type 'Trap' and contain similar fields, but the right message includes additional fields like 'Request ID', 'Error status', and 'Error index'.

```
SNMP: ----- Simple Network Management Protocol (Version 1) -----
SNMP:
SNMP: SNMP Version = 1
SNMP: Community = public
SNMP: Command = Trap
SNMP: Enterprise = {1.3.6.1.6.3.1.1.5}
SNMP: Network address = [0.0.0.10]
SNMP: Generic trap = 3 (Link up)
SNMP: Specific trap = 0
SNMP: Time ticks = 139691700
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.1.1} (ifIndex.1)
SNMP: Value = 1
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.7.1} (ifAdminStatus.1)
SNMP: Value = 1 (up)
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.8.1} (ifOperStatus.1)
SNMP: Value = 1 (up)
SNMP:

SNMP: ----- Simple Network Management Protocol (Version 2) -----
SNMP:
SNMP: SNMP Version = 2
SNMP: Community = public
SNMP: Command = SNMPv2-trap
SNMP: Request ID = 0
SNMP: Error status = 0 (No error)
SNMP: Error index = 0
SNMP:
SNMP: Object = {1.3.6.1.2.1.1.3.0} (sysUpTime.0)
SNMP: Value = 140530800 hundredths of a second
SNMP:
SNMP: Object = {1.3.6.1.6.3.1.1.4.1.0} (internet.6.3.1.1.4.1.0)
SNMP: Value = {1.3.6.1.6.3.1.1.5.3}
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.1.1} (ifIndex.1)
SNMP: Value = 1
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.7.1} (ifAdminStatus.1)
SNMP: Value = 1 (up)
SNMP:
SNMP: Object = {1.3.6.1.2.1.2.2.1.8.1} (ifOperStatus.1)
SNMP: Value = 2 (down)
SNMP:
```

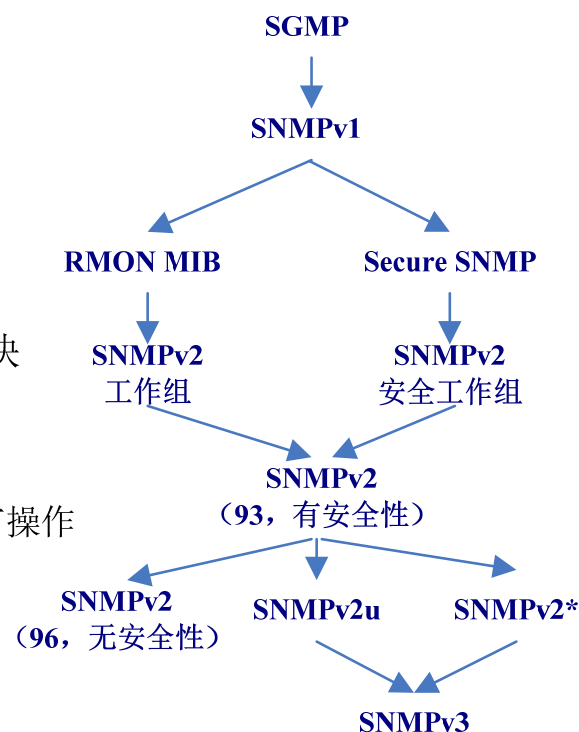
Why SNMPv3

- SNMP的发展
- SNMPv3设计目标
 - 满足安全设置的需要
 - 定义一种体系结构以有利于SNMP长期发展
 - 允许体系结构中不同部分的标准化速度不同
 - 支持将来的扩展
 - 支持SNMP的最小实现
 - 尽量保持SNMP的简单性
 - 尽可能地利用已有的规范
 - 支持大网络中需要的更复杂的特征
- SNMPv3等于SNMPv2加上安全和管理

USM和VACM

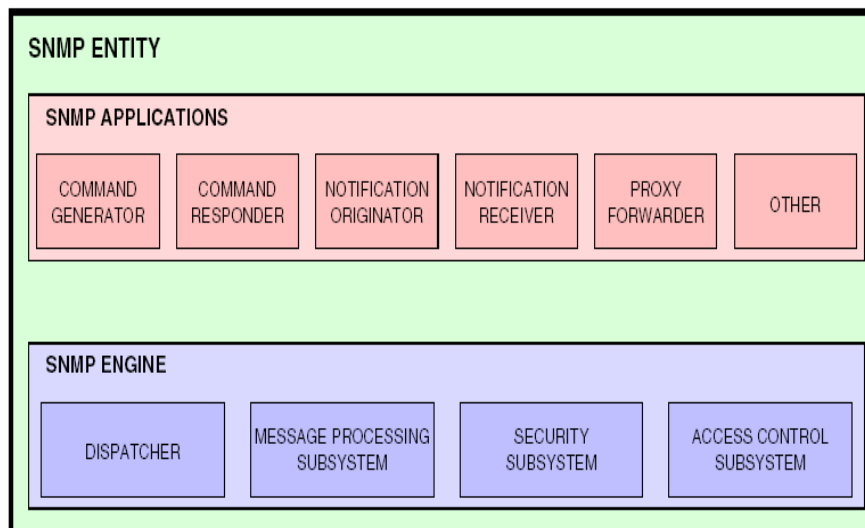
模块化体系结构；
并且定义了各模块
之间的原语

保持SNMPv2原有操作



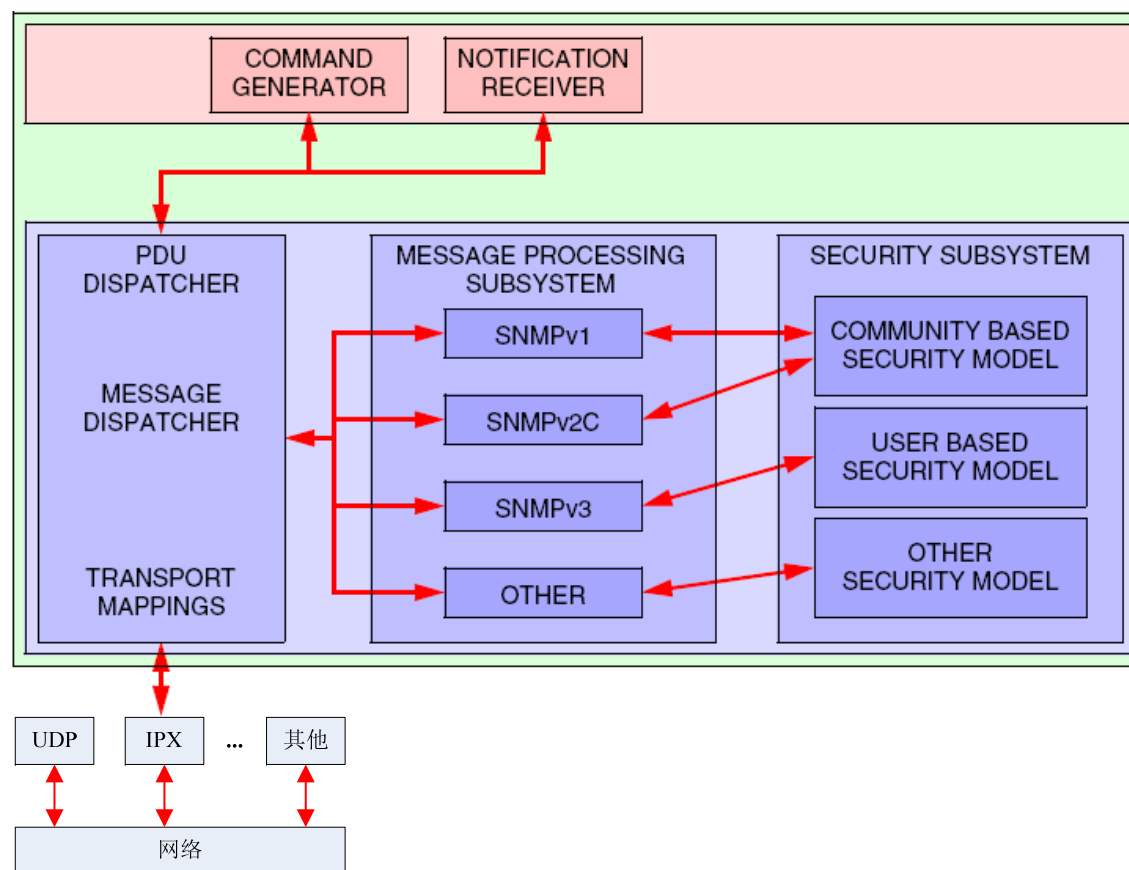
SNMPv3体系结构

- SNMP实体
 - SNMP应用
 - 命令发生、接收器
 - 通告发生、接收器
 - 代理及其他
 - SNMP引擎
 - PDU分配
 - 消息处理
 - 安全子系统
 - 访问控制子系统

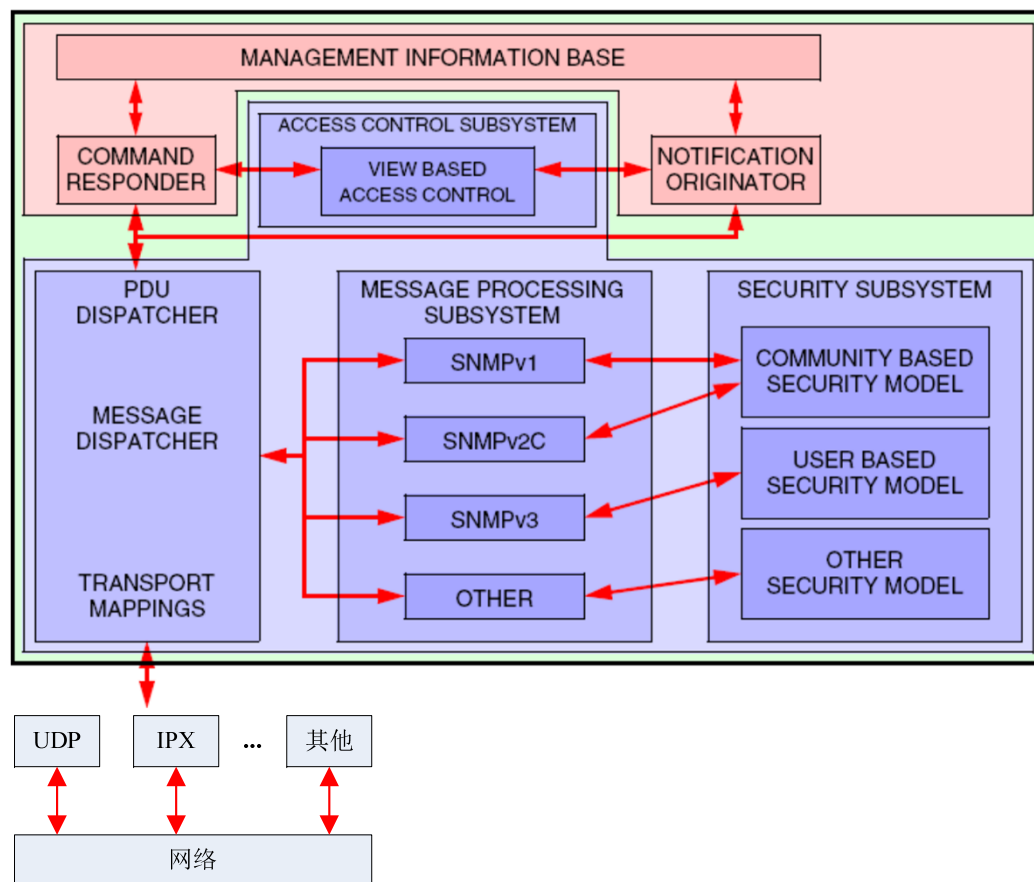


- snmpEngineID
 - 标识SNMP引擎，是SNMP引擎以及该引擎相对应的SNMP实体的唯一且无歧义标识符。
 - 取值方式
 - 手动配置
 - 自动算法：目前公司采用0x800022B603+mac地址

典型SNMP管理站



典型SNMP代理



SNMP受到的安全威胁

- 主要威胁

- 信息修改： 是指一些非授权实体可能改变由另一个授权实体产生的传输中的消息，包括伪造一个对象的值。
- 伪装： 是指一个实体伪装成另一个授权实体来进行它无权进行的操作。

- 第二层次的威胁

- 泄密： 是指窃听代理和管理站之间的数据交换。
- 消息流修改： 消息流修改是指消息可能被恶意地重排、延迟或者重发。

- 第三层次的威胁

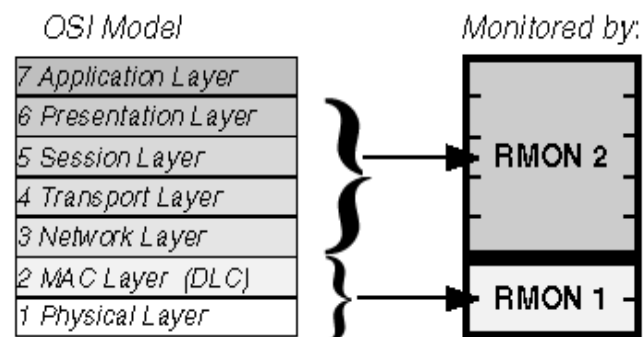
- 拒绝服务： 攻击者阻止管理站和代理之间的信息交换。
- 流量分析： 攻击者观测管理站和代理之间流量的一般模式。

RMON

- 为何要RMON?
 - 流量分析：SNMP要不断轮询，而轮询负担重
- RMON代理的实现方式
 - 分布式RMON
 - 使用一个具有完整的软硬件系统的独立设备作为远程监视器（RMON探测器）。
 - 工作在“混杂”模式，查看分析LAN的每一个包。
 - 嵌入式RMON
 - 嵌入到网络中的关键设备如集线器、交换机中。
 - 一般只实现RMON的基本四组（统计、历史、警报和事件组）功能

RMON (二)

- RMON1与RMON2
- RMON的四个基本组



	功能	元素
统计量	包括探测器为该设备每个监控的接口测量的统计值。	数据包丢弃、数据包发送、广播数据包、CRC错误、大小块、冲突以及计数器的数据包。范围从64~128、128~256、256~512、512~1024以及1024~1518字节。
历史	定期地收集统计网络值地记录并存储起来以便日后提取。	取样周期、样品数目和项目。提供有关网段流量、错误包、广播包、利用率以及碰撞次数等其他统计信息的历史数据。
告警	定期从探测器的变量选取统计例子。并与前面配的阈值相比较。	告警类型、间隔、阈值上限、阈值下限
事件	控制在此处事件的产生和报告。	事件类型、描述、事件最后一个发送的时间

Mib Browser

- 工具软件
 - MG MibBrowser (更广泛使用)
 - AdventNet Web NMS (界面粗糙, 但编译检查严格)
- 主要演示操作
 - SNMPv2、v3建立连接
 - Get、Set、GetNext
 - TableView: GetBulk实现
 - Walk
 - 多变量绑定Set/Get
 - SNMPv2和SNMPv3 Trap接收设置
 - Mib 编译

[返回](#)