INSTITUTO DE PESQUISAS TECNOLÓGICAS DO ESTADO DE SÃO PAULO

EDUARDO DE LIMA BRITO

Implementação de gerenciamento em ambientes sem gerenciamento nativo: Extensão de funcionalidades em agentes SNMP

São Paulo

2004

EDUARDO DE LIMA BRITO

Implementação de gerenciamento em ambientes sem gerenciamento nativo: Extensão de funcionalidades em agentes SNMP

Dissertação apresentada ao Instituto de Pesquisas Tecnológicas do Estado de São Paulo – IPT, para obtenção do título de Mestre em Engenharia de Computação.
Área de concentração: Redes de Computadores

Orientador: Dr. Antonio Luiz Rigo

São Paulo
2004

Brito, Eduardo de Lima
Implementação de gerenciamento em ambientes sem gerenciamento nativo: extensão de funcionalidades em agentes SNMP. / Eduardo de Lima Brito. São Paulo, 2004.
112p.

Dissertação (Mestrado em Engenharia de Computação) - Instituto de Pesquisas Tecnológicas do Estado de São Paulo. Área de concentração: Redes de Computadores.

Orientador: Prof. Dr. Antonio Luiz Rigo

1. Gerenciamento de redes de computadores 2. SNMP 3.Tese
I. Instituto de Pesquisas Tecnológicas do Estado de São Paulo. Centro de Aperfeiçoamento Tecnológico II. Título

CDU 004.7(043)
B862i

RESUMO

Desenvolveu-se um estudo de caso para implementar gerenciamento nos sistemas de movimentação e de aquisição de dados de um laboratório de ensaios do Agrupamento de Sistemas de Controle do IPT.

Os equipamentos utilizados no laboratório de ensaios não contemplam, em seu projeto original, recursos de gerenciamento. A implementação de um agente extensível SNMP, parte do escopo deste trabalho, permite atribuir valores ao estado destes equipamentos que possibilitam seu gerenciamento por qualquer aplicação SNMP. Analisaram-se algumas opções de extensibilidade e optou-se pela implementação do protocolo AgentX, utilizando o software NET-SNMP.

Selecionaram-se quatro situações típicas de gerenciamento com o propósito de demonstrar a utilização de operações básicas do SNMPv1 (get, set e trap), que servirão de modelo para a implementação futura de novas funções, visando ampliar o conjunto de situações gerenciáveis, através do desenvolvimento de procedimentos e funções análogas.

Com base nestas premissas, desenvolveu-se a solução, composta por uma MIB e um subagente que contém, respectivamente, as definições e a implementação do gerenciamento. Desenvolveram-se também funções que traduzem os valores dos objetos SNMP em comandos para a controladora dos equipamentos do laboratório e vice-versa, sendo possível, desta forma, a implementar o gerenciamento à distância, oferecendo flexibilidade a seus técnicos para acompanhamento e atuação no sistema por acesso remoto, de qualquer local da instituição.

Palavras-chave: Gerenciamento de Redes de Computadores; Extensão de funcionalidades em agentes SNMP; SNMP

ABSTRACT

It was developed a case study to implement a network management solution at the laboratory of the Agrupamento de Sistemas de Controle located at IPT.

Equipments used at the laboratory don't offer management facilities in its original concepts. The implementation of an extensible SNMP agent enables the assignment of values in the status of those equipments through any SNMP application. It was analyzed some extensibility options and selected the AgentX protocol implementation using, for this, the NET-SNMP software.

Four manageable situations were defined to demonstrate the three basic SNMP operations (get, set and trap) and to provide guidance for future new manageable situations implementation, by developing procedures and functions with the same structure and using the same processes.

Based on this, it was developed the new IPT's lab management solution composed by a MIB and a subagent that contains the definitions and the implementations of the management in SNMP world. It was also developed some functions to map SNMP object values in commands to/from the IPT's lab equipment controller board, being possible the SNMP-based management solution implementation for the IPT's laboratory, offering the flexibility to its technicians to monitor and act in the system remotely, from any place inside the institution.

Keywords: Computer Network Management; SNMP agent's extensibility; SNMP.

Lista de figuras

Figura 1 Arquitetura do sistema de gerenciamento.....…01
Figura 2 Arquitetura de Gerenciamento de Redes…….…..….....08
Figura 3 Relacionamento entre um gerenciador e um agente…….....09
Figura 4 Ilhas de Gerenciamento…….....13
Figura 5 Gerenciador dos Gerenciadores…….….....14
Figura 6 Plataforma integrada de gerenciamento…….....…….....14
Figura 7 Arquitetura SNMP…….....17
Figura 8 Hierarquia do SNMP…….…….....…..22
Figura 9 Árvore de objetos da SMI….…….…….....23
Figura 10 Nós filhos da MIB-II…….....24
Figura 11 Laboratório – IPT Sistemas Digitais …….…….26
Figura 12 Arquitetura do AgentX ……..37
Figura 13 Arquitetura - Gerenciamento do Laboratório – IPT……...…..…..50
Figura 14 Arquitetura – Alteração do estado da antena (Ligado/Desligado) 55
Figura 15 Arquitetura – Verificação da temperatura da antena 58
Figura 16 Arquitetura – Verificação de erros e warnings no log 59
Figura 17 Formato de mensagem SNMP……...…….…….…Anexo A
Figura 18 Operações Get, GetNext, Response e Set..…….…….…Anexo A
Figura 19 SNMPv1 TRAP PDU……....…….…….…Anexo A
Figura 20 SNMPv2 PDU para as operações Get, GetNext, Inform, Response e Set e Trap ……...…….…….…Anexo A
Figura 2118 SNMPv2 GetBulk PDU……...Anexo A

Lista de abreviaturas, siglas e símbolos

ASN.1	*Abstract Syntax Notation One*
ATM	*Automated Teller Machine*
BER	*Basic Encoding Rules*
CMIP	*Common Management Information Protocol*
IANA	*Internet Assigned Numbers Authority*
ICMP	*Internet Control Message Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IP	*Internet protocol*
IPT	Instituto de Pesquisas Tecnológicas do Estado de São Paulo
ISO	*International Standardization Organization*
MIB	*Management Information Base*
NMS	*Network Management System*
OID	*Object Identifier*
OSI	*Open Systems Interconnection*
PDU	*Protocol Data Unit*
RFC	*Request for Comment*
RMON	*Remote Monitoring*
SMI	*Structure of Management Information*
SGMP	*Simple Gateway Management Protocol*
SNMP	*Simple Network Management Protocol*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
HTTP	*Hypertext Transfer Protocol*

Sumário

Resumo.....

Abstract.....

Lista de figuras.....

Lista de abreviaturas, siglas e símbolos.....

1 Introdução 1

1.1 Propósito da dissertação..... 2

2 Gerenciamento de Redes 3

2.1 Funções do gerenciamento..... 4
2.2 Arquitetura de gerenciamento..... 7
2.3 Evolução do gerenciamento..... 12

3 SNMP 15

3.1 A Estrutura de uma MIB (SMI)..... 20
3.2 Identificadores de objetos (OID) 22
3.3 MIB-II a MIB padrão para a Internet..... 24

4 Estudo de Caso: Implementação do gerenciamento no laboratório de testes ... 26

4.1 Análise do ambiente para a confirmação da adequação do SNMP 29
4.2 Mapeamento dos produtos compatíveis com SNMP..... 33
4.3 Análise das opções baseadas em SNMP..... 37

5 Construção de mecanismos necessários à implementação do gerenciamento no laboratório 49

5.1 Definição dos objetos a serem gerenciados 49
5.2 Definição da arquitetura de gerenciamento 50
5.3 Criação do arquivo de MIB..... 51
5.4 Criação do arquivo de código dos novos objetos..... 53
5.5 Criação dos subagentes implementando os novos objetos 55
5.6 Implantação do gerenciamento no laboratório..... 55

6 Conclusão..... 61

7 Sugestão para trabalhos futuros..... 63

Referências..... 64

Anexo A: Mensagens e Operações SNMP 69

Anexo B: Códigos e testes desenvolvidos para o estudo de caso..... 76

Anexo C: O software NET-SNMP..... 92

1 Introdução

O interesse por SNMP surgiu de um trabalho que consistia no desenvolvimento de uma interface entre o sistema de gerenciamento da rede de uma empresa de telecomunicações e seu sistema de atendimento ao cliente.

A informação sobre o estado da rede era capturada, utilizando seus sistemas de gerenciamento proprietários que alimentavam de informações um gerenciador comum através do SNMP.

Se um nó falhasse, seu gerenciador enviava uma TRAP SNMP ao gerenciador comum. Este registravadisponibilizava a informação da falha em um diretório, verificado constantemente pelo sistema de atendimento, que, ao. Ao identificar a falha, o sistema de atendimento abria uma ordem de serviço de rede externa, de forma que a central de atendimento ao cliente visualizasse o tipo do problema e a região afetada.

A interface permitia a identificação dos nós de rede inoperantes e, caso um cliente reclamasse, a central de atendimento contava com o apoio desse sistema de informação para se antecipar, demonstrando ao reclamante que a empresa já estava ciente do problema, havia tomado providências e a solução estava em andamento.

A figura 1 ilustra a arquitetura do sistema.

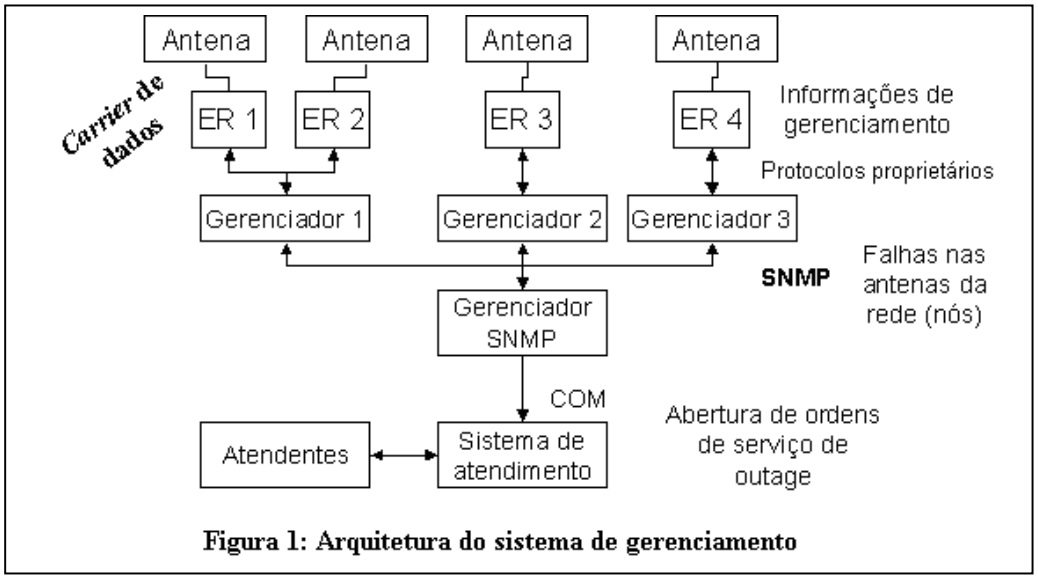

Figura 1: Arquitetura do sistema de gerenciamento

Esta experiência mostrou algumas vantagens da utilização do SNMP e a importância da consolidação do gerenciamento do hardware e software, pois sem esta consolidação o gerenciamento da rede apresentaria maior complexidade tanto em sua operação quanto em custo de manutenção. distribuídos em ambientes diversificados em um único aplicativo, no qual produtos de fornecedores diferentes se comunicam por diferentes protocolos.

Motivado pela experiência acima e interessado em um aprofundamento no tema SNMP, iniciou-se a prospecção de casos práticos que admitissem a aplicação deste protocolo. Surgiram dois casos nos quais a implementação do gerenciamento era necessária.

O primeiro, tratava da definição de MIBs SNMP para uma central de telefonia para a qual o IPT estava desenvolvendo uma solução de gerenciamento proprietária.

O segundo caso, tinha como foco a criação de mecanismos que permitissem o gerenciamento dos elementos que compõem o laboratório de testes do Agrupamento de Sistemas de Controle do IPT.

Elegeu-se a implementação do gerenciamento no laboratório, que permitiu uma aplicação mais ampla dos conceitos do SNMP, pois para resolver este caso foi necessário promover a extensão de agentes que passaram a manipular informações de elementos, até então, não eram gerenciáveis através do SNMP.

1.1 Propósito da dissertação

O objetivo deste trabalho foi implementar o gerenciamento no laboratório de testes do Agrupamento de Sistemas de Controle do IPT. Através desta implementação foi possível explorar o gerenciamento de redes por SNMP, utilizando-se de um estudo de caso para aplicação dos conceitos estudados.

O estudo de caso adotado apresentou o desafio de oferecer um ambiente com recursos computacionais que não dispunham de quaisquer mecanismos pré-existentes de gerenciamento, obrigando o autor a conhecer toda a estrutura e funcionalidade de uma implementação SNMP.

2 Gerenciamento de Redes

Uma das primeiras redes de comutação de pacotes (ARPANET) foi desenvolvida pelo Departamento de Defesa Norte Americano em 1969 [01]. No final da década de 70 o protocolo TCP/IP foi padronizado pelo Comitê de Arquitetura de *Internet* (*Internet Architecture Board* - IAB) para a comunicação com fins militares. A partir da difusão deste protocolo, e com a expansão das redes, o gerenciamento de redes tornou-se um item importante para a manutenção da operabilidade das mesmas. No final dos anos 80 e início dos 90, a quantidade de equipamentos conectados a *Internet* cresceu de forma explosiva. Um número crescente de sub-redes incorporou a necessidade das atividades de monitoração.

No surgimento do gerenciamento de redes, os administradores ou especialistas em protocolos utilizavam recursos básicos do *TCP/IP* como os comandos *ICMP (Internet Control Message Protocol)* e o *PING* [02]. O gerenciamento de rede consistia na utilização de mensagens *ICMP echo/echo-reply* para testar se havia comunicação entre dispositivos. Outra técnica utilizada era o acesso remoto do administrador em estações ou sub-redes. Com o crescimento da utilização das redes, estas técnicas já não eram mais eficientes para monitorar seus dispositivos.

A necessidade de gerenciamento de uma rede de computadores é proporcional a seu tamanho, complexidade e quantidade de aplicações críticas relativas ao negócio que a rede em questão atende.

O crescimento das redes heterogêneas favorece a presença de equipamentos de diferentes fornecedores, software distribuído, diferentes tecnologias de comunicação como, por exemplo, redes locais, satélites, ATMs e outros.

Dada esta pluralidade, os administradores de rede se deparam com um grande número de ferramentas de gerenciamento presentes no mercado com funcionalidades diversas e com novos conceitos e terminologias sendo agregados constantemente ao dia a dia.

Todo gerenciamento de rede parte, ou deve partir, de um axioma fundamental: o impacto do gerenciamento no funcionamento da rede deve ser mínimo.

Entende-se por gerenciamento, a arte de regulação das diversas ações que influenciam a evolução da rede, através da habilidade de controle e monitoração da rede por um ponto central.

Alguns exemplos destas ações são a atualização tecnológica, novas necessidades de aplicações, índices de qualidade, índices de segurança, limitações de custo e etc.

2.1 Funções do gerenciamento

As funções do gerenciamento de rede dividem-se em cinco grupos, são eles [03,04]:

- Gerenciamento de falhas
- Gerenciamento de desempenho
- Gerenciamento de segurança
- Gerenciamento de configuração
- Gerenciamento de contabilização

2.1.1 Gerenciamento de falhas

O gerenciamento de falhas fornece meios para os administradores de rede descobrirem falhas em elementos gerenciáveis, na rede e na operação, possibilitando determinar as causas e tomar as medidas corretivas necessárias.

Um gerenciador de falhas deve oferecer minimamente as seguintes funcionalidades:

- Relatórios de ocorrência de falhas
- Criação de *logs*
- Realização de testes e diagnósticos
- Mecanismos para correção de falhas (dependendo do caso a correção deve ser automática)
- Indicação da falha e apontamento do componente em falha, origem da falha, prováveis causas e estratégias corretivas.

Exemplo de falhas:

- Canal de comunicação interrompido
- Interfaces de meio físico defeituosas ou com mau contato
- Negociação de protocolos fracassada (*timeout* e perdas de mensagens)
- Configuração do hardware de estações de trabalho
- Acesso a disco e dispositivos de impressão

2.1.2 Gerênciamento de desempenho

Objetiva a medição de vários aspectos de desempenho da rede, incluindo a captura e análise de dados estatísticos sobre os sistemas. Dentre outras funções, o gerenciador de desempenho deve obter indicadores de utilização e erros para cada elemento na rede e fornecer um nível consistente de medida de desempenho, de forma que cada dispositivo corresponda adequadamente à operação da rede.

Abaixo são listados alguns exemplos de medição necessária pelo gerenciador de desempenho:

- Vazão nas linhas de comunicação
- Atraso de transmissão de mensagem
- Utilização de disco e de *buffer* nos servidores de disco
- Ocupação de impressoras
- Índices de utilização de aplicação e de outros recursos

2.1.3 Gerênciamento de segurança

Permite o controle de acessos a recursos da rede de forma que a informação não seja obtida sem autorização, limite o acesso e forneça notificações para os administradores

caso determinadas condições sejam satisfeitas (ex. Alteração de dados de controle de um servidor)

Outras funcionalidades que um gerenciador de segurança deve oferecer são:

- Atualização das senhas
- Freqüência de uso dos mecanismos de distribuição de chaves
- Acesso do *log* de alarmes
- Instalação de aplicativos antivírus, etc.

2.1.4 Gerênciamento de configuração

Objetiva o monitoramento da configuração da rede de forma que um hardware e um software específico seja gerenciado. Ela fornece mecanismos para inicializar, reconfigurar e desligar os elementos gerenciáveis.

Exemplos de elementos gerenciáveis pela gerência de configuração pertencentes à configuração física e lógica de uma rede:

- Configuração Física (hardware)
  - Roteadores e pontes da rede com as respectivas ligações
  - Estações de trabalho
  - Servidores
  - Hubs
- Configuração Lógica (software)
  - Endereços atribuídos
  - Protocolos utilizados e configurações
  - Rotas
  - Sistemas operacionais
  - Aplicativos
  - Parametrização de equipamentos
  - Distribuição dos serviços

2.1.5 Gerênciamento de contabilização

Objetiva medir quanto recurso de rede cada usuário utiliza.

A contabilização pode ter objetivo de análise do comportamento da rede ou de efetiva tarifação dos recursos utilizados.

Foco da medição:

- Volumes de dado trafegados por usuário, grupo de usuários e áreas de interesse de tráfego
- Áreas de memória ocupadas
- Volumes de disco ocupados
- Volumes de impressão

A gerência de contabilização deve permitir medir o custo de cada operação realizada. Ex.: custo de acesso a um arquivo no servidor, custo de uma mensagem trocada via correio eletrônico.

2.2 Arquitetura de gerenciamento

Os sistemas de gerenciamento de redes têm uma arquitetura básica semelhante à da figura 2 [05]:

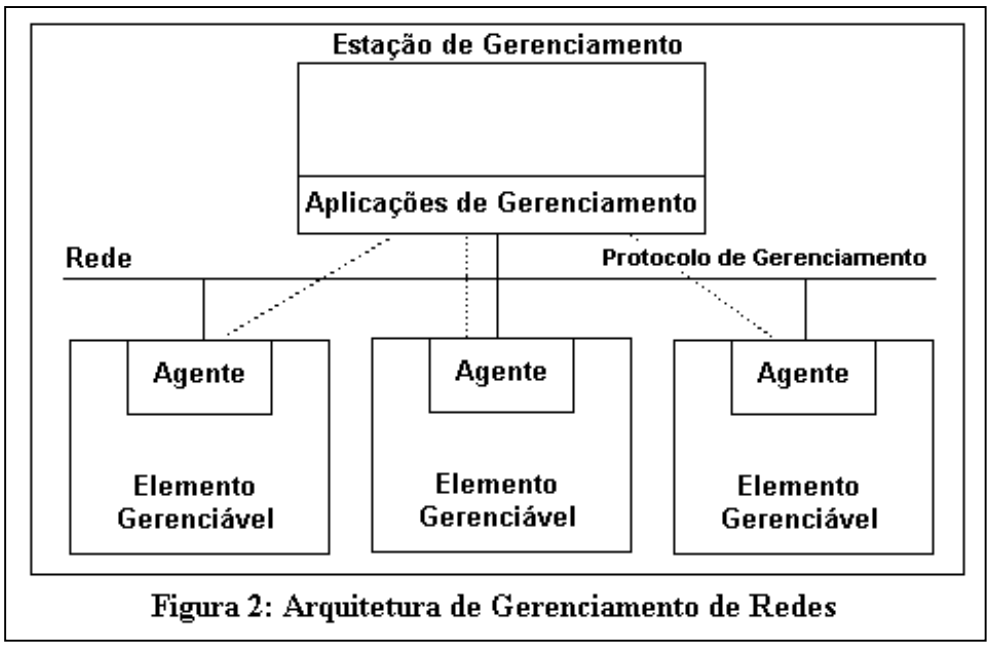

Figura 2: Arquitetura de Gerenciamento de Redes

Os seguintes elementos fazem parte da arquitetura básica de gerenciamento de redes:

- Estação de gerenciamento (ou gerenciador)
- Protocolo de gerenciamento
- Agente
- Elementos gerenciáveis

2.2.1 Estação de gerenciamento

O gerenciador (conhecido em inglês como NMS – *Network Management System*) [12] é um servidor executando algum tipo de software que pode lidar com tarefas de gerenciamento de uma rede, interagindo, para isso, com os agentes e consolidando as informações recebidas e enviadas aos mesmos. Normalmente os gerenciadores possuem um console com interface gráfica que permite que os operadores e administradores visualizem graficamente sua rede, controlem os elementos gerenciáveis e realizem suas programações.

Os gerenciadores podem ser programados para realizar consultas automaticamente e receberem notificações dos agentes da rede. Através destas consultas o gerenciador sabe se há alguma falha associada aos elementos gerenciados, qual o tipo, severidade e outras características da falha.

Uma notificação é um método utilizado por um agente para informar ao gerenciador que alguma coisa aconteceu. Como quem envia a notificação é o agente e o faz somente quando é detectado um problema, este comando tem natureza assíncrona. No SNMP versão 1 a operação de notificação existente é conhecida como *TRAP*. A partir da versão 2 também existe um outro tipo de notificação conhecida como *INFORM*.

Um gerenciador deve ter a capacidade de interpretar todas as informações dos agentes com o qual se relaciona. O elemento que viabiliza esta interpretação é conhecido como MIB (*Management Information Base* ou base de informações gerenciáveis). Quando se deseja adicionar funcionalidades a um agente, como é o caso deste trabalho, deve-se desenvolver uma MIB para refletir estas funcionalidades, compilá-la e adicioná-la ao gerenciador, além de, é claro, recompilar o agente com o código necessário para a implementação das funcionalidades em questão. Desta forma, torna-se possível uma comunicação entre gerenciador e agente baseada nas funcionalidades implementadas.

Nem todos os gerenciadores têm a flexibilidade de incorporar MIBs que implementem qualquer funcionalidade. A maioria dos gerenciadores trata de MIBs específicas e privadas de forma a atender somente um conjunto de produtos de determinado fabricante. Porém, os gerenciadores que trabalham com o SNMP como protocolo padrão são, em sua maioria, abertos para a inserção de novas MIBs.

A figura 3 ilustra o relacionamento entre agente e gerenciador:

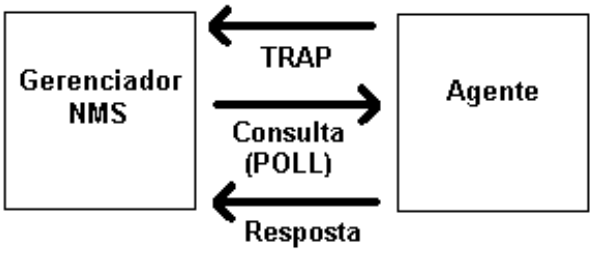

Figura 3: Relacionamento entre um gerenciador e um agente

Para o estudo de caso desenvolvido foi utilizado o gerenciador NET-SNMP, porém, pode-se encontrar no mercado um número variado de gerenciadores. Alguns exemplos de gerenciadores são: Netcool, Tivoli Netview, HP OpenView, Ciscoworks e etc.

2.2.2 Protocolo de Gerenciamento

A adoção destes protocolos muitas vezes não é uma questão de escolha, pois o dispositivo de rede pode conter protocolos de gerenciamento proprietários, ou seja, protocolos que as empresas fabricantes de equipamentos de rede desenvolvem para gerenciar seus produtos.

Os protocolos SNMP e CMIP são os padrões de gerenciamento [13] adotados em todo o mundo.

O SNMP, detalhado no capítulo posterior, é o protocolo mais utilizado, principalmente em ambientes LAN. O CMIP é utilizado principalmente em ambientes de telecomunicações para gerenciar redes grandes e complexas.

O CMIP (*Common Management Information Protocol* ou Protocolo Comum de Gerenciamento), desenvolvido com o apoio do governo e empresas americanas, é um protocolo de gerenciamento baseado na arquitetura OSI *(Open Systems Interconnection)* e utiliza um mecanismo de transporte que, diferentemente do SNMP, é orientado à conexão suportando controle de acesso, autorização e *logs* de segurança.

A especificação do CMIP para redes TCP/IP é conhecida como CMOT (CMIP *over* TCP/IP). A versão do IEEE 802 para LANs é conhecida como CMOL (CMIP *over* LLC)

A maior vantagem do uso do CMIP frente ao SNMP é que:

- Os métodos de objetos CMIP não somente guardam informações mas também são utilizados para realizar tarefas, o que não pode ser realizado pelo SNMP.
- O CMIP é um sistema mais seguro e foi construído para suportar autorização, controle de acesso e *logs* de segurança.
- O CMIP fornece um melhor reporte de condições atípicas da rede.

Apesar de apresentar estas vantagens, o CMIP é raramente implementado em ambientes de LAN, devido ao baixo número de elementos de rede que suportam seu uso. Além disso, segundo Rajesh Sharma em seu livro *Cisco Security Bible* [33], publicado em 2002, o consumo de recursos em uma utilização de CMIP é 10 vezes maior que na utilização do SNMP, o que inviabiliza seu uso em uma expressiva quantidade de casos. Outra dificuldade é a complexidade de sua programação necessitando de profissionais especializados, tanto para a implementação quanto para a manutenção dos sistemas gerenciadores.

Na arquitetura do CMIP, a aplicação de gerenciamento pode iniciar transações com os agentes utilizando as seguintes operações:

ACTION – Requisita que uma ação ocorra de acordo com o que foi definido pelo objeto gerenciado.
CANCEL_GET – Cancela uma requisição *GET*.
CREATE – Cria uma instância no objeto gerenciado.
DELETE –Exclui uma instância no objeto gerenciado.
GET – Requisita o valor da instância do objeto gerenciado.
SET – Altera o valor do objeto gerenciado.

2.2.3 Agente

O agente é um software que geralmente reside no mesmo hardware do objeto gerenciado. A função do agente é interagir com o objeto gerenciado de forma a executar algum procedimento ou receber informações solicitadas pelo gerenciador.

Cabe ao agente implementar todas as funcionalidades necessárias ao gerenciamento de seu respectivo objeto.

Um agente pode contar também com subagentes ou chamadas a funções externas para tornar-se mais abrangente.

Através do estudo de caso desenvolvido, foi possível estender as funcionalidades de um agente para gerenciar um sistema que não era gerenciável através do SNMP.

2.2.4 Elementos Gerenciáveis

Um elemento gerenciável pode ser qualquer tipo de dispositivo residente na rede gerenciável através de um agente. Normalmente quando um elemento sai da fábrica, já contém um agente que implementa um protocolo comum (CMIP ou SNMP) e/ou um protocolo proprietário (especificado e implementado pelo próprio fabricante), tornando-se gerenciável através da rede.

Exemplos de elementos que podem ser gerenciáveis são:

- Computadores
- Impressoras
- Controladoras
- Roteadores

2.3 Evolução do gerenciamento

Conforme notas de aula do prof. Geraldo Lino Campos - programa de Mestrado Profissional do IPT - 2002, oO gerenciamento de redes de computadores passou por três estágios durante sua evolução [03]. Devido à diversidade de sistemas e realidades das empresas e instituições, é possível encontrar estas três configurações coexistindo atualmente, são elas:

- Ilhas de gerenciamento
- Gerenciador de Gerenciadores
- Plataforma integrada de gerenciamento

2.3.1 Ilhas de gerenciamento

Em um estágio inicial o gerenciamento de redes era realizado através de protocolos e gerenciadores proprietários denominados ilhas de gerenciamento pois era comum em grandes redes encontrarem-se diversos sistemas sendo gerenciados separadamente e de forma particular. A figura 4, ilustra este esquema de gerenciamento:

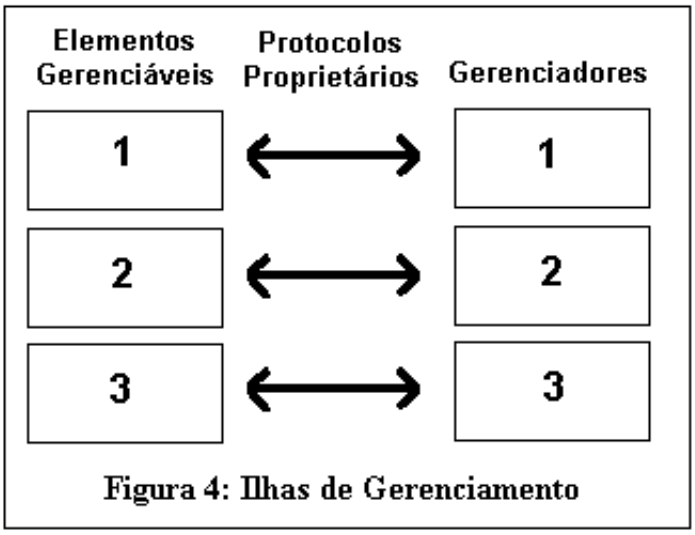

Figura 4: Ilhas de Gerenciamento

2.3.2 Gerenciador de Gerenciadores

A dificuldade e alto custo de monitoração dado o crescimento de ilhas de gerenciamento e o aparecimento protocolos comuns impulsionaram a implementação de gerenciadores que consolidavam as informações de cada ilha em um único console. Esta arquitetura é predominantemente utilizada atualmente devido à presença ainda grande de protocolos e gerenciadores proprietários nas redes de comunicação. Esta presença se justifica pela necessidade de gerenciamento específico que os protocolos comuns (SNMP e CMIP) ainda não comportam.

A figura 5, ilustra este esquema de gerenciamento:

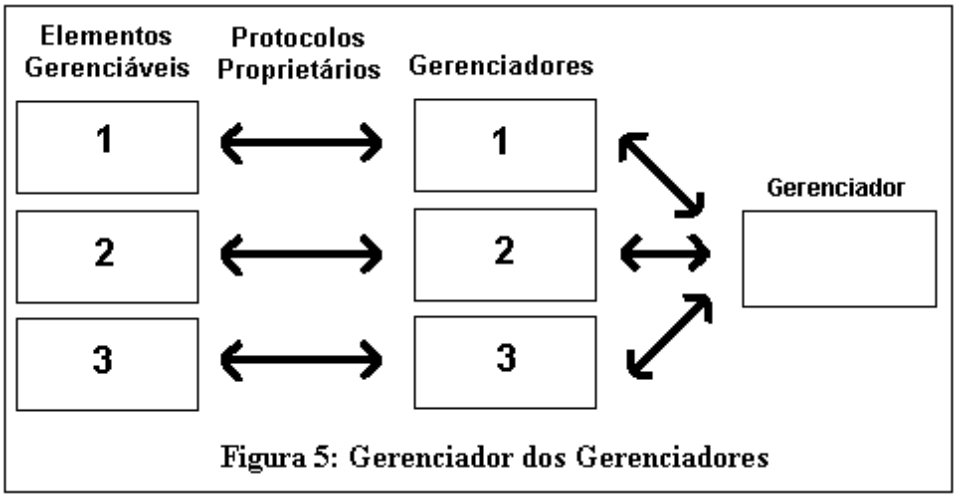

Figura 5: Gerenciador dos Gerenciadores

2.3.3 Plataforma Integrada de Gerenciamento

A tendência atual é ter um único gerenciador capaz de interagir com todos os elementos gerenciados da empresa ou instituição. Para que isto seja possível, é necessário que as aplicações gerenciadoras sejam implementadas de forma a suprir a necessidade de gerenciamento de todos os elementos gerenciáveis e que estes possuam agentes que possam interagir com os protocolos comuns abertos. A figura 6, ilustra este esquema de gerenciamento:

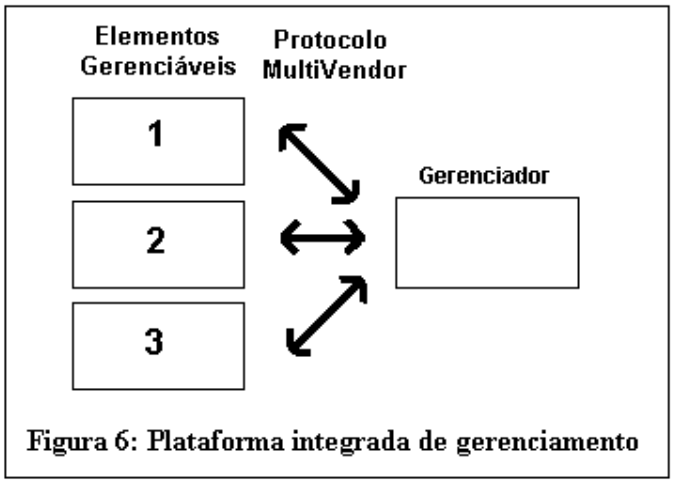

Figura 6: Plataforma integrada de gerenciamento

3 SNMP

Depois que o TCP/IP se tornou o protocolo padrão para a internet em 1983[06], as redes tomaram uma abrangência global, não sendo viável depender de um pequeno número de especialistas em rede para solucionar problemas de gerenciamento.

O que parecia necessário era a adoção de um protocolo padronizado com funcionalidades que permitissem ir além de comandos ICMP (ex. *ping*), e que fosse facilmente aprendido e utilizado pelas pessoas que trabalhassem com gerenciamento de redes.

Com este intuito, o SNMP foi desenvolvido em 1988 [06] pela IETF *(Internet Engineering Task Force)* e adotado como padrão de gerenciamento de redes e dispositivos de redes baseado em TCP/IP. Em sua concepção original[31], o SNMP foi projetado para ser utilizado sobre redes *Ethernet*, baseando-se no protocolo SGMP *(Simple Gateway Management Protocol)* e incorporando padrões ISO como o uso de objetos, estrutura hierárquica baseada em árvore e etc.

O SGMP foi desenvolvido em março de 1987 e aceito como padrão em novembro do mesmo ano (RFC 1028)[37]. Após a criação do SGMP, por volta de 1988, surgiram três protocolos de gerenciamento de redes:

- *High-Level Entity Management System (HEMS)*
- *Simple Network Management Protocol (SNMP)*
- *CMIP sobre TCP/IP (CMOT)*

Devido aos requisitos da implementação de um sistema de gerenciamento baseado no modelo OSI, o SNMP superou o CMOT como a principal opção de gerenciamento de redes. A RFC 1098 recomenda o SNMP como o protocolo preferencial para gerenciamento em redes TCP/IP.

A sigla SNMP significa *Simple Network Management Protocol*, ou em português, Protocolo Simples de Gerenciamento de Redes [07]. A palavra “simples” vem da

concepção do protocolo, no qual o agente SNMP requer uma parte muito pequena de *software*, delegando a maior parte do processamento e controle para o gerenciador.

O SNMP básico é amplamente utilizado. Os maiores fabricantes de computadores, estações, *bridges*, roteadores e *hubs* oferecem, em seus produtos, suporte ao SNMP. O aparecimento RMON [16,17] adicionou um conceito fundamental ao gerenciamento baseado em SNMP, pois o RMON permite a monitoração de sub-redes como uma visão integrada, ao invés de tratá-las separadamente.

A solução de gerenciamento refere-se a um conjunto de padrões, incluindo um protocolo, uma especificação da estrutura de base de dados e um conjunto de objetos. Podem ser utilizados para ações passivas (ex.: monitoração) ou ativas (ex.: modificação de estados dos dispositivos).

A arquitetura SNMP define uma relação cliente-servidor na qual o servidor (NMS – *Network Management Station*) executa aplicações de gerenciamento ou diretivas que gerenciam e controlam os elementos de rede. Clientes (elementos de rede gerenciáveis) são elementos que possuem agentes de gerenciamento responsáveis por atender às funções de gerenciamento de rede acionadas pelo servidor.

O SNMP utiliza o UDP como protocolo de transporte de dados entre os gerenciadores e agentes. Diferentemente do TCP, o UDP não estabelece conexão ponto-a-ponto, introduz menos tráfego e interfere o mínimo possível no desempenho da rede gerenciada. O UDP não controla a perda de datagramas, passando essa tarefa para a aplicação SNMP que, para viabilizar o controle, trabalha com o uso de retransmissão e temporizadores.

O SNMP utiliza a porta 161 do UDP para enviar e receber solicitações e porta 162 para receber *traps* dos dispositivos gerenciáveis[18]. Qualquer dispositivo que implementa o SNMP deve utilizar estes números de porta como padrão pois os gerenciadores já são configurados desta forma. Caso algum fornecedor utilize uma outra porta, esta deve ser mencionada em sua documentação, de forma que as estações de gerenciamento possam ser configuradas para trabalhar com seu produto.

A figura 7 , encontrada em diversas literaturas como, por exemplo, no livro *Network manager's handbook* de Nathan J. Muller, 2003, representa a arquitetura do SNMP[32]:

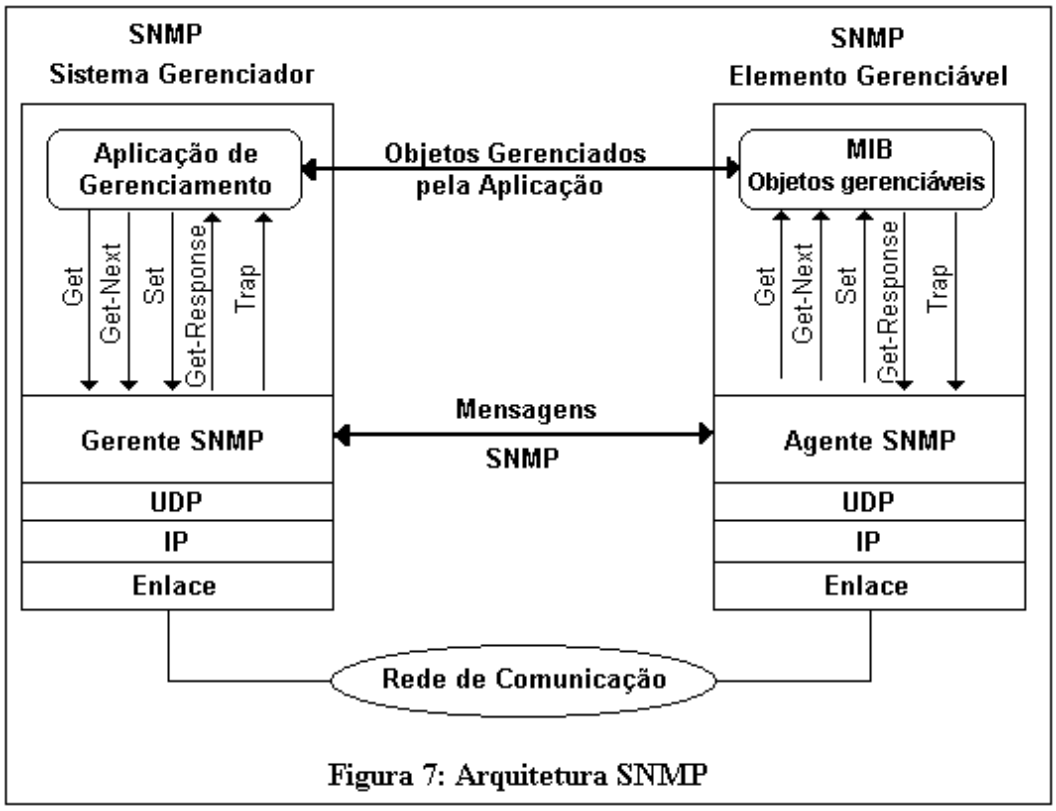

Figura 7: Arquitetura SNMP

O SNMP utiliza o conceito de comunidade para garantir a segurança do acesso à informação em um processo análogo ao estabelecimento de permissões a determinados grupos de usuários, no qual usuários podem ser outros elementos de rede. Fazendo isto, é possível restringir o acesso à leitura (*read-only*), gravação (*read-write*) e *trap* somente para quem é autorizado, de forma a não comprometer a integridade da informação necessária ao gerenciamento [19].

A maioria dos fornecedores produz seus equipamentos com *strings* de comunidades padrão, geralmente *public* para a comunidade *read-only* e *private* para a comunidade *read-write*.

A funcionalidade do SNMP é determinada por um conjunto de comandos (ex.: *get, get-next, set, trap*), que atuam de acordo com as funcionalidades implementadas em cada agente e descritas por MIBs (*Management Information Base*, ou base de informações de

gerenciamento). Segundo Nathan J. Muller em seu livro *Network manager's handbook*, publicado pela McGraw-Hill em 2003, estas MIBs descrevem o que gerenciar e como o elemento será gerenciado, constituindo a inteligência da implementação do SNMP em um determinado ambiente. Todo elemento gerenciável mantém uma base de dados com valores para cada definição descrita em uma MIB [11].

Em uma implementação, MIBs devem ser carregadas nos gerenciadores SNMP de forma adequada a fim de torná-los aptos a interpretar mensagens recebidas em resposta às solicitações enviadas aos agentes SNMP, que também implementam MIBs.

Para garantir a portabilidade do SNMP entre os elementos gerenciáveis e garantir que informações básicas ao gerenciamento como Localização do Elemento, Descrição, Nome do Contato e outras, estejam presentes em todas as implementações SNMP, algumas MIBs são classificadas como padrão, como por exemplo a MIB-II (um conjunto de MIBs que todos os agentes e gerenciadores SNMP devem implementar). Isto possibilita a comunicação com qualquer agente SNMP sem que seja preciso conhecer os detalhes que ele implementa.

A IETF *(Internet Engineering Task Force)* é responsável pela definição de protocolos padrão que controlam o tráfego na Internet, incluindo o SNMP. A IETF publica RFCs *(Requests for Comments)* que são especificações para os diversos protocolos existentes no mundo IP. Primeiramente, a IETF submete os documentos com o status de "sugerido", depois passam para o status de "experimental" (*draft)*. Quando um *draft* final é aprovado, a RFC recebe o status de "completo" *(standard).*

As Conforme Douglas R. Mauro em seu livro SNMP Essencial, existem três principais versões de SNMP são [08]:

SNMPv1 (versão 1) → Definida na RFC 1157 pela IETF, foi intencionalmente simplificada devido a urgência de um protocolo de gerenciamento. A segurança do SNMPv1 baseia-se em comunidades que funcionam de forma análoga à utilização de senhas de acesso. Estas senhas são *strings* de texto puro que permitem qualquer aplicativo baseado em SNMP ter acesso a informações de gerenciamento de um dispositivo. Existem três tipos de permissões de acesso para as comunidades no SNMPv1: *read-only*, *read-write* e *trap*.

Esta versão foi incorporada a muitos produtos e plataformas de gerenciamento. A camada de transporte, utilizando UDP, expõe o sistema a eventuais perdas de dados em uma rede instável e pode limitar sua utilização, dependendo da criticidade do gerenciamento. O esquema de segurança citado restringe o uso do SNMPv1 em redes não críticas, como a rede do laboratório do IPT dentro do contexto apresentado no estudo de caso.

SNMPv2 (versão 2) → SNMPv2 não foi muito utilizado em sua concepção original. Existem versões diferentes do SNMPv2 utilizadas, como as versões SNMPv2c e SNMPv2u.

Uma das razões pela qual o SNMPv2 não ter sido muito utilizado, é que nesta versão, não foram endereçadas as principais preocupações com segurança relativas a versão anterior. Esta versão teve como melhoria a comunicação eficiente de NMS para NMS, a autenticação e encriptação de dados, a simplificação do mecanismo de transferência de dados. A MIB-II foi significativamente revisada e alterada para atender a versão 2 do SNMP. Esta revisão contempla as RFCs 2863[38], 2011[39], 2012[40] e 2013[41].

SNMPv3 (versão 3) → O desenvolvimento da versão 3 iniciou-se em 1997 e sua primeira proposição como padrão através da RFC 2261[54] ocorreu em janeiro de 1998. Dentre as principais melhorias propostas pelo SNMPv3 pode-se citar:

- Reforça a questão de segurança, a qual é considerada a maior deficiência nas versões anteriores
- Define uma arquitetura que permite a portabilidade das aplicações SNMP que foram ou serão definidas
- Mantém o SNMP tão simples quanto possível
- Mantém o SNMP com baixo custo de implementação
- Torna possível a atualização de partes do SNMP de forma escalável sem agredir a arquitetura

Além do SNMP propriamente dito, outros elementos importantes ao funcionamento do mesmo contam com RFCs. Uma lista com todas as RFCs pode ser encontrada no site do IETF (http://www.ietf.org/rfc.html).

3.1 A Estrutura de uma MIB (SMI)

De acordo com o paradigma atual de gerenciamento de redes, os objetos gerenciáveis em uma rede devem estar logicamente acessíveis. Isto significa que a informação importante ao gerenciamento deve estar armazenada em algum lugar, podendo ser recuperada, lida e modificada. O SNMP permite tanto a recuperação quanto a modificação destas informações.

A *Structure of Management Information*, ou, em português, Estrutura de Informações de Gerenciamento conhecida como SMI (RFC 1155)[55] organiza, nomeia e descreve as informações de modo que o acesso lógico a elas possa ocorrer [11].

Em linhas gerais, a SMI dita que a definição de objetos gerenciados, através das MIBs, pode ser fragmentada em três atributos[27]:

Nome

O nome é o identificador de objeto (OID – *Object Identifier*) que define com exclusividade um único objeto gerenciado. Estes nomes podem ser exibidos tanto em formato numérico quanto alfanumérico, ou melhor, para cada objeto existe um identificador numérico e outro respectivo alfanumérico.

Tipo e sintaxe

O tipo de dado de um objeto gerenciado é definido por meio de um subconjunto da *Abstract Syntax Notation One* (ASN.1). A ASN.1 é um meio de especificar o modo como os dados são representados e transmitidos entre gerenciadores e agentes, no contexto do SNMP. A notação é independente da máquina e plataforma.

Codificação

Uma única instância do objeto gerenciado é codificada em uma *string* de octetos por meio do método *Basic Encoding Rules* (BER). O método BER define o modo de codificação e decodificação dos objetos para que sejam transmitidos através de um meio de transporte (Ex.: *Ethernet*).

As MIBs são então definidas através das regras estipuladas na SMI. No exemplo abaixo é apresentado o código de um objeto contido na MIB-II.

(objeto SysUpTime contido na MIB-II)

SysUpTime OBJECT-TYPE
SYNTAX TimeTicks
ACCESS read-only
STATUS mandatory
DESCRIPTION
"O tempo (em centésimos de segundo) desde que o gerenciamento de rede foi reinicializado"
::={system 3}

OBJECT-TYPE é a sintaxe utilizada para nomear o objeto. Na prática, os agentes SNMP farão referência a este objeto pelo OID (*Object Identifier*)

SYNTAX define o tipo de informação armazenada no objeto. Por exemplo, uma MIB pode ser "*integer*" (inteira), "*counter*" (contador), "*DisplayString*" (texto) e etc.

ACCESS informa se uma entidade externa pode alterar o valor do objeto. "*read-only*" significa que somente o agente pode alterar o valor, ou seja, este não pode ser alterado por um comando. Se o valor é "*read-write*" então pode ser modificado.

STATUS é o estado do objeto na comunidade SNMP. Os valores para STATUS podem ser *"obsolete"*, *"current"* ou *"mandatory"*.

DESCRIPTION descreve o objeto.

::={system 3} é o *OID*

3.2 Identificadores de objetos (OID)

Para garantir a ordem e a universalidade do gerenciamento através do SNMP, os objetos gerenciáveis são organizados em uma hierarquia semelhante a uma árvore. Essa estrutura é base do esquema de atribuição de nomes do SNMP. Um ID de objeto é formado por uma seqüência de inteiros baseada nos nós da árvore, separada por pontos. Ex.: 1.3.6.1.4

Na árvore de objetos, o nó posicionado no início da árvore é denominado raiz. Os filhos formarão uma subárvore e o nó que não tiver filhos será chamado nó de folha[06]. A figura 8 exemplifica o conceito de hierarquia da árvore SNMP.

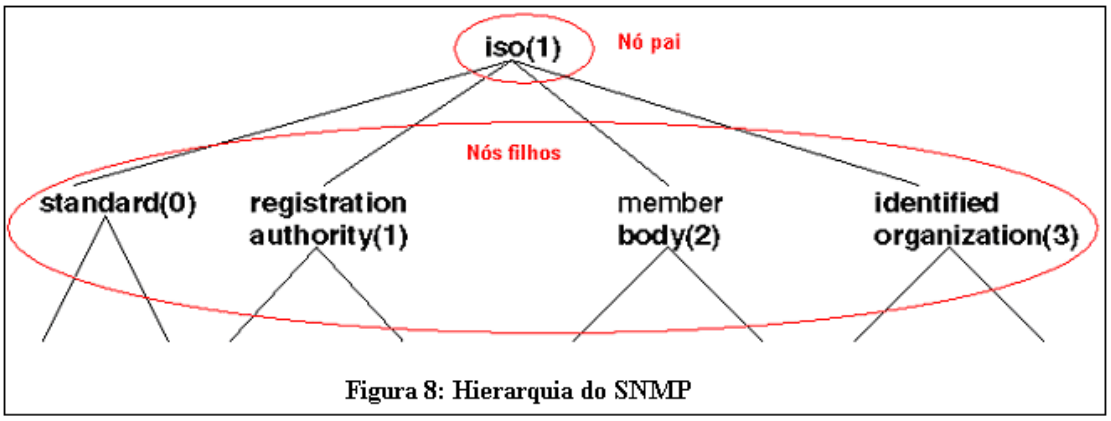

Figura 8: Hierarquia do SNMP

De acordo com as regras da árvore, o nome de um objeto é dado através de seu nome na árvore precedido pelos antecedentes. Um objeto possui uma OID numérica e um nome, em formato texto, associado (Ex.: OID 1.3.6.1 equivale a OID iso.org.dod.internet).

Os nós filhos de um mesmo pai recebe valores inteiros únicos dentro de uma subárvore, ou seja, não podem ser identificados pelo mesmo número.

Os nós filhos podem ser pais de outras sub-árvores onde o esquema de numeração é recursivamente aplicado. Um exemplo disto é a sub-árvore 1.3.6.1.1 (O número 1 se repete ao final do OID).

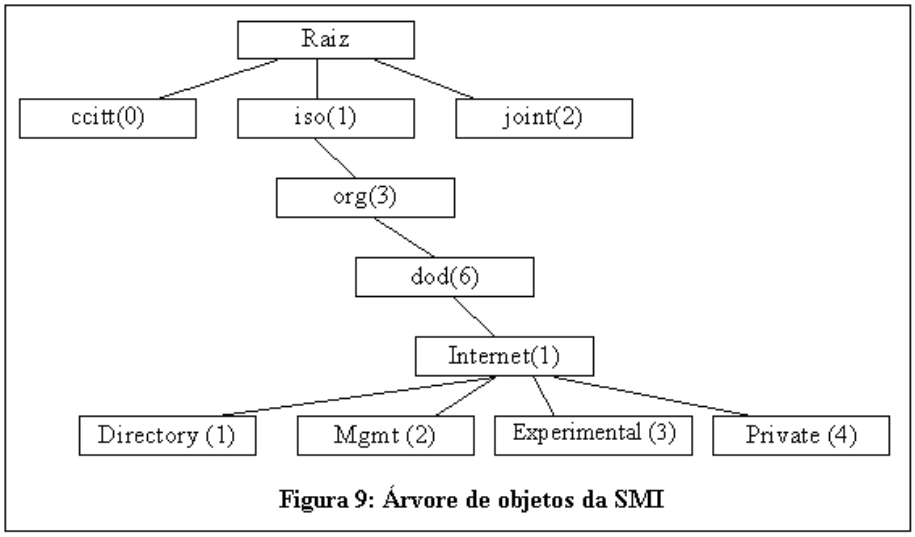

Figura 9: Árvore de objetos da SMI

A árvore pode crescer a uma profundidade arbitrária.

A figura 9 destaca a ramificação *dod.internet*, a ramificação mais utilizada e a que tem mais crescido desde a concepção da arquitetura.

Dentro da ramificação *iso.org.dod.internet* observa-se as seguintes ramificações:

Directory – não é usada no momento.

Management ou Mgmt – define um conjunto padrão de objetos de gerenciamento da Internet. Esta ramificação conta com a definição de MIBs importantes para o gerenciamento, como é o caso da MIB-II.

Experimental – Reservada para fins de teste e pesquisa.

Private – A ramificação *Private* conta com uma sub-ramificação *Enterprises* que é o nó raiz disponibilizado para as empresas privadas registrarem a informação de suas MIBs. Cada empresa recebe um OID próprio a partir do qual pode criar quantos OIDs necessitar.

Atualmente a *Internet Assigned Numbers Authority* (IANA) gerencia todas as atribuições de números de empresa privada para indivíduos, instituições, organizações e empresas. Por exemplo, o número da empresa *Cisco Systems* é 9, e a OID base para o respectivo espaço de objetos é definida como:

iso.org.dod.internet.private.enterprises.cisco ou 1.3.6.1.4.1.9

Para escrever a MIB utilizada no estudo de caso apresentado, foi solicitado ao IANA um número de "*Enterprise*", cujo valor atribuído foi 15784, ou seja, a MIB desenvolvida para o estudo de caso posiciona-se no ramo da árvore, abaixo do seguinte OID: 1.3.6.1.4.1.15784.

3.3 MIB-II a MIB padrão para a Internet

A MIB-II, definida como *iso.org.dod.internet.mgmt.1* ou 1.3.6.1.2.1, é um grupo de gerenciamento muito importante, porque todo dispositivo com suporte para o SNMP deve aceitar, pelo menos, a MIB-II [20]. Esta MIB é definida na RFC1213.

Aceitar a MIB-II significa que o agente deve conter a implementação dos objetos definidos pela MIB em seu código e o gerenciador deve saber interpretar as mensagens vindas com suas informações.

Por exemplo, qualquer gerenciador deverá ser capaz de consultar e interpretar a informação *system.sysDescr.0* vinda de um agente e, qualquer agente SNMP deverá retornar a descrição do objeto gerenciado quando for requisitado sobre *system.sysDescr.0*, uma vez que *system* é um nó da MIB-II e que *sysDescr* é nó de *system*.

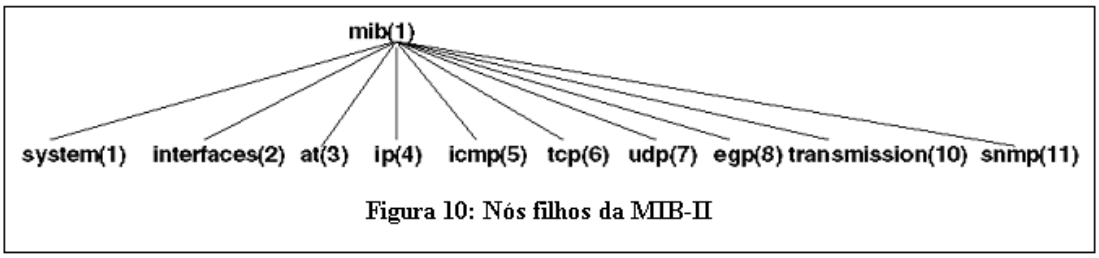

Figura 10: Nós filhos da MIB-II

Abaixo são mostrados os 10 grupos básicos implementados na MIB-II[57].

- *system* (1.3.6.1.2.1.1) – Define uma lista de objetos pertencentes à operação do sistema, tais como: tempo de funcionamento, contato e nome do sistema.
- *interfaces* (1.3.6.1.2.1.2.) – Rastreia o status de cada interface em uma entidade gerenciada.
- *at* (1.3.6.1.2.1.3) – *at* significa *address translation* e sua função é manter compatibilidade com versões anteriores
- *ip* (1.3.6.1.2.1.4) – Rastreia os diversos aspectos do IP
- *icmp* (1.3.6.1.2.1.5) – Rastreia os diversos aspectos do ICMP
- *tcp* (1.3.6.1.2.1.6) – Rastreia o estado das conexões TCP
- *udp* (1.3.6.1.2.1.7) – Rastreia os diversos aspectos do UDP
- *egp* (1.3.6.1.2.1.8) - Rastreia diversos dados estatísticos do EGP
- *transmission* (1.3.6.1.2.1.10) – Não existem atualmente objetos definidos para este grupo, porém, algumas MIBs específicas são definidas a partir desta subárvore
- *snmp* (1.3.6.1.2.1.11) - Avalia e rastreia aspectos do SNMP

Cada um dos grupos acima também representa a raiz de uma árvore com nós específicos, devendo estar presentes em qualquer implementação de SNMP por se tratar da MIB-II.

4 Estudo de Caso: Implementação do gerenciamento no laboratório de testes

O propósito deste estudo de caso foi resolver uma necessidade de gerenciamento de um processo não convencional, aplicando-se conceitos de gerenciamento de redes e SNMP, pois acredita-se que, além da contribuição oferecida pela solução ao laboratório e da coerência da apresentação de um resultado prático como dissertação em um mestrado profissional, a experimentação auxilia a fixação do conteúdo estudado.

O trabalho teve como objetivo a implementação do gerenciamento nos sistemas de movimentação e de aquisição de dados de um laboratório de ensaios do Agrupamento de Sistemas de Controle do IPT, pois no ano de 2002 seu laboratório foi credenciado pela Agência Reguladora dos Sistemas de Transportes Públicos Delegados do Estado de São Paulo, passando a exercer a função de uma entidade certificadora de sistemas de cobrança automática de tarifas de pedágio.

Há no Agrupamento de Sistemas de Controle do IPT, um mecanismo capaz de simular situações que acontecem entre os sistemas de cobrança automática de tarifas e os automóveis que passam por uma estação de pedágio, construído especialmente para certificar este tipo de sistema. Através das situações simuladas é possível mensurar a confiabilidade de um determinado subsistema para que lhe seja outorgada a certificação.

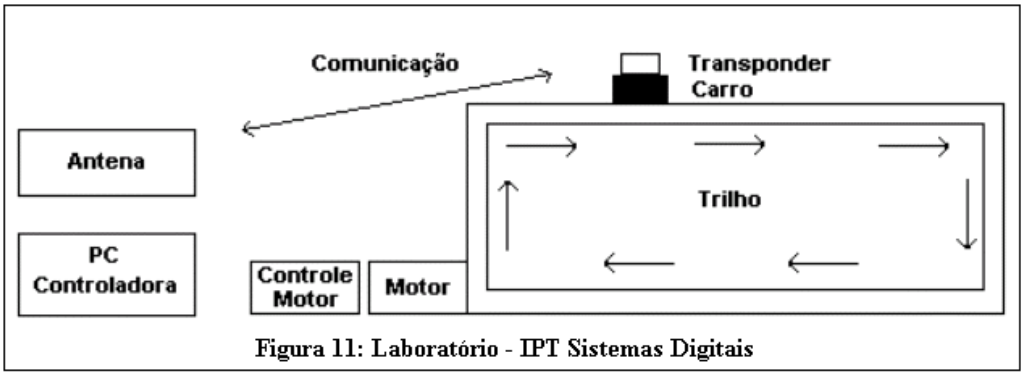

Figura 11: Laboratório - IPT Sistemas Digitais

Conforme a figura 11, este protótipo é integrado por dois sistemas com propósitos distintos. Um é o sistema de movimentação composto por um conjunto de trilhos, carros

e motor, que tem como objetivo simular uma rodovia com automóveis passando por ela. O trilho é feito de madeira e os carros são placas de madeira, colados em uma esteira de borracha movimentada por um motor. Este motor pode ser acelerado ou desacelerado dependendo da necessidade do teste ser realizado.

Outro sistema integrante do protótipo é o sistema de comunicação que trata do sistema de cobrança a ser avaliado. Em uma situação real, estes sistemas são compostos de uma ou mais antenas, variável determinada pela largura da rodovia, porém, para as situações simuladas no laboratório é necessário apenas uma antena. Além da antena há também uma placa controladora e um sistema para interface. A antena tem a função de interagir com os *transponders*, dispositivos colados aos veículos. Estes dispositivos contêm toda a identificação do veículo como dados do motorista, número do chassis, placa e outros. A placa controladora é a parte do hardware que traduz as informações provenientes da antena para o sistema de interface, o qual permite que os usuários e funcionários do pedágio coletem os dados de cobrança.

Os testes realizados pelo IPT, para a certificação, consistem em inúmeras situações como:

- Acelerar o veículo de acordo com vários perfis de velocidade e avaliar o comportamento do sistema de registro quando este passa pelo ponto de medição
- Alterar o ângulo de incidência do feixe de rádio freqüência emitido pela antena, a altura da mesma ao solo, obstruir a passagem do sinal com algum obstáculo e verificar como isto é resolvido pelo sistema
- Anexar transponders com codificações inválidas ou não codificados ao veículo para verificar se o sistema acusa as irregularidades.
- Manter o veículo por longo período no ponto de medição simulando avaria no mesmo e observar se o sistema registra múltiplas passagens.

Alguns destes testes consistem em deixar os veículos passando seguidamente pelo trilho. O sistema torna-se sujeito a erros ou a falhas tais como: problemas eletro-mecânicos, como aquecimento da antena, desalinhamento das correias que movem a

esteira, quebra do suporte do veículo, falhas na controladora, etc. Os testes podem acontecer durante dias, exigindo constante supervisão técnica.

Uma das soluções possíveis para este problema é a criação de condições para que o ambiente do laboratório seja gerenciado remotamente, através de qualquer terminal conectado à rede do IPT. Os dados obtidos no ensaio seriam trabalhados pelo sistema de interface e os dados referentes ao funcionamento do ambiente, gerenciados por esta solução.

Baseado neste conceito, o presente estudo de caso foi desenvolvido visando o gerenciamento do ambiente do laboratório. Dentre os desafios encontrados, e que motivaram um aprofundamento no estudo da arquitetura do SNMP, constatou-se que os equipamentos presentes no laboratório não eram gerenciáveis, impondo o desenvolvimento de mecanismos que alterassem tal condição.

Para endereçar as questões relativas ao gerenciamento do laboratório foi utilizada a seguinte metodologia:

1. Análise do ambiente para a confirmação da adequação do SNMP
2. Mapeamento dos produtos compatíveis com SNMP

2.3.Estudo das opções de gerenciamento através do SNMP

3.4.Construção de mecanismos necessários à implementação do gerenciamento no laboratório

A seguir, são detalhados as atividades e os produtos de cada etapa.

4.1 Análise do ambiente para a confirmação da adequação do SNMP

Identificou-se os padrões de gerenciamento mais comuns, dentre os protocolos e arquiteturas disponíveis, com a finalidade de selecionar a melhor solução de gerenciamento para o laboratório.

Foram consideradas as seguintes opções de gerenciamento:

- Baseado em CLI
- Baseado em HTTP
- Baseado em SNMP
- Baseado em CMIP

4.1.1 Gerenciamento baseado em CLI

De acordo com Tony Kenyon em seu livro *Data Netorks*, publicado pela *Digital Press* no ano de 2002, oO CLI (Interface por Linha de Comando) é uma possibilidade que alguns equipamentos fornecem para acessar suas configurações através de uma porta serial ou de uma interface *Telnet*[27]. Cada equipamento apresenta sua linguagem interna CLI.

O usuário aplica os comandos e uma tarefa ou agente os verifica e executa.

A comunicação equipamento-gerente acontece através de alertas enviados a partir dos equipamentos para a aplicação gerente da rede. Estes alertas são impressos nos consoles dos usuários.

Uma vantagem desta técnica é que ela é rápida, consome poucos recursos da rede, podendo até ser utilizada através de linhas discadas quando a rede está inoperante.

Em um gerenciamento baseado em CLI, o administrador da rede configura programas de uma forma centralizada e os envia para cada equipamento através de *Telnet* ou de uma conexão tipo console. Para que isto ocorra, é necessário que cada equipamento tenha um endereço IP associado.

O gerenciamento baseado em CLI tem servido aos administradores por vários anos, mas apresenta algumas desvantagens como [27]:

- O usuário deve aprender algumas CLIs diferentes, pois elas não são padronizadas. Cada equipamento possui a sua própria CLI.
- Os mecanismos de segurança são muito simples, pois a senha de acesso é apenas um texto ASCII transferido sem criptografia pelo meio físico. Alguns protocolos de segurança como SSL e SSH podem ser disponibilizados, porém dependem do fornecedor do equipamento.

Alguns administradores de rede ainda preferem utilizar CLIs, pois, uma vez aprendidos, podem ser muito eficientes para configuração e diagnóstico rápidos. Geralmente as CLIs suportam lotes de instruções sendo executados de uma vez.

O agente CLI tem algumas funções pré-definidas como:

- Autenticação do usuário e determinação de seus acessos
- Validação dos comandos CLI
- Determinação da ação a ser realizada
- Realização de chamadas para o sistema

Ao analisar esta possibilidade para o gerenciamento dos equipamentos do laboratório percebeu-se que não suportam o CLI, por não se tratarem de equipamentos desenvolvidos propriamente para rede. O fabricante do sistema de comunicação fornece funções para acesso à controladora, mas estas dependem do desenvolvimento de programas específicos, extrapolando o conceito do CLI.

4.1.2 Gerenciamento baseado em HTTP

Segundo T. Sridhar em seu livro *Designing Embeded Communication Software* publicado pela CMP Books em 2003 [34], Oo gerenciamento baseado em HTTP oferece a

possibilidade de gerenciar uma rede independentemente da localização desta, em tempo real, utilizando algum meio de acesso, seja remoto ou local[34]. Tem como pré-requisito um servidor *web* executado geralmente na mesma rede que o equipamento gerenciado, o qual recebe e envia dados em HTML ou XML com informações para seu gerenciamento. Desta forma, o administrador da rede pode gerenciá-la remotamente através de qualquer conexão IP utilizando um aplicativo (*browser*) padrão de fácil utilização.

Esta opção tem se tornado comum para o gerenciamento de sistemas distribuídos, sendo quase sempre utilizada no lugar do CLI para configuração e controle, pelo fato de permitir a configuração de equipamentos através de campos com dados pré-definidos que limitam a manipulação de valores, reduzindo, assim, a possibilidade de erros.

Dentre inúmeras vantagens encontra-se a facilidade de utilização, a possibilidade de consolidação do gerenciamento de muitos equipamentos através do mesmo aplicativo *(browser)*, a independência de plataforma e a existência de mecanismos padrão de segurança, como por exemplo, protocolos SSH e SSL.

A principal desvantagem desta abordagem é que tende a ser lenta e sem escala, quando estiver sendo gerenciando centenas de equipamentos.

Ao ser analisada a opção de gerenciamento baseado em HTTP para o laboratório, constata-se que o conceito é aplicável e oferece as facilidades desejadas para o monitoramento, porém, mesmo com a implantação de uma solução que permita a visualização das informações em HTTP, ainda é necessário implementar um protocolo que controle os equipamentos, não projetados com mecanismos de supervisão, e que permita comandar ações (consulta ou alteração) de estado dos mesmos.

4.1.3 Gerenciamento baseado em SNMP

Conforme exposto no capítulo “SNMP” e, analisando este protocolo como possível solução para o gerenciamento do laboratório, constataram-se as seguintes vantagens:

- Permite o gerenciamento através de rede IP possibilitando o uso através de aplicativos (*browsers*) (englobando o conceito de gerenciamento por HTTP).

- É de domínio público, possibilitando acesso a RFCs e facilitando o entendimento do protocolo.
- Existem vários produtos para gerenciamento SNMP no mercado, inclusive fornecidos gratuitamente.
- Sua concepção é simples, de fácil entendimento e implantação.
- Consome baixo volume de banda e recursos (comparado com modelos de gerenciamento orientados à conexão).
- Os maiores fornecedores de equipamentos para rede incluem o suporte ao SNMP em seus produtos.

A utilização do SNMP durante estes anos permitiu que diversas MIBs fossem desenvolvidas para ampliar suas funcionalidades, como é o caso do *RMON* [16,17] *MIB*, *Script MIB, Remote Ping, Traceroute, and Lookup Operations MIB* e muitas outras. Estas MIBs poderão ser utilizadas, com a finalidade de adicionar recursos ao gerenciamento do laboratório, quando a complexidade do mesmo justificar este tipo de solução, pois, o laboratório pode ter suas configurações alteradas de acordo com o tipo de teste realizado. Desta forma, a solução de gerenciamento do mesmo deve ser escalável.

4.1.4 Gerenciamento baseado em CMIP

Conforme exposto anteriormente no capítulo “SNMP”, pode-se considerar que a maior vantagem do uso do CMIP frente ao SNMP é que:

- Os métodos de objetos CMIP além de guardar informações, também são utilizados para realizar tarefas, o que não acontece com o SNMP.
- O CMIP é um sistema mais seguro e foi construído para suportar autorização, controle de acesso e *logs* de segurança.
- O CMIP fornece melhor reporte de condições atípicas da rede.

Apesar destas vantagens, o CMIP é raramente implementado em ambientes de LAN devido à pequena quantidade de equipamentos de rede que suportam seu uso. Além disso,

também requer muitos recursos de processamento e armazenamento de informações. Outra dificuldade é a complexidade de sua programação, necessitando de profissionais especializados, tanto para a implementação quanto para a manutenção dos sistemas gerenciadores.

Portanto, a utilização do CMIP como base para o gerenciamento do laboratório não se mostra vantajosa, dada a necessidade de recursos adicionais e a adição de certa complexidade ao gerenciamento, o qual não se justifica pela simplicidade dos equipamentos a serem gerenciados.

4.2 M

Eapeamento dos produtos compatíveis com SNMPstudo das opções de gerenciamento através do SNMP

Para definir a melhor forma de implementar o SNMP no laboratório, estudaram-se algumas opções de aplicações disponíveis, pois são diversas as aplicações que implementam o SNMP. Estas aplicações variam desde bibliotecas de programação que permitem criar utilitários exclusivos, até plataformas completas e dispendiosas de gerenciamento de rede.

O Software de gerenciamento é enquadrado em cinco categorias [10]:

- Agentes SNMP
- Suítes NMS
- Gerenciadores de componentes (específicos do fornecedor – proprietários)
- Software de análise de tendências
- Software de suporte

4.2.1 Agentes SNMP

Um agente é um software que controla todas as comunicações SNMP com qualquer dispositivo compatível com o protocolo [19,06]. Em alguns dispositivos, o software do agente é incorporado ao próprio dispositivo e não requer instalação. Porém, em outros é necessário instalar o agente como parte de um de software adicional.

O que diferencia o uso dos agentes são itens como capacidade de extensão, confiabilidade, portabilidade de plataforma e etc, que variam de importância de acordo com o tipo de rede e elemento a ser gerenciado e como se pretende gerenciar.

Segue alguns agentes existentes e o endereço na internet onde maiores informações podem ser encontradas:

- HP extensible SNMP Agent – http://www.openview.hp.com
- Sun Microsystems SNMP Agent – http://www.sun.com

- Concord SystemEDGE – http://www.empire.com
- Microsoft SNMP Agent – http://www.microsoft.com
- Net-SNMP – http://net-snmp.sourceforge.net

4.2.2 Suítes NMS

Uma Suíte de NMS, neste contexto, é um software que engloba vários aplicativos em um único produto. No cenário do gerenciamento a NMS é um dos elementos mais importantes. Os produtos NMS permitem uma visão geral da rede de seus servidores, roteadores, comutadores etc. Na maioria dos casos, essa visão é uma representação gráfica da rede, com vários rótulos e ícones interessantes. A facilidade de representação gráfica da informação gerenciada, será um dos fatores considerados para a implementação final do gerenciamento no laboratório. Abaixo são listados algumas possibilidades, que poderão ser consideradas.

- HP OpenView NNM e ITO – http://www.openview.hp.com
- Tivoli Netview – http://www.tivoli.com/products/index/netview/
- Castle Rock SNMPc – http://www.castlerock.com
- BMC – http://www.bmc.com
- Computer Associates Unicenter TNG Framework – http://www.cai.com
- Veritas NerveCenter – http://www.veritas.com
- OpenRiver – http://www.riversoft.com
- GxSNMP – http://www.gxsnmp.org
- Tkined – http://www.home.cs.utwente.nl/~schoenw/scotty/
- OpenNMS – http://www.opennms.org

4.2.3 Gerenciadores de Componentes

É um software voltado para um determinado tipo de fornecedor ou função, por exemplo, um gerenciador de componentes pode ser um produto para gerenciar um conjunto de modens.

Como alguns destes produtos são específicos do fornecedor, é fácil comprar algo que se tornará menos útil do que o esperado. Por exemplo, o CiscoView, um produto da Cisco que faz muitas coisas interessantes como exibir a parte traseira dos roteadores. Porém, este produto simplesmente não funciona com um roteador Nortel, 3Com etc.

São vários os gerenciadores proprietários existentes no mercado atualmente. Um destaque pode ser dado para os seguintes:

- Sun Management Center – http://www.sun.com/sysmon/
- Cisco Works 2000 – http://www.cisco.com
- 3Com Total Control – http://www.3com.com
- Aprisma – http://www.spectrummgmt.com

4.2.4 Análise de tendências

Estes produtos fornecem um registro histórico permitindo a análise da desempenho da rede, dos erros e alertas obtidos. Permitem também uma análise mais criteriosa para a detecção de problemas recorrentes ou em potencial.

Atualmente os mais utilizados produtos de análise de tendência são:

- Concord eHealth – http://www.concord.com
- Trinagy TREND – http://www.desktalk.com
- MRTG – http://www.mrtg.org
- Cricket – http://cricket.sourceforge.net
- InfoVista – http://www.inforvista.com

4.2.5 Software de Suporte

O software de suporte é um pacote que contém todo o tipo de recurso para ser utilizado em conjunto com os outros sistemas citados anteriormente. Alguns deles podem ser utilizados para escrever aplicativos de gerenciamento independente.

Alguns destes utilitários são:

- Perl – http://www.perl.com
- SNMP Support for Perl – http://www.cpan.org
- WILMA
- NET-SNMP C Library – http://net-snmp.sourceforge.net
- NET-SNMP Perl Module – http://www.cpan.org
- A3Com – http://www.kernel.org/software /A3Com/
- SNMP++ - http://rosegarden.external.hp.com/snmp++/
- Netcool – http://www.micromuse.com

1.4.3 Análise das opções baseadas em SNMP

De acordo com a arquitetura de gerenciamento idealizada para o laboratório, é necessário, em um primeiro momento, utilizar um sistema de NMS para permitir a interação, através da internet, entre os profissionais do laboratório e os equipamentos. Também é necessário utilizar um agente SNMP para realizar a comunicação com os equipamentos e sistemas a serem monitorados, porém, através de uma análise da situação do laboratório percebeu-se que os equipamentos e o sistema utilizados não implementam o SNMP, tornando necessária a utilização de ferramentas que permitam a extensão das funcionalidades dos agentes SNMP para contemplar seus gerenciamentos.

Dentre as opções de ferramentas existentes com esta característica foram analisadas:

- Implementação do AgentX (RFC 2257)
- Utilização de MIBs específicas
- Extensão do agente principal NET-SNMP
- Extensão do agente SystemEDGE
- Extensão do agente principal HP Openview

4.3.1 Implementação do AgentX

O protocolo AgentX (RFC2257)[42] fornece uma solução padronizada para o caso de extensibilidade de agentes. Ele é concebido para ser, independente da versão do SNMP, um método para que os subagentes se comuniquem com o agente principal. Cada subagente trabalha em seu próprio espaço, fornecendo uma alternativa mais robusta aos monolíticos agentes SNMP. Com a complexidade dos processos executados nos servidores e aplicações diariamente, a utilização do AgentX assume grande importância.

Um agente SNMP é um software residente em um dispositivo gerenciado, que responde a solicitações SNMP e gera trap assíncronas. A necessidade do AgentX surge da impossibilidade de adição e remoção de objetos da MIB[23] enquanto um agente estiver em execução, ou seja, da falta de um método padrão para a extensão de funcionalidade de um agente. O SNMP *Multiplexing Protocol* (SMUX, RFC 1227)[43] foi uma tentativa anterior de fornecer padronização para a extensão de agentes, mas é considerado deficiente.

Em um ambiente SNMP AgentX, o agente é composto por uma única entidade de processamento, denominada agente principal e uma ou mais entidades de processamento denominadas subagentes [06]. O agente principal e os subagentes podem residir no mesmo dispositivo ou podem se comunicar através de TCP ou *sockets*.

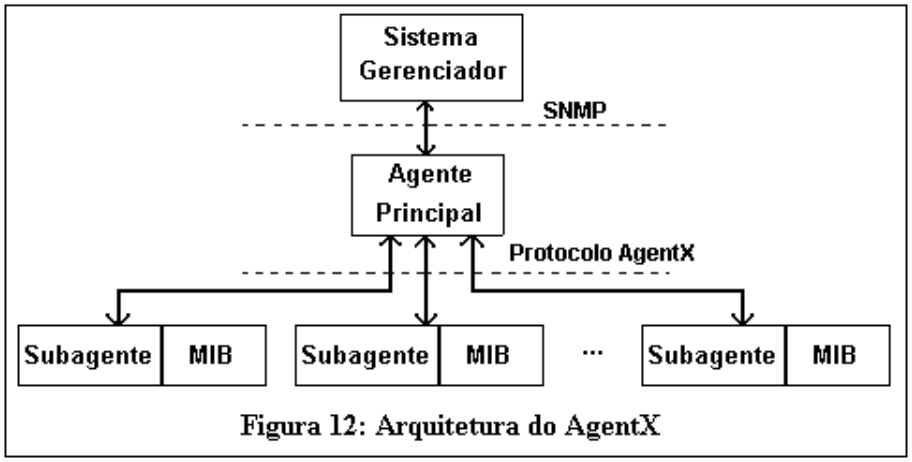

Figura 12: Arquitetura do AgentX

Na [illegible] SNMP quanto por AgentX. A função do agente principal é manter uma tabela associando cada subagente com as MIBs que ele gerencia. Quando o agente principal recebe uma

requisição SNMP, encontra o subagente responsável para informação e inicia uma requisição AgentX.

Os subagentes AgentX são responsáveis por fornecer acesso à informação. Quando um subagente é inicializado, contacta o agente principal e informa ao mesmo, as MIBs que gerencia. O subagente não tem conhecimento de SNMP e nem da existência de outros subagentes. O protocolo AgentX também define como o agente principal deve evitar e resolver conflitos.

Segundo o artigo *An Overview of the AgentX Protocol* de Mat White, membro da *Carnegie Mellon University*, publicado no *The Simple Times* volume 6 em Março de 1998, sSem uma abordagem padronizada, seria muito difícil aos fornecedores rastrearem as possíveis extensões de agentes aceitas pelas diversas plataformas[25]. O AgentX tenta endereçar esta questão, oferecendo aos fornecedores uma interface consistente para extensão de agentes. Também define o conceito de regiões MIB ou conjunto de variáveis gerenciadas. Subagentes são responsáveis por registrarem MIBs junto a um agente principal. Na prática, isso significa que os fornecedores criam um subagente para cada MIB que implementam; por exemplo, um subagente de RMON, um subagente de MIB-II, um subagente de recursos de equipamento e outros, oferecendo uma maneira de adicionar e remover funcionalidades de um agente, sem prejudicar a real operação entre uma NMS e o agente.

A utilização do AgentX no laboratório demonstra-se vantajosa pois, havendo intensão de crescimento do número de equipamentos, a criação de subagentes ocorre à medida que mais equipamentos são incorporados ao gerenciamento. Outra vantagem é a sua fácil implementação através do NET-SNMP que oferece um método bastante simples para a extensão de agentes através do protocolo AgentX.

4.3.2 Utilização de MIBs especiais

Algumas MIBs foram desenvolvidas para o laboratório do IPT, com a finalidade de adicionar funcionalidades ao SNMP oferecendo mais flexibilidade no gerenciamento. Analisou-se:

- Script MIB (RFC 2593)[48]
- Remote Ping, Traceroute, and Lookup Operations MIB (RFC 2925)[56]
- RMON I e II[46,45]

4.3.2.1 Script MIB

O Script MIB (RFC 2593)[48] foi criado pelo grupo de trabalho de gerenciamento distribuído do IETF. Seu propósito é fornecer mecanismos para distribuição de *scripts* de gerenciamento pré-definidos em equipamentos ou sistemas remotos. A possibilidade de executar os *scripts* em equipamentos ou sistemas remotos pode reduzir a carga de processamento na NMS central possibilitando, também, a verificação local dos estados dos equipamentos.

No artigo *Introduction to the Script MIB* de David Levi, um dos membros do grupo de trabalho de gerenciamento distribuído do IETF, publicado no volume 7 do *The Simple Times* em novembro de 1999, é colocado que o O modelo tradicional do SNMP consiste em uma única estação de gerenciamento centralizando a informação proveniente dos agentes, o que restringe o funcionamento à medida que as redes crescem e que o Script MIB endereça uma solução para este problema através da distribuição de um conjunto de processamento e dados para outros equipamentos na rede[26].

A distribuição dos *scripts*, conforme o Script MIB, pode ser realizado através de um modelo no qual o equipamento captura os *scripts* a serem processados por ele através de um endereço pré-definido ou por um modelo no qual a estação de gerenciamento "empurra" os scripts através da operação SNMP *SetRequest*. Estes *scripts* são então armazenados em uma tabela que o equipamento acessa.

A execução dos *scripts* é controlada por uma tabela que contém também um objeto encarregado de iniciá-la.

De acordo com o presente caso de estudo, a utilização do Script MIB pressupõe a existência de agentes SNMP se comunicando com os equipamentos, tratando-se de uma alternativa que pode ser considerada no futuro.

Outro fator que inviabiliza a utilização do Script MIB no laboratório, é que esta tem como foco grandes redes de comunicação, não sendo o caso atual, pois uma única estação é capaz de gerenciar todo o ambiente.

Porém, a utilização do Script MIB poderá ser aplicada ao gerenciamento de sistemas de cobrança automática de tarifas de casos reais onde várias estações de pedágio podem ser gerenciadas por uma central de operações.

4.3.2.2 Remote Ping, Trace route, and Lookup Operations MIB

A *Remote Ping, Trace route e Lookup Operations MIB* (RFC 2925) define as operações de *ping*, *traceroute* e *lookup* aplicadas a partir de um equipamento remoto. As operações de *ping* e *traceroute* são úteis para o gerenciamento de uma rede, pois permitem determinar se existe conexão entre dois equipamentos ou sistema, identificando o caminho trilhado por esta conexão. A operação de *lookup* permite converter endereços IPs em nomes de máquina, e vice-versa, realizando consultas a DNSs.

O *Ping* utiliza a facilidade "*ECHO*" fornecida pelo ICMP *(Internet Control Message Protocol),* mas também é possível implementá-lo por outros métodos:

- Utilizando a porta echo UDP
- Programando uma consulta SNMP
- Programando uma tentativa de conexão TCP

É importante para a estação de gerenciamento conhecer o método utilizado. Diferentes métodos levarão a diferentes resultados, uma vez que o protocolo e o processo resultante será diferente.

A operação de *Lookup* permite obter o nome de equipamentos remotos *(hosts)* e seus respectivos endereços IP.

Uma requisição SNMP é iniciada em um equipamento que se comunica com um equipamento remoto onde o agente SNMP está implementado. Este recebe a requisição e executa funções *ping*, *traceroute* ou *lookup* e retorna seus valores ao equipamento solicitante.

A utilização desta MIB para o caso do laboratório não se justifica em um primeiro momento, devido pois outras ferramentas oferecem funções de gerenciamento mais apropriadas ao tamanho e a simplicidade do ambiente. Porém, para casos onde existam muitos equipamentos a serem gerenciados, como pode ser o caso de aplicações reais de sistemas de cobrança automático de tarifas em pedágios, esta MIB pode ser de grande valia.

4.3.2.3 RMON I e II

Segundo Rajesh Sharma em seu livro *Cisco Security Bible* [33], publicado em 2002, o O protocolo de gerenciamento de rede SNMP permite que estações de gerenciamento obtenham periodicamente informações dos elementos gerenciáveis para a verificação de seu funcionamento e uso[33]. Porém, o SNMP não estava preparado para o gerenciamento de tráfego e monitoração de variações de desempenho. Foi necessário desenvolver uma tecnologia que permitisse antever potenciais problemas e atuar preventivamente. A solução encontrada foi implementar uma MIB e, mais tarde, um conjunto de MIBs, conhecido como RMON, abreviação de *Remote Monitoring* (monitoramento remoto).

O RMON é executado nos equipamentos locais, em software conhecido como agente inteligente (implementado em equipamentos tais como roteadores e switches) de forma a coletar dados mais eficientemente.

Estes agentes são sempre referidos como Módulos de Coleta de Dados (DCMs) ou *probes*. *Probes* podem ser configuradas pelo gerente da rede para ativar determinados mecanismos como *traps* SNMP, a partir de uma estação remota, e monitorar eventos específicos.

O RMON define nove grupos de informações relevantes, aplicáveis em todas as LANs, incluindo objetos especialmente desenvolvidos para monitorar segmentos *Ethernet* remotos.

Há duas versões de RMON, chamadas de RMONv1 (RFC2819)[46] e RMONv2 (RFC2021)[45].

A versão 1 do RMON fornece um conjunto de funcionalidades, para administradores de rede, que não era possível obter com as ferramentas baseadas apenas em SNMP.

Permite o armazenamento de histórico de desempenho dos equipamentos e o ativamento automático de alarmes em determinadas condições de rede.

A versão 2 do RMON fornece dados sobre o tráfego na camada de rede. Isto possibilita aos administradores analisarem o tráfego por tipo de protocolo. O RMONv2 permite verificar o desempenho no nível de porta em roteadores ou switches. Também permite que uma única *probe* de RMON monitore muitos tipos de protocolo trafegando no segmento e tenha muito mais flexibilidade na consulta de dados históricos e medição do tempo de resposta da rede.

O RMONv1 é caracterizado por implementações de alto custo e elevado índice de incompatibilidades entre fornecedores. Os requisitos de sistema podem ser significativos, especialmente se as funcionalidades do RMON forem habilitadas para todas as interfaces. Estes inconvenientes não desapareceram no RMONv2.

Os agentes RMON operam em modo *offline*, um monitoramento não interativo com os administradores. Um agente RMON é configurado remotamente e ativado através de uma aplicação de gerenciamento. O agente lê os parâmetros monitorados e registra cada informação em seu segmento local, atualizando conteúdos e contadores. A informação fica armazenada no agente até que a aplicação de gerenciamento remoto a requisite.

Através do RMON é possível detectar erros ou alterações no perfil de tráfego antes que um problema se torne crítico. O agente pode ser configurado para reconhecer uma condição indesejável e gerar uma *trap* SNMP.

As MIBs de RMON podem armazenar informações detalhadas sobre as operações da LAN. Um alto nível de detalhe pode ser utilizado para isolar problemas de desempenho ou analisar tendências para eventuais replanejamentos. Um único agente RMON pode fornecer informações para muitas aplicações de gerenciamento. Uma tabela mantida dentro do agente informa qual dado e qual intervalo deve ser enviado para um endereço IP específico.

O RMON foi concebido para operações de monitoramento contínuo onde a comparação histórica de desempenho se mostra relevante. Os ensaios no laboratório do IPT são eventos discretos e pontuais, com pouco apelo para implementações de RMON nos moldes convencionais. Além disso, a ausência de agentes de monitoração nos instrumentos do laboratório, desestimulou sua equipe de operação a formular modelos de

gerenciamento que fizessem uso intensivo de automação. Embora não incluída nesse trabalho, pela ausência de um modelo de gerenciamento durante a definição do escopo do projeto, sugere-se que o laboratório invista nessa tecnologia, seja na exploração de usos não convencionais aplicaveis ao próprio laboratório, seja em implementações de gerenciamento nos sistemas de pedágios reais, onde a opção pelo RMON torna-se interessante pois as atividades são em período integral e onde o histórico de desempenho dos equipamentos e a capacidade de previsibilidade tornam-se fundamentais para garantir alta disponbilidade do serviço.

4.3.3 Extensão do agente SystemEDGE

O SystemEDGE oferece um agente extensível que, através de uma MIB privada, define uma tabela denominada *extensionGroup* (OID é 1.3.6.1.4.1.546.14), cujo objetivo é permitir que novos objetos sejam adicionados.

Os novos objetos são definidos em um arquivo de configuração *(sysedge.df)* que informa ao agente as OIDs estendidas a que ele deve responder. O formato da linha a ser adicionada no *sysedge.cf* é:

"extension número do objeto tipo acesso 'comando'"

onde:

- ***extension*** informa que é uma extensão pertencente ao *extensionGroup*,
- **número do objeto** será responsável por identificá-lo através da árvore SNMP em *1.3.6.1.4.1.546.14.número do objeto*,
- **tipo** corresponde ao tipo de dado envolvido,
- **acesso** informa se é um objeto somente leitura ou leitura e gravação
- **comando** é o programa que o agente executará quando essa OID for consultadada por uma NMS.

Alguns exemplos de configuração de objetos estendidos são:

- *extension 1 Integer Read-Only '/usr/local/Script.sh'* (neste exemplo, o programa Script.sh será executado quando a OID 1.3.6.1.4.1.546.14.1 for consultada)
- *extension 54 Counter Read-Write '/usr/local/Program.pl'*

Estender o agente permite criar programas exclusivos para fazer o que é necessário, possibilitando obter informações sobre dispositivos ou programas sem suporte ao SNMP, desde que seja escrito um programa que consulte o status destes dispositivos ou programas.

Para que a extensão funcione o programa precisa atender a duas exigências:

- Todas as solicitações de *set*, *get* e *getnext* devem gerar saídas. Para *get* e *getnext*, a saída do programa deve ser o valor real do objeto solicitado. Isto significa que o programa que busca as informações necessárias deve retornar um único valor. Para uma solicitação de *set*, o programa deve retornar o novo valor do objeto. A solicitação falhará se não existir uma saída.
- O programa deve imprimir as informações a serem retornadas (com base no tipo de solicitação), seguidas por um caractere da nova linha. O agente analisará somente até esse caractere. Se o caractere de nova linha for o primeiro encontrado, o agente gera um erro e retorna esse erro para a aplicação do NMS ou SNMP.

O agente extensível SystemEDGE é acompanhado por um documento contendo as diretivas em *perl* necessárias para a codificação do programa de extensão dos agentes. A única ação necessária é adicionar a lógica pertinente ao objeto gerenciado.

Também é necessário desenvolver uma MIB para o novo objeto gerenciado a adicioná-la à NMS.

Dado o exposto, o agente extensível do SystemEDGE atende às necessidades de extensão de agentes para gerenciar os equipamentos e sistema do laboratório do IPT.

Porém o SystemEDGE, ao contrário do NET-SNMP, não pode ser adquirido gratuitamente, o que o inviabiliza para os desenvolvimentos iniciais visto que, em maio deste ano, foi promulgada uma resolução do Governo do Estado que regula a aquisição de software no âmbito do setor público estadual declarando a preferência por produtos classificados na categoria de software livre.

4.3.4 Extensão do agente HP Openview

O Openview fornece um arquivo para configuração de seu agente extensível. Este arquivo *snmp.extend* permite definir novos objetos gerenciáveis.

O conceito é muito similar ao agente SystemEDGE pois consiste em informar quais são os objetos gerenciados, seguindo a estrutura de definição de uma MIB. A principal diferença entre estes dois agentes extensíveis é que o Openview permite que seja criados OIDs abaixo de qualquer ramificação na árvore SNMP, fato que com o systemEDGE não é possível, pois qualquer extensão deve respeitar a ramificação do agente systemEDGE (1.3.6.1.4.1.546.14)

O Openview possibilita incluir diretivas que executam determinados comandos ou que retornam valores caso o objeto definido seja consultado ou alterado.

Segue um exemplo de definição de um objeto denominado *fmailListMsgs*

fmailListMsgs OBJECT-TYPE
SYNTAX DisplayString
ACCESS read-only
STATUS mandatory
DESCRIPTION
“Lista de mensagens da fila de correio.”
READ-COMMAND: /usr/lib/sendmail –bp
READ-COMMAND-TIMEOUT: 10
:: = {fmail 2}

Segundo essa definição, quando *fmailListMsgs* for consultada, o comando *sendmail –bp* será executado e um resumo da fila do correio será impresso. Assim, através de uma estação de gerenciamento, é possível obter, por uma operação *snmpget* no objeto *fmailListMsgs*, o status da fila de saída do correio eletrônico.

O agente extensível do Openview também é adequado ao gerenciamento dos equipamentos e sistemas do laboratório, porém, também é um sistema comercializado, o que inviabiliza sua utilização no momento. Outra desvantagem é o fato de não ser adaptável a um hardware com poucos recursos de memória e processamento, pois seu funcionamento depende da instalação de todo o pacote HP Openview.

4.3.5 Extensão do agente principal NET-SNMP

Assim como o SystemEDGE e o Openview, o NET-SNMP permite que o agente SNMP tenha suas funcionalidades extendidas.

No NET-SNMP, a extensão de funcionalidades pode ser realizada através da extensão do agente principal ou da criação de subagentes utilizando-se, para isto, o protocolo AgentX.

A extensão do agente principal consiste em compilar o novo módulo contendo os objetos a serem estendidos em conjunto com o agente NET-SNMP. Para o desenvolvimento deste módulo são fornecidas algumas bibliotecas em linguagem C, que facilitam o desenvolvimento e garantem a compatibilidade com o SNMP.

A criação de subagentes segue o mesmo princípio para o desenvolvimento do módulo que contém o novo objeto. A diferença está na compilação onde o NET-SNMP fornece um aplicativo para a geração dos subagentes. A criação de subagentes apresenta a vantagem de não alterar o agente principal, de forma que é possível acrescentar objetos sem comprometer o sistema em funcionamento.

No próximo capítulo encontram-se mais detalhes sobre a extensão do agente NET-SNMP

4.3.6 Análise das opções baseadas em SNMP - Resumo

Dado o exposto, adotou-se o agente extensível NET-SNMP para a implementação do gerenciamento SNMP no laboratório estendendo-o com a criação de subagentes que utilizam o protocolo AgentX.

A utilização do AgentX no laboratório demonstra-se vantajosa pois, como se trata de um piloto, a flexibilidade de criação de objetos de uma forma faseada, respeitando as características mutáveis de um laboratório de testes, passa a ser um requisito fundamental.

Optar pelo NET-SNMP faz sentido, uma vez que este produto fornece as ferramentas necessárias para a geração de subagentes e, principalmente, porque, diferentemente dos demais, trata-se de um produto distribuído gratuitamente. Outra vantagem é a disponibilização dos códigos fontes e acesso fácil à documentação do produto. Também há algumas comunidades de interesse do NET-SNMP nas quais a troca de informações são frequentes, sendo bastante útil em um ambiente de experimentação como o laboratório no IPT. Apesar do NET-SNMP apresentar a informação de forma simples ao usuário final, algum software específico para a visualização da informação *(browser)* deverá ser utilizado.

Vale ressaltar que as características do gerenciamento mudam quando se analisa a utilização de ferramentas SNMP para um ambiente real de sistemas de cobrança de tarifa automática de pedágio. Neste contexto, gerenciamento distribuído, análise de desempenho através de histórico e alta disponibilidade assumem maior importância que os adotados para a implementação do gerenciamento no laboratório de ensaios.

4.45 Construção de mecanismos necessários à implementação do gerenciamento no laboratório

Para a implementação do gerenciamento no laboratório, os seguintes passos foram executados:

- Definição dos objetos a serem gerenciados
- Definição da arquitetura de gerenciamento
- Criação do arquivo de MIB
- Criação do arquivo de código dos novos objetos
- Criação dos subagentes implementando os novos objetos
- Implantação do gerenciamento no laboratório

4.4.15.1 Definição dos objetos a serem gerenciados

Definiu-se como escopo para este trabalho, um conjunto de ações representativas dos principais eventos SNMP passíveis de ocorrerem em ambiente de controle e um primeiro passo para o efetivo gerenciamento do laboratório do IPT:

- Ligar ou desligar a antena que se comunica com os dispositivos anexados aos carros
- Consultar o status da antena (ligada ou desligada)
- Consultar o valor da temperatura da antena
- Verificar mensagens de erro de comunicação nos logs do sistema

A ação de ligar e desligar a antena pressupõe uma ação que parte de um comando do gerenciador para alterar o valor do dispositivo gerenciado, que no caso, é a antena. Esta situação, além de implementar um mecanismo de utilização eficiente ao laboratório, permite demonstrar o conceito de *SET* (atribuir) do SNMP.

As ações de consultar o status da antena e o valor da temperatura estão associadas a uma operação de verificação, sem alteração de valor ou estado do dispositivo gerenciado. Esta ação implementa o conceito de *GET* (obter) do SNMP.

A verificação de mensagens de erro localizadas no *log* do sistema implementa o conceito de notificação *TRAP*(armadilha), no qual o agente informará ao gerenciador, caso um erro seja detectado, sem que este o consulte.

A implementação das ações definidas acima para o caso do laboratório, além de permitir a replicação do conceito para a implementação de outras funcionalidades, por simples analogia, permite a utilização dos conceitos básicos do gerenciamento SNMP.

5.2 Definição da arquitetura de gerenciamento

A definição da arquitetura de gerenciamento mostrou-se necessária de forma a garantir escalabilidade para a solução e possibilitar a sua implementação em ambientes heterogênios, pois o atual sistema de interface com a placa controladora é executado sobre o sistema operacional windows e o servidor que contém o ambiente gerenciador (NET-SNMP, agentes, mibs e desenvolvimentos), executa o sistema operacional Linux.

De acordo com o guia do Servidor Conectiva Linux 8 [28], o SAMBA é um aplicativo que torna possível o compartilhamento de recursos com máquinas executando o Windows. O nome SAMBA é derivado do protocolo utilizado pelo Windows para compartilhar discos e impressoras: o protocolo SMB, *Server Message Block*[29].

Através da utilização do SAMBA é possível criar redes mistas, utilizando servidores Linux e clientes Windows. O SAMBA também permite que o Linux acesse discos exportados de máquinas Windows. É através desta funcionalidade que o SAMBA viabiliza a comunicação entre o ambiente da controladora e o ambiente do gerenciador, pois pode-se compartilhar um diretório no ambiente windows e ambos os sistemas acessarem as informações contidas em arquivos.

Esta solução apresenta algumas desvantagens, como a possibilidade de corrompimento do arquivo acessado e a não sincronização entre os movimentos de leitura e gravação realizados por ambientes diferentes em momentos diferentes. Porém, nos testes realizados durante o presente estudo de caso, a implementação do SAMBA mostrou-se simples e atendeu aos objetivos perseguidos.

Sendo assim, o SAMBA foi utilizado na arquitetura de gerenciamento do laboratório para o desenvolvimento do estudo de caso apresentado, porém, para implementações em

ambientes mais críticos ou com um maior volume de dados, recomenda-se a consideração de alternativas como Sockets ou FTP[30].

A arquitetura definida para o gerenciamento do laboratório é ilustrada na figura 13.

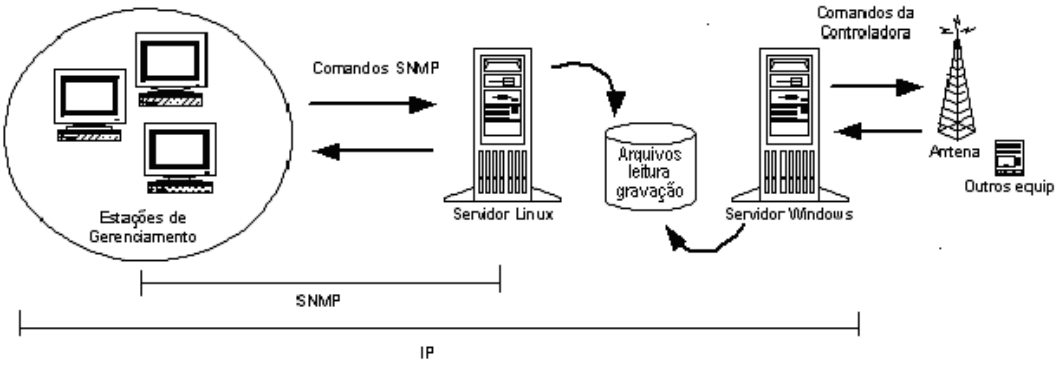

Figura 13: Arquitetura - Gerenciamento do Laboratório - IPT

5.2
4.4.35.3 Criação do arquivo de MIB

O aquivo MIB contém as definições das funcionalidades a serem processadas [22,24]. Além disso, a MIB:

- Fornece a especificação inicial do que será implementado, facilitando o desenvolvimento do agente.
- Possibilita a identificação dos objetos testados pelos seus nomes, ao invés de somente tratá-los por números. Esta vantagem é especialmente útil quando os arquivos MIB utilizam funcionalidades de outras MIBs e o tratamento através de números prejudica a depuração de erros.
- Possibilita a utilização do aplicativo *mib2c*, parte integrante da suíte NET-SNMP, para auxiliar na confecção do arquivo de cabeçalho e de codificação. Este aplicativo lê um arquivo de MIB e gera os arquivos citados já com as funções básicas, restando ao programador incluir as chamadas relativas as funcionalidades específicas.

A criação do arquivo de MIB deve seguir as diretivas da SMI *(Structure of Management Information)* (Nome, Tipo, Sintaxe e Codificação)

Um pré-requisito para a definição de uma MIB é adquirir um número dentro da árvore SNMP.

Para se atribuir o número da MIB na árvore SNMP, deve-se requerer um número ao IANA *(Internet Assigned Numbers Authority)*. É necessário preencher um formulário no site *www.iana.org* e escolher a opção de atribuição de números de protocolo *(private enterprise number)*. Este formulário contém dados básicos como nome, endereço, telefone e email da empresa e do contato requisitante. Foi cedido, para fins de realização deste trabalho, o número 15784 enviado por email quatro dias após a solicitação.

De posse do número cedido pelo *IANA*, desenvolveu-se uma MIB para implementar os seguintes objetos:

- *antena_status*
- *temp_status*
- *errox (trap)*

O objeto *antena_status* contém o valor do estado da antena que poderá ter somente dois valores: *ligada(1)* ou *desligada(0)*. Este objeto será alterado quando houver mudança no estado da antena, seja por uma solicitação do gerenciador ou por qualquer fator externo como pane, rompimento de cabo de força, tomada com problemas etc. Na implementação, o agente SNMP deverá consultar a antena periodicamente para manter este objeto atualizado.

O objeto *temp_status* terá o valor da temperatura em que a antena se encontra no momento, tratando-se portanto de um valor numérico. Este objeto será alterado pelo próprio agente que periodicamente estará verificando a temperatura da antena e atualizando o objeto.

O objeto *errox* precisa ser implementado na MIB pois possibilitará o recebimento de *traps* SNMP, quando for detectada a presença de erros no *log*. Será desenvolvida uma função, explicada posteriormente, para verificar a ocorrência destes erros e inicializar as traps.

Com base na definição dos objetos, foi desenvolvida uma MIB que está descrita no item 1 do Anexo B .

Após a definição da MIB executou-se o comando *snmptranslate*, que integra o NET-SNMP e serve para verificar a consistência da árvore da MIB e evitar problemas de reconhecimento da mesma pelo software e pelos objetos que serão criados a seguir.

A sintaxe utilizada foi *snmptranslate -M+. –mLABORATORIO -Tp -IR*

O resultado é listado abaixo:

```
+--iso(1)
   +--org(3)
      +--dod(6)
         +--internet(1)
            +--directory(1)
            +--mgmt(2)
            |  +--mib-2(1)
            |     +--transmission(10)
            +--experimental(3)
            +--private(4)
            |  +--enterprises(1)
            |     +--laboratorioMib(15784)
            |        +--antena(1)
            |        |  +-- -RW- INTEGER   antenastatus(1)
            |        |  +-- -RW- INTEGER   tempstatus(2)
            |        +--testetrap(2)
            |           +--testetrap#(0)
            |              +--errox(1)
            +--security(5)
            +--snmpV2(6)
               +--snmpDomains(1)
               +--snmpProxys(2)
               +--snmpModules(3)
```

Analisando-se o resultado listado percebe-se que a MIB criada foi listada e contém inclusive os valores *RW – INTEGER* à frente das duas variáveis criadas indicando que são variáveis inteiras e que permitem a alteração/atualização de seus valores.

4.4.45.4 Criação do arquivo de código dos novos objetos

Da mesma forma que o arquivo de MIB define as funcionalidades para as aplicações de gerenciamento, os agentes NET-SNMP necessitam de arquivos que definam suas

funcionalidades. Estes arquivos são conhecidos por arquivos de cabeçalho, que depois de compilados, funcionam como bibliotecas que o agente utiliza para seu funcionamento.

O arquivo de cabeçalho é também utilizado para informar ao compilador se o módulo criado depende de outros módulos. Recomenda-se que o código contido nos arquivos de cabeçalho seja simples, basicamente, a definição de rotinas públicas. O código pode ser gerado integral ou parcialmente pelo aplicativo *mib2c*, componente do NET-SNMP. Este aplicativo não foi utilizado neste experimento, pois reproduziu-se na íntegra o que o mesmo geraria.

A implementação de subagentes resultou na criação de um arquivo de cabeçalho denominado *antena.h*.

O código deste arquivo de cabeçalho está descrito no item 2 do Anexo B.

Os arquivos de cabeçalho possui estrutura simples. Sua função é apenas definir protótipos de funções durante a compilação.

Uma vez definido os arquivo de cabeçalho, o próximo passo foi criar os arquivos de código C que contem a implementação do subagente. Criou-se o arquivo implementando os objetos *antenastatus* e *tempstatus*.

O código deste arquivo é apresentado no item 3 do Anexo B.

A estruturação do arquivo foi realizada com o objetivo de inicializar os objetos *antenastatus* e *tempstatus*. Utilizaram-se bibliotecas do NET-SNMP no código compilado para a geração do subagente.

O objeto *trap errox* criado na MIB não foi implementado como agente pois sua função é somente traduzir o conteúdo de uma *trap* que, neste caso, é inicializada pela função de verificação do *log*, não passando, portanto, pelo agente SNMP.

4.4.55.5 Criação dos subagentes implementando os novos objetos

Definido o código do subagente foi necessário compilá-lo. Para a compilação do subagente, o NET–SNMP fornece uma rotina que, em conjunto com o código C correspondente, produz o executável que, via protocolo AgentX, se comunica com o agente principal e age conforme o determinado.

A rotina utilizada para a compilação respeita a seguinte sintaxe:

> *net-snmp-config --compile-subagent laboratorio1 antena.c*

Através desta compilação identificou-se o subagente como laboratório 1 e utilizou-se o código dos objetos descritos no arquivo *antena.c*

Ao final da compilação obteve-se uma mensagem de êxito, apresentada no item 5 do Anexo B.

Assim, foi criado o executável *laboratorio1*, utilizado como subagente do agente SNMP principal.

Para utilizar este subagente foi necessário incluir a diretiva *"master agentx"* no *snmpd.conf*, arquivo de configuração do agente do NET-SNMP. Quando o agente é inicializado este arquivo é lido. A diretiva *"master agentx"* permite suporte para o AgentX.

Testou-se os objetos criados através do comando SNMPGET do NET-SNMP. O resultado detalhado deste teste está no item 4 do Anexo B.

4.4.65.6 Implantação do gerenciamento no laboratório

Criados os subagentes, desenvolveu-se a interface entre o SNMP e o protocolo proprietário da placa controladora do sistema de comunicação do pedágio. Esta placa

implementa algumas funções que podem ser utilizadas para prover informações aos objetos recém implementados.

Desenvolveu-se, então, uma função para cada situação gerenciada

4.4.6.15.6.1 Situação I: Ligar ou desligar a antena que se comunica com os dispositivos anexados aos carros

Esta função é executada continuamente, verificando se o objeto antena_status foi alterado, caso isto tenha acontecido, é ativada uma função da controladora que força a nova situação da antena (ligar ou desligar). A função também atualiza o objeto antena_status com o estado atual da antena de forma a garantir que estas duas variáveis tenham o mesmo valor. Isto é necessário pois pode acontecer da antena estar desligada por outros meios não comandados pelo gerenciador.

A figura 14 ilustra a arquitetura utilizada para a implementação da função que permite ligar ou desligar a antena.

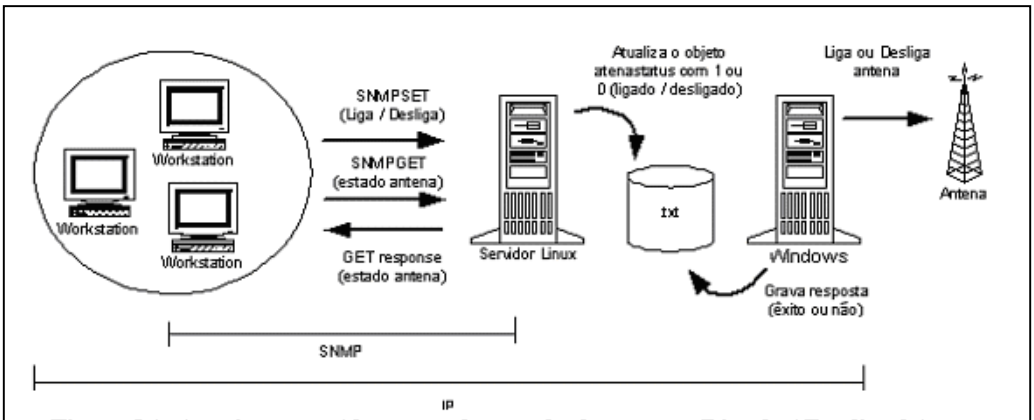

Figura 14: Arquitetura - Alteração do estado da antena (Ligado / Desligado)

Abaixo, é apresentada a estrutura da função em linguagem natural. A versão codificada é listada no apêndice C.

Função Altera_valor_antena( )

/ Esta função é implementada no ambiente Linux e tem como objetivo gravar o valor do objeto SNMP antena_status no arquivo status.txt. Este arquivo será lido por uma outra função implementada no ambiente windows que irá executar o comando correspondente de acordo com a linguagem da controladora da antena */*

{ repita sempre

{ verifica se objeto SNMP antena_status foi alterado

se sim então

{antena_status tem valor "ligado" ? se sim grava valor 1 em status.txt;

se não grava valor 0 em status.txt }

} // fim da função Altera_valor_antena( )

Função complementar( )

/ Esta função é implementada no ambiente windows. Ela lê o arquivo status.txt e executa o comando correspondente */*

{ repita sempre

{ lê arquivo status.txt

Se valor == 1 então controladora_liga_antena();

Senão controladora_deliga_antena();

} }

4.4.6.25.6.2 Situação II: Consultar o status da antena (ligada ou desligada)

Se o objeto *antena_status* tem o valor referente à posição atualizada do estado da antena, garantido pela função descrita acima, a consulta ao estado da antena poderá ser feita através do comando SNMPGET ou acessado por qualquer gerenciador SNMP de mercado.

A sintaxe para para acesso ao valor do antena_status através do comando SNMPGET do NET-SNMP é:

> snmpget –c public ipdoservidor antena_status

4.4.6.35.6.3 Situação III: Consultar o valor da temperatura da antena

A temperatura da antena poderá ser consultada através de uma operação GET do SNMP para requisitar o valor do objeto *temp_status*. Este objeto será atualizado através da função *Consulta_temperatura( )*, descrita abaixo em linguagem natural. A versão codificada é listada no apêndice C.

Função Consulta_temperatura( )

/ Esta função é implementada no Linux. Lê o arquivo temperatura.txt e atualiza (comando SET) o valor do objeto SNMP tempstatus. */*

{ repita sempre
{ Lê arquivo temperatura.txt;
verifica temperatura foi alterada
se sim então temp_status = novo valor contido em temperatura.txt}
}

Função Complementar();

/ Esta função é implementada no ambiente windows e verifica o valor da temperatura da antena através de uma linguagem própria da controladora e escreve seu valor no arquivo temperatura.txt */*

{ repita sempre
{ verifica temperatura da controladora;
escreve valor no arquivo temperatura.txt;
} }

A figura 15 ilustra a arquitetura utilizada para a implementação da função que permite verificar a temperatura da antena.

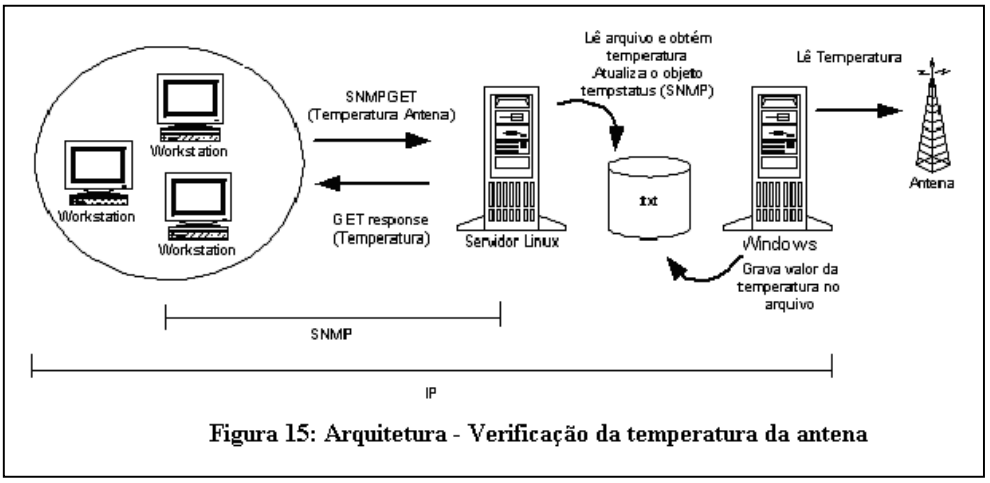

Figura 15: Arquitetura - Verificação da temperatura da antena

4.4.6.45.6.4 Situação IV: Verificar mensagens de erro de comunicação nos logs do sistema

A verificação de erros no *log* do sistema, é realizada através de uma função que lê o log e quando identifica uma letra E após o símbolo > (identificação do erro) envia uma trap SNMP com a descrição do erro localizada na linha abaixo.

Abaixo é apresentado um extrato de um log do sistema:

<20000615 14.54.08.836> A CAS>
LiC Msg 2450 Data 0x 01 00 2f 00 13 c0
<20000615 14.54.17.253> A ROM>
Released Lid 0x01 after 45ms
<20000615 14.54.17.287> E CAS> cas_Config_and_Supervision_Task
Link Controller turned off!
<20000615 14.54.17.290> W CAS>
LiC Msg 2450 Data 0x 00 00 2f 00 13 c0
<20000615 14.54.17.776> A CAS>
LiC Msg 2457 Data 0x 00 03 00 00 1c c0

No extrato apresentado, a linha com o conteúdo:

“<20000615 14.54.17.287> E CAS> cas_Config_and_Supervision_Task

Indica, através da letra E, um erro.

Desta forma, a função desenvolvida para atender a situação IV, em linguagem natural, é descrita abaixo. No apêndice C, a mesma é apresentada em forma de código.

verifica_erro_log ( )

{

Faça enquanto arquivo não acabou

{ procure por “ >” se achou “> “ veja a letra subsequente.

Se letra for “E” então envie uma trap SNMP com a mensagem que consta na linha abaixo” } }

Para a implementação desta função foram adicionados alguns controles que asseguram que o mesmo registro de log não seja verificado novamente.

A figura 16 ilustra a arquitetura utilizada para a implementação da função que permite verificar o acontecimento de erros e avisos referentes ao ambiente do laboratório.

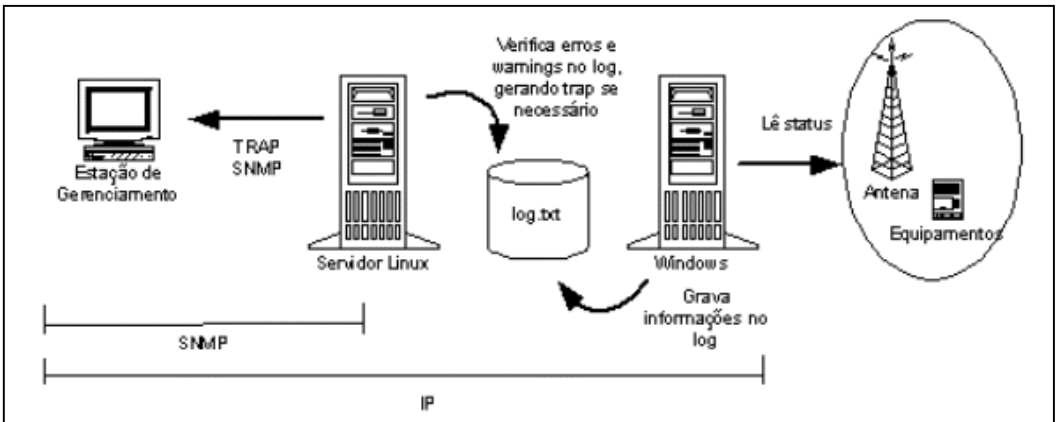

Figura16: Arquitetura - Verificação de erros e warnings no log

6 Conclusão

A imImplementaçãor do gerenciamento no laboratório de ensaios localizado no Agrupamento de Sistemas de Controle da Divisão de Mecânica e Eletricidade do Instituto de Pesquisas Tecnológicas teve como desafio o fato de que os equipamentos instalados no Agrupamento de Sistemas de Controle não dispunham de recursos nativos para suportar tarefas de gerenciamento. O Trata-se de um laboratório em questão é semelhante a outros oitenta laboratórios existentes na sede e, portanto, o sucesso obtido significa melhoria nas condições de acompanhamento de ensaios de muitas outras unidades laboratoriais. O desafio: os equipamentos instalados no Agrupamento de Sistemas de Controle não dispõem de recursos nativos para suportar tarefas de gerenciamento.

A primeira etapa foi marcada pela análise de algumas arquiteturas de gerenciamento com a finalidade de se adquirir maior conhecimento do gerenciamento de redes através do protocolo SNMP e a fixação de importantes conceitos da sua implementação. A utilização do SNMP foi confirmada como uma boa alternativa, pois a tecnologia mostrou-se aderente às características atuais e futuras do laboratório. Mas, pela ausência de suporte nativo, dominar a tecnologia não era suficiente para atingir o objetivo.

Como os equipamentos do laboratório não eram compatíveis com o SNMP, foi necessária a implementação de um agente extensível que pudesse traduzir seus estados em valores manipuláveis por qualquer aplicação SNMP. Analisaram-se algumas opções de extensibilidade utilizadas no mercado e optou-se pela implementação de subagentes através do protocolo AgentX, utilizando o NET-SNMP.

A opção pelo AgentX levou em consideração a escalabilidade proposta por este método pois a implementação do gerenciamento no laboratório deve ocorrer em fases. A opção pelo NET-SNMP baseou-se em sua disponibilidade gratuita e por oferecer mecanismos de fácil utilização para a implementação do AgentX.

Foram selecionadas quatro ações para demonstrar a implementação das operações básicas do SNMPv1 (get, set e trap), As ações selecionadas foram:

- Ligar ou desligar a antena que se comunica com os dispositivos anexados aos carros
- Consultar o status da antena (ligada ou desligada)

- Consultar o valor da temperatura da antena
- Verificar mensagens de erro de comunicação nos *logs* do sistema

Desenvolveu-se, então, uma MIB com três objetos e um subagente que se comunica com o agente SNMP principal através do protocolo AgentX que permite o acesso ao estado e a temperatura da antena através de qualquer gerenciador SNMP. Novas funcionalidades de gerenciamento poderão ser implementadas através da replicação de funções análogas, baseando-se no trabalho realizado e aproveitando a solução desenvolvida.

As informações disponibilizadas aos agentes passam por tratamento prévio obtido através de alterações no código fonte da aplicação que acessa o *firmware* do produto gerenciado. O desenvolvimento não se limitou à simples aplicação da suíte SNMP. Evoluiu para a integração com o legado e nesse sentido permitiu desvendar todos os segredos de uma implementação do processo de gerenciamento.

A utilização do SNMP para o caso em questão mostrou-se vantajosa devido à aderência de seu funcionamento com as características atuais e futuras do laboratório. Um dos resultados imediatos é a provável utilização das operações unitárias já desenvolvidas. Além disso, a equipe de operadores participou deste estudo de caso e está apta a ampliar o seu leque de funcionalidades. A visualização pelo usuário final das informações gerenciadas através de forma amigável, embora não tenha sido objeto desta dissertação, está sendo implementada por uma equipe do laboratório. Em decorrência da experiência adquirida, outros laboratórios do IPT podem ser beneficiados pelos resultados deste trabalho, considerando-se que a equipe de operadores treinada pode colaborar na capacitação de outros grupos de profissionais do instituto e na implementação do sistema de gerenciamento de outras unidades.

7 Sugestão para trabalhos futuros

Aos colegas do curso de Mestrado Profissional em Engenharia da Computação do IPT ou de outras instuições, segue uma lista com sugestões de trabalhos afins:

- Análise de outros protocolos (XML, Corba, COM) orientados ao gerenciamento deste tipo de aplicação, comparando-os com o SNMP.
- Desenvolvimento de solução de gerenciamento para equipamento específico, incluindo o desenvolvimento de MIBs e agentes
- Análise detalhada do funcionamento do protocolo AgentX e suas aplicações
- Adequação dos resultados obtidos neste trabalho à versão 3 do protocolo SNMP, de modo a tartar questões de segurança da informação.
- Estudo sobre a MIB-II visando maximizar sua utilização apontando os casos em que tende a ser subutilizada em determinadas implementações.
- Desenvolvimento de uma funcionalidade específica do software NET-SNMP em conjunto com seus desenvolvedores.
- A ampliação da solução desenvolvida neste trabalho para novas situações de gerenciamento

Referências

01 - Stallings, William. **SNMP, SNMPv2, and RMON**. 3a. Ed. Addison-Wesley, 1998

02 – Nasa, **Página do grupo de estudos de SNMP (http://www.nas.nasa.gov/Groups/LAN/ClassNotes/snmp/)**, website, 2003

03 – Campos, Geraldo Lino. **Conceitos Gerais de Gerenciamento**.
Apostila - mestrado do IPT, São Paulo: 2001

04 - Douglas W. Stevenson, **Network Management What it is and what it isn't**.
Network management whitepaper, Apr 1995

05 - Cisco Systems. **Internetworking Technologies Handbook**. Cisco Press, 3a.ed, cap 56

06 - Mauro, Douglas R**. SNMP Essencial**; tradução de Teresa Cristina Feliz de Souza, Rio de Janeiro:Campus, 2001

07- Shine, Edgar Hideyuki - **Revista PC&Cia, Ano 1 – no.9**, Editora Saber, Abril/2002.

08 - Case ,Jeff. **Trends in Management using the SNMP-based Internet Standard Management Framework**. SNMP Research, 2001

09 - Zeltserman, David. **Practical Guide to Snmpv3 and Network Management**, Editora Prentice Hall, 1999

10 - University of Michigan, **Future Computing Environment Monitoring Team Final Repor**twhite paper, Michigan, 1994

11 – Perkins, David. **Understanding SNMP MIBS**. Editora Prentice Hall, 1 ed , 2002

12 - Feit, Sidnie. **SNMP A Guide to Network Management**, McGraw-Hill, 2002

13 – Black, Uyless D. **Network Management Standards: SNMP, CMIP, TMN, MIBs and Object Libraries.** McGraw-Hill, 2a. ed., 1994

14 - Thing, Lowell. **Site WhatIs.com (http://www.whatis.com)**, website, 2003

15 - Harnedy, Sean J. **Total SNMP: Exploring the Simple Network Management Protocol.** Prentice Hall, 2ed, 1998

16 – Perkins, David T. **RMON: Remote Monitoring of SNMP-Managed LANs**. Prentice Hall, Hardcover,1998

17 – Wiley, Gilbert Held. **LAN Management with SNMP and RMON**, Paperback, 1996

18 – Miller, Mark A. **Managing Internetworks with SNMP, M&T IP Library**, 3a.ed, 1999

19 - Rose, Marshall T. **How to Manage Your Network using SNMP: The Network Management Practicum**. Prentice Hall, Paperback, 1995

20 - Steinberg, Louis A. **Troubleshooting SNMP:Analyzing MIBs.** McGraw-Hill Osborne Media, 1ed., 2000

21- Saperia, Jonathan. **SNMP at the Edge: Building Effective Service Management Systems**. McGraw-Hill, 2002

22 - NET-SNMP Project, **Tutorial NET-SNMP**, The NET-SNMP Project home page (http://www.net-snmp.com), 2003

23 - IETF Agentx Working Group, **IETF Agentx Working Group - Home page.** http://www.ietf.org, 2003

24 - Townsend, Robert L. **SNMP Application Developer's Guide**. John Wiley & Sons; Book and Disk edition, 1997

25 - White, Mat. **An Overview of the AgentX Protocol**. The Simple Times volume 6, 1998

26 – Levi, David. **Introduction to the Script MIB.** The Simple Times volume 7, 1999

27 – Kenyon, Tony. **Data Networks.** Digital Press, 2002

28 – Conectiva S.A. **Guia do Servidor Conectiva Linux.** Conectiva S.A, 3ª. edição, 2002

29 – Deroest, James W. **SAMBA Unix & NT internetworking**. McGraw-Hill, 2000

30 – Sharp, John. **Microsoft Visual J# . NET (core reference) – Basic Network Programming**. Microsoft Press, 2003

31 – Lane, Patrick. **CIW: Internetworking Professional Study Guide.** Sybex, 2002

32 – Muller, Nathan J. **Network manager's handbook.** McGraw-Hill, 2003

33 - Sharma, Rajesh Kumar. **Cisco Security Bible.** Hungry Minds, 2002

34 - T. Sridhar. **Designing Embeded Communication Software**. CMP Books, 2003

35 -. Udupa, Divakara K. **Telecommunications Management Network.** McGraw-Hill, 1999

36 – Network Working Group. **RFC 1028 – Simple Gateway Monitoring Protocol**, 1987

37 – Network Working Group. **RFC 1098 – Simple Network Management Protocol**, 1989

38 – Network Working Group. **RFC 2863 – The Interfaces Group MIB**, 2000

39 - Network Working Group. **RFC 2011 - SNMPv2 Management Information Base for the Internet Protocol using SMIv2**, 1996

40 - Network Working Group. **RFC 2012 - SNMPv2 Management Information Base for the Transmission Control Protocol using SMIv2**, 1996

41 – Network Working Group. **RFC 2013 - SNMPv2 Management Information Base for the User Datagram Protocol using SMIv2,** 1996

42 – Network Working Group. **RFC 2257 - Agent Extensibility (AgentX) Protocol Version 1,** 1998

43 – Network Working Group. **RFC 1227 - SNMP MUX protocol and MIB,** 1991

44 – Network Working Group. **RFC 2825 - A Tangled Web: Issues of I18N, Domain Names, and the Other Internet protocols**, 2000

45 – Network Working Group. **RFC 2021 - Remote Network Monitoring Management Information Base Version 2 using SMIv2**, 1997

46 – Network Working Group. **RFC 2819 - Remote Network Monitoring Management Information Base,** 2000

47 – Network Working Group. **RFC 2578 - Structure of Management Information Version 2 (SMIv2),** 1999

48 – Network Working Group. **RFC 2593 - Script MIB Extensibility Protocol Version 1.0,** 1999

48 – Network Working Group. **RFC 2741 - Agent Extensibility (AgentX) Protocol Version 1,** 2000

49 – Network Working Group. **RFC 2571 - An Architecture for Describing SNMP Management Frameworks,** 1999

50 - Network Working Group. **RFC 2564 - Application Management MIB,** 1999

51 – Network Working Group. **RFC 3411 - An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks,** 2002

52 – Network Working Group. **RFC3395 - Remote Network Monitoring MIB Protocol Identifier Reference Extensions**, 2002

53 – Network Working Group. **RFC3434 - Remote Monitoring MIB Extensions for High Capacity Alarms**, 2002

54 – Network Working Group. **RFC2261 – An Architecture for Describing SNMP Management Frameworks**, 1998

55 – Network Working Group. **RFC1155 – Structure and Identification of Management Information for TCP/IP-based Internets**, 1990

56 – Network Working Group. **RFC2925** – **Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations**, 2000

57 – Network Working Group. **RFC1213 - Management Information Base for Network Management of TCP/IP-based internets: MIB-II**, 1991

58 – Network Working Group. **RFC1157 - A Simple Network Management Protocol (SNMP)**, 1990

Anexo A: Mensagens e Operações SNMP

Mensagens e Operações SNMP

As mensagens SNMPv1 e SNMPv2 contém duas partes: o cabeçalho da mensagem e um PDU (*protocol data unit* ou unidade de dados de protocolo) [05].

Figura 17: Formato de mensagem SNMP

Protocol Data Unit (PDU) é o formato de mensagem que os gerenciadores e agentes utilizam para enviar e receber informações.

O Cabeçalho da mensagem SNMPv1 e SNMPv2

O cabeçalho da mensagem SNMP (versões 1 e 2) contém dois campos:

- Número da versão – especifica a versão de SNMP utilizada
- Nome da comunidade – define o acesso para grupos de gerenciadores

SNMPv1 PDU (*Protocol Data Unit*)

Os PDUs SNMPv1 englobam os comandos e operações que indicam as instâncias envolvidas na transação. Os campos do PDU são variáveis em tamanho de acordo com o ASN.1

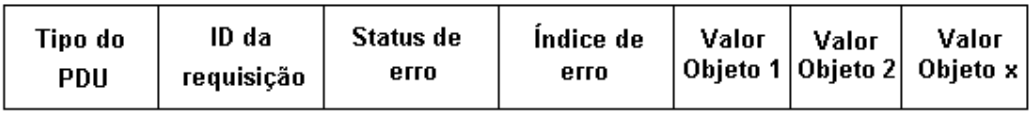

Figura 18: As operações Get, GetNext, Response e Set contém os mesmos campos

- Tipo do PDU – Especifica o tipo do PDU transmitido (Get, Get-next, Set ou Response).
- ID da requisição – Associa a requisição SNMP com respostas.
- Status de erro – Indica se um erro o correu e o tipifica.
- Índice do erro – Associa o erro com uma instância de objeto em particular.
- Valor do objeto – Serve como campo de dados. Associa uma instância com seu valor.

SNMPv1 TRAP PDU

O TRAP no SNMPv1 tem um formado de PDU com oito campos, diferentemente dos demais comandos.

Empresa	Endereço do Agente	Tipo de Trap Genérica	Código de Trap Específica	Tempo	Valor Objeto 1	Valor Objeto 2	Valor Objeto x

Figura 19: SNMPv1 TRAP PDU

- Empresa – Identifica o tipo do elemento gerador da trap.
- Endereço do agente – Fornece o endereço do elemento gerador da trap.
- Tipo de Trap genérica – Indica um dos tipos existentes de trap genérica
- Código de Trap específica – Indica um dos códigos de trap específicas.
- Tempo – Fornece o período de tempo entre a última reinicialização da rede e a geração do trap.
- Valor do objeto – Campo de dados da trap. Cada campo é associado a uma instância de um objeto em particular com seu valor.

Os PDUs acima referem-se a operações do SNMP versão 1. São elas:

1. *Get*
2. *Get-next*

3. *Set*
4. *Get-response*
5. *Trap*

SNMPv2 PDU (*Protocol Data Unit*)

Da mesma forma que para o SNMPv1, os PDUs podem variar em tamanho. Os PDUs do SNMPv2 tem o mesmo formato para as operações *Get*, *GetNext*, *Inform*, *Response*, *Set*, e *Trap*. A operação *GetBulk* tem um formato próprio diferente dos demais.

Tipo do PDU	ID da requisição	Status de erro	Índice de erro	Valor Objeto 1	Valor Objeto 2	Valor Objeto x

Figura 20: SNMPv2 PDU para as operações Get, GetNext, Inform, Response, Set e Trap

- Tipo do PDU – Especifica o tipo do PDU transmitido (Get, GetNext, Inform, Response, Set ou Trap).
- ID da requisição – Associa a requisição SNMP com respostas.
- Status de erro – Indica se um erro o correu e o tipifica. Somente respostas atualizam este campo. Outras requisições transmitem o valor 0.
- Índice do erro – Associa o erro com uma instância de objeto em particular. Somente respostas atualizam este campo. Outras requisições transmitem o valor 0.
- Valor do objeto – Serve como campo de dados. Associa uma instância com seu valor.

SNMPv2 *Getbulk* PDU

A operação *Getbulk* possui um formato próprio de PDU.

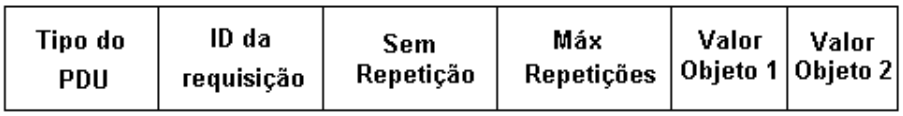

Figura 21: SNMPv2 GetBulk PDU

- Tipo do PDU – Identifica o PDU como uma operação GetBulk.
- ID da requisição – Associa as requisições SNMP com as respostas.
- Sem Repetição – Especifica o número de instâncias que devem ser recuperadas não mais que uma vez.
- Máx. Repetições – Define o número máximo de repetição de uma variável além daquelas especificadas no campo Sem Repetição.
- Valor do Objeto – Serve como um campo de dados para cada instância associa seu valor.

Operação *get* e *get-response*

A solicitação de *get* é iniciada pela NMS que a envia para o agente. O agente recebe a solicitação e a processa para obter o máximo proveito. Dispositivos com carga excessiva, como os roteadores, talvez não consigam responder à solicitação e precisarão excluí-la. Se o agente obtiver as informações solicitadas, retornará um *get-response* para a NMS, onde a resposta é processada [06].

O comando *get* serve para recuperar um único objeto de MIB de cada vez. Entretanto, tentar gerenciar dessa maneira pode ser uma perda de tempo, momento em que o comando *get-next* entra em cena e permite recuperar mais de um objeto de um mesmo dispositivo após um intervalo de tempo.

No NET-SNMP a operação *get* é realizada pelo comando *snmpget*.

Operação *get-next*

A operação de *get-next* permite emitir uma seqüência de comandos para recuperar um grupo de valores de uma MIB. Em outras palavras, para cada objeto MIB a ser recuperado, são gerados separadamente uma solicitação de *get-next* e um *get-response*.

Operação *getBulk*

O SNMPv2 define a operação de *getBulk*, que permite que um aplicativo de gerenciamento recupere uma grande seção de uma tabela, de uma só vez. A operação *get* padrão pode tentar recuperar mais de um objeto MIB de uma vez, mas os tamanhos de mensagens são limitados pelos recursos do agente. Se o agente não puder retornar todas as respostas solicitadas, retornará uma mensagem de erro sem dados. A operação *getBulk* por outro lado, instrui o agente a retornar o máximo da resposta possível. Isto significa que são possíveis respostas incompletas.

Ao executar um comando *getBulk*, é necessário definir dois campos: *"nonrepeaters"* e *"max-repetitions"*. O campo *"nonrepeaters"*informa ao comando *getBulk* que é possível recuperar os primeiros N objetos com uma simples operação de *get-next*. O campo *"max-repetitions"* instrui o comando *getBulk* a tentar até M operações *get-next* para recuperar os demais objetos

No NET-SNMP a operação *getBulk* é realizada através do comando *snmpwalk*.

Operação *set*

O comando *set* é utilizado para modificar o valor de um objeto gerenciado ou para criar uma nova linha em uma tabela. Os objetos MIB definidos como *read-write* ou *write-only* podem ser alterados ou criados com esse comando. Uma NMS pode definir mais de um objeto de cada vez.

No NET-SNMP a operação *set* é possível através do comando *snmpset*.

***Traps* do SNMP**

Uma *trap* é um meio de um agente informar à NMS que aconteceu algo errado. A *trap* originada no agente, é enviada ao destino configurado no próprio agente. Geralmente o destino da *trap* é o endereço IP da NMS. Nenhuma confirmação é enviada da NMS para o agente, de modo que este não sabe se a *trap* alcançou a NMS. Como o SNMP usa o UDP e as *traps* são geradas para informar problemas ocorridos na rede, estão propensas a não alcançar seus destinos.

Quando uma NMS recebe uma *trap*, ela deve saber interpretá-la, isto é, a NMS deve saber o significado da *trap* e o que fazer com as informações que a *trap* transmitiu.

No NET-SNMP o comando *snmptrap* permite o envio de *traps* através da linha de comando.

Anexo B: Códigos e testes desenvolvidos para o estudo de caso

Introdução

O propósito deste anexo é possibilitar a continuidade da implementação do gerenciamento no laboratório do Agrupamento de Sistemas de Controle do IPT , através da documentação dos códigos desenvolvidos durante o estudo de caso.

Também será possível, por analogia, o desenvolvimento de outros casos, utilizando-se das funções apresentadas.

MIB desenvolvida para atender às situações gerenciadas

LABORATORIO DEFINITIONS::=BEGIN

--A diretiva IMPORTS: Inclui definições de outras mibs permitindo que as mesmas sejam utilizadas nesta. No caso abaixo, as definições utilizadas foram aproveitadas na MIB SNMPv2-SMI. A diretiva IMPORTS deve estar presente no primeiro item do arquivo de MIB

IMPORTS
MODULE-IDENTITY, OBJECT-TYPE, Integer32, enterprises FROM SNMPv2-SMI;

--A diretiva MODULE-IDENTITY tem como função principal a definição do módulo que está sendo implementado. Observe que a declaração do módulo faz o link com a arvore SNMP associando a variável 15784 (número fornecido pelo IANA).

laboratorioMib MODULE-IDENTITY

LAST-UPDATED "300210180000Z"
ORGANIZATION "ELB Consulting"
CONTACT-INFO

"Eduardo de Lima Brito
Country: Brazil
State: SP
Phone: +55 +11 9152 3827
email: elimabrito@hotmail.com"
DESCRIPTION
"MIB experimental para o mestrado"
REVISION "300210180000Z"
DESCRIPTION
"Idem a anterior"
::= { enterprises 15784}

-- OBJECT IDENTIFIER declara um nó da árvore abaixo de laboratorioMib que é o nó raiz. O objeto serve como uma espécie de “nó master” onde os OBJECT-TYPE ficam dependurados. O objeto não possui valores.

antena OBJECT IDENTIFIER ::= { laboratorioMib 1 }

-- OBJECT-TYPE declara as variáveis antenastatus e tempstatus. O OID que está sendo criado é enterprises.15784.1.1 para antena.antenastatus e enterprises.15784.1.2 para antena.tempstatus.

antenastatus OBJECT-TYPE
SYNTAX INTEGER
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Informa o estado da antena - se está ligada ou desligada"
::= { antena 1 }

```
tempstatus OBJECT-TYPE
SYNTAX       INTEGER
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
"Informa a temperatura da antena"
::= { antena 2 }

testetrap OBJECT IDENTIFIER ::= { laboratorioMib 2 }

- TRAP-TYPE permite a criação de um objeto trap

errox TRAP-TYPE
STATUS current
ENTERPRISE testetrap
VARIABLES { sysLocation }
DESCRIPTION "This is just a demo"
::= 1
}   - - Finaliza a MIB
```

Arquivo de cabeçalho (antena.h) dos objetos criados

```
#ifndef ANTENA_H
#define ANTENA_H

/* function declarations */
void init_antenastatus(void);
void init_tempstatus(void);
#endif /* ANTENA_H */
```

Arquivo de código para a implementação dos objetos antenastatus e tempstatus

```
#include <net-snmp/net-snmp-config.h>
#include <net-snmp/net-snmp-includes.h>
#include <net-snmp/agent/net-snmp-agent-includes.h>
#include "antena.h"

static int antenastatus = 1;
static int tempstatus = 60;

/** Inicializa o módulo antena */

void init_antena(void)
{
static oid antenastatus_oid[] = {1, 3, 6, 1, 4, 1, 15784, 1,1,0};
static oid tempstatus_oid[] = {1, 3, 6, 1, 4, 1, 15784, 1,2,0};
DEBUGMSGTL (("antena","Inicializando o módulo antena\n"));
DEBUGMSGTL(("antenastatus",
"Inicializando antenastatus. Valor padrão = %d\n",

antenastatus));

DEBUGMSGTL(("tempstatus",
"Inicializando tempstatus. Valor padrão = %d\n",

tempstatus));

netsnmp_register_int_instance("antenastatus",antenastatus_oid,
                OID_LENGTH(antenastatus_oid), &antenastatus, NULL);
netsnmp_register_int_instance("tempstatus",
tempstatus_oid,
```

```
OID_LENGTH(tempstatus_oid),

&tempstatus, NULL);

DEBUGMSGTL(("antena", "Módulo antena inicializado\n"));
}
```

Testes executados para verificação dos objetos criados

- O teste abaixo mostra a existência do objeto antenastatus e seu valor inicial 1

  Comando: [root@ELB01 mibgroup]# snmpget -c public localhost LABORATORIO::antenastatus.0

  Retorno: LABORATORIO::antenastatus.0 = INTEGER: 1

- O teste abaixo mostra a existência do objeto tempstatus e seu valor inicial 60

  Comando: [root@ELB01 mibgroup]# snmpget -c public localhost LABORATORIO::tempstatus.0

  Retorno: LABORATORIO::tempstatus.0 = INTEGER: 60

- O teste abaixo mostra a possibilidade de alteração do valor do objeto antenastatus

  Comando: [root@ELB01 snmp]# snmpset -c public localhost LABORATORIO::antenastatus.0 i 2

Retorno: LABORATORIO::antenastatus.0 = INTEGER: 2

- O teste abaixo mostra a possibilidade de alteração do valor do objeto *tempstatus*

  Comando: [root@ELB01 snmp]# snmpset -c public localhost LABORATORIO::tempstatus.0 i 100

  Retorno: LABORATORIO::tempstatus.0 = INTEGER: 100

Resultado da compilação do subagente laboratorio1

[root@ELB01 mibgroup]# net-snmp-config --compile-subagent laboratorio1 antena.c
generating the tmporary code file: netsnmptmp.1268.c
checking for init_antena in antena.c
void init_antena(void)
running: gcc -g -O2 -Dlinux -I. -I/usr/local/include -o laboratorio1 netsnmptmp.1268.c antena.c -L/usr/local/lib -lnetsnmpagent -lnetsnmphelpers -lnetsnmpmibs -lnetsnmp -ldl -lcrypto -lm
removing the tmporary code file: netsnmptmp.1268.c
subagent program laboratorio1 created

Código-fonte, em linguagem C, da função implementada para ligar/desligar a antena (implementada no servidor Linux)

```
// Primeiro deve-se incluir as bibliotecas do net-snmp
#include </usr/local/share/snmp1/include/net-snmp/net-snmp-config.h>
#include </usr/local/share/snmp1/include/net-snmp/net-snmp-includes.h>
#include <string.h>
#include <stdio.h>
```

```
/* O objetivo da função valor_antenastatus é obter o valor da variável antenastatus, através de uma solicitação GET do SNMP */

int valor_antenastatus( )
{
   struct snmp_session session, *ss;
   struct snmp_pdu *pdu;
   struct snmp_pdu *response;
   oid anOID[MAX_OID_LEN];
   size_t anOID_len = MAX_OID_LEN;
   struct variable_list *vars;
   int status;
   int count=1;
   int my_saved_value;
  // Inicializa a biblioteca SNMP
    init_snmp("altera_antena");
 // Inicializa a sessão que define o host a ser pesquisado. (Foi definido o host com o IP 10.0.0.5)
   snmp_sess_init( &session );
   session.peername = strdup("10.0.0.5");
   // Informa a versão do SNMP (No caso utiliza-se a versão 1)
   session.version = SNMP_VERSION_1;
   // Informa o nome da comunidade utilizado para a autenticação
   session.community = "public";
   session.community_len = strlen(session.community);
    // A sessão é então aberta
   SOCK_STARTUP;
   ss = snmp_open(&session);   /* estabelece a sessão */
   if (!ss) {
      snmp_perror("ack");
      snmp_log(LOG_ERR, "erro ao abrir a sessão!\n");
```

```
    exit(2);
   }
   // Cria-se o PDU para a requisição GET, pois deseja-se saber o valor do objeto. Informa-se a oid  do antenastatus.0
    pdu = snmp_pdu_create(SNMP_MSG_GET);
  read_objid(".1.3.6.1.4.1.15784.1.1.0", anOID, &anOID_len);
    snmp_add_null_var(pdu, anOID, anOID_len);
    // Envia a requisição
     status = snmp_synch_response(ss, pdu, &response);
   // Processa a resposta
   if (status == STAT_SUCCESS && response->errstat == SNMP_ERR_NOERROR) {
   // SUCCESSO: Armazena o valor do objeto antenastatus.
  for (vars = response->variables; vars; vars = vars->next_variable)
  my_saved_value = *vars->val.integer;
  // Descarta-se o PDU e fecha-se a sessão
  }
    if (response)
      snmp_free_pdu(response);
    snmp_close(ss);
    SOCK_CLEANUP;
    return (my_saved_value);
 } /* valor_antenastatus() */
 // A função main lê o objeto SNMP antenastatus e grava no arquivo status.txt seu valor. A função residente no ambiente windows irá ler este valor e executar o comando correspondente – 1 (liga antena) / 0 (desliga antena)
  int main()
 {
  FILE *fp;
  int antenastat=0;
  int continua = 1;
```

```
// Repete continuamente
 do
 {
 // Lê o objeto SNMP antenastatus
  antenastat = valor_antenastatus();
// Grava no arquivo status.txt o valor do objeto
  fp = fopen("status.txt", "wt");
  fprintf(fp, "%d", antenastat);
  wait(1000);
  fclose(fp);
 } while (continua ==1);
 return(0);
}
```

Código-fonte, em linguagem C, da função implementada para atualização da temperatura da antena (implementada no servidor Linux)

```
// Primeiro deve-se incluir as bibliotecas do net-snmp
#include </usr/local/share/snmp1/include/net-snmp/net-snmp-config.h>
#include </usr/local/share/snmp1/include/net-snmp/net-snmp-includes.h>
#include <string.h>
#include <stdio.h>
/* O objetivo da função altera_tempstatus é alterar o valor da variável tempstatus, através de uma solicitação SET do SNMP */
int altera_tempstatus(char *val)
{
   struct snmp_session session, *ss;
   struct snmp_pdu *pdu;
   struct snmp_pdu *response;
   oid anOID[MAX_OID_LEN];
   size_t anOID_len = MAX_OID_LEN;
```

```
    struct variable_list *vars;
    int status;
    int count=1;
    char *argumento = "i";
  // Inicializa a biblioteca SNMP
    init_snmp("altera_antena");
  //Inicializa a sessão que define o host a ser pesquisado. (Foi definido o host com o IP 10.0.0.5)
    snmp_sess_init( &session );
    session.peername = strdup("10.0.0.5");
    session.version = SNMP_VERSION_1;
    session.community = "public";
    session.community_len = strlen(session.community);
   //Abre a sessão
    SOCK_STARTUP;
    ss = snmp_open(&session);   /* estabelece a sessão */
    if (!ss) {
       snmp_perror("ack");
       snmp_log(LOG_ERR, "erro ao abrir a sessão! \n");
       exit(2);
    }
   /*Cria-se o PDU para a requisição SET, pois deseja-se saber o valor do objeto. Informa-se a oid do tempstatus.0. (Observe que a OID do tempstatus termina em 2.0, diferentemente de antenastatus que termina em 1.0)    */
    pdu = snmp_pdu_create(SNMP_MSG_SET);
    read_objid(".1.3.6.1.4.1.15784.1.2.0", anOID, &anOID_len);
    snmp_add_var(pdu, anOID, anOID_len, *argumento, val);
    // Envia a requisição
    status = snmp_synch_response(ss, pdu, &response);
    //Libera o PDU e encerra a sessão
    if (response)
```

```
   snmp_free_pdu(response);
   snmp_close(ss);
   SOCK_CLEANUP;
   return (0); }
```

/ O objetivo da função main é atualizar o valor do objeto SNMP tempstatus, através de uma solicitação SET, de acordo com o valor da temperatura da antena que consta no arquivo temperatura.txt. Este arquivo é alimentado pela função implementada no ambiente windows que lê o valor da temperatura diretamente na controladora da antena./

```
 int main()
{
FILE *fp;
char s[3];
int continua = 1;
// Repete continuamente ....
do
 {   fp = fopen("temperatura.txt", "r");
 fgets(s,4,fp);
 altera_tempstatus(s);  //invoca a função para alterar o objeto tempstatus
 wait(1000);
 fclose(fp);
 } while(continua ==1);
 return(0); }
```

Código-fonte, em linguagem C, complementar da função implementada para a execução dos comandos de leitura da temperatura e ligar/desligar a antena (implementada no Windows)

```
#include <iostream>
#include <fstream>
#include "Comando.h"
```

```
#include "control.h"
namespace
{   void detecta_le_temperatura()
 {  Control(TRX_READ_TEMPERATURE);
    }
   void detecta_liga_antena()
 {  int val_status;
   {  std::ifstream fp("status.txt");
      if (!fp)
       return;
        if (!(fp >> val_status))
       return;     }
   static int old_status = 1;
    if (val_status != old_status) {
    if (val_status == 1) {
     static const Comando sequence_on[] = {
       TRX_OUTPUT_POWER,
       TRX_MY_POWER_MODE_ON,
       PROG_CTRL_PAUSE,
       LIC_SET_DSRC_LINK_MODE_ON,
       END_OF_LIST
      };
      Control_System(sequence_on);
    }
    else {
     static const Comando sequence_off[] = {
       LIC_SET_DSRC_LINK_MODE_OFF,
       TRX_MY_POWER_MODE_OFF,
       END_OF_LIST        };
      Control_System(sequence_off);   }
    old_status = val_status;       }    }
```

```
  }
int control_file()
{
  static int times = 0;
  ++times;
  if (times == 20) {
    times = 0;
    detecta_liga_antena();
    detecta_le_temperatura();
  }    return 0; }
```

Código-fonte da função que verifica a ocorrência de erros e warnings no log e gera um TRAP quando isto acontece (implementada no servidor Linux)

```
procedure TForm1.Button1Click(Sender: TObject);
var arquivo: Textfile;
    ch: char;
    deu,stop,acha_marcacao:boolean;
    i:byte;
{objetivo: identificar mensagens de erro no log e enviar um TRAP para o
gerenciador SNMP}
begin
// Identifica o arquivo de log da controladora
AssignFile(arquivo,'c:\controladora\log\logv01.txt');
Reset(arquivo);
// Seta variáveis de controle
deu:=false;
stop:=false;
acha_marcacao:=false;
i:=0;
// Inicializa o objeto SNMP com os dados do gerenciador e do objeto
```

```
// SNMP nome dado ao Objeto TipwSNMP da IP*works que permite a comunicação SNMP
// através do Delphi em Windows, ambiente do software de gerenciamento da controladora
SNMP.ACTIVE:=TRUE;
SNMP.OBJID[0]:='1.3.6.1.4.1.15784.2.0.1';
SNMP.SNMPVersion:=snmpverV01;
SNMP.Community:='public';
SNMP.LocalHost:='10.5.55.0';
SNMP.RemoteHost:='10.5.55.7';
// Identifica a região do log que ainda não foi pesquisada através da
// busca de asteriscos que são escritos para marcar a região já pesquisada
 while not EOF(arquivo)do begin
 Read(arquivo,ch);
 if ch = '*' then i:=i+1;
 end;
 Reset(arquivo);
//pára o cursor depois do último asterisco - vai procurar onde não foi alterado
 while i > 0 do begin
 Read(arquivo,ch);
 if ch = '*' then i:=i-1; end;
//Inicia a procura do status do log
While not EOF(arquivo) do begin
while not stop do begin
Read(arquivo,ch);
if EOF(arquivo) then begin stop:=true; deu:=true; end;
if ch = '<' then stop:=true;
end; //while not stop
// acha o caracter > dentro da linha - o status da linha se encontra duas posições
```

```
// à frente deste caracter
while not deu do begin
Read(arquivo,ch);
if EOF(arquivo) then begin deu:=true; end;
If ch = '>'then deu:= true;
end; // while not deu
// avança duas posições para verificar o status da linha do log
Read(arquivo, ch);
Read(arquivo, ch);
// caso a letra E apareça significa que houve um erro e um trap é enviado
ao // gerenciador
if not EOF(arquivo)and ch = 'E' then SNMP.SendTrap;
stop:=false;
deu:=false
end; //While not eof(arquivo)
CloseFile(arquivo);
// abre o arquivo para gravação e escreve o caracter * para idenficar
// a região pesquisada
append(arquivo);
write(arquivo,'*');
flush(arquivo);
CloseFile(arquivo); //Fecha o arquivo
end;
```

Anexo C: O software NET-SNMP

Introdução

O NET-SNMP é um pacote que contém ferramentas relacionadas com o uso do SNMP[22]. Estas ferramentas incluem:

Um agente extensível
Uma biblioteca SNMP
Ferramentas para requisitar ou enviar informações de/para agentes SNMP
Ferramentas para gerar e capturar traps SNMP
Uma versão do comando unix "*netstat*" utilizado através de SNMP
Um browser de mib gráfico baseado em Perl/Tk e SNMP

Este pacote foi originalmente baseado na implementação de SNMP na Universidade de Carnegie Mellon porém foi melhorado significativamente depois disto.

O NET-SNMP fornece uma série de comandos que permitem a monitoração e interação com os elementos de rede através do protocolo SNMP. Os comandos principais do NET-SNMP são:

- snmpget
- snmpwalk
- snmpset
- snmpgetnext
- snmpbulkget
- snmpbulkwalk
- snmptrap
- snmpstatus
- snmptest
- snmptable
- snmpnetstat

- snmpusm
- snmpdelta

snmpget

É uma aplicação SNMP que utiliza a requisição GET do SNMP para buscar informações em um elemento de rede. Através da linha de comando podem ser pesquisados os status de um ou mais objetos (OIDs).

A sintaxe do snmpget é: *snmpget [COMMON OPTIONS] [-Cf] OID [OID]...*

Abaixo listamos um exemplo deste comando:

Comando:*snmpget -c public zeus system.sysDescr.0*

A resposta obtida é o valor da variável sysDescr.0 na máquina zeus.

Resposta: *system.sysDescr.0 = "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"*

Caso o elemento de rede apresente algum erro ao processar a requisição, uma mensagem de erro será mostrada. Caso haja outras variáveis na requisição, esta será reenviada sem a variável com problema.

snmpwalk

É uma aplicação SNMP que utiliza as requisições SNMP GETNEXT para consultar os elementos de rede e obter uma árvore de informações.

Um identificador de objeto (OID) deve ser informado na linha de comando. Este OID especifica qual a parte do objeto será pesquisada com a utilização das requisições GETNEXT

Todas as variáveis na subárvore abaixo dos objeto informado serão apresentados pelo usuário.

Caso nenhum OID seja informado, o snmpwalk procurará as informações da MIB-2.

Como no snmpget, mensagens de erro serão apresentadas caso haja problemas com a requisição do snmpwalk.

A sintaxe do snmpget é: *snmpwalk [APPLICATION OPTIONS] [COMMON OPTIONS] [OID]*

Abaixo listamos um exemplo deste comando:

Comando: *snmpwalk -Os -c public -v 1 zeus system*

A resposta obtida é uma lista de valores das variáveis abaixo do sistema

Resposta:

sysDescr.0 = STRING: "SunOS zeus.net.cmu.edu 4.1.3_U1 1
sun4m"
sysObjectID.0 = OID: enterprises.hp.nm.hpsystem.10.1.1
sysUpTime.0 = Timeticks: (155274552) 17 days, 23:19:05
sysContact.0 = STRING: ""
sysName.0 = STRING: "zeus.net.cmu.edu"
sysLocation.0 = STRING: ""
sysServices.0 = INTEGER: 72

snmpset

É uma aplicação que utiliza a requisição SET do SNMP para atualizar/alterar informações no elemento de rede. Um ou mais identificadores de objeto (OIDs) devem ser passados como argumentos através da linha de comando.

A sintaxe do snmpset é: *snmpset [COMMON OPTIONS] OID TYPE VALUE [OID TYPE VALUE]...*

O Type (tipo) é um caracter representando o tipo da variável:

i INTEGER
u UNSIGNED
s STRING
x HEX STRING
d DECIMAL STRING
n NULLOBJ
o OBJID
t TIMETICKS
a IPADDRESS
b BITS

Abaixo listamos um exemplo deste comando:

Comando: *snmpset -c private -v 1 test-hub system.sysContact.0 s dpz@noc.rutgers.edu ip.ipforwarding.0 = 2*

Este comando irá alterar o valor das variáveis *system.sysContact.0* e *ip.ipForwarding.0.*

snmpgetnext

O snmpget é uma aplicação SNMP que utiliza a requisição SNMP GETNEXT para obter informações de um elemento de rede. Um ou mais objetos (OIDs) podem ser informados como parâmetros através da linha de comando. Para cada objeto informado, a variável que é lexicograficamente subsequente terá seu valor retornado.

A sintaxe do snmpgetnext é: *snmpgetnext [COMMON OPTIONS] [-Cf] OID [OID]...*

Abaixo listamos um exemplo deste comando:

Comando: *snmpgetnext -c public zeus interfaces.ifTable.ifEn- try.ifType.1*

A resposta obtida é o valor da variável subsequente que é a interfaces.ifTable.ifEntry.ifType.2.

Resposta: *interfaces.ifTable.ifEntry.ifType.2 = softwareLoopback(24)*

snmpbulkget

É uma aplicação SNMP que utiliza a requisição SNMP GET BULK para consultar o elemento de rede eficientemente para obter a informação. Um ou mais objetos (OID) podem ser passados como argumentos através da linha de comando.

A sintaxe do snmpbulkget é: *snmpbulkget [APPLICATION OPTIONS] [COMMON OPTIONS] OID [OID]*

Abaixo listamos um exemplo deste comando:

Comando: *snmpbulkget -v2c -Cn1 -Cr5 -Os -c public zeus system.ifTable*

A resposta obtida é o valor da variável subsequente que é system.sysDescr.0 e os próximos cinco objetos na ifTable

sysDescr.0 = STRING: "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"
ifIndex.1 = INTEGER: 1
ifIndex.2 = INTEGER: 2
ifDescr.1 = STRING: "lo0" et cetera.

Como o snmpbulkget utiliza a mensagem get bulk do snmp, esta não está diponível no SNMPv1.

snmpbulkwalk

É uma aplicação SNMP que utiliza a requisição GETBULK do SNMP para consultar os elementos de rede e obter uma árvore de informações.

Um identificador de objeto deve ser informado através da linha de comando. Este OID especifica qual parte do objeto será pesquisado utilizando as requisições GETBULK

Caso nenhum argumento seja informado o snmpwalk procurará as informações referentes a MIB-2.

A sintaxe do snmpbulkwalk é: *snmpbulkwalk [APPLICATION OPTIONS] [COMMON OPTIONS] [OID]*

Abaixo listamos um exemplo deste comando:

Comando: *snmpbulkwalk -v2c -Os -c public zeus system*

A resposta obtida é uma lista de valores das variáveis abaixo de *system*

sysDescr.0 = STRING: "SunOS zeus.net.cmu.edu 4.1.3_U1 1 sun4m"
sysObjectID.0 = OID: enterprises.hp.nm.hpsystem.10.1.1
sysUpTime.0 = Timeticks: (155274552) 17 days, 23:19:05
sysContact.0 = STRING: ""
sysName.0 = STRING: "zeus.net.cmu.edu"
sysLocation.0 = STRING: ""
sysServices.0 = INTEGER: 72

O comando snmpbulkwalk é mais eficiente do que o snmpwalk pois o valor das variáveis são capturados através de uma única transação. O snmpwalk captura o valor de cada variável em uma transação diferente.

Assim como o snmpbulkget, o snmpbulkwak utiliza a mensagem get bulk do snmp, e não está diponível no SNMPv1.

snmptrap

É uma aplicação SNMP que utiliza a operação TRAP do SNMP para enviar informações para o gerenciador da rede. Um ou mais objetos (OIDs) podem ser passados como argumentos na linha de comando. O tipo e o valor devem acompanhar cada objeto.

Quando invocado como SNMPINFORM ou quando o argumento –Ci é adicionado a linha de comando do snmptrap, Um INFORM-PDU é enviado aguardando uma resposta do receptor do trap. Se não, é enviado um TRAP-PDU ou TRAP2-PDU.

O parâmetro TYPE (tipo) é um caracter representando:

- i INTEGER
- u UNSIGNED
- c COUNTER32
- s STRING
- x HEX STRING

- d DECIMAL STRING
- n NULLOBJ
- o OBJID
- t TIMETICKS
- a IPADDRESS
- b BITS

O snmptrap deve respeitar uma das três sintaxes:

snmptrap -v 1 [COMMON OPTIONS] [-Ci] enterprise-oid agent generic-trap specific-trap uptime [OID TYPE VALUE]...

snmptrap -v [2c|3] [COMMON OPTIONS] [-Ci] uptime trap-oid [OID TYPE VALUE]...

snmpinform -v [2c|3] [COMMON OPTIONS] uptime trap-oid [OID TYPE VALUE]...

Abaixo listamos um exemplo deste comando:

Comando: *snmptrap -v 1 -c public manager enterprises.spider test-hub 3 0 " interfaces.iftable.ifentry.ifindex.1 i 1*

Este comando irá enviar um *trap* genério de *linkUp* para a interface1 para o gerenciador

snmpstatus

É uma aplicação SNMP que recupera informações estatísticas importantes de um elemento de rede.

A informação retornada é:

O endereço IP do elemento de rede.

Uma descrição da entidade (sysDescr.0)

O uptime do agente SNMP da entidade (sysUpTime.0)

A soma dos pacotes recebidos em todas as interfaces(ifInUCastPkts.* + ifInNUCastPkts.*)

A soma dos pacotes transmitidos por todas as interfacesnterfaces (ifOutUCastPkts.* + ifOutNUCastPkts.*)

O número dos pacotes IPs recebidos (ipInReceives.0)

O número dos pacotes IPs enviados (ipOutRequests.0)

Abaixo listamos um exemplo deste comando:

Comando: *snmpstatus -c public -v 1 netdev-kbox.cc.cmu.edu*

A resposta obtida será semalhante a seguinte lista:

[128.2.56.220]=>[Kinetics FastPath2] Up: 1 day, 4:43:31
Interfaces: 1, Recv/Trans packets: 262874/39867 | IP:31603/15805

O snmpstatus também verifica o status operacional de todas as interfaces e, se alguma não está rodando a mensagem será parecida com a seguinte:

2 interfaces are down!

snmptest

O snmptest é uma aplicação que permite monitorar a gerenciar informações sobre uma entidade de rede.

Um exemplo deste comando é:

snmptest -c public -v 1 zeus Variable: system.sysDescr.0 Variable:

Como retorno da requisição teremos informações dos pacotes conforme abaixo:

requestid 0x5992478A errstat 0x0 errindex 0x0 system.sysDescr.0 = STRING: "Unix 4.3BSD"

snmptable

É uma aplicação SNMP que utiliza repetidamente as requisições GETNEXT e GETBULK do SNMP para consultar informações de um elemento de rede. O parâmetro TABLE-OID deve ser especificado na tabela SNMP.

A sintaxe do snmpbulkwalk é: *snmptable [COMMON OPTIONS] TABLE-OID*

Abaixo listamos dois exemplos deste comando:

Comando: *snmptable localequipamento public at.attable*

A resposta para este comando será semalhante a seguinte lista:

SNMP table: at.atTable atIfIndex atPhysAddress atNetAddress 1 8:0:20:20:0:ab 130.225.243.33

Comando: *snmptable localequipamento public -Cf + at.attable*

A resposta para este comando será semalhante a seguinte lista:

SNMP table: at.atTable atIfIndex+atPhysAddress+atNetAddress 1+8:0:20:20:0:ab+130.225.243.33

snmpnetstat

O comando snmpnetstat mostra simbolicamente os valores de informações de sistemas remotos que utilizam o protocolo SNMP.

Há um número de saídas, dependendo da opção da informação apresentada. A primeira forma do comando retorna uma lista de *sockets* ativos.

A segunda forma apresenta os valores relacionados à rede conforme a opção selecionada.

A terceira forma, dado um intervalo especificado, fará com que o snmpnetstat mostre as informações de pacotes trafegados nas interfaces de rede configuradas.

A Quarta forma retorna estatísticas sobre o protocolo informado.

A especificação do equipamentoname pode ser um nome de equipamento ou um endereço internet especificado (dot notation)

As comunidades da versão 1 e 2c especificam o nome da comunidade para a transação com o sistema remoto.

O comando snmpnetstat deve respeitar uma das quatro sintaxes abaixo:

snmpnetstat [common arguments] [-a] [-n] equipamento
snmpnetstat [common arguments] [-iors] [-n] equipamento
snmpnetstat [common arguments] [-i] [-n] [-I interface] equipamento [interval]
snmpnetstat [common arguments] [-a] [-n] [-P protocol] equipamento

snmpusm

É uma aplicação que pode ser utilizada para a manutenção dos agentes SNMP onde é possível criar, deletar, clonar e alterar os atribuitos dos usuários configurados em um agente SNMP.

A epecificação do USM no SNMPv3 USM (ver RFC2574) dita que os usuários são criados e mantidos através da adição e modificação de linhas na tabela MIB usmUserTabledictatethat

Para a criação de um novo usuário deve-se simplesmente criar uma linha utilizando o snmpset.

A senha para o novo usuário é inicialmente criada através de um clone da senha de um outro usuário.

Para alterar a senha deve-se saber a senha antiga e utilizar o subcomando passwd do comando snmpusm.

O agente NET-SNMP deve primeiramente ser inicializado de forma que pelo menos um usuário é iniciado antes da utilização do comando snmpusm.

Abaixo ilustraremos um exemplo do comando *snmpusm*:

Vamos assumir que para nossos exemplos que as configurações do VACM e do USM estão contidas no snmpd.conf.

Estas linhas configuram um usuário padrão chamado "initial" com a autenticação "setup_passphrase" então podemos realizar a configuração inicial do agent:

 # Entradas de configuração do VACM configuration entries

rwuser initial

 # Vamos adicionar um usuário novo:

rwuser wes

 # Entradas de configuração do USM

createUser initial MD5 setup_passphrase DES

Criação de um usuário novo

snmpusm -v3 -u initial -n "" -l authNoPriv -a MD5 -A setup_passphrase localequipamento create wes initial

O comando acima, cria um usuário novo, com o nome"wes" utilizando o usuário "initial" para este processoto de forma que a senha também será a mesma.

snmpusm -v 3 -u wes -n "" -l authNoPriv -a MD5 -A setup_passphrase localequipamento passwd setup_passphrase new_passphrase

O comando acima, altera a senha do usuário para “new_passphrase”.

Teste do novo usuário

snmpget -v 3 -u wes -n "" -l authNoPriv -a MD5 -A new_passphrase localequipamento sysUpTime.0

snmpdelta

O snmpdelta irá monitorar o OID especificado e relatar mudanças ocorridas com ele a todo o tempo.

A sintaxe do snmpdelta é: *snmpdelta [common arguments] [-Ct] [-Cs] [-CS]-Cm] [-CF configfile] [-Cl] [-CL SumFileName] [-Cp period] [-CP Peaks] [-Ck] [-CT] oid [oid]*

Abaixo ilustraremos um exemplo do comando *snmpdelta*:

$ snmpdelta -c public -v 1 -Cs localequipamento IF-MIB::ifinucastpkts.3 IF-MIB::ifoutucastpkts.3

Este comando retornará a seguinte resposta:

[20:15:43 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:43 6/14] ifOutUcastPkts.3 /1 sec: 158
[20:15:44 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:44 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:45 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:45 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:46 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:46 6/14] ifOutUcastPkts.3 /1 sec: 158
[20:15:47 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:47 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:48 6/14] ifInUcastPkts.3 /1 sec: 184
[20:15:48 6/14] ifOutUcastPkts.3 /1 sec: 184
[20:15:49 6/14] ifInUcastPkts.3 /1 sec: 158
[20:15:49 6/14] ifOutUcastPkts.3 /1 sec: 158
^C