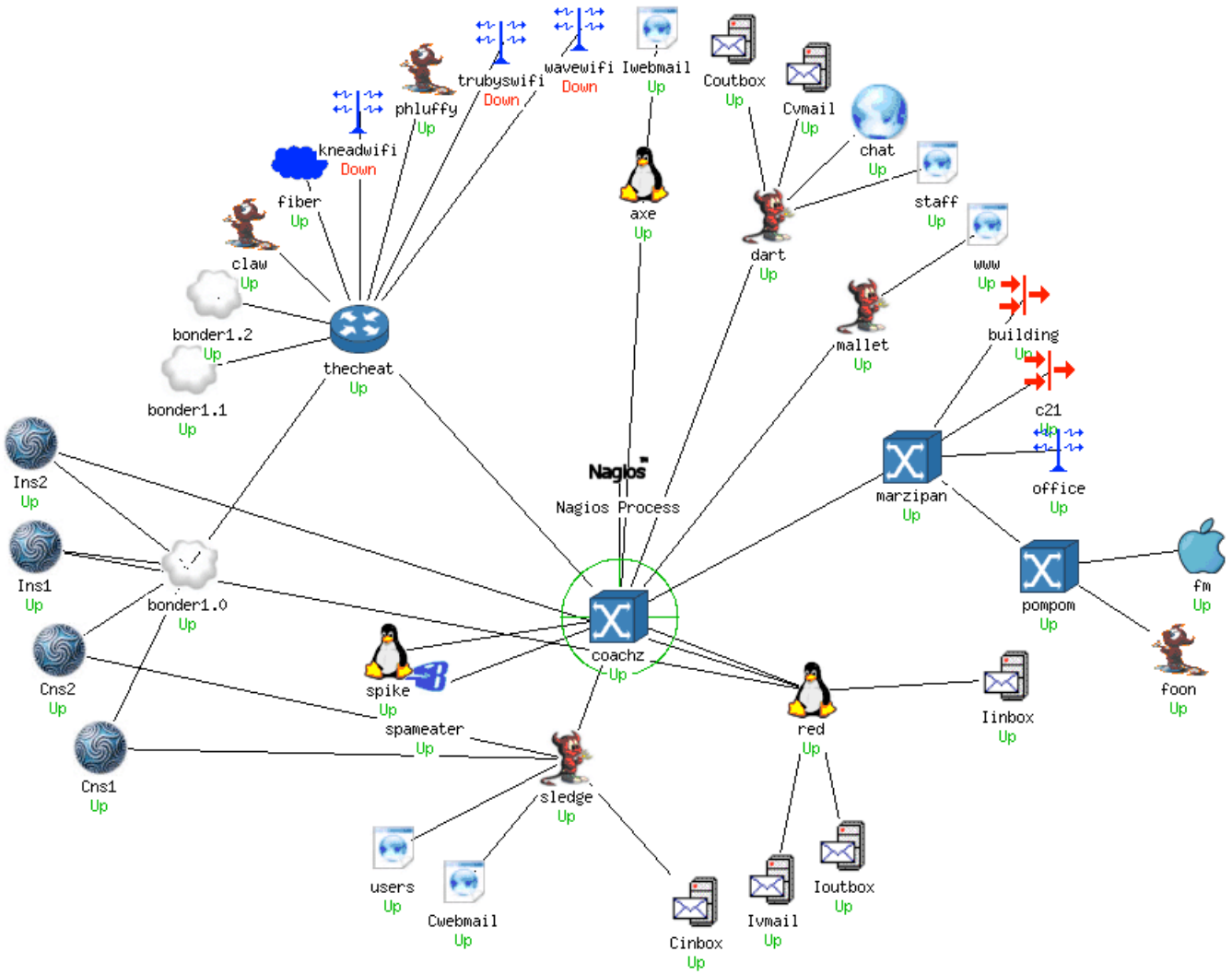
# Administração de Redes 2019/20

Simple Network Management Protocol (SNMP)

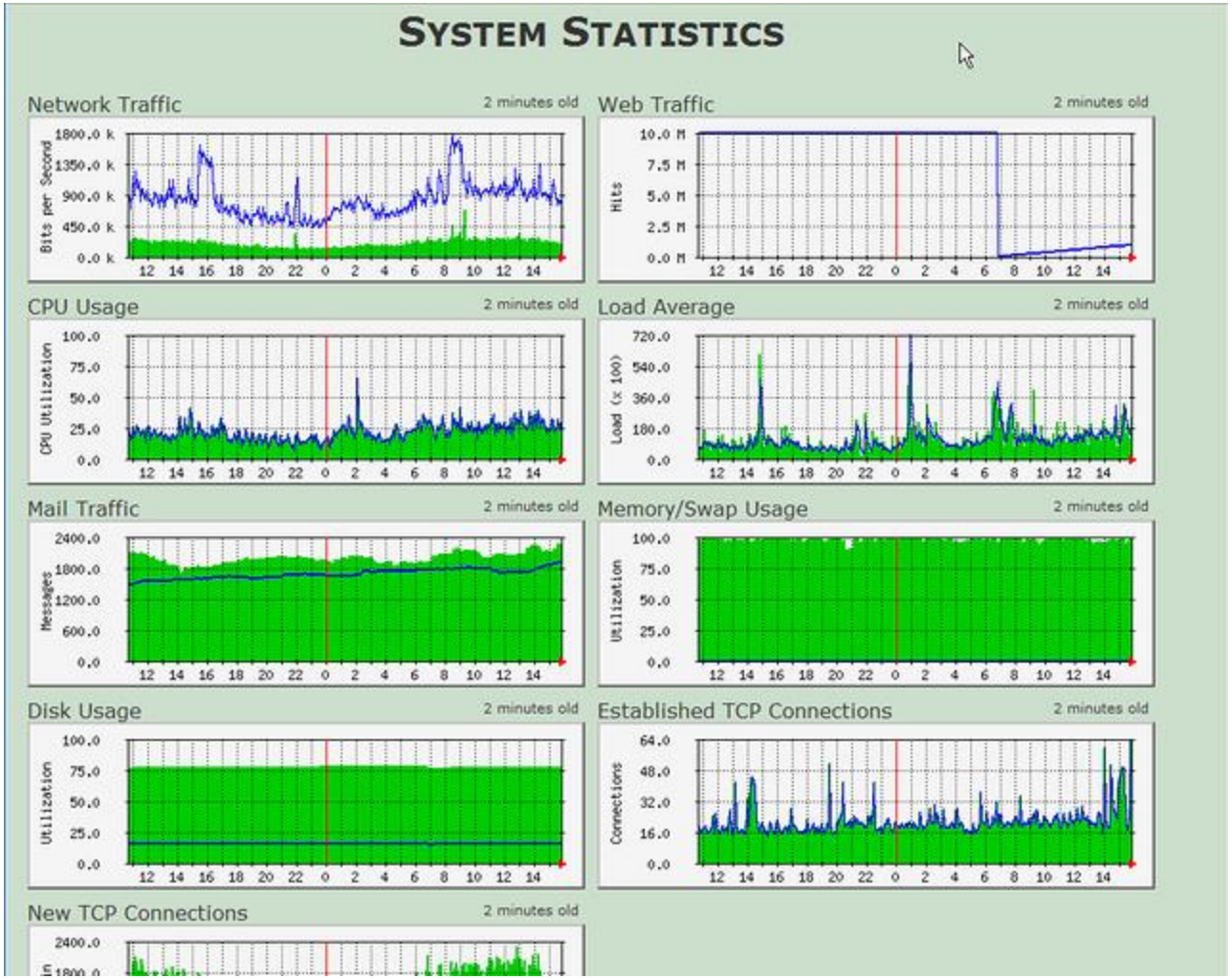
# Introdução

- Redes de grandes dimensões têm demasiados componentes para que a sua gestão possa fazer-se apenas com intervenção humana
- Localização dispersa ➔ necessárias ferramentas para monitorização e configuração remotas
- Muitos tipos diferentes de equipamento para gerir, de diferentes fabricantes
  - Terminais, servidores, impressoras, *routers*, comutadores, …
- Toda esta diversidade dificulta o desenvolvimento de ferramentas integradas de gestão
- São necessários *standards* que
  - Definam procedimentos genéricos para monitorização e configuração
  - Sejam suficientemente flexíveis para poder gerir qualquer dispositivo
- SNMP é o standard *de facto* para a gestão em redes IP

# Monitorização de equipamento



# Obtenção de estatísticas



# Registo e análise de eventos

| | | | | | | |
|---|---|---|---|---|---|---|
| Mon May 7 15:37:00 2012 | .1.3.6.1.6.3.1.1.5.1 | 169.144.105.9 | Status Events | Normal | A coldStart trap signifies that the SNMP entity, |
| Mon May 7 15:36:30 2012 | .1.3.6.1.6.3.1.1.5.1 | 169.144.105.8 | Status Events | Normal | A coldStart trap signifies that the SNMP entity, |
| Wed Dec 28 16:20:124 2011 | .1.3.6.1.4.1.2343.13.990.0.17 | 192.168.5.49 | Other Event | ok | Do not forget to eat. |
| Wed Dec 28 16:20:123 2011 | .1.3.6.1.4.1.9996.13.991.3.4 | 192.168.5.51 | Other Event | warning | Chassis fan is dead. |
| Wed Dec 28 16:20:122 2011 | .1.3.6.1.4.1.2021.45.33.5.6 | 192.168.5.50 | Status Event | normal | Another sample message |
| Wed Dec 28 16:20:121 2011 | .1.3.6.1.4.1.2343.2.993.1.17 | 192.168.5.48 | Closure | warning | Out of pizza, send backup. |
| Wed Dec 28 16:20:120 2011 | .1.3.6.1.4.1.6879.13.991.3.4 | 192.168.5.48 | Other Event | ok | Coldstart detected. |
| Wed Dec 28 16:20:119 2011 | .1.3.6.1.4.1.6879.45.33.5.6 | 192.168.5.48 | Status Event | warning | Smoke alarm detected. |
| Wed Dec 28 16:20:118 2011 | .1.3.6.1.4.1.6879.45.33.5.6 | 192.168.5.54 | Status Event | normal | Coldstart detected. |
| Wed Dec 28 16:20:117 2011 | .1.3.6.1.4.1.5675.2.993.1.17 | 192.168.5.52 | Status Event | ok | Smoke alarm detected. |
| Wed Dec 28 16:20:116 2011 | .1.3.6.1.4.1.6879.2.993.1.17 | 192.168.5.50 | Other Event | normal | Doggon it, people like me. |
| Wed Dec 28 16:20:115 2011 | .1.3.6.1.4.1.5675.45.33.5.6 | 192.168.5.47 | Status Event | critical | Oh no the fire hydrant blew up |
| Wed Dec 28 16:20:114 2011 | .1.3.6.1.4.1.2021.13.990.0.17 | 192.168.5.45 | Status Event | warning | Do not forget to eat. |
| Wed Dec 28 16:20:113 2011 | .1.3.6.1.4.1.6879.45.33.5.6 | 192.168.5.46 | Closure | normal | Oh no the fire hydrant blew up |
| Wed Dec 28 16:20:112 2011 | .1.3.6.1.4.1.9996.45.33.5.6 | 192.168.5.52 | Closure | critical | Coldstart detected. |
| Wed Dec 28 16:20:111 2011 | .1.3.6.1.4.1.5675.45.33.5.6 | 192.168.5.44 | Status Event | normal | Oh no the fire hydrant blew up |
| Wed Dec 28 16:20:110 2011 | .1.3.6.1.4.1.2021.2.993.1.17 | 192.168.5.48 | Status Event | ok | Another sample message |
| Wed Dec 28 16:20:109 2011 | .1.3.6.1.4.1.9996.13.991.3.4 | 192.168.5.52 | Other Event | ok | Doggon it, people like me. |

# Simple Network Management Protocol

- Conjunto de tecnologias para administrar remotamente dispositivos em redes TCP/IP
- Dois significados para SNMP
  - Estrutura de gestão SNMP
    - *Internet Standard Management Framework*
  - O protocolo usado para troca de informação de gestão entre dispositivos
- Objectivos do SNMP
  - Forma universal de definir informação de gestão
  - Separação de funções entre o *software* de gestão e o protocolo usado para trocar informação de gestão
  - Simplicidade
    - Número limitado de operações, fáceis de compreender
    - Facilidade de implementação para os fabricantes de equipamentos

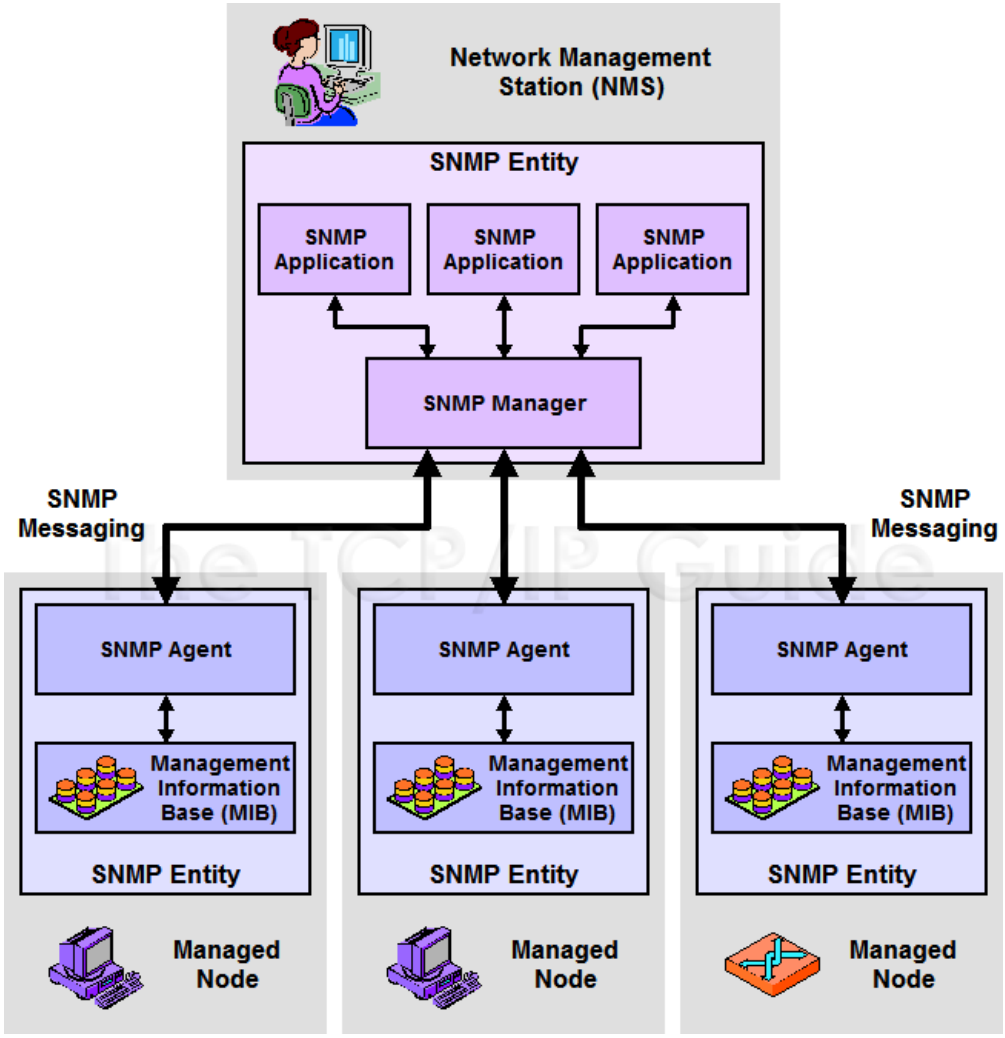
# Versões do protocolo SNMP

- SNMPv1 tinha limitações significativas, nomeadamente a nível de
  - Segurança
  - Fiabilidade nas notificações assíncronas
- SNMPv2 pretendia colmatar essas limitações; contudo
  - Muitas variantes e demasiada complexidade
  - Apenas a versão 2c *(community-based)* teve algum sucesso
    - Resolve o problema de fiabilidade
    - A nível de segurança é semelhante ao SNMPv1
- SNMPv3 veio resolver a confusão do SNMPv2
  - Incorpora diversos modelos de segurança
  - Suporta
    - Autenticação segura
    - Integridade das mensagens
    - Privacidade

# Tipos de dispositivos

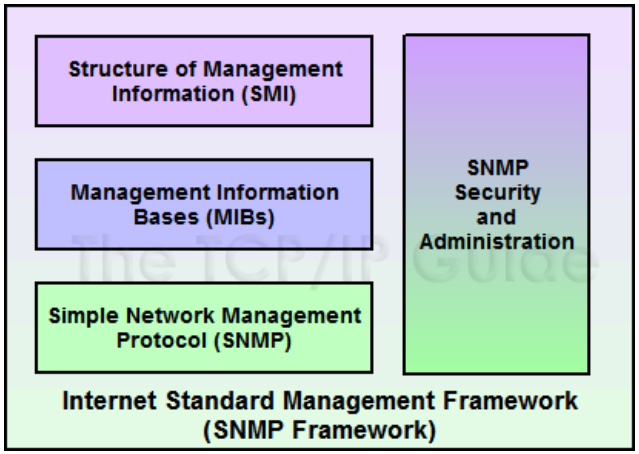
- O SNMP define dois tipos básicos de dispositivos
  - Dispositivos geridos
    - Contêm objectos geridos organizados numa base de dados com estrutura em árvore, designada MIB *(Management Information Base)*
    - Contêm um agente SNMP
      - *Software* que implementa o protocolo e comunica com a estação de gestão
  - Estações de gestão
    - Dispositivos usados para gerir os dispositivos geridos
    - Contêm um gestor SNMP
      - *Software* que implementa o protocolo e comunica com os dispositivos geridos
    - Aplicações SNMP usadas pelo administrador para gerir a rede
- Existe um terceiro tipo de dispositivo, o *proxy*
  - Usado para possibilitar a gestão por SNMP de dispositivos que não o suportam

# Visão geral da gestão SNMP



# Componentes da estrutura de gestão SNMP

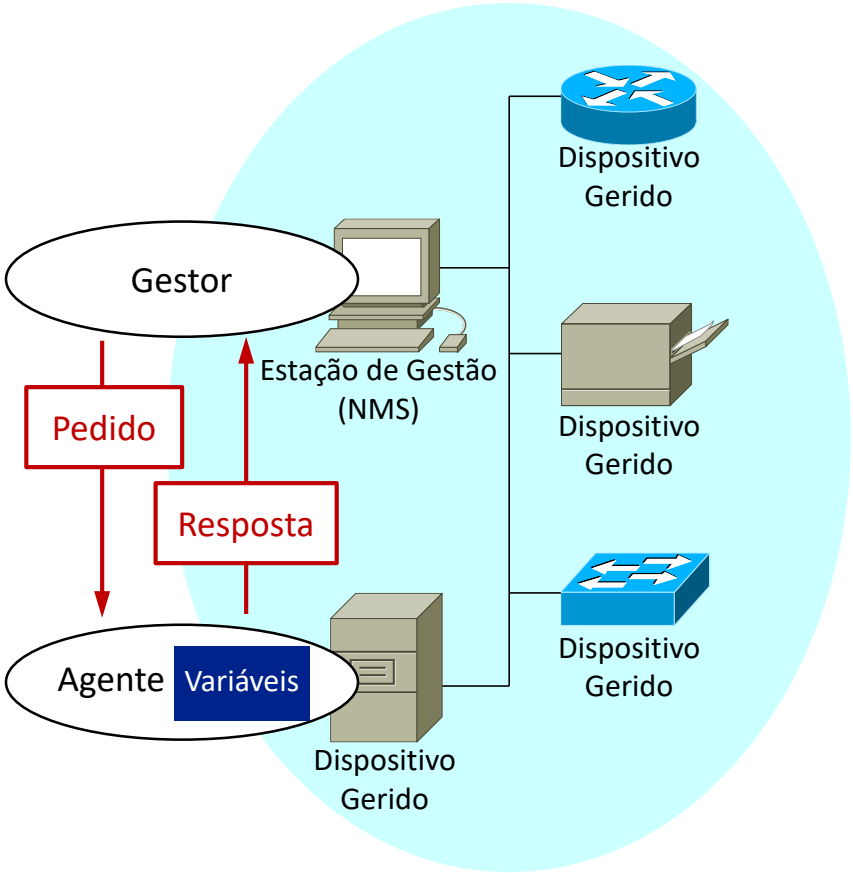
- Structure of Management Information (SMI)
  - Linguagem para definir de forma consistente os objectos geridos
  - Define estrutura, sintaxe e características da informação de gestão
- Management Information Base (MIB)
  - Base de dados de objectos geridos, descrita usando SMI
  - Estrutura em árvore
    - Sub-árvores definidas em diferentes módulos MIB
- Protocolo SNMP
  - Define as mensagens e como as utilizar
- Segurança e Administração
  - Aspectos de segurança
  - Questões de implementação
  - Transição entre versões
  - …



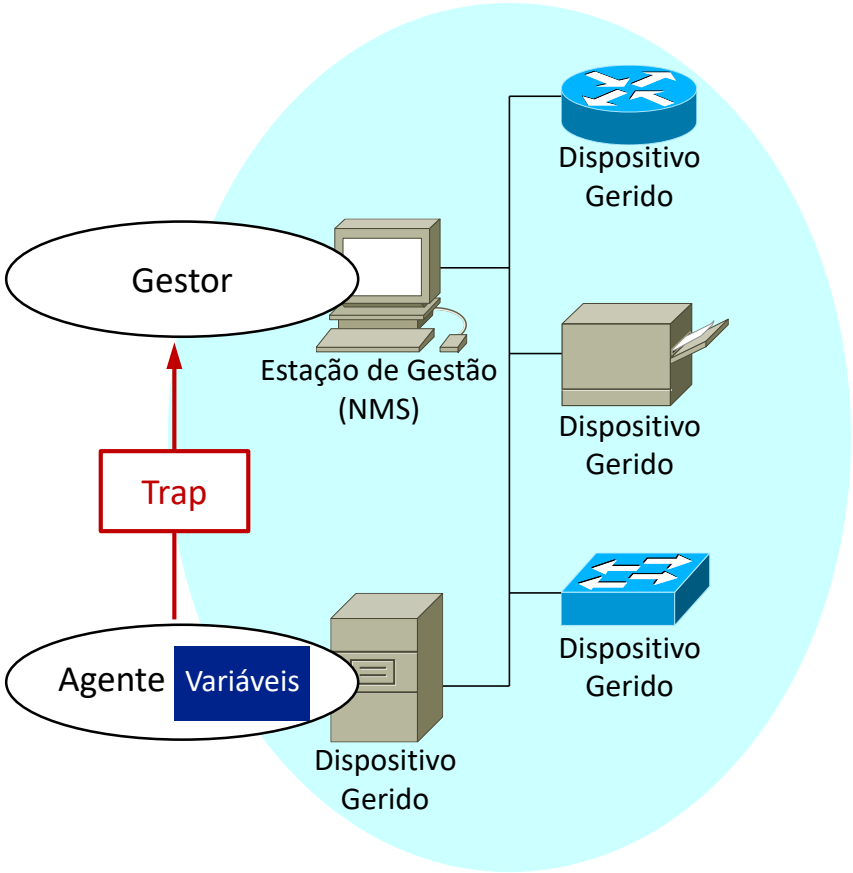
# Concepção orientada à informação

- O SNMP implementa um modelo de gestão orientado à informação
  - Um modelo orientado a comandos seria problemático
    - Dispositivos diferentes necessitam de comandos diferentes
    - Suporte de novos dispositivos obrigaria a alterações ao protocolo
- MIB contém variáveis (objectos) que podem ser lidas e/ou escritas
  - Modelo simples e muito flexível
  - Execução de comandos através da escrita em variáveis. Exemplos:
    - Alteração do endereço IP de uma interface através da escrita numa variável para o endereço IP dessa interface
    - Colocação de um dispositivo em modo teste através da escrita numa variável representando o modo corrente
    - Ordem de *reboot* através de uma variável representando o tempo até ao próximo *reboot* — a escrita nessa variável inicia uma contagem decrescente
- Além da leitura e escrita de variáveis, suporta também notificações assíncronas (*traps*)

# Transferência de informação: Polling vs. Traps

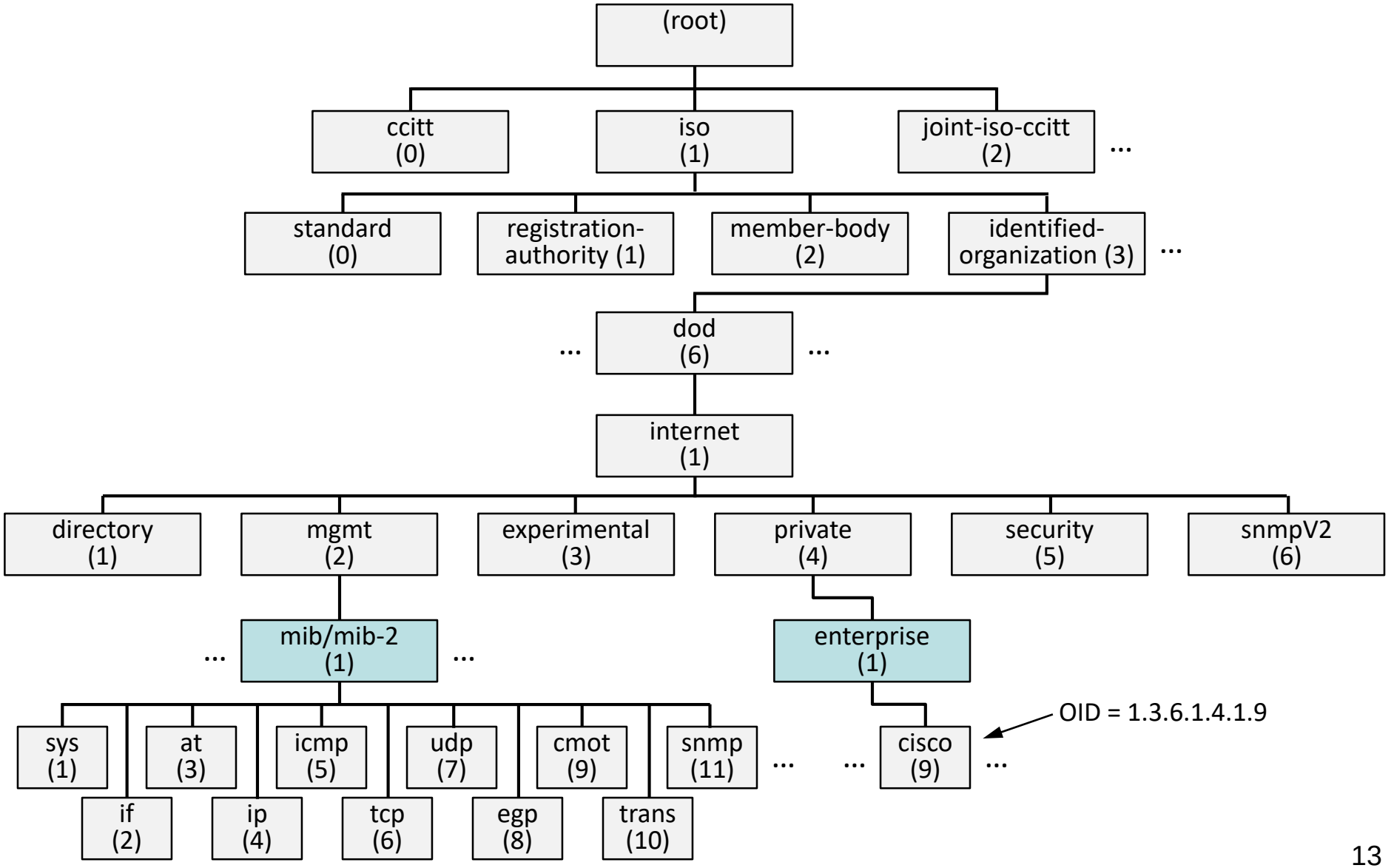



Polling
(Pedido/Resposta)




Traps
(Notificações Assíncronas)

# Organização dos objectos na MIB



# Organização da MIB

- SNMPv1 definia toda a MIB
- Versões posteriores organizadas em módulos
  - Módulos MIB correspondem a sub-árvores na hierarquia de objectos
    - Contêm conjuntos de objectos relacionados
  - Extensível: é possível definir novos módulos
  - Flexível: dispositivos suportam apenas os módulos que fazem sentido
- SMI usada para definir os objectos nos módulos e a sua localização na hierarquia

# Abstract Syntax Notation 1 (ASN.1)

- Linguagem formal para descrição de dados
- Define tipos de dados básicos
  - E.g., BOOLEAN, INTEGER, OCTET STRING, …
- Permite definir tipos de dados derivados
  - E.g., SEQUENCE (sequência de valores de tipos diferentes), SEQUENCE OF (sequência de valores do mesmo tipo), …
- A SMI, usada na definição da MIB, baseia-se no ASN.1
  - Alguns tipos definidos pela SMI: IpAddress, Counter32, TimeTicks, …

- ASN.1 define também formas não ambíguas de codificar, transmitir e descodificar dados de tipos definidos
- As mensagens SNMP são codificadas usando ASN.1 BER *(Basic Encoding Rules)*
  - Codificação TLV (Type/Length/Value)

# Definição de objectos na MIB

- Características mandatórias
  - Nome do objecto, humanamente entendível (e.g., `sysDescr`)
  - Sintaxe (tipo do objecto)
  - Acesso (acesso máximo na SMIv2)
    - read-create, read-write, read-only, accessible-for-notify, not-accessible
  - Status
    - mandatory, optional, obsolete (SMIv1); current, obsolete, deprecated (SMIv2)
  - Definição (descrição textual do significado do objecto)
  - Localização (relativa) na hierarquia de objectos
- Características opcionais
  - Unidades, referência, índice (para tabelas), aumenta (alternativa ao índice), valor-padrão
- Convenções textuais permitem redefinir tipos para lhe dar semântica mais específica
  - TimeInterval é um INTEGER, mas com o significado preciso de "Período de tempo medido em centésimos de segundo"

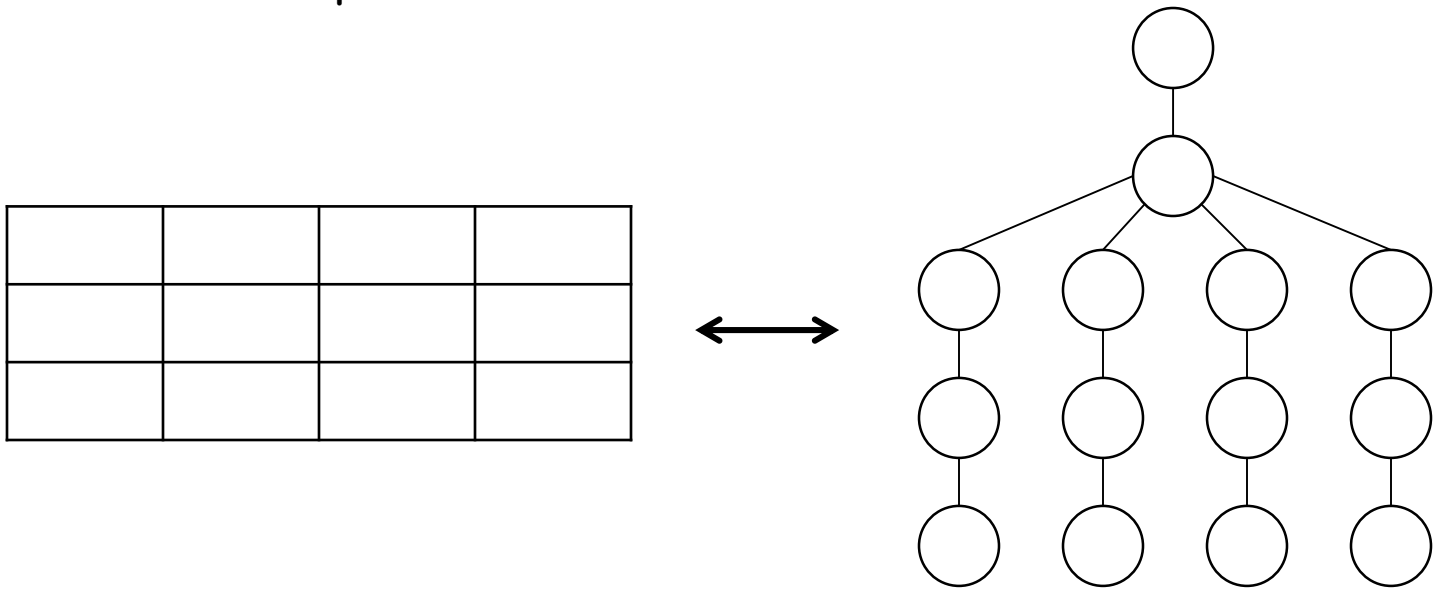
# Definição de objectos na MIB

- Exemplo de objecto (sysLocation) definido em SMIv2:

```
sysLocation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
STATUS current
DESCRIPTION "The physical location of this node (e.g., 'telephone
closet, 3rd floor'). If the location is unknown, the value is the zero-
length string."
::= { system 6 }
```

# Tabelas

- Alguma informação é inerentemente tabular, e.g., tabela de encaminhamento
- É possível mapear tabelas na estrutura em árvore
  - Necessário definir parte da MIB como tabular
  - Mapeamento da linha e coluna da tabela no Object ID (OID) que contém a respectiva entrada



# Tabelas — Exemplo

```
-- The IP routing table contains an entry for each route
-- presently known to this entity.
ipRouteTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF IpRouteEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
              "This entity's IP Routing table."
    ::= { ip 21 }
```

Com maiúscula (tipo definido no slide seguinte)

```
ipRouteEntry OBJECT-TYPE
    SYNTAX  IpRouteEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
            "A route to a particular destination."
    INDEX   { ipRouteDest }
    ::= { ipRouteTable 1 }
```

Coluna a usar como índice

# Tabelas — Exemplo

Índice (pode não ser a primeira coluna)

```
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
    }
```

# Tabelas — Exemplo

- Note-se que a coluna pode ser sempre identificada por um número — o número da coluna
- A linha é identificada pelo índice, que pode ter mais de um número — neste caso é um prefixo IP, que tem quatro
- No mapeamento para o OID, primeiro aparece a coluna e depois a linha
- E.g., a *default gateway* (próximo salto para o prefixo 0.0.0.0) estará em

`ip.ipRouteTable.ipRouteEntry.ipRouteNextHop.0.0.0.0`

Coluna

Linha (índice)

# Tipos de PDU*

| Classe de PDU (SNMPv3) | Descrição | PDU SNMPv1 | PDU SNMPv2/3 |
|---|---|---|---|
| Read | Mensagens para ler informação de gestão de um dispositivo gerido usando o mecanismo de *polling* | GetRequest<br>GetNextRequest | GetRequest<br>GetNextRequest<br>GetBulkRequest |
| Write | Mensagens que alteram a informação de gestão de um dispositivo gerido | SetRequest | SetRequest |
| Response | Mensagens enviadas em resposta a um pedido | GetResponse | Response |
| Notification | Mensagens usadas para enviar uma notificação assíncrona a um gestor | Trap | Trapv2<br>InformRequest |

* PDU = Protocol Data Unit

# Tipos de PDU

- GetRequest
  - Serve para pedir o valor de uma ou mais variáveis (instâncias)
- GetNextRequest
  - Serve para pedir o valor das instâncias de variáveis cujos OID são imediatamente seguintes em ordem lexicográfica aos OID indicados
  - Usada para fazer o *walk* da MIB
    - Percorrer por ordem lexicográfica a sub-árvore debaixo de um dado OID
- GetBulkRequest (v2/3)
  - Especificados $N+M$ OIDS, e indicando que são $N$ *non-repeaters* e $R$ *max-repetitions*, obtém até $N + (M \times R)$ variáveis como se fosse
    - Um GetNextRequest para cada um dos $N$ primeiros OID
    - $R$ GetNextRequests sequenciais para cada um dos $M$ OID seguintes
  - Usada para acelerar a obtenção de tabelas e o *walk*

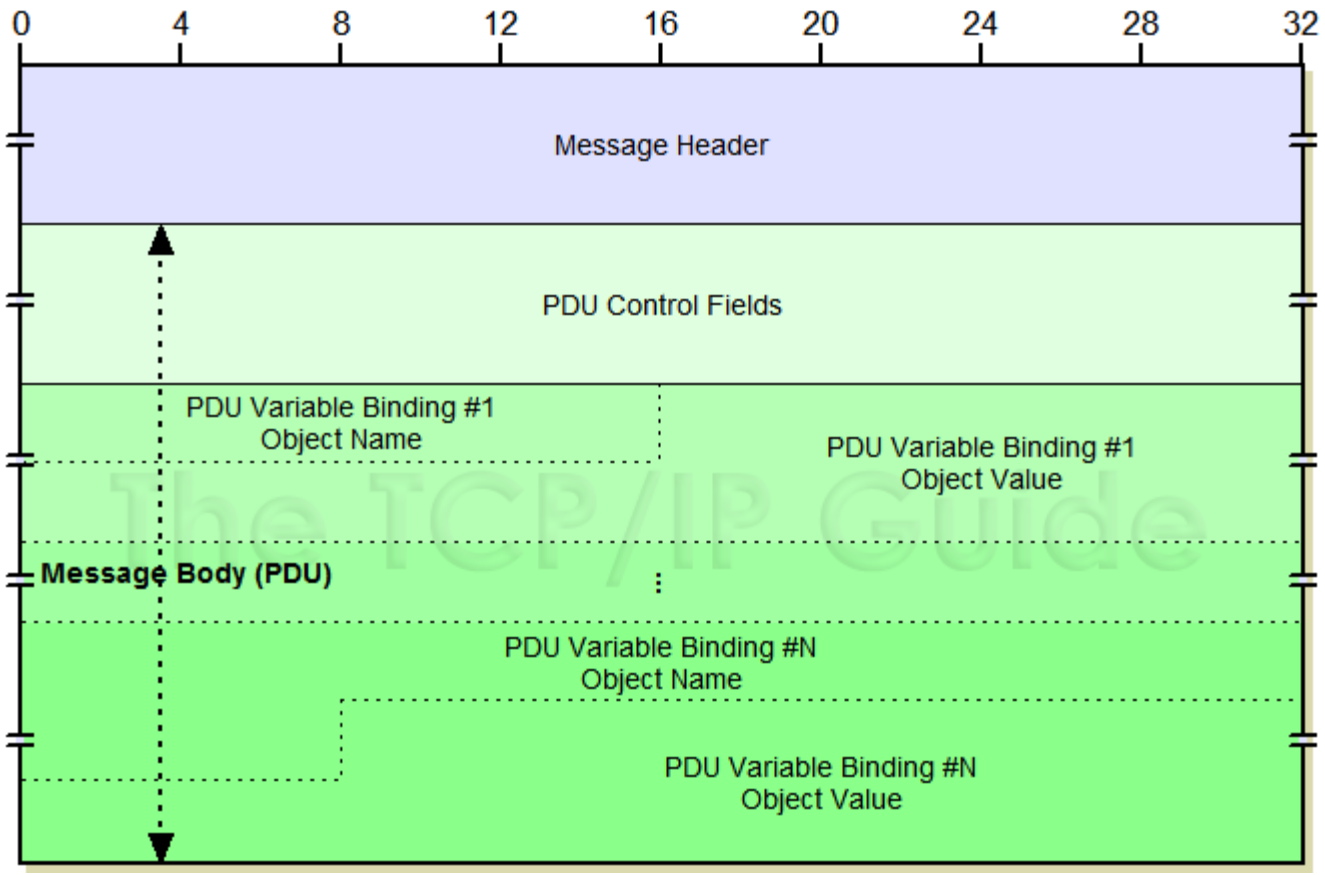
# Tipos de PDU

- SetRequest
  - Serve para modificar o valor de uma ou mais variáveis (instâncias)
- Response
  - Serve para responder a um pedido
- Trap
  - Serve para notificar (de forma assíncrona / não solicitada) o gestor da ocorrência de um dado evento
    - Em reacção à recepção de uma *trap*, o gestor pode fazer pedidos para obter mais variáveis que lhe dêem informação adicional
  - Mensagem não confirmada ➔ entrega não-fiável
- InformRequest
  - Semelhante à *trap*, mas com recepção confirmada / retransmissão
  - Concebida para comunicação entre gestores, mas pode também ser enviada por um agente a um gestor

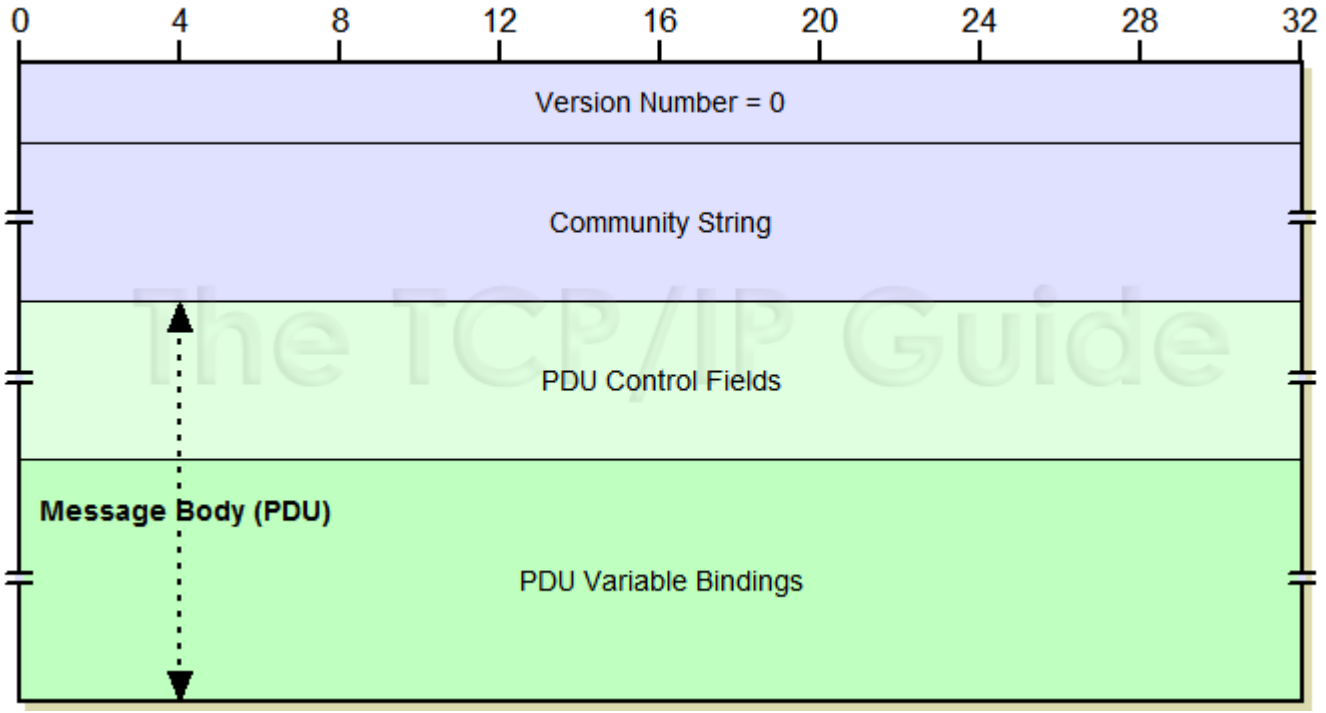
# Formato das mensagens

- Normalmente sobre UDP/IP, mas desde a v2 que existem *transport mappings* para outras pilhas protocolares
- Portas
  - 161 para a generalidade das mensagens, 162 para receber *traps*

# Formato geral das mensagens SNMP

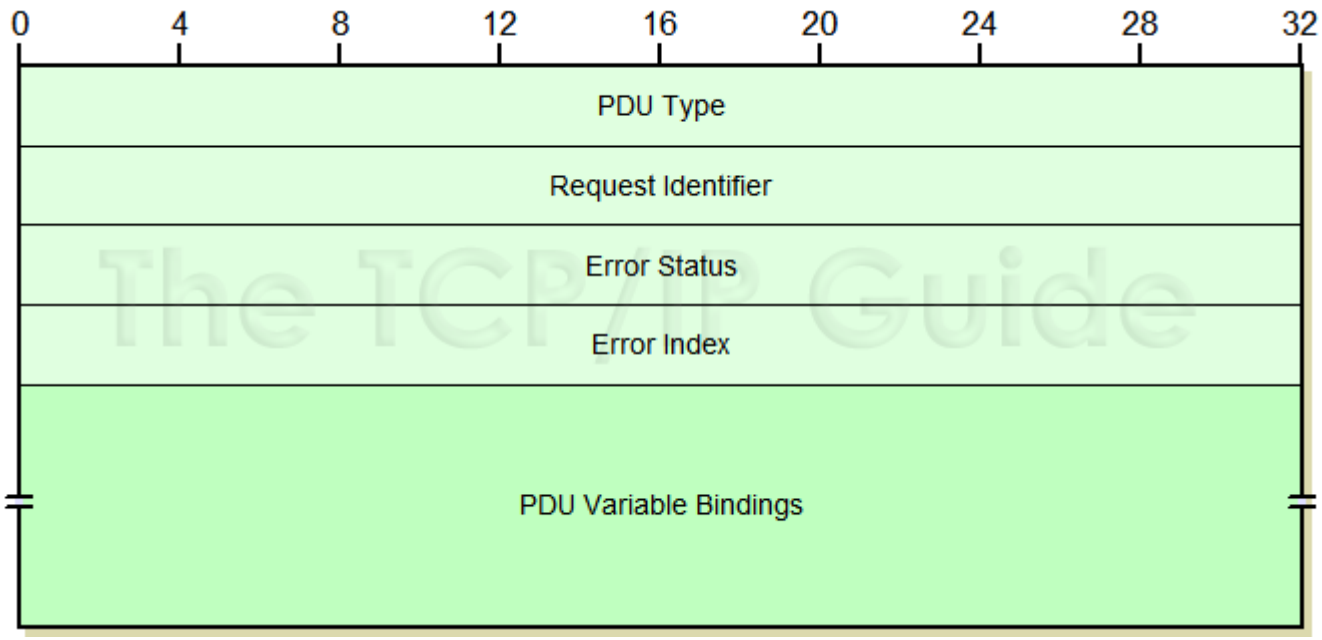


# Formato geral das mensagens SNMPv1




- Mensagens SNMPv2c diferem apenas no número da versão (1)

# Formato geral das PDU SNMP



# Exemplo de mensagem SNMPv1

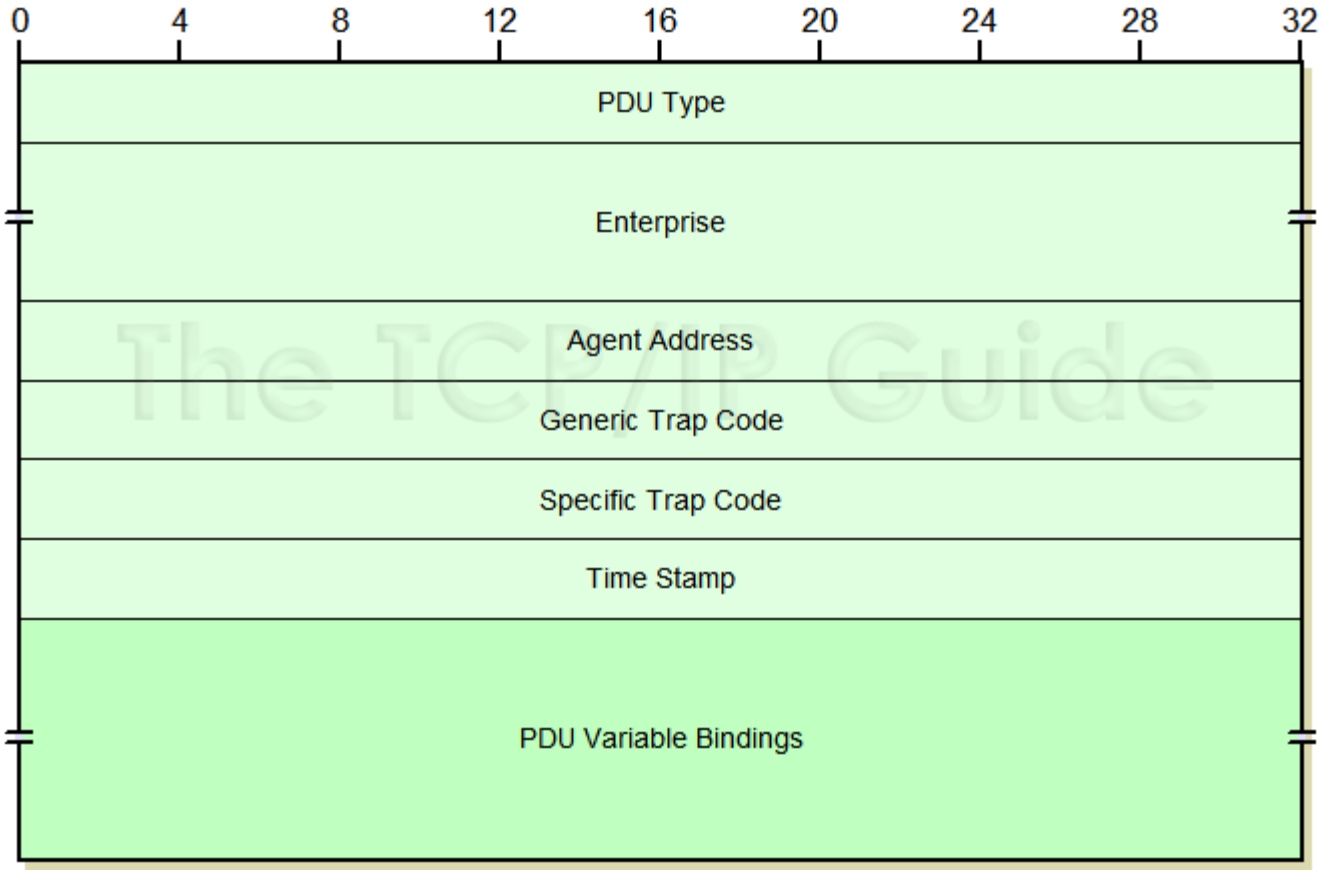
- Get-Request para o objecto sysDescr (1.3.6.1.2.1.1.1)

Bytes (hex) → 30 29 02 01 00

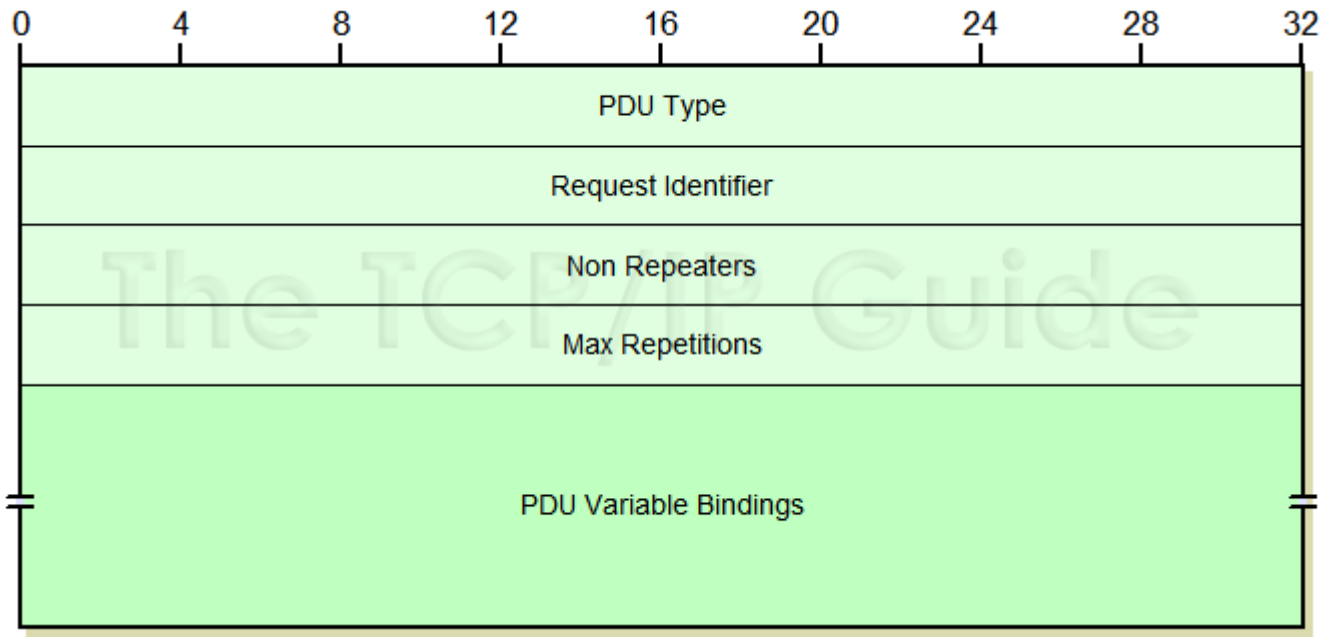
Significado → SEQUENCE len=41 INTEGER len=1 vers=0

```
   04      06    70   75   62   6C   69   63
string  len=6    p    u    b    l    i    c

   A0      1C     02      04    05 AE 56 02
getReq len=28 INTEGER len=4    -req ID-

   02      01     00       02      01    00
INTEGER len=1 status INTEGER len=1 index

   30       0E       30       0C    06    08
SEQUENCE len=14 SEQUENCE len=12 OID len=8

 2B   06   01   02   01   01   01   00
1.3 . 6 . 1 . 2 . 1 . 1 . 1 . 0

 05     00
NULL len=0
```

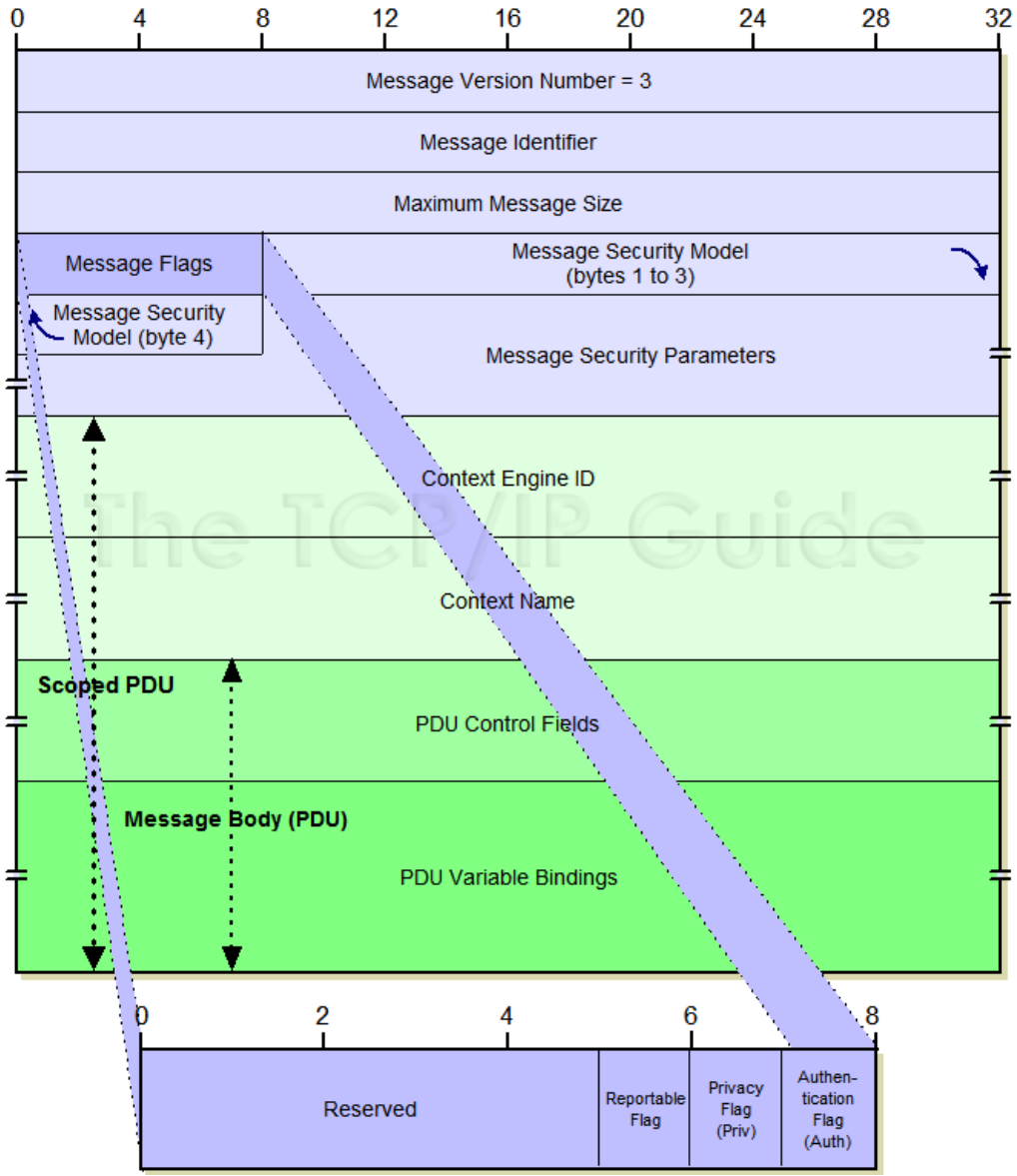
# Formato geral das traps SNMP



# Formato das GetBulkRequest SNMPv2



# Formato geral das mensagens SNMPv3



# Segurança

- SNMPv1
  - Mensagens não cifradas
    - Conteúdo visível por quem conseguir escutar os pacotes
    - Fáceis de forjar
  - Autenticação por *string* de comunidade
    - Espécie de *password* partilhada
    - Circula em *clear text*
- Por estes motivos, o SNMPv1
  - Não se usava para escrita
    - Excepto em redes fisicamente isoladas
  - Baseava-se no uso de "objectos fracos"
    - O seu uso não originaria danos significativos
    - E.g., uma variável com o *uptime* é aceitável, mas uma com a *password* de root não é

# Segurança

- SNMPv2
  - Muitas variantes com diferentes modelos de segurança
    - Party-based, user-based, view-based access control
  - Pouco suportadas na prática
    - A única versão amplamente suportada é a v2c
      - Baseada em *strings* de comunidade,
      - Nível de segurância idêntico à v1…

# Segurança

- SNMPv3
  - Mensagens incluem um *hash* criptográfico calculado sobre
    - O conteúdo da mensagem
    - A *password* do utilizador
    - O instante de transmissão
  - Suporta autenticação segura
    - A *password* nunca é incluída nas mensagens
    - A *password* é por utilizador
    - Permite definir diferentes níveis de acesso a diferentes partes da MIB para diferentes utilizadores
  - Garante autenticidade e integridade das mensagens
    - Verifica que a mensagem não foi alterada nem é forjada
  - Impede a (re)utilização de mensagens antigas
  - Suporta cifragem completa das mensagens para garantir privacidade

# Segurança

- Níveis de segurança no SNMPv3
  - noAuthNoPriv
    - Inseguro
  - AuthNoPriv
    - Autenticação segura, por utilizador
    - Garantia de integridade das mensagens
    - Sem cifragem — alguém que escute os pacotes na rede pode obter informações sobre os dispositivos
  - AuthPriv
    - Mensagens cifradas
    - Nível mais seguro