Universidade Federal do Rio de Janeiro

Núcleo de Computação Eletrônica

Fábio Nunes Paulino de Souza

SOFTWARE LIVRE NA GERÊNCIA DE REDES: Uma implementação de ferramentas gratuitas e de código aberto como solução na gerência de redes.

Rio de Janeiro

2007

Fábio Nunes Paulino de Souza

SOFTWARE LIVRE NA GERÊNCIA DE REDES: Uma implementação de ferramentas gratuitas e de código aberto como solução na gerência de redes.

Monografia apresentada para obtenção do título de Especialista em Gerência de Redes de Computadores no Curso de Pós-Graduação Lato Sensu em Gerência de Redes de Computadores e Tecnologia Internet do Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro - NCE/UFRJ.

Orientador:

Professor João Carlos Peixoto de Almeida da Costa, M.Sc., NCE-IM/UFRJ, Brasil

Rio de Janeiro

2007

Fábio Nunes Paulino de Souza

SOFTWARE LIVRE NA GERÊNCIA DE REDES:
Uma implementação de ferramentas gratuitas e de código aberto como solução na gerência de redes.

Monografia apresentada para obtenção do título de Especialista em Gerência de Redes de Computadores no Curso de Pós-Graduação Lato Sensu em Gerência de Redes de Computadores e Tecnologia Internet do Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro - NCE/UFRJ.

Aprovada em abril de 2007.

p/ Professor João Carlos Peixoto de Almeida da Costa,
M.Sc., NCE-IM/UFRJ, Brasil

Este trabalho é dedicado a todos aqueles que encaram as dificuldades como provações. Com o conhecimento obtido, criam novas situações e novas provações. Assim, crescemos, fazemos alguma coisa de útil para a sociedade e nos sentimos vivos.

AGRADECIMENTOS

Primeiramente, ao Pai Celestial, a Jesus Cristo e ao Espírito Santo.

Aos meus pais, por todo o sacrifício e dedicação sem a menor hesitação.

Aos meus familiares que estando perto ou longe, me apóiam em tudo e sempre estão presentes seja qual for o momento.

Aos meus amigos e amigas que conquistei durante todos esses anos. São eles que de alguma forma marcaram a minha vida, e são as minhas referências para nunca desistir.

Aos professores que realmente dedicam sua vida em compartilhar o conhecimento, e fazem isso com amor.

A todos aqueles que direta ou indiretamente me auxiliaram de alguma forma no meu crescimento pessoal e profissional.

E a todos aqueles que se colocaram a disposição de ler as idéias aqui expressas e que de alguma forma tiraram algum proveito de alguma coisa em algum lugar, em alguma situação.

RESUMO

SOUZA, Fábio Nunes Paulino de. **SOFTWARE LIVRE NA GERÊNCIA DE REDES: Uma implementação de ferramentas gratuitas e de código aberto como solução na gerência de redes**. Monografia (Especialização em Gerência de Redes e Tecnologia Internet). Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2007.

Estudo de uma solução de gerenciamento com ferramentas gratuitas de código aberto em um ambiente corporativo, com a finalidade de administrar os componentes de uma rede de grande porte. Com o crescimento das redes, sua maior complexidade e o número crescente de dispositivos utilizando os recursos nelas disponíveis, a necessidade das empresas de manter seus serviços digitais disponíveis todo o tempo possível se tornou cada vez mais vital aos ramos de negócio. Atualmente, existem ferramentas de gerenciamento para todos os tamanhos de redes e de negócios. Entre essas soluções, algumas têm custo baixo enquanto outras podem comprometer alguns milhares de dólares. Através de uma experimentação, foi analisado um conjunto de ferramentas que, integradas, compreendem uma solução de gerenciamento para uma rede heterogênea e crítica. O conhecimento obtido com o desenvolvimento deste trabalho pretende servir como argumentação para seu uso em um ambiente de produção, ou justificar um investimento maior com ferramentas comerciais.

ABSTRACT

SOUZA, Fábio Nunes Paulino de. **SOFTWARE LIVRE NA GERÊNCIA DE REDES: Uma implementação de ferramentas gratuitas e de código aberto como solução na gerência de redes**. Monografia (Especialização em Gerência de Redes e Tecnologia Internet). Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. Rio de Janeiro, 2007.

This abstract contains a study of a free and open-source-tool management solution within a corporate network. Its objective is to manage several components of a wide network. Due to the number of devices and the complexity of networks increasing faster and faster, companies' needs for keeping their digital systems working all the time are essential for business. Nowadays, there are many tools for all kinds of companies and business, some of which may be low-cost while others may demand thousands of dollars. This work was an experience to analyze an integrated solution of tools for managing a heterogeneous and critical network. The acknowledgement gathered with the development of this work intends to serve as basis for its use in a production environment or, perhaps, to justify a bigger investment with commercial tools.

LISTA DE FIGURAS

Página

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AMD	Advanced Micro Devices
ATM	Asynchronous Transfer Mode
CMIP	Commom Management Information Protocol
DNS	Domain Name System
FAQ	Frequently Asked Questions
FTP	File Transfer Protocol
GNU	GNU's Not UNIX
GPL	General Public License
HP	Hewlett-Packard
HTML	HyperText Markup Language
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISO	International Organization for Standardization
ITU-T	International Telecommunications Union - Telecommunication Standardization Sector
MIB	Management Information Base
MRTG	Multi Router Traffic Grapher
NMS	Network Management Station
OID	Object Identifier
OSI	Open Systems Interconnection
Perl	Practical Extraction and Report Language
PHP	Hypertext Preprocessor
PNG	Portable Network Graphics
RAM	Random Access Memory
RFC	Request for Comments
RRD	Round Robin Database
RRDTool	Round Robin Database Tool
SMI	Structure of Management Information
SMS	Short Message Service
SNMP	Simple Network Management Protocol
SNMPv1	Simple Network Management Protocol version 1
SNMPv2	Simple Network Management Protocol version 2
SNMPv3	Simple Network Management Protocol version 3
SQL	Structured Query Language
SSH	Secure Shell
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TI	Tecnologia da Informação
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
WAN	Wide Area Network

SUMÁRIO

1 **INTRODUÇÃO**

1.1 OBJETIVO

A gerência de redes tornou-se uma atividade cada vez mais necessária em pequenas, médias e grandes organizações e cada uma delas requer um nível de gerência compatível com a criticidade de suas informações. A complexidade no controle dos recursos de tecnologia de redes cresceu em grande parte graças à Internet, que implicou no desenvolvimento de novas tecnologias, novos protocolos, interfaces e padrões. Os procedimentos e ferramentas de gerência surgiram a fim de adequar, simplificar e aperfeiçoar esse trabalho acompanhando a tendência sempre crescente desse ambiente e ao mesmo tempo, tendo que se adequarem ao tamanho de qualquer organização.

Hoje em dia, existe uma variedade muito grande de ferramentas que tratam especificamente de gerência de redes e muitas delas exigem um grande investimento por parte das empresas que as adquirem. Existem organizações que utilizam soluções caríssimas, porém muitos dos recursos disponíveis nestas soluções (detalhes de configuração, módulos específicos, adequação ao ambiente etc) não são aproveitados, e com isso, muito do investimento é jogado fora. Esses resultados negativos acabam impactando em um estudo de viabilidade de uma solução de gerenciamento e em muitos desses casos o investimento não é justificado. Assim como também existem empresas que preferem reservar seus investimentos em outras áreas, deixando a gerência de redes e às vezes a própria área de TI em outro plano. Sem perceber, essas empresas acabam por colocar toda a sua estrutura de negócios em risco.

A principal questão é a seguinte: É possível administrar uma rede, seja ela de pequeno ou de grande porte, utilizando os procedimentos normatizados e os principais protocolos de gerência utilizados no mercado, com ferramentas e sistemas ao custo próximo de zero?

O objetivo deste trabalho é verificar a viabilidade de uma solução de gerência pró-ativa que utilize um conjunto de ferramentas gratuitas, de código aberto, reconhecidas no mercado e que possuem a capacidade de gerenciar redes de diversos tamanhos.

Para tal análise, este trabalho irá abordar em um ambiente real de produção de uma empresa de grande porte, duas ferramentas de gerência de código aberto (*open source*) e gratuitas. Para viabilizar esta solução de gerenciamento, foi utilizado como cenário o ambiente de produção da área de TI de uma grande empresa pública, localizado no Rio de Janeiro. As ferramentas de gerência são o **Zabbix** e o **Cacti**. Cada uma delas, com suas respectivas funcionalidades e configurações, foram instaladas e otimizadas para atender uma rede corporativa com diferentes plataformas, serviços e protocolos, com o intuito de justificar a utilização dessas ferramentas, auxiliar gerentes de rede a manter o ambiente produtivo e ganhar tempo na prevenção de problemas, os quais são quase sempre ocasionados por falta de monitoramento adequado.

Um dos fatores mais relevantes à gerência de redes é o fato de ela ter um comportamento reativo onde se preocupa, basicamente, em corrigir os danos causados por alguma falha no sistema. Estudar os erros, os eventos diferenciados da rede e anteceder os momentos que possam causar quaisquer danos é o comportamento ideal. A utilização de ferramentas que fornecem esses recursos cria o diferencial em um gerente ou equipe de gerência, associando a estes, um perfil pró-ativo.

1.2 RELEVÂNCIA

O porquê de “uso de ferramentas ao custo próximo de zero”?

A utilização de qualquer ferramenta de gerência seja ela comercial ou não, necessita de processos e metodologias, além de pessoal com algum conhecimento técnico desejável. Tudo isso, para que as principais funções de gerência de redes, assim como o uso de uma ferramenta específica, sejam conduzidas da melhor maneira. A própria organização desses procedimentos e funções requer algum custo e tempo. Por outro lado, esse investimento pode ser, muitas vezes, inferior ao investimento inicial de compra ou licenciamento de uma ferramenta comercial.

O desenvolvimento deste trabalho, apesar de focado nas ferramentas de gerência, preocupa-se também na viabilidade de toda a solução (plataforma, bibliotecas, scripts, etc) ser de código aberto e gratuita. Por esse motivo, desde o sistema operacional, passando pelas ferramentas que auxiliarão na montagem da solução – servidor Web, linguagem de programação, banco de dados – e as

ferramentas de gerência propriamente ditas, seguem a mesma filosofia de distribuição e livre utilização dos códigos fonte, sem a necessidade de custo de compra ou licenciamento comercial sobre elas, filosofia essa baseada no conceito de GPL (General Public License) [1].

Este trabalho descreve as características relevantes das ferramentas e as análises dos resultados obtidos a fim de justificar o uso de ferramentas gratuitas e de código aberto com o propósito de gerência.

1.3 ORGANIZAÇÃO

O trabalho inicialmente aborda os conceitos de gerência estratégica de redes e protocolos padrões de Internet disponíveis, de forma a fundamentar a utilização da solução baseada nas normas utilizadas no mercado e nos padrões definidos pelos órgãos responsáveis. Essa parte inicial não pretende ser extensa, deixando o foco principal na descrição e análise da solução. Contudo, essa abordagem não pode ser superficial, pois o entendimento desses conceitos é fundamental na implementação das ferramentas e na análise com base nesses conceitos e padrões.

A seção seguinte se inicia descrevendo o ambiente utilizado no desenvolvimento do trabalho, suas características gerais e específicas. A descrição das ferramentas, seu funcionamento e características serão vistos em seguida e, para ilustrar cada um deles, são utilizados como exemplos, alguns fatores e circunstâncias existentes no ambiente estudado, permitindo assim que se possa ter uma visão dos recursos de cada ferramenta em funcionamento real e não somente em teste.

Com base na descrição e estudo de cada ferramenta, é feita a análise de toda a solução, baseando-se em um modelo de gerência padronizado, o qual é ilustrado no capítulo anterior. Esse modelo define os diferentes tipos de funções desejáveis em ferramentas de gerência, e serão essas funções, utilizadas como critérios para análise, que determinarão a viabilidade ou não da solução em diferentes tipos de ambiente. Também serão utilizados os fatores e suas variáveis do ambiente utilizado, para auxiliar nas justificativas e fundamentação da análise.

O capítulo seguinte conclui a análise da solução e cita outros fatores importantes identificados durante o desenvolvimento do trabalho. Também são identificadas as dificuldades encontradas e outros fatores que impediram a

realização ou estudo de alguma característica na solução ou no ambiente. Esta parte por sua vez, servirá de oportunidade para novos estudos e trabalhos dedicados à solução descrita ou à gerência de redes de forma geral, com o desenvolvimento e contínua avaliação dos fatores aqui mostrados.

Os procedimentos de instalação, configuração das diversas partes da solução, assim como exemplos de otimização e exemplos propostos com base na solução e no ambiente analisado, estão documentados e disponíveis em documento anexo a este trabalho.

2 **REFERENCIAL TEÓRICO**

2.1 ASPECTOS DA GERÊNCIA ESTRATÉGICA DE REDES

O crescimento da Internet, refletido no desenvolvimento de novas tecnologias de equipamentos e de meios de transmissão, a formulação de novos padrões e o grau de confiabilidade e dependência nas redes de dados, fazem com que a tecnologia de redes de computadores se desenvolva em uma velocidade muito rápida, e com isso o número de aplicações e serviços nela suportados também aumenta em proporção igual ou superior. As redes de dados deixaram de ser, há muito tempo, apenas mecanismos que interligavam universidades e governos, mas passaram a ser também um meio de ligação com o mundo e o principal ramo de negócio de inúmeras empresas. A grande diversidade de padrões e protocolos surgiu para atender essa demanda sempre crescente, e com isso a complexidade no controle de todos esses recursos também tende a aumentar. Foi necessário então, o desenvolvimento de serviços e protocolos que ajudassem os gerentes na tarefa de monitorar, analisar e tomar providências frente ao comportamento das redes. Como foi dito anteriormente, a demanda faz com que os conceitos utilizados na tecnologia de comunicação de dados evoluam, assim os próprios conceitos e procedimentos utilizados na gerência de redes também seguem essa tendência. O trabalho primordial de todo gerente de redes é garantir o desempenho, disponibilidade e estabilidade de um sistema e ter o pensamento pró-ativo com o intuito de impedir ou minimizar problemas que possam paralisar ou colocar o desempenho de uma rede em níveis inaceitáveis [2].

Um bom trabalho de gerenciamento não consiste apenas em utilizar as melhores ferramentas disponíveis, mas também na aplicação correta de metodologias – ou filosofias – que agregadas ao conhecimento técnico e procedimentos, provêm uma gerência de redes mais capacitada, adaptada ao ambiente e acompanhando o seu crescimento.

2.2 GERENCIAMENTO OSI

A necessidade na padronização dos diversos procedimentos e serviços de gerência, assim como os inúmeros objetos e informações que podem ser

gerenciados, fez com que a ISO (International Organization for Standardization) e a ITU-T (International Telecommunications Union - Telecommunication Standardization Sector), elaborassem uma arquitetura de ferramentas e serviços voltados ao gerenciamento de ambientes que seguem a padronização da ISO. Essa arquitetura é conhecida como Modelo de Gerenciamento de Redes OSI (Open Systems Interconnection), mostrado na figura 1 [3].

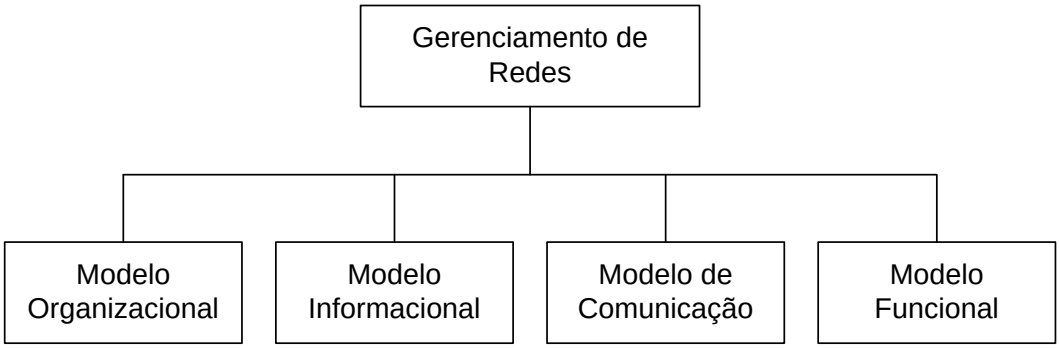

Figura 1 – Modelo de Gerenciamento de Redes OSI [4].

O **Modelo Organizacional** descreve quais são os componentes de um Sistema de Gerenciamento de Redes e como esses componentes se correlacionam. O modelo define o **objeto** (que pode ser gerenciável ou não), o **agente** (que é chamado através de processos internos do objeto) e o **gerente** que interpreta e armazena as informações trocadas com o agente [4]. Na figura 2, é possível relacionar graficamente essas entidades.

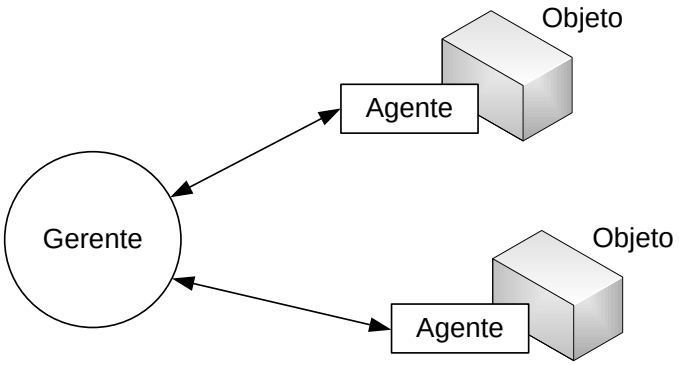

Figura 2 – Componentes do Modelo Organizacional.

O **Modelo Informacional** descreve como as informações trocadas entre agentes e gerentes são organizadas e como são armazenadas. As informações mantidas no agente seguem uma estrutura hierárquica conhecida como SMI (Structure of Management Information). Essa definição da ISO descreve a estrutura da informação gerenciada (sintaxe) e o significado (semântica) da informação dentro do agente. O agente armazena essas informações localmente em um banco de informações denominado MIB (Management Information Base). Ver figura 3. O gerente mantém uma MIB local com as informações extraídas dos MIBs dos agentes. Opcionalmente, o gerente pode permitir que sua MIB seja consultada por outros gerentes [4].

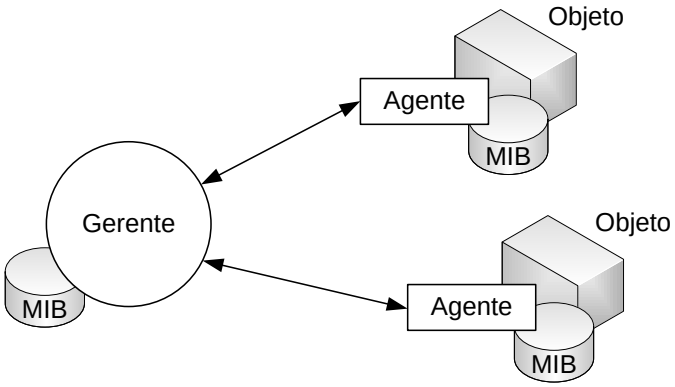

Figura 3 – Estrutura de MIBs.

A RFC (Request for Comments) 1066 apresentou a primeira versão da MIB, que definiu a base de informações necessária para gerenciar dispositivos baseados na pilha de protocolos do TCP/IP [5]. Em março de 1991, a RFC 1213 definiu a evolução da MIB anterior e, denominada de MIB-II [6], se tornou o padrão utilizado na Internet.

A MIB passa então a ser descrita como um conjunto de variáveis denominadas como objetos, e cada um deles possuem atributos. É importante que tanto o agente quanto o gerente utilizem o mesmo esquema conceitual de MIB, para que a troca de informações seja viável. Esses objetos são organizados dentro da MIB de forma hierárquica como uma árvore de registros [7], como apresentado na figura 4.

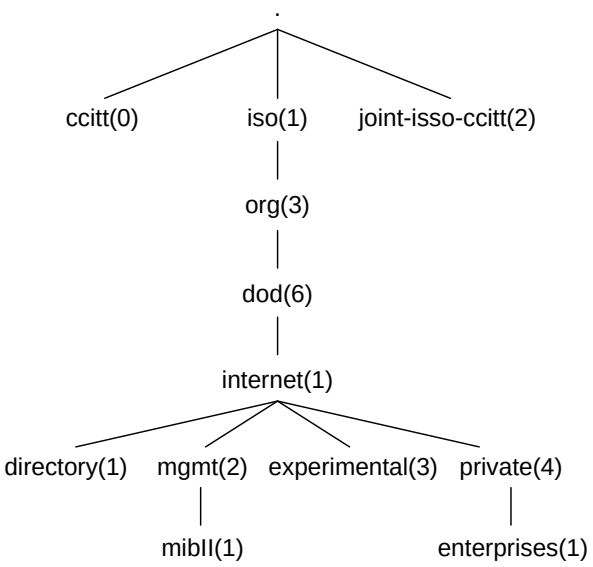

Figura 4 – Estrutura da árvore MIB [8].

Cada nó da árvore possui um OID (Object Identifier), que consiste em uma seqüência de números separados por pontos a partir da raiz. Esse tipo de organização hierárquica é semelhante à estrutura de árvore do sistema DNS (Domain Name System) no modo de leitura e localização dos objetos dentro da árvore. Baseado nesse conceito, o nó **mibII** pode ser identificado na árvore de duas formas. Na forma numérica (1.3.6.1.2.1) ou na forma a identificar cada uma dos nós a qual pertence cada um dos objetos (iso.org.dod.internet.mgmt.mibII) [8].

Dentro do nó mibII, existe uma sub-árvore onde estão definidos os objetos separados por grupos de variáveis que possuem um contexto em comum, como mostrado na tabela 1. Cada variável disponibiliza um tipo de informação relacionado ao objeto gerenciado.

Tabela 1 – Grupos da MIB-II [8].

Grupo	**Tipo de informações disponibilizadas pelas variáveis dentro de cada grupo**
system(1)	Informações básicas sobre o objeto gerenciável (dispositivo de rede)
interfaces(2)	Informações sobre as interfaces de rede
at(3)	Tradução de endereços IP em endereços físicos
ip(4)	Informações sobre o protocolo IP
icmp(5)	Informações sobre o protocolo ICMP
tcp(6)	Informações sobre o protocolo TCP
udp(7)	Informações sobre o protocolo UDP
egp(8)	Informações sobre o protocolo EGP
transmission(10)	Informações sobre meios de transmissão
snmp(11)	Informações sobre o protocolo SNMP

As variáveis contidas em cada grupo de objetos são utilizadas para armazenar as informações obtidas na gerência, durante a troca de informações entre agentes e gerentes. As variáveis pertencentes ao grupo ip(4), por exemplo, fornecem informações relevantes sobre o funcionamento do protocolo IP (Internet Protocol) no objeto gerenciável, como a tabela de roteamento, endereços das interfaces, gateways, se este funciona como um roteador entre diversas outras informações. Lembrando que cada uma dessas variáveis é identificada por um único OID. Como exemplo, para saber se um dispositivo funciona como um roteador, basta obter do agente, o valor da variável que na estrutura da MIB, é referenciado pelo OID .3.6.1.2.1.4.1 ou iso.org.dod.internet.mgmt.mibII.ip.ipFowarding. Se for obtido o valor 1, significa que o dispositivo funciona como um roteador. Um valor diferente não denota a função de roteamento.

O **Modelo de Comunicação** define como serão trocadas as informações entre gerentes e agentes. Mais especificadamente, define os protocolos utilizados e o formato das informações. Nesse aspecto, são definidos os protocolos de aplicação, no caso o SNMP (Simple Management Network Protocol), que será abordado mais à frente, o protocolo de transporte, e o formato das requisições, respostas, assim como as notificações assíncronas geradas pelos agentes. Neste modelo, os gerentes enviam Requisições aos agentes, e estes enviam Respostas. Os agentes também podem enviar mensagens – Traps – sem solicitações, para notificar o gerente quanto a algum evento que mereça atenção.

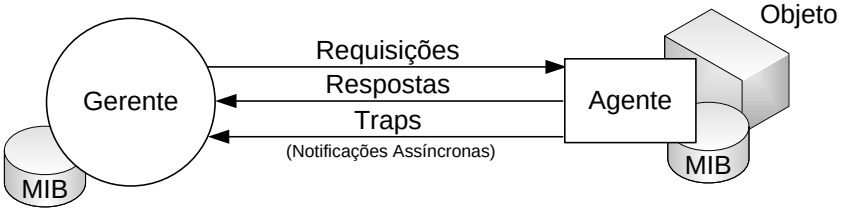

Figura 5 – Comunicação entre gerentes e agentes.

O **Modelo Funcional** padroniza e define os requisitos necessários às ferramentas que são utilizadas na estação de gerenciamento (NMS – Network

Management Station). A ISO define cinco áreas funcionais para essas ferramentas, como mostradas na figura a seguir [4].

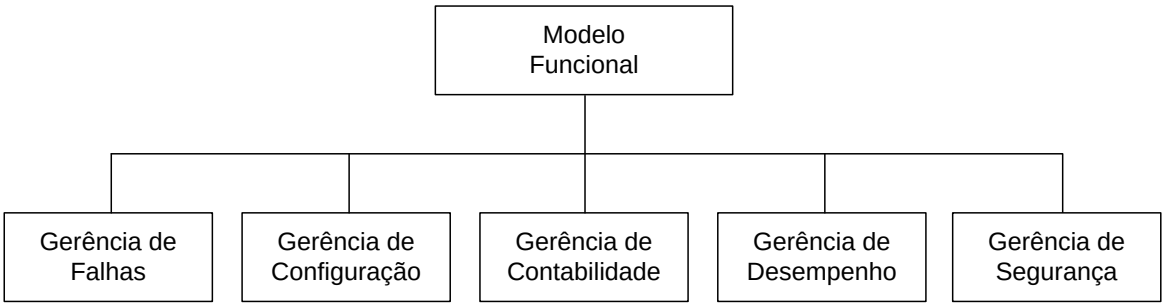

Figura 6 – As cinco áreas do modelo funcional [4].

Justamente por se tratarem das áreas de atuação padronizadas e definidas para as ferramentas de gerências, também são utilizadas como critérios de avaliação e análise da solução desenvolvida neste trabalho.

Gerência de Falhas – Descreve as funções que detectam problemas no funcionamento de algum recurso da rede, tratam de isolar o problema e possivelmente corrigir alguma falha. Executam monitoramento constante dos elementos de uma rede através de testes de diagnóstico. Os eventos geram notificações ou alertas que são registrados (logados) e armazenados para futura análise. Com base nesses eventos, procedimentos são adotados para que a rede volte ao funcionamento normal o quanto antes [9].

Gerência de Configuração – Define as funções de controle lógico e físico dos recursos de uma rede. Trata de procedimentos para controlar os elementos, suas configurações, seus registros, documentação, histórico de eventos e mudanças na topologia de um sistema. [2].

Gerência de Contabilidade – Determina e administra os custos e utilização dos recursos de rede. Os critérios estabelecidos permitem faturar a utilização desses recursos e de serviços, como por exemplo, utilização de um enlace físico, ou uso de um serviço de VoIP [9].

Gerência de Desempenho – O comportamento da rede é constantemente monitorado e dados estatísticos são gerados e armazenados com base em parâmetros estabelecidos. Esses dados são úteis na tomada de decisão como a atualização (*upgrade*) de um elemento ou prevenção de algum evento crítico. Como

exemplo, pode-se fazer um estudo de aumento de banda de um enlace com base nos dados colhidos em um determinado período [9].

Gerência de Segurança – Fornece suporte às aplicações e ferramentas com relação à segurança. São definidas as políticas de segurança, controle no acesso aos elementos da rede e suas respectivas configurações. Notificações quanto à tentativa de acessos indevidos são utilizados. Também trata do uso de criptografia e auditoria [2].

2.3 PADRÕES DE GERENCIAMENTO

Com a complexidade na gerência de dispositivos e na quantidade de informações possíveis de serem administradas, foi preciso definir a forma como as ferramentas utilizam essas informações. Uma breve descrição do CMIP (Common Management Information Protocol) será vista adiante, porém o SNMP será abordado com mais detalhes por ser o principal protocolo utilizado no desenvolvimento deste trabalho, além de ser o padrão de facto no gerenciamento de redes baseadas no protocolo IP.

2.3.1 **O protocolo CMIP**

O protocolo de gerenciamento na camada de aplicação definido pela ISO se baseia no conceito de gerente-agente e define o formato das mensagens trocadas entre essas entidades. O CMIP tinha a proposta de ser um protocolo robusto e seguro, contudo isso o tornava complexo de se configurar e consumia bastante processamento dos dispositivos. O ambiente da Internet demandava um protocolo mais simples e que poderia atender às necessidades de gerenciamento de forma mais accessível e rápida. Foi então que em 1988, o IETF (Internet Engineering Task Force) começou trabalhar em um protocolo que atendesse a essas características e fosse voltado para a pilha de protocolos do TCP/IP, o SNMP [2].

2.3.2 **O protocolo SNMP**

Um protocolo de nível de aplicação que foi largamente aceito por diversos fabricantes, e também utiliza o conceito de gerente-agente, além do conjunto de mensagens (Requisições, Respostas e Traps), da utilização de variáveis e uma base de informações padronizada (MIB). A primeira especificação do protocolo surgiu em Agosto de 1998 [10]. Sua facilidade de implementação e configuração, aliados ao baixo processamento e pouca necessidade de recursos de rede, o tornaram padrão de mercado. A versão seguinte do protocolo (SNMPv2), definida em Abril de 1993, tenta solucionar alguns problemas de performance, segurança e eficiência da primeira versão. Essa versão faz algumas melhorias no acesso às variáveis contidas na MIB, porém não tem ainda a preocupação com a segurança na troca das mensagens e utiliza a mesma forma de autenticação trivial definida na versão anterior. O SNMPv1 utiliza o conceito de *Communities*, que funcionam como senhas compartilhadas entre os agentes e gerentes na troca de mensagens. A definição do SNMPv3, surgida em Janeiro de 1998, adiciona algumas funcionalidades à especificação, além de melhorias nos parâmetros de segurança [10]. O SNMPv3 consolida o conhecimento e experiências obtidas com as versões anteriores, faz as melhorias e ao mesmo tempo se preocupa em atualizar os sistemas de gerenciamento sem a necessidade de mudanças na estrutura da arquitetura, além de manter a simplicidade do protocolo.

2.3.3 **Arquitetura SNMP**

Na figura 7, é mostrada a arquitetura básica do SNMP. O **gerente** é a entidade que implementa o protocolo de gerenciamento, através de suas requisições para os agentes, e normalmente utiliza uma ferramenta que serve de interface entre o administrador da rede e o sistema de gerenciamento. Existem ferramentas capazes de mapear a rede e descobrir seus dispositivos de forma automática e/ou controlada. Algumas soluções geram gráficos e utilizam recursos visuais e sonoros para monitorar e reagir a algum evento. O gerente também possui uma base de dados com as informações extraídas das bases de dados dos agentes.

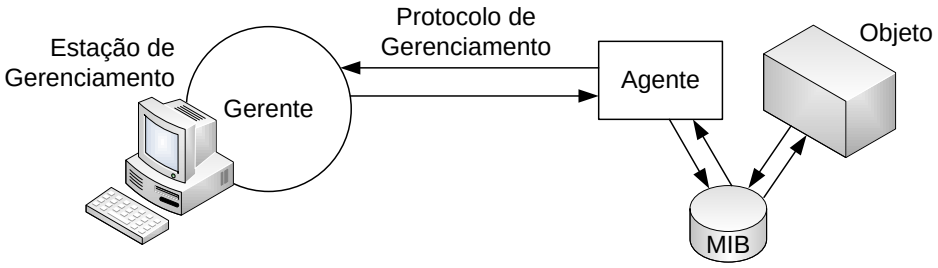

Figura 7 – Arquitetura SNMP.

Os **agentes** são processos alocados internamente nos dispositivos gerenciados, podendo ser componentes adicionais de hardware ou módulos específicos do sistema operacional e são capazes de responder às requisições dos gerentes e também enviar notificações sobre eventos quando configurados para tal [2]. Esses dispositivos podem ser estações, servidores, switches, roteadores, hubs ou qualquer dispositivo que trabalhe com a pilha de protocolos do TCP/IP e implemente SNMP. Alguns dispositivos como displays, equipamentos industriais e elétricos não trabalham com o TCP/IP ou utilizam um protocolo de comunicação diferente, mas podem ser gerenciados pelo SNMP. Para isso, precisam de um agente Proxy, que ao mesmo tempo entende o protocolo de comunicação desse dispositivo e o converte para o TCP/IP [10]. Em um sistema baseado no protocolo SNMP, os agentes implementam MIBs internamente.

Apesar de o SNMP permitir ser usado com diversos protocolos, ele utiliza como protocolo de transporte o UDP (User Datagram Protocol). As razões para não utilizar o TCP (Transfer Control Protocol) como protocolo de transporte, são diversas, mas entre as relevantes podem ser citadas, o baixo overhead inerente ao protocolo UDP, baixa necessidade de processamento dispensada e principalmente, não há o estabelecimento de conexão que ocorre no TCP. Problemas na rede afetam sensivelmente aplicações como Telnet e FTP (File Transfer Protocol) orientados à conexão que o TCP fornece, mas podem ser restabelecidas em outro momento. Todavia, isso não deve acontecer com aplicações baseadas no uso do SNMP, pois são exatamente nesses momentos críticos que elas são necessárias, trocando informações entre dispositivos e identificando problemas. Nesse âmbito, as trocas de informações entre agentes e gerentes são realizadas através do par **endereço IP / porta UDP**. Os agentes aguardam pelas requisições SNMP na porta UDP 161, enquanto as notificações (Traps) são recebidas pelo gerente na porta

UDP 162 [7]. O relacionamento entre as entidades SNMP (agentes e gerentes) baseados no modelo TCP/IP é mostrado na figura 8.

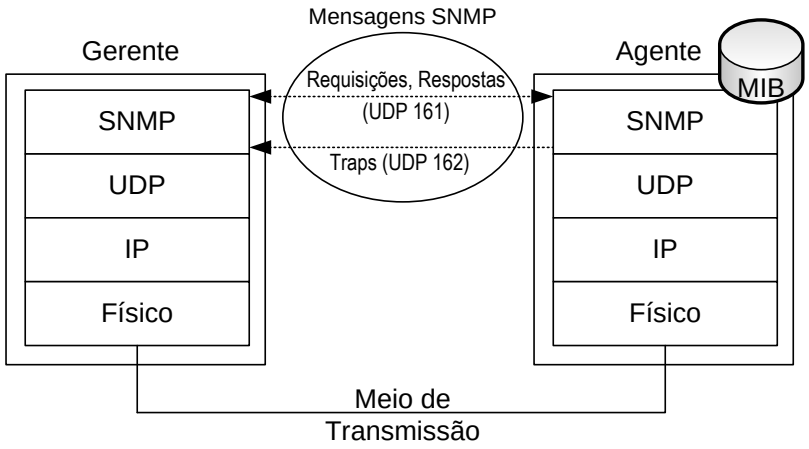

Figura 8 – O SNMP baseado no modelo TCP/IP [8].

Os agentes respondem às requisições que são **primitivas síncronas** enviadas pelos gerentes periodicamente (*pollings*) ou quando solicitadas. Como visto na figura 8, essas primitivas podem obter informações da MIB para consulta e monitoração, como verificar o estado de uma interface ou de um serviço, a taxa de pacotes, taxa de processamento, etc ou podem alterar as informações da MIB para controle e gerenciamento. Estas mensagens trocadas entre as duas entidades são executadas através de operações do protocolo SNMP. Nas primitivas síncronas são utilizadas basicamente três tipos de operação, **Get**, **GetNext**, e **Set**. As operações Get e GetNext, são utilizadas para ler as variáveis da MIB, sendo que a operação Get, retorna o valor da variável especificada na Requisição do gerente (*GetRequest*), enquanto a operação GetNext, retorna o valor da variável posterior à variável especificada. Esta última é útil na leitura de um conjunto de variáveis, quando não se tem conhecimento da última variável dentro de um conjunto de objetos [8]. A operação Set é utilizada quando o gerente necessita alterar o valor de uma variável na MIB do agente. Por exemplo, quando o gerente quer alterar o estado de uma interface de rede (de *up* para *down*) ou quer reinicializar um dispositivo, ele envia uma primitiva específica com a operação Set (*SetRequest*), que altera o valor da variável na MIB correspondente ao estado da interface ou ao processo de inicialização do dispositivo. A mensagem do tipo Response, está associada à

Resposta das operações Get, GetNext e Set. Na figura 9, estão relacionadas as operações básicas do SNMP, com base no modelo TCP/IP, visto anteriormente.

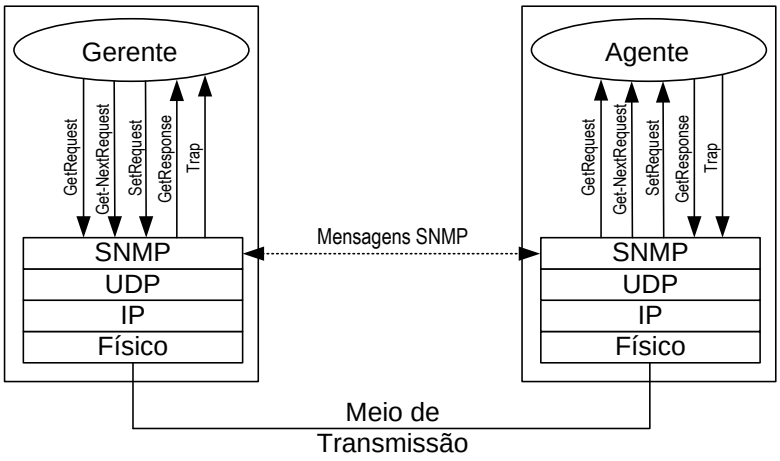

Figura 9 – Operações básicas do protocolo SNMP [9].

Os agentes também podem enviar **primitivas assíncronas** utilizando a operação Trap do SNMP. Quando configurados adequadamente, os agentes podem enviar a qualquer momento mensagens Traps aos gerentes informando algum evento ocorrido no dispositivo, que por alguma definição de critérios estabelecida pelo gerente de rede, mereceu ser notificado. Por exemplo, é possível configurar o agente para mandar Traps nos momentos em que a utilização do processador de um dispositivo gerenciado chegue próximo a 100%, ou então notificar que uma tentativa de acesso não autorizado ao dispositivo ocorreu. A comunicação entre agentes e gerentes através de Traps, é vista na figura 10.

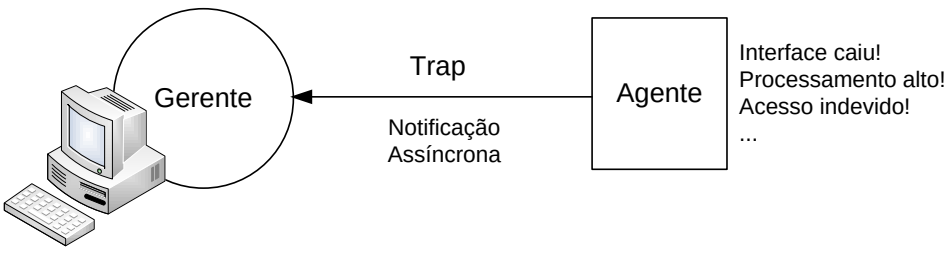

Figura 10 – Traps enviados ao gerente.

Os *pollings* são requisições controladas, mas podem congestionar a rede quando o número de variáveis consultadas na MIB e a quantidade de informações resgatadas são demasiadamente grandes. Todavia, as respostas às requisições são

consideradas como uma forma de confirmação à entrega destas requisições, já que se utiliza como transporte o protocolo UDP, enquanto os Traps não são confirmados, ou seja, não há garantia da chegada desses eventos ao gerente.

Durante o desenvolvimento do trabalho, poderão ser vistos gráficos do ambiente que mostram a utilização dos principais enlaces de rede, impactados pelas requisições geradas pelas ferramentas e pelas respostas dos dispositivos gerenciáveis.

3 **DESENVOLVIMENTO**

3.1 AMBIENTE GERENCIADO

Foi escolhido, como ambiente para gerência do Zabbix e do Cacti, o setor de TI de uma empresa pública, localizado na cidade do Rio de Janeiro. O local mantém grande quantidade e diversidade de servidores e ativos de rede. Durante o desenvolvimento deste trabalho, são citadas situações reais encontradas no setor, porém, em si tratando de ambiente confidencial, os nomes de equipamentos, serviços e endereçamentos foram preservados.

Neste ambiente foram selecionados diversos equipamentos de rede, com funcionalidades distintas, onde alguns são utilizados apenas no nível de distribuição (*camada 2*) e outros no nível de backbone de rede local, Wan e roteamento, sendo todos estes com suporte ao protocolo SNMPv1. Alguns servidores com plataformas e sistemas operacionais diferentes também foram escolhidos, entre eles servidores Windows, Linux, FreeBSD (Unix) e Sun Solaris, sendo que alguns têm o SNMPv1 funcionando e outros não, o que não inviabiliza o uso das ferramentas nesses casos. Na tabela 2, estão descritos os dispositivos gerenciados para se ter uma idéia geral do ambiente utilizado.

Tabela 2 – Descrição do ambiente gerenciado.

Dispositivos gerenciados	**Descrição**
02 Roteadores Cisco 3700	Família de roteadores utilizados no Backbone Wan ATM/Frame Relay.
05 Roteadores Cisco 3600	Família de roteadores utilizados no Backbone Wan ATM/Frame Relay.
09 Roteadores Cisco 1700	Família de roteadores utilizados no Backbone Wan ATM/Frame Relay.
02 Switches 3Com 4007	Switches modulares com backplane de 48 Gbps, módulos de portas ópticas Gigabit e módulos de portas RJ45 Fast Ethernet. Estes Switches são utilizados como concentradores na rede Gigabit Ethernet e suportam mais de 30 Vlans.
05 Switches 3Com 4070	Switches com backplane de 56 Gbps com 24 portas SFP (MiniGbic) Gigabit. Estes Switches também são utilizados como concentradores na rede Gigabit Ethernet e suportam mais de 30 Vlans.
53 Switches 3Com 4400	Switches (camada 2) com backplane de 8,8 Gbps e utilizados na distribuição da rede para os usuários. Possui 24 portas Fast Ethernet e módulos de expansão para interconexão com os

	concentradores através de links Gigabit.
04 Estações Windows XP Professional	Estações de usuários.
08 Servidores Windows 2003 Server	Servidores com aplicações corporativas diversas.
02 Servidores Unix	Servidores Compaq utilizados para aplicações corporativas.
01 Servidor Unix	Servidor Intel Pentium 4 de 3Ghz com 1 GB de RAM e plataforma Unix FreeBSD.
06 Servidores Linux Red Hat	Servidores AMD 64 Opteron (Biprocessados) Utilizados para aplicações corporativas.
02 Servidores NetAppliance	Servidores Storage com aproximadamente 1 TB de dados em disco cada um.
03 Servidores Sun	Servidores com Sistema Operacional Solaris.
01 Servidor Apple	Servidor com Mac OS utilizado para aplicações diversas.
16 Corebuilders ATM 3Com 7000HD	Switches modulares com backplane de 5 Gbps, módulos de portas ópticas OC-3 (155 Mbps) e módulos de portas RJ45 Fast Ethernet. Estes Switches são utilizados como concentradores da rede ATM e suportam mais de 50 Vlans.
10 Switches 3Com 1000/1100	Switches (camada 2) com backplane de até 2 Gbps e utilizados na distribuição da rede para os usuários. Possui 24 portas Ethernet e módulos de expansão para interconexão com os concentradores através de links ATM OC-3 (155 Mbps).

3.2 SOLUÇÃO DE GERENCIAMENTO

3.2.1 **Hardware utilizado**

Foi utilizado um computador da marca HP com processador AMD de 2 GHz e 512 MBytes de memória RAM. Dentre o Zabbix e o Cacti, as duas ferramentas que foram instaladas, somente o Zabbix informa no site oficial, os requisitos mínimos de hardware, que por acaso estão abaixo das configurações iniciais deste computador. Tudo levaria a crer que a estação suportaria com alguma folga a carga de processamento, ou pelo menos a carga de processamento do Zabbix. Porém, com o crescimento da base de dados das ferramentas, e o grande número de requisições e processos na estação, devido à inclusão crescente de dispositivos, verificou-se que a quantidade de memória era insuficiente para suportar as duas ferramentas simultaneamente de forma estável. A quantidade de memória foi dobrada, passando para 1 GByte. A memória Swap configurada passou então para 2 GBytes e com isso a estação passou a ficar estável. Com exceção dessa característica específica, as demais configurações do computador (disco rígido, processador, memória de vídeo,

etc) foram mantidas, como pode ser visto mais à frente. Foi adicionada também mais uma interface de rede, pois este computador precisou ter acesso às redes administrativa (da empresa) e de gerenciamento (dos equipamentos de rede), ambas separadas por vlans distintas. Na tabela 3, é vista a configuração detalhada do computador dedicado que será referenciado em todo este trabalho como estação de gerenciamento, estação NMS, ou simplesmente NMS.

Tabela 3 – Descrição da estação de gerenciamento.

Estação NMS	
Processador	AMD Athlon XP 2800 (2,08 GHz – 512KB Cache L2).
Memória RAM	1 GBytes.
Plataforma	Debian GNU/Linux release 31r2. Kernel 2.4.27-2-386.
Vídeo	Onboard Geforce de 64 MBytes.
Unidade de disco	Maxtor IDE de 40 GBytes.
Interfaces de rede	Duas interfaces Fast Ethernet.

Cada uma das interfaces de rede desta estação fica ligada a uma das principais redes existentes. A rede administrativa onde ficam localizados os servidores, estações e roteadores e a rede de gerenciamento onde ficam ligados as interfaces de gerenciamento de todos os switches Fast Ethernet, Gigabit Ethernet e ATM (AsynchronousTransfer Mode), como pode ser visto na figura 11.

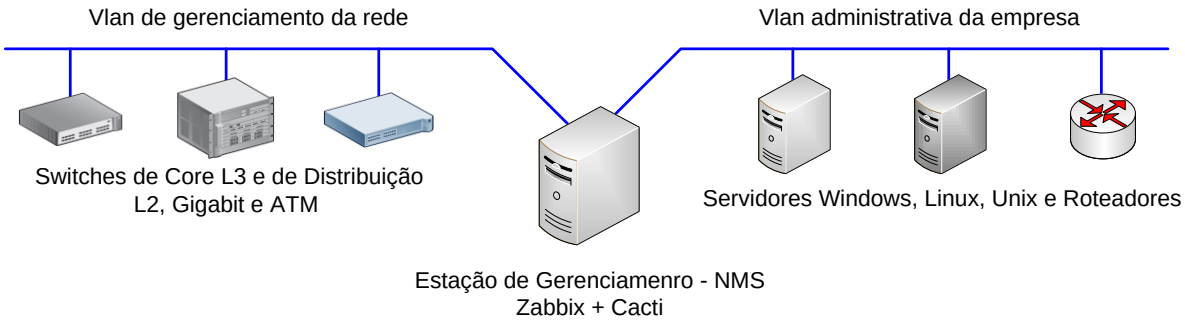

Figura 11 – Arquitetura de gerenciamento do ambiente.

Essas redes não são ligadas diretamente, possuem endereços de rede diferentes e não existe roteamento que permita uma rede ser acessível à outra, por esse motivo a estação possui duas interfaces. Um detalhe que poderia ser aplicado é habilitar o protocolo 802.1Q em uma interface desta estação, e assim acessar as duas redes com apenas uma interface, pois os switches já têm esse protocolo

padrão em funcionamento, mas o ajuste deste recurso não foi o foco na instalação do sistema operacional no NMS. Alguns cuidados foram tomados com relação ao endereçamento e segurança da estação NMS. Utilizou-se endereçamento IP fixo nas duas interfaces e todo o acesso à estação é feito através de aplicação que utilize o protocolo SSH (Secure Shell), para gerenciamento do console do sistema operacional e através de interface Web para gerenciamento e configuração das ferramentas.

3.2.2 **Sistema operacional**

Foi escolhido o sistema operacional Debian GNU/Linux e instalado com os recursos básicos de uma estação, porém sem nenhum tipo de servidor pré-instalado (com exceção do servidor de SSH) e foi desativada a interface gráfica Gnome, na tentativa de impedir que outros programas ou serviços além das ferramentas de gerenciamento consumam os recursos de memória e processamento na estação NMS. O disco de 40 GBytes se mostrou suficiente para armazenar todo o sistema e ferramentas, e o particionamento foi feito com base nas utilizações das bases de dados necessárias para o funcionamento da solução. Alguns testes foram levados em consideração, antes de se chegar ao particionamento final utilizado. Foram separadas as partições dedicadas aos diretórios */var* e */usr*. Duas partições especiais também foram criadas, sendo uma para o diretório */www* onde será a partição de trabalho do servidor Web e onde foram instalados os arquivos da interface de configuração Web do Zabbix e do Cacti, e uma outra partição para o diretório */downloads* onde foram organizados todos os pacotes de instalação das ferramentas, facilitando a localização dos pacotes necessários. Nesta partição também ficaram guardados os conteúdos dos 3 primeiros cds do Debian, para então, ser criado um repositório local. Esse recurso é facilmente implementado através de documentação disponível no site oficial do Debian. Esses procedimentos ajudam no momento de instalação e configuração do sistema operacional, das ferramentas e dos programas necessários. Na tabela 4, pode ser visto o particionamento utilizado e suas respectivas funções.

Tabela 4 – Particionamento de disco da estação NMS.

Partição	Descrição
/boot	50 MB – boot do sistema operacional.
swap	2 GB – duas vezes a memória física.
/downloads	10 GB – utilizada para armazenar os pacotes de instalação do Debian (.deb) e pacotes de instalação das ferramentas de gerenciamento, bibliotecas e plugins.
/var	4 GB – arquivos temporários e logs do sistema.
/usr	3 GB – instalação de aplicativos do sistema.
/www	5 GB – utilizada para armazenar os arquivos de trabalho das ferramentas.
/	5 GB – demais arquivos e diretórios do sistema.

Algumas medidas de segurança foram adotadas, no que se refere ao acesso à estação NMS. São ajustes nos arquivos de configuração *securetty* e *sshd.conf*, para impedir acessos na estação através do login de root, tanto fisicamente no console da estação, quanto no acesso através de conexão via SSH. Também foi criado um procedimento para backup dos bancos de dados do Cacti e do Zabbix e dos arquivos de configuração para um servidor externo através de transferência via FTP. Os detalhes de configuração de segurança como os procedimentos de backup e restauração dos bancos de dados e scripts para backup remoto estão disponíveis em documento anexo.

Levando em consideração que a função de gerenciar uma rede precisa ser estável, segura e com desempenho aceitável, o Debian GNU/Linux foi escolhido por ser exaustivamente testado e somente disponibilizado em versão final depois de meses de testes e feedback constantes de desenvolvedores de todo o mundo [11].

3.2.3 **Ferramentas de apoio**

Para tornar possível a utilização do Zabbix e do Cacti na estação, foram utilizados alguns programas e ferramentas que serviram de apoio ao desenvolvimento do trabalho. A utilização dessas ferramentas adicionais também seguiu a filosofia de licenciamento GPL.

O servidor Web escolhido foi o Apache que irá servir de interface entre o gerente de rede e as ferramentas de gerenciamento. É compatível com diversos sistemas operacionais e arquiteturas além de ser gratuito e de código aberto. É uma ferramenta flexível altamente otimizável e robusta, além de ser um dos servidores Web mais utilizados no mundo.

O MySQL é um banco de dados relacional, gratuito e um dos requisitos de instalação para o funcionamento do Zabbix e do Cacti. É um banco de dados rápido e escolhido por diversas aplicações por ser eficiente e preferido no desenvolvimento para a Web, além de permitir tabelas com até 4 GBytes de dados. É compatível com diversos sistemas operacionais além de Linux como Windows, Unix, etc e com diversas linguagens de programação além do PHP (Hypertext Preprocessor), como Java, C, Visual Basic entre outras [12].

A linguagem PHP é utilizada para criar páginas Web dinâmicas e interagir com diversos tipos de banco de dados como Oracle, Postgresql, Sybase além do MySQL. O PHP consome pouco recurso de hardware, permitindo que aplicações complexas sejam desenvolvidas e aproveitem os recursos de processamento da melhor forma. Esta linguagem tem a capacidade de processar e manipular imagens dinamicamente baseadas nas informações de um banco de dados [13]. Tais características também foram possíveis com a compilação de algumas bibliotecas disponíveis livremente como o GD Library e o renderizador de fontes FreeType entre outras que permitem a criação de imagens e gráficos mais profissionais.

3.2.4 **Ferramentas de gerenciamento**

3.2.4.1 Zabbix

O Zabbix é uma ferramenta de monitoramento centralizado, escrita em PHP e desenvolvida pela Zabbix SAI, empresa fundada em 2005 na Letônia. Consiste em monitorar diversos parâmetros de rede (disponibilidade de servidores, equipamentos de rede, serviços, etc), através de *polling* (requisições periódicas de informações) ou através de *trapping* (notificações enviadas pelos objetos monitorados).

O Zabbix possui diversos alertas que, devidamente configurados, podem enviar e-mails, mensagens SMS (Short Message Service), avisos sonoros, visuais, ou executar scripts. Esses alertas são disparados quando algum parâmetro nos objetos gerenciados mude de estado, ou alcancem um determinado valor. Exemplos, o monitoramento do serviço HTTP em um servidor e a utilização de um enlace Wan de um roteador. O Zabbix poderá disparar um dos alarmes citados acima, caso o serviço HTTP pare de funcionar ou quando a taxa de utilização desse enlace Wan chegue a 70%.

A ferramenta utiliza um banco de dados onde as informações da rede, além dos dados estatísticos, são armazenados. Esses dados podem ser utilizados para gerar relatórios e gráficos em tempo real. Todas estas funções são configuradas e acessadas através da interface Web do Zabbix.

3.2.4.1.1 Componentes do Zabbix

A ferramenta possui três componentes principais que acompanham o pacote da ferramenta [14]. Figura 12.

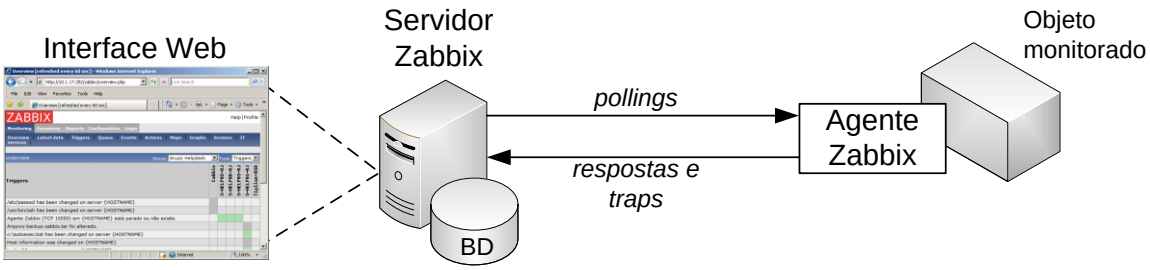

Figura 12 – Componentes do Zabbix.

O **servidor Zabbix** é o processo que fica carregado na memória da estação NMS e é a entidade que faz o monitoramento propriamente dito dos objetos na rede, realizando as verificações periódicas nesses objetos ou tratando as notificações (*traps*) enviadas pelos mesmos. O servidor Zabbix também armazena essas informações em um banco de dados (que pode ser MySQL, PostgreSQL, Oracle ou SQLite) e também gera os alertas baseados nas mudanças da rede.

O **agente Zabbix** é o processo instalado e carregado no objeto monitorado. É ele que responde às requisições ou envia notificações (*traps*) ao servidor Zabbix, informando valores de parâmetros locais do objeto (estatísticas do processador, unidades de disco, memória, integridade de arquivos, atividade na interface de rede, etc). Para obter essas informações, o agente Zabbix utiliza diretamente as *systems calls* (procedimentos de interação entre a aplicação e o núcleo do sistema operacional do objeto [15]). O agente é compatível tanto com sistemas baseados em Unix, como Win32.

Nos objetos onde não é possível a instalação deste agente, podem ser utilizados outros dois mecanismos de monitoramento. Através do protocolo SNMP

ou do monitoramento simples, que verifica a disponibilidade de objetos na rede, além de serviços TCP. Estes mecanismos serão vistos mais adiante.

A **interface Web** é utilizada, local ou remotamente, para realizar todas as configurações, geração de relatórios e de gráficos no servidor Zabbix. O acesso à interface pode ser feito através de qualquer browser, em qualquer plataforma. O Apache é o serviço necessário na estação NMS, para implementar a interface Web.

3.2.4.1.2 Parâmetros especiais

Durante todo o trabalho de monitoramento, são utilizados dois parâmetros especiais dentro do Zabbix: ***Item*** e ***Trigger***. São esses parâmetros que determinarão o comportamento do Zabbix e a reação frente às mudanças na rede.

Item é uma variável que armazena uma informação específica coletada de um objeto através do mecanismo de *polling*. Essa informação pode ser uma string (texto) ou um valor numérico. Além de ser coletada, essa informação é armazenada no banco de dados da ferramenta.

Trigger é a expressão lógica que testa a informação ou valor do *item* sob uma condição lógica, para determinar se o alarme deve ser disparado ou não. Se a expressão for verdadeira, o alarme é disparado.

Exemplo de *item*: ***taxa_utilização_processador***

Essa variável armazenará periodicamente a percentagem de utilização do processador de um determinado objeto.

Exemplo de *trigger*: ***(taxa_utilização_processador = 100)***

A cada verificação, o servidor Zabbix substitui o *item*, pelo valor obtido do objeto monitorado. Nestes exemplos, se a expressão se tornar verdadeira, ou seja, se no momento da coleta da informação, o processador do objeto alcançar 100% de utilização, o alarme será disparado.

Os *items* e *triggers* podem ser criados pelo usuário, contudo, uma série de parâmetros pré-definidos acompanha a ferramenta, agilizando o trabalho de monitoramento. Cada um dos mecanismos de monitoramento utiliza *items* e *triggers* com funções específicas utilizadas em cada um dos mecanismos.

3.2.4.1.3 Mecanismo de monitoramento através do agente Zabbix

O agente Zabbix localmente instalado em um objeto é capaz de obter informações solicitadas e enviar notificações (*traps*) ao servidor Zabbix. A troca de informações entre o agente e o servidor é feita, por padrão, pelas portas TCP 10051 no servidor e 10050 no objeto onde o agente está instalado.

Ao instalar o agente Zabbix em um objeto, é necessário fazer o cadastro deste objeto no servidor Zabbix, informando o endereço (IP ou hostname), mecanismo de monitoramento, porta TCP utilizada, entre outras informações. Dependendo do tipo de sistema encontrado no objeto (Unix ou Win32), pode-se associá-lo a um modelo (*template*), que atribuirá *items* e *triggers* pré-definidos baseados no sistema do objeto. Alguns exemplos de *items* podem ser vistos na tabela 5.

Tabela 5 – *Items* pré-definidos utilizados com o agente Zabbix.

agent.version	Versão do agente Zabbix que está sendo executado no objeto monitorado.
kernel.maxfiles	Número máximo de arquivos abertos suportados pelo sistema operacional do objeto.
net.tcp.dns[ip, zone]	Verifica se o serviço DNS do objeto está funcionando (up). Tem valor 1, se funcionando, enquanto o valor 0 significa que não está.
net.tcp.port[<ip>, port]	Verifica se é possível abrir conexões TCP com a porta especificada em *port*. Tem valor 0, se não é possível abrir conexão, enquanto 1 informa que é possível. Se o IP não for especificado, é assumido o IP 127.0.0.1. Exemplo: *net.tcp.port[200.100.50.1,80].*
system.hostname	Retorna o nome do objeto (*hostname*).
system.localtime	Retorna a hora do objeto.
system.uptime	Retorna o tempo de *uptime* (sec) do objeto.
vfs.file.cksum[file]	Calcula o *checksum* de um arquivo informado. Exemplo: *vfs.file.cksum[/etc/passwd]*.
vfs.file.exists[file]	Verifica se um dado arquivo existe. Tem valor 0 se não existir e 1 se existir.
vfs.file.size[file]	Retorna o tamanho de um arquivo em Bytes. O arquivo deve ter permissões de leitura. Exemplo: *vfs.file.size[/var/log/syslog].*
vfs.fs.size[fs <,mode>]	Calcula o espaço em disco de um volume informado em Kbytes. Se *mode* não for informado, o valor total será utilizado. Caso seja volume montado, retorna o espaço em disco não utilizado. Exemplo: *vfs.fs.size[/tmp,free]*.
vm.memory.size[<mode>]	Retorna a quantidade de memória em Bytes. Se *mode* não for informado, retorna o valor total de memória.

Pode-se citar como exemplo, a necessidade de monitorar em um objeto com sistema Unix, o arquivo **/etc/passwd**. Assim, o servidor Zabbix irá gerar alarmes, se no objeto, algum procedimento relacionado à criação ou alteração de informações de usuários ocorrer.

No cadastro do objeto no servidor Zabbix, atribui-se a ele o *item* ***vfs.file.cksum[/etc/passwd]***. Isso fará com que o servidor solicite ao agente Zabbix, o *checksum* do arquivo /etc/passwd periodicamente.

Ao associar o objeto a um perfil (*template*) para sistemas Unix, também são atribuídos automaticamente alguns *triggers* pré-definidos. Um dos *triggers* que pode ser utilizado neste exemplo tem o seguinte formato: ***{HOST:vfs.file.cksum[/etc/passwd].diff(0)}>0***, onde, "HOST" é o nome do objeto monitorado, seguido do *item*, da função ***diff()***, e da condição para que a expressão seja verdadeira (***>0***). Este é o formato da expressão onde é testado o valor do *item.* Assim, a cada solicitação do servidor Zabbix, o valor do *checksum* do arquivo passwd é consultado, armazenado e testado através da função ***diff()***. Esta função testa o valor obtido, com o valor da solicitação armazenada anteriormente, e retorna 1 se for encontrada diferença entre as solicitações. Se, no objeto Unix, houver alguma operação do tipo, alteração de senha de um usuário, o arquivo passwd será alterado. Na solicitação seguinte, o *checksum* do arquivo terá outro valor, fazendo que a expressão descrita no *trigger*, seja verdadeira. Desse modo o alarme é disparado. É certo dizer que, o alarme é sempre disparado na primeira vez que o objeto é monitorado, pois o *item* armazenado no servidor Zabbix ainda não possui valor, somente após a solicitação seguinte, o monitoramento segue normalmente. O processo de monitoramento é ilustrado, em instantes distintos, nas figuras 13 e 14.

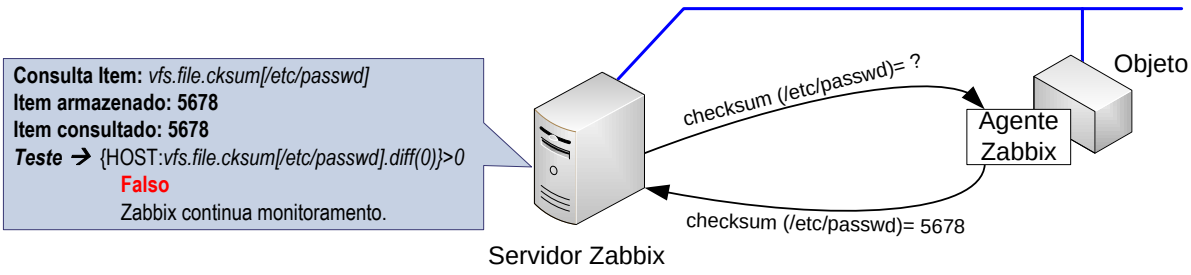

Figura 13 – Solicitação ao agente Zabbix no instante Δt^1.

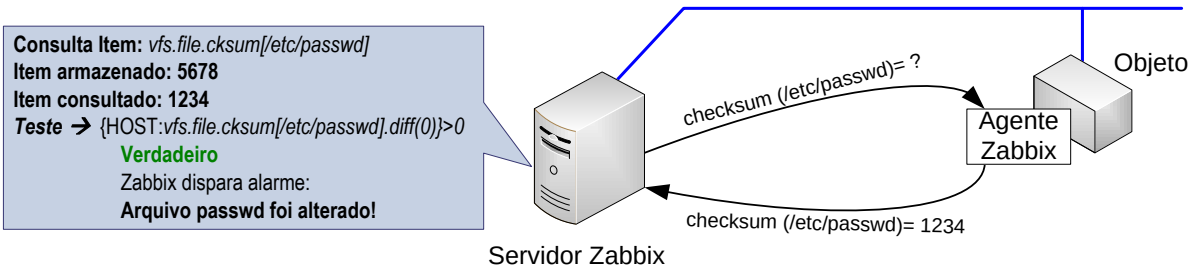

Figura 14 – Solicitação ao agente Zabbix no instante Δt^2.

3.2.4.1.4 Mecanismo de monitoramento através do protocolo SNMP

A utilização deste mecanismo é uma alternativa para o monitoramento de objetos onde não é possível a instalação do agente Zabbix, como roteadores, switches, etc. Contudo, o protocolo não deixa de ser tão eficaz quanto a utilização do agente Zabbix. A limitação nesse caso fica a cargo da quantidade de informações (OIDs) que podem ser obtidas do objeto. Neste mecanismo, o servidor Zabbix utiliza as portas e mensagens inerentes ao protocolo SNMP, sendo este compatível com as versões 1, 2 e 3.

Do mesmo modo que no mecanismo anterior, o objeto contendo o agente SNMP precisa ser cadastrado no servidor Zabbix, atribuindo-lhe um *item*, referente ao recurso que se deseja monitorar. O *item*, neste caso, é um dos OIDs contidos na MIB do agente SNMP. Muitos dos OIDs disponíveis no formato da mib-II, também estão pré-definidos no Zabbix, não necessitando que o usuário da ferramenta defina o formato e a identificação dos OIDs. Estes *items* pré-definidos são facilmente aplicáveis, associando-se o objeto ao modelo (*template*) para objetos SNMP.

Por exemplo, para monitorar o tráfego de entrada de uma interface Fast Ethernet de um determinado servidor, atribui-se a este objeto, o *item* ***ifInOctets1***, que corresponde ao OID *interfaces.ifTable.ifEntry.ifInOctets.1*. Neste exemplo, faz-se necessário gerar um alerta, caso a taxa do tráfego de entrada da interface ultrapasse 70%. Assim, deve-se atribuir um *trigger* ao objeto. O *trigger* utilizado neste caso é definido da seguinte forma: ***{HOST:IfInOctets2.delta(60)}>8750000***. Com estes parâmetros definidos, o servidor Zabbix irá consultar o valor em bytes do

tráfego de entrada da interface periodicamente. Como o valor consultado será sempre cumulativo, pois os contadores na MIB do agente somente são zerados quando o objeto é reiniciado, é necessário fazer a diferença do valor consultado, com o valor da consulta anterior. Essa diferença é feita pela função ***delta()***, que dará o valor real da utilização na interface. Este valor será também o *item* armazenado e testado com a condição (>8750000). Se a taxa real do tráfego de entrada ultrapassar 8,75 MBytes (70% de uma interface Fast Ethernet), a expressão definida pelo *trigger* será verdadeira e o alarme será disparado. O processo de monitoramento através do SNMP é ilustrado, em instantes distintos, nas figuras 15 e 16.

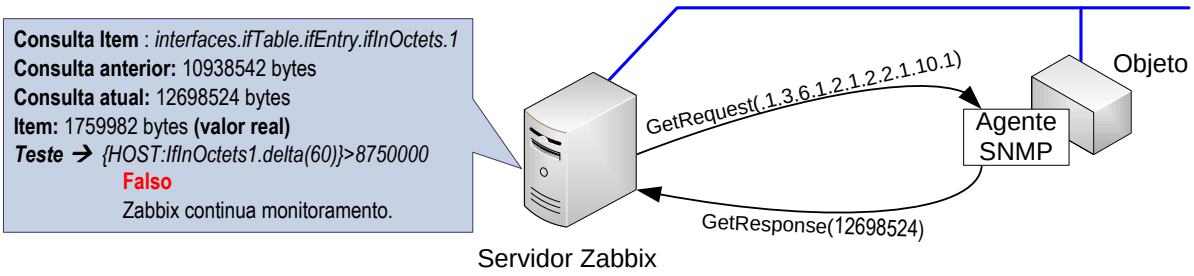

Figura 15 – Verificação com agente Zabbix no instante Δt^1.

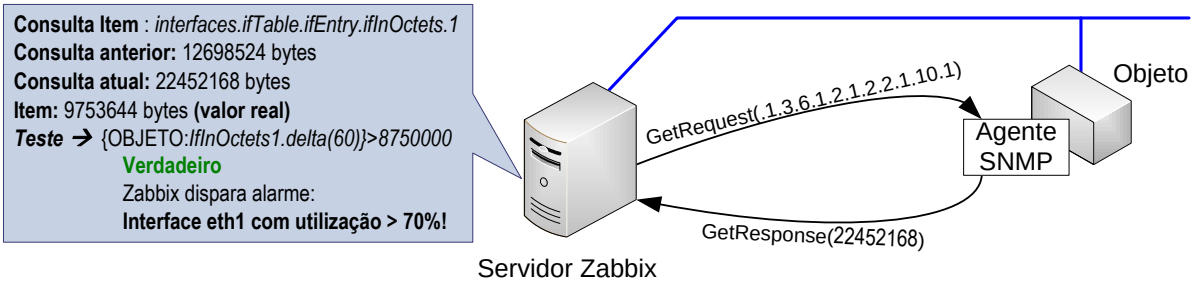

Figura 16 – Verificação com agente Zabbix no instante Δt^2.

3.2.4.1.5 Mecanismo de monitoramento simples

Este mecanismo fornece a possibilidade de se monitorar objetos que não possuem um agente, Zabbix ou SNMP, ou quando se quer apenas monitorar a disponibilidade e tempos de resposta de dispositivos ou serviços que utilizem portas TCP. Apenas o servidor Zabbix fica encarregado de fazer o monitoramento utilizando o protocolo ICMP (Internet Control Message Protocol) e TCP. Na tabela 6,

são mostrados alguns exemplos de *items* pré-definidos utilizados nesse mecanismo, sendo alguns deles customizáveis, permitindo a entrada de parâmetros pelo usuário.

Tabela 6 – *Items* pré-definidos utilizados no monitoramento simples.

icmpping	Verifica se um objeto é alcançável usando ICMP (ping). Se valor for 0, objeto não está acessível, se valor 1, objeto respondeu ao ping.
icmppingsec	Retorna o tempo (sec) de resposta do ICMP (ping).
smtp<,port>	Verifica se o servidor SMTP está funcionando e aceitando conexões. Retorna 0, se servidor parado e 1, se servidor funcionando.
ssh<,port>	Verifica se o servidor SSH está funcionando e aceitando conexões. Retorna 0, se servidor parado e 1, se servidor funcionando.
tcp,port	Verifica se o serviço TCP informado está funcionando e aceitando conexões. Retorna 0, se serviço parado e 1, se servidor funcionando.
http_perf,<ip>,<port>	Verifica se o servidor HTTP está funcionando e aceitando conexões. Retorna 0, se servidor parado, ou o tempo de resposta (sec).
smtp_perf,<ip>,<port>	Verifica se o servidor SMTP está funcionando e aceitando conexões. Retorna 0, se servidor parado, ou o tempo de resposta (sec).

Um exemplo de monitoramento, utilizando um dos *items* acima, pode ser descrito da seguinte maneira: Faz-se necessário monitorar a disponibilidade de um objeto (servidor, roteador, switch, etc) a cada momento. Este objeto é cadastrado no servidor Zabbix e atribui-se a ele o *item* ***icmpping***. Com isto, o Zabbix fará uma verificação periódica de alcançabilidade deste objeto utilizando o protocolo ICMP (*echo-request*). Será então disparado um alarme quando o servidor Zabbix não tiver a resposta (*echo-reply*) do objeto à requisição ICMP, e para que isso funcione, atribui-se a ele o *trigger* definido pela expressão ***{HOST:icmpping.last(0)}#1***. Como descrito na tabela anterior, o *item icmpping* retornará 1, se o objeto responder. Caso o objeto não responda à requisição ICMP, o valor do item será 0 e a expressão torna-se verdadeira. Nesse momento, o alarme então é disparado. Este mecanismo descrito é ilustrado em diferentes instantes nas figuras 17 e 18.

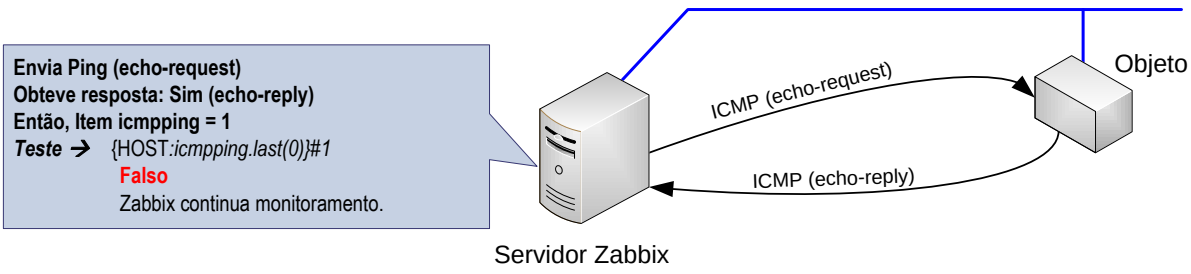

Figura 17 – Verificação simples no instante Δt^1.

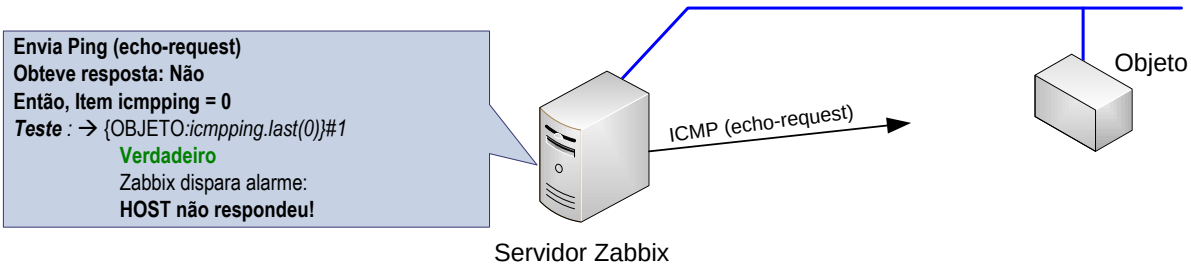

Figura 18 – Verificação simples no instante Δt^2.

A utilização destes mecanismos vai depender da necessidade atribuída a cada objeto, permitindo que combinações de mecanismos sejam utilizadas. Modelos (*templates*) podem ser criados utilizando os *items* e *triggers* de modelos já existentes, assim o usuário pode criar modelos para tipos de objetos existentes em cada ambiente.

A utilização dos *triggers* e ativação dos alarmes são constantemente monitoradas e relatórios podem ser gerados, informando a atividade de cada objeto e suas referentes estatísticas.

A figura 19 mostra o resumo (*overview*) do Zabbix, onde os *triggers* são monitorados.

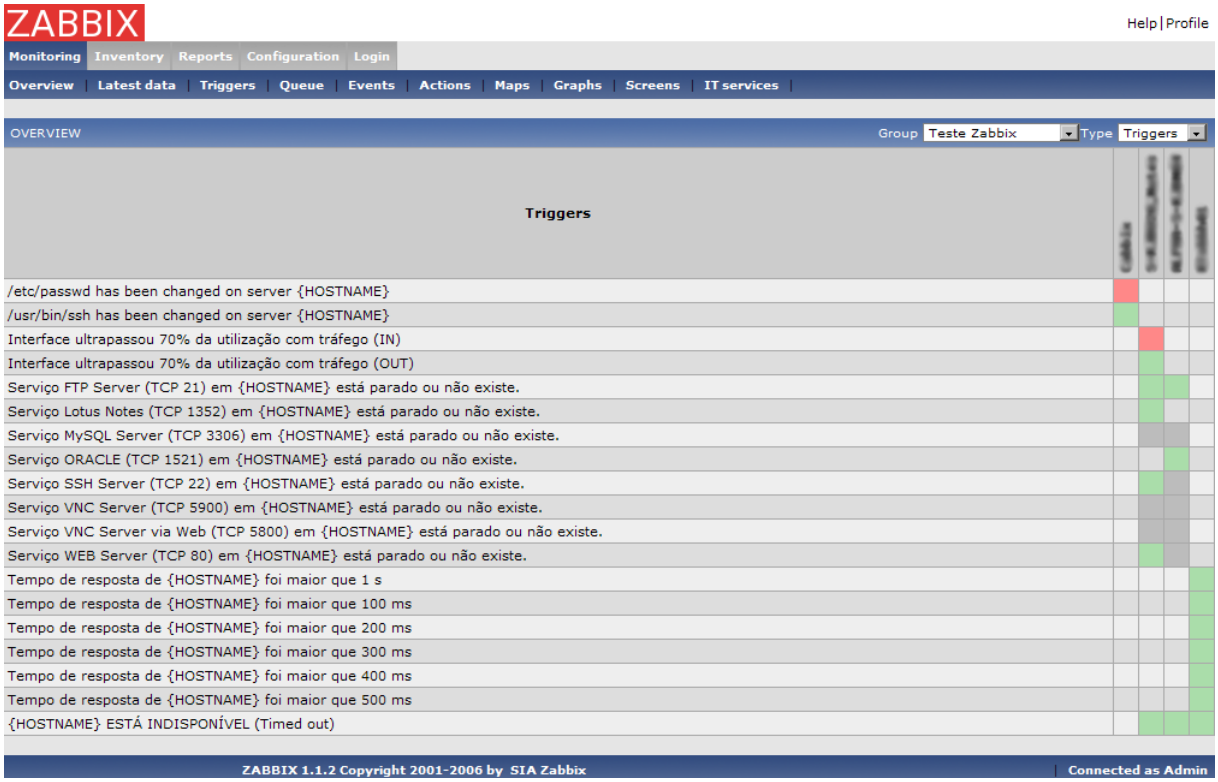

Figura 19 – Resumo dos alarmes no console do Zabbix.

Nesta figura acima, retirada da implementação no ambiente descrito no item 4.1, estão sendo monitorados entre diversos tipos de serviços, o servidor Oracle, a taxa de utilização das interfaces Fast Ethernet de alguns servidores, o tempo de resposta dos roteadores e a integridade dos arquivos de senhas (*passwd*) dos servidores Unix. Os objetos são mostrados na vertical, à direita. Os *triggers* que, eventualmente se tornam verdadeiros e disparam os alarmes, são mostrados à esquerda.

3.2.4.2 Cacti

3.2.4.2.1 História

Em 1995, Tobi Oetiker desenvolveu um script em Perl que pudesse mostrar graficamente a taxa de utilização de um link de Internet. Nos anos seguintes, foi adicionada a esse script, uma agilidade conseguida com o uso da linguagem C, e foi assim que surgiu o MRTG (Multi Router Traffic Grapher), uma ferramenta que consiste em utilizar o protocolo SNMP para consultar periodicamente os dados de dispositivos que suportem esse protocolo e representar graficamente esses valores, como no exemplo da figura abaixo [16].

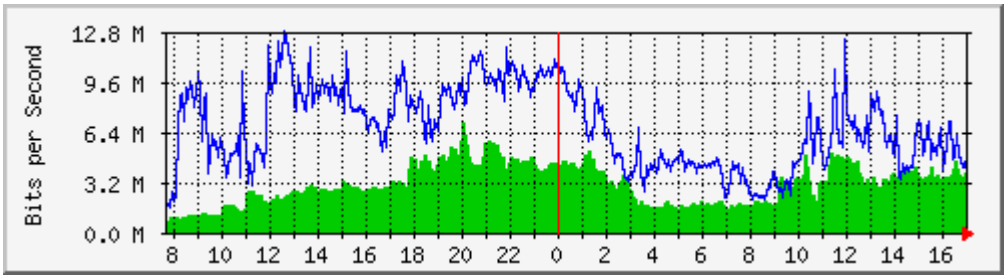

Figura 20 – Exemplo de gráfico gerado pelo MRTG.

O método de consulta e armazenamento dessas informações em séries temporais, foi desenvolvido pelo mesmo criador do MRTG, e chamado de RRD (Round Robin Database), implementado em uma ferramenta conhecida como RRDtool. Esta ferramenta pode ser carregada como um módulo adicional para suprir algumas limitações do MRTG clássico, que gera páginas HTML (HyperText Markup Language) e os gráficos, baseados em arquivos do tipo .log. Essas limitações não

permitem uma flexibilização dos gráficos e das páginas, além da performance prejudicada pela geração periódica das páginas HTML. Com a utilização do RRDTool, o MRTG ficou apenas encarregado de consultar os dispositivos e atualizar os arquivos de dados do tipo .rrd, convertidos do formato anterior .log. A geração dos gráficos e das páginas é realizada apenas sob demanda, tornando o MRTG mais rápido, além de permitir que os gráficos sejam otimizáveis e mais elegantes [17]. A utilização do MRTG não se restringiu apenas no monitoramento de interfaces de roteadores, mas também no monitoramento da carga de um sistema, utilização de processador, memória disponível, espaço em disco, etc [18].

3.2.4.2.2 O Cacti

O Cacti se propôs a ser uma ferramenta *frontend*, ou seja, uma interface completa entre o usuário e o sistema RRD, capaz de gerar os gráficos e também armazenar essas informações em um banco de dados MySQL. O Cacti também pode utilizar o SNMP para obter os dados dos dispositivos gerenciados. Por utilizar um banco de dados, pode armazenar as informações recolhidas de forma permanente, permitindo assim que possa ser feita uma consulta e gerar um gráfico de qualquer instante dentro do período de monitoramento. Essa funcionalidade é possível com o Cacti em conjunto com o MySQL e esta característica foi um dos pontos importantes levados em consideração para a escolha da ferramenta.

É uma ferramenta totalmente voltada para o gerenciamento via Web e em conjunto com a linguagem PHP, permite que todas as configurações necessárias como inclusão de dispositivos, geração e organização hierárquica dos gráficos e a criação de usuários para gerenciar determinadas áreas do Cacti, sejam realizadas através do browser.

O funcionamento básico do Cacti consiste em fazer requisições periódicas através do protocolo SNMP, com suporte às versões 1, 2 e 3 do protocolo, aos dispositivos que suportam um agente SNMP ativo e que devidamente configurados respondam a essas requisições. Com os dados obtidos, o sistema RRD funcionando no Cacti, utiliza esses dados em série e monta gráficos para visualizar o tipo de informação necessária. Figura 21.

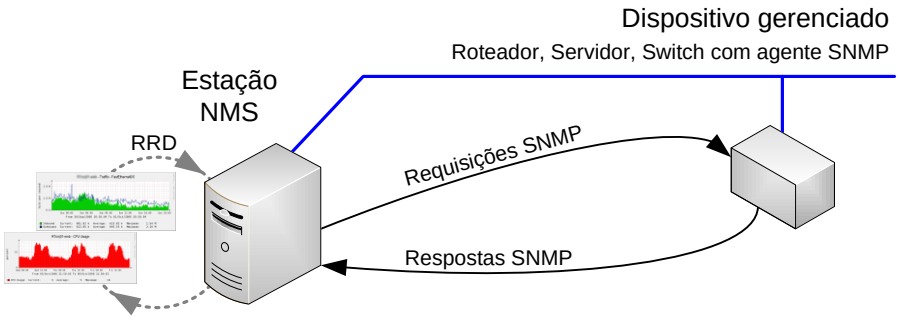

Figura 21 – Mecanismo de geração de gráficos do Cacti.

Por exemplo, a cada 5 minutos, o Cacti solicita o valor da variável na MIB do dispositivo gerenciado correspondente ao tráfego de uma determinada interface, desse modo, é possível para o sistema RRD montar um gráfico do tipo bits/segundo a cada intervalo de tempo, que neste caso é de 5 minutos. A possibilidade de armazenar esses dados em um banco de dados permite que sejam criados também gráficos com períodos diários, semanais, mensais e anuais. Diversos gráficos podem ser gerados como o da figura 22, onde um gráfico do tipo processamento(%) X tempo foi gerado. Em teoria, é possível para o Cacti, gerar gráficos com qualquer tipo de variável.

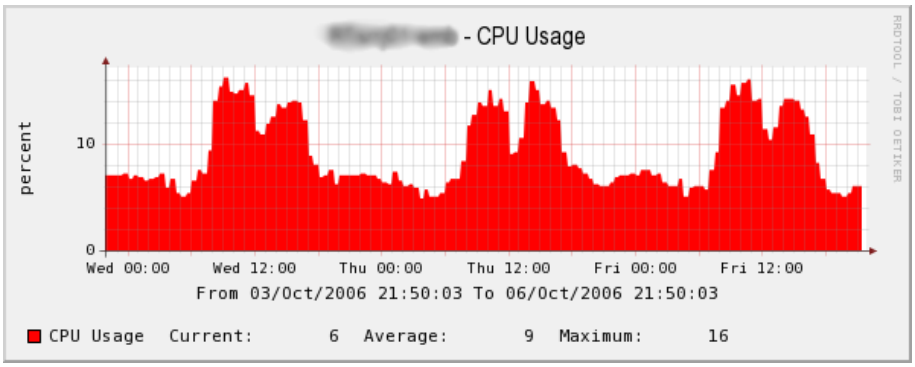

Figura 22 – Exemplo de gráfico gerado pelo RRD no Cacti.

Outra característica levada em consideração é a possibilidade de incluir módulos ao Cacti, também conhecidos como *Plugins*, que adicionam funcionalidades extras à ferramenta.

Dentre os diversos tipos de plugins existentes, uma delas conhecida como *Discovery*, adiciona a capacidade de descobrir elementos na rede e verificar se estes podem ser gerenciados através de protocolo SNMP. Outro plugin cria uma

página no Cacti para exibir graficamente os elementos gerenciados e seus respectivos status (Up/Down).

Um dos plugins bastante recomendados é o Ntop, que como o próprio nome sugere, adiciona a ferramentas de análise de tráfego e protocolos de rede como uma das opções do Cacti. Mesmo não sendo focado neste trabalho, o Ntop como plugin é recomendado para explorar e ampliar a capacidade de gerência da rede com o próprio Cacti. A maioria dos plugins disponíveis é facilmente encontrada e disponibilizada para cópia em um dos diversos sites disponíveis sobre o Cacti [19].

Mesmo com toda a gama de plugins disponíveis e diversas funções possíveis a serem agregadas ao Cacti, foi utilizada outra ferramenta de gerência de redes, que tem a habilidade de se integrar ao Cacti, funcionando assim como um plugin. Neste trabalho será estudado o PHP Network Weathermap.

3.2.4.2.3 PHP Network Weathermap

O PHP Network Weathermap é uma ferramenta que utiliza os dados gerados pelo MRTG, Cacti ou qualquer outra ferramenta que utilize o RRDToll em seu funcionamento, para gerar mapas de utilização da rede, utilizando objetos em cores baseadas no tráfego dos links, possibilitando ter uma visão geral de utilização da rede [20]. A solução aqui proposta utiliza a ferramenta escrita em PHP, facilitando assim sua utilização integrada ao Cacti como um plugin adicional.

Abaixo a figura 23 mostra um mapa criado para atender o ambiente descrito nesta solução, onde é possível ter uma visão geral da rede Wan, onde os enlaces dos roteadores são mostrados conforme a utilização de cada um em relação às suas capacidades. Na figura 24, o mesmo gráfico é mostrado com uma característica específica. Quando o ponteiro do mouse é colocado sobre um determinado enlace, um gráfico do próprio Cacti é visualizado sobre o mapa, Esse recurso somente é possível graças à integração do PHP Weathermap com o Cacti.

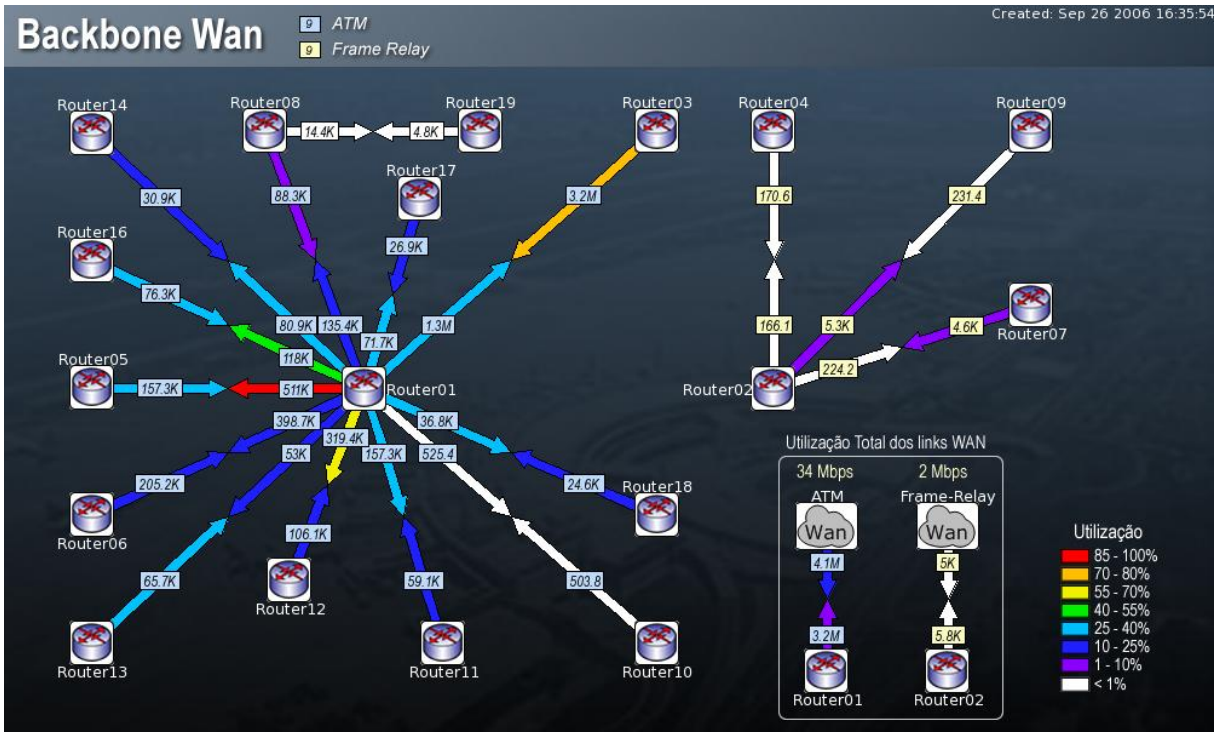

Figura 23 – Mapa de utilização da rede Wan.

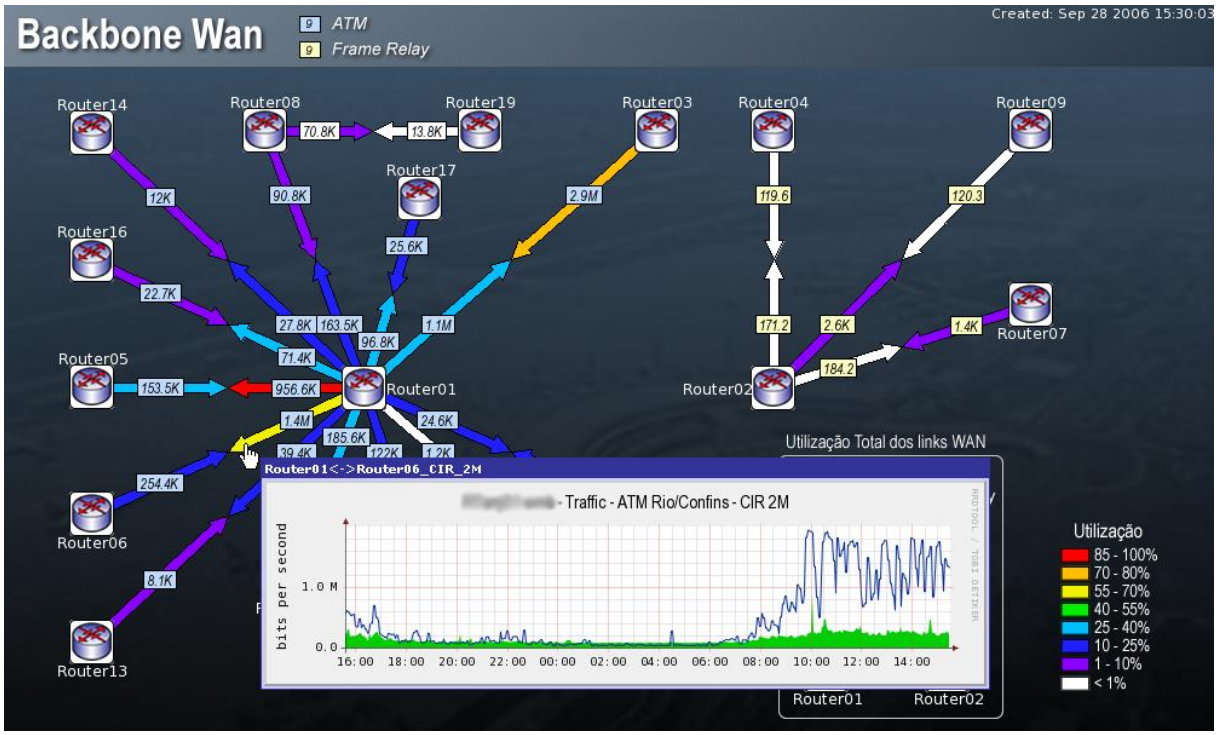

Figura 24 – Mapa da rede Wan com gráfico do Cacti.

Os gráficos são criados através de arquivos de configuração, no formato texto, totalmente customizáveis onde parâmetros são utilizados para definir os dispositivos, as ligações que resultam nos enlaces, e diversos parâmetros que definem os

critérios para mudança de cores dos enlaces e das etiquetas que mostram a percentagem de utilização ou a utilização real em bits/s, entre outros parâmetros que deixam o mapa o mais dinâmico e informativo possível.

Em documento anexo, pode ser visto um exemplo de arquivo de configuração que cria um dos mapas utilizados no ambiente em produção.

4 **ANÁLISE DA SOLUÇÃO DE GERENCIAMENTO**

Após todo o trabalho de implementação, documentação de configurações, detalhes de instalação e dificuldades encontrados na solução colocada em prática no ambiente antes descrito, foram analisadas algumas características de cada ferramenta. Essa análise tem como base a definição do OSI de funcionalidades desejadas em ferramentas de gerência de redes, descritas no modelo Funcional. Estas definições serão utilizadas como critérios para determinar a viabilidade da solução aqui mostrada como ferramenta de gerência suficiente em um ambiente de grande porte, ou se tal solução necessita de algum complemento, tendo em vista algum ponto da análise que denotou alguma deficiência ou falta de algum recurso.

4.1 A SOLUÇÃO SOB O FOCO DA GERÊNCIA DE FALHAS

O Cacti instalado com as funções básicas e funcionando com os recursos aqui apresentados, além do plugin utilizado – o PHP Weathermap – não possui, por padrão, opções para gerar alertas. Tal função pode ser obtida com a utilização de plugins disponíveis nos sites relacionados. Por outro lado, o Zabbix possui um número considerável de alertas e opções de configuração frente a eventos na rede. O uso da interface Web também ajudou bastante na edição de elaboração dos alertas (triggers) ou definição de níveis críticos. Os tipos de testes de verificação, os alertas e as ações são totalmente otimizáveis. O Zabbix permitiu a elaboração de mapas de rede, onde os dispositivos e os eventos da rede são facilmente visualizados e podem fornecer um nível maior de informação ao se clicar em um dispositivo.

No esquemático da rede apresentado na figura 25, é possível visualizar alguns dos dispositivos gerenciados como servidores e roteadores.

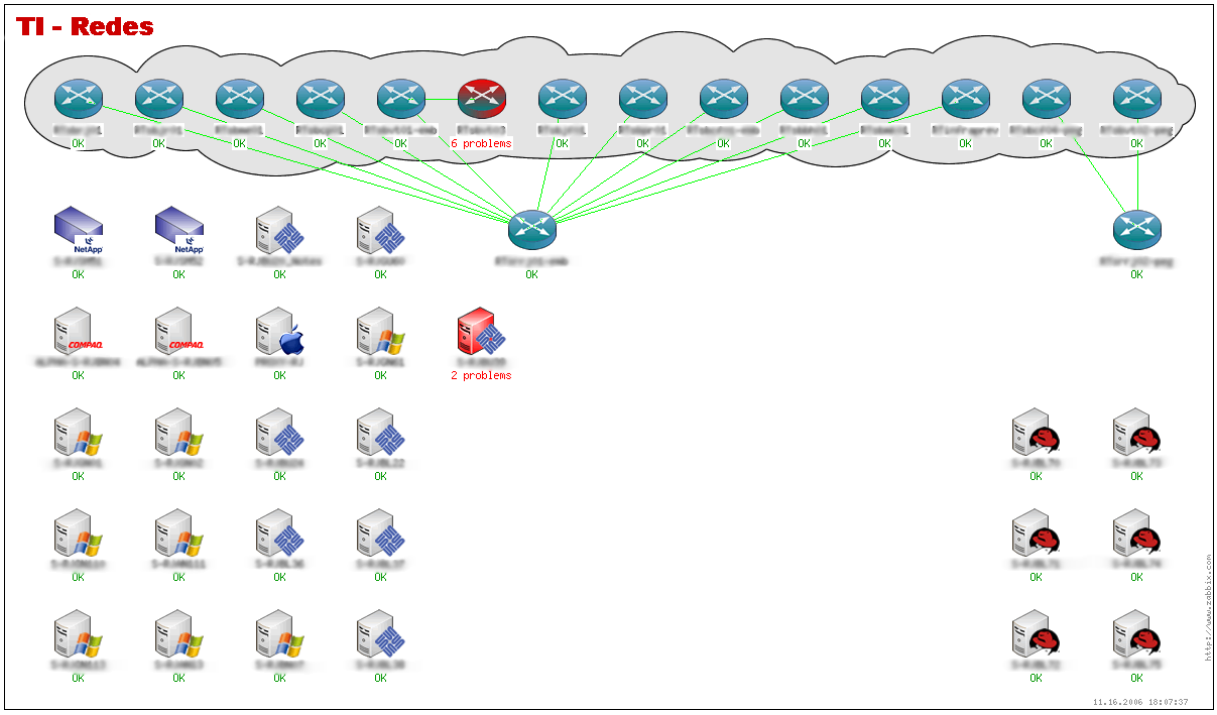

Figura 25 – Diagrama da rede gerenciada pelo Zabbix.

Neste exemplo, um dos servidores apresenta problemas. Os ícones representando os dispositivos são imagens do tipo PNG, onde uma imagem representa o dispositivo em funcionamento normal e outra imagem (de cor vermelha) ilustra o dispositivo com algum alerta ativado. Algumas imagens já vêm disponibilizadas junto com a ferramenta, mas optou-se por definir ícones que ilustrassem os roteadores e tipos de servidores por sistema operacional. Ao se clicar no dispositivo com problema, uma nova página é carregada com a relação dos alarmes disparados e um descritivo de cada um, como pode ser visto na figura 26, quando se clicou sobre o servidor problemático.

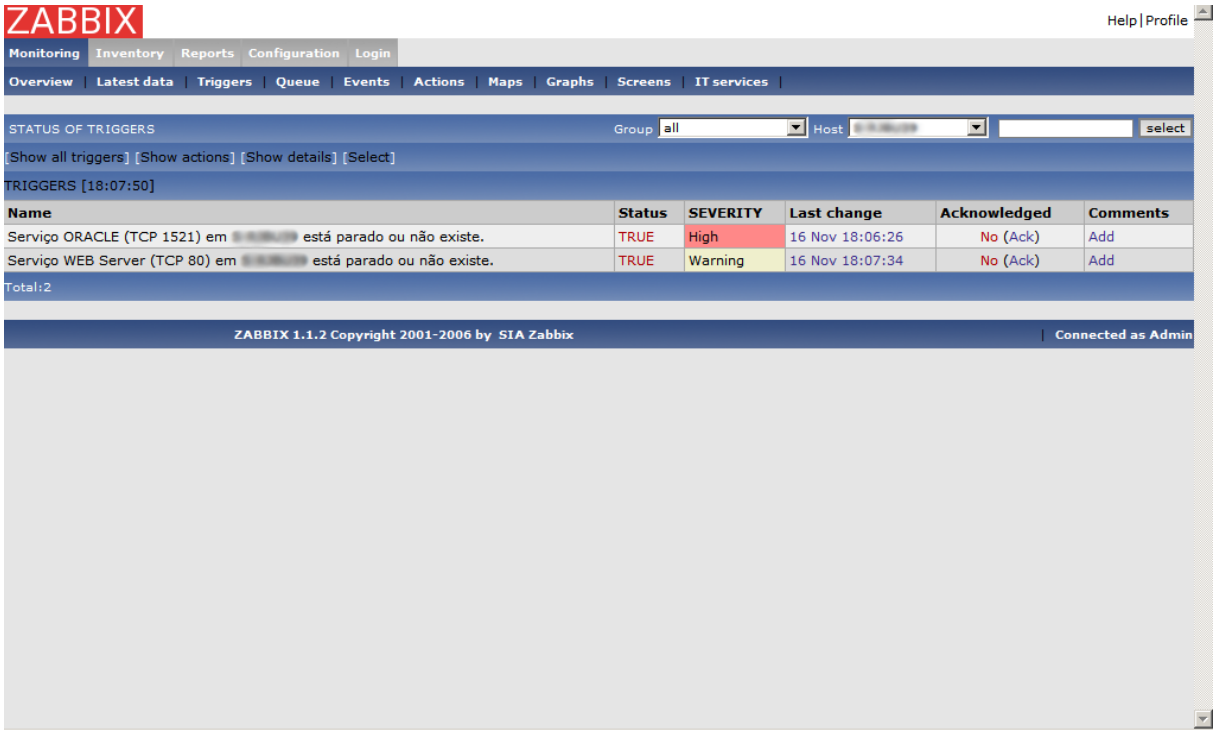

Figura 26 – Descritivo do alarme disparado.

4.2 A SOLUÇÃO SOB O FOCO DA GERÊNCIA DE CONFIGURAÇÃO

4.2.1 **Configuração de sistemas operacionais**

O Zabbix também mostrou ser eficiente quanto à mudança de configuração dos dispositivos. Com determinados dispositivos onde o agente do Zabbix foi instalado, o Zabbix pode identificar e gerar alertas em mudanças de arquivos de sistemas Unix, como os arquivos de senhas (passwd) ou de serviços (/etc/services) e ainda é capaz de detectar mudanças em qualquer tipo de arquivo. Esse mecanismo faz com que o *checksum* de um determinado arquivo seja verificado a cada processo de verificação e se alguma mudança ocorrer, o alerta definido é disparado. Também podem ser identificadas as criações e exclusões de arquivos e para isso o Zabbix monitora também número de inodes dos discos entre outros detalhes. O mesmo também acontece para sistemas Windows com o agente do Zabbix funcionando como Serviço.

Abaixo, as figuras 27 e 28 ilustram a configuração do *item* e do *trigger*, respectivamente, utilizados para monitorar um arquivo específico, que neste exemplo, é o arquivo de backup do Zabbix. A mudança padrão desse arquivo é feita

semanalmente para uma estação Windows XP, quando o script de backup na estação NMS é disparado, mas o teste do item (*checksum*) é realizado diariamente. O requisito básico para este tipo de teste funcionar é que o arquivo deve ter permissão de leitura pelos usuários comuns do sistema onde o arquivo se encontra.

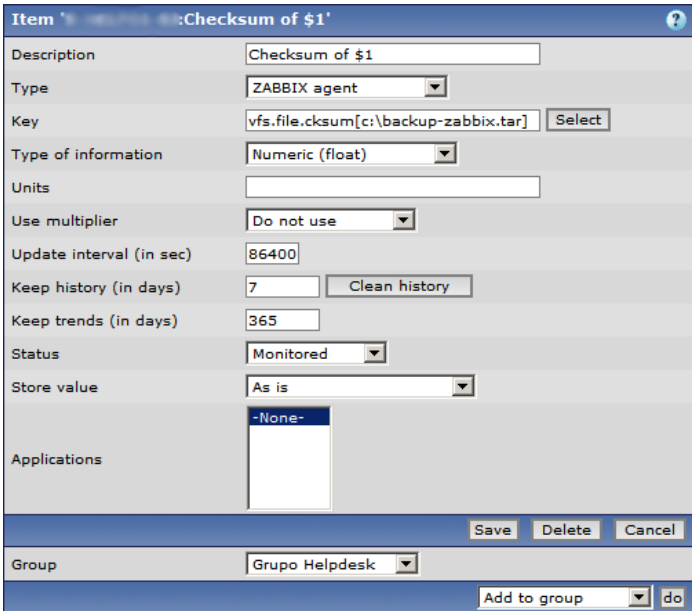

Figura 27 – Configuração do Item que verifica o checksum de arquivos.

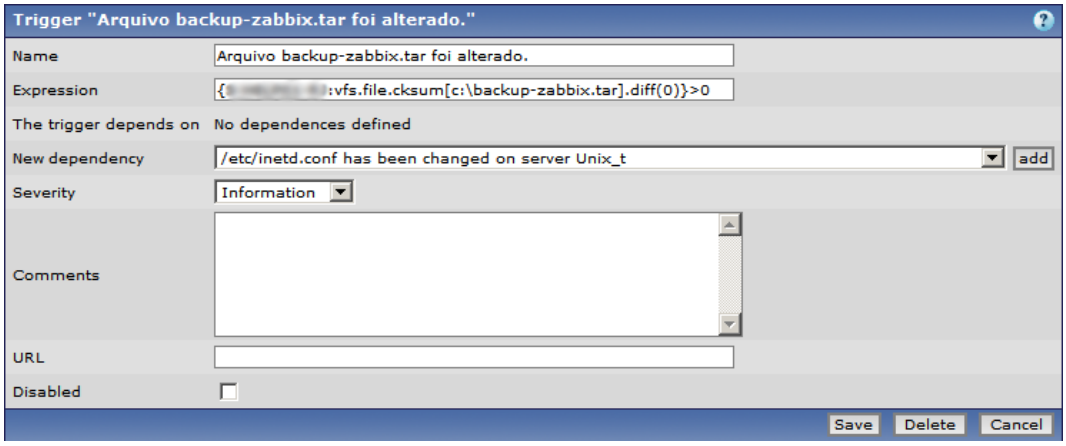

Figura 28 – Configuração do Alarme correspondente.

4.2.2 **Configuração de dispositivos de rede**

Em dispositivos que não permitiam a instalação do agente, a gerência foi beneficiada com a possibilidade de monitorar os dispositivos pelo Zabbix através do protocolo SNMP. Alertas puderam ser criados tendo como parâmetros os valores

retornados das consultas SNMP. Uma das aplicações dessa funcionalidade foi utilizada para monitorar interfaces que deveriam ficar em estado *Blocking* como resultado do protocolo SpanningTree configurado nos switches. Nos switches 4400 da 3Com, as portas ópticas 25 e 26 foram utilizadas como *uplinks*, sendo a porta 25 a única ativa. A porta 26 somente será ativada (colocada em estado *Forwarding*) quando a porta 25 estiver indisponível. Para detectar quaisquer mudanças de estado dessas portas, foi preciso criar um item no Zabbix para monitorar o valor da variável na MIB descrita como *.interfaces.ifTables.ifEntry.ifOperStatus.126*. A última parte da variável descreve a porta 26, que na notação utilizada pela 3Com, utiliza números centesimais, como 101 para a porta 1, 110 para a porta 10 e assim por diante. Em seguida foi criado um alarme para ser disparado quando o valor desta variável for diferente de 5, o que significa que a porta em questão deixou o estado *Blocking*.

4.2.3 **Configuração da estação NMS**

Um dos problemas encontrados na própria estação de gerenciamento foi a reinicialização da estação de tempos em tempos. No período inicial de implementação, a estação estava monitorando cerca de 90 dispositivos entre computadores e equipamentos de rede, e cada um desses gerando entre 2 a 5 tipos de gráficos diferentes dependendo do tipo de informação que era requisitada de cada um. Depois da inclusão de aproximadamente 40 novos dispositivos (switches ATM com mais de 10 interfaces cada e gerando gráficos para essas interfaces) , notou-se que nas semanas seguintes em intervalos de 3 a 4 dias a estação simplesmente reinicializava sem explicação aparente. Com a análise dos gráficos do Cacti, mais especificadamente os gráficos de memória Swap disponível e de processos, notou-se que depois da inclusão dos últimos dispositivos o número de processos aumentava da mesma forma que a quantidade de memória Swap disponível caia, e quando esta chegava próximo de zero, como pode ser visto nas figuras 29 e 30, a estação reinicializava.

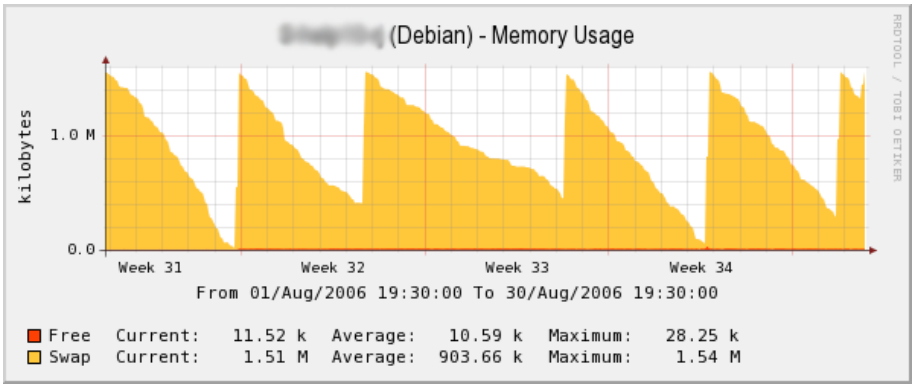

Figura 29 – Swap disponível na estação NMS.

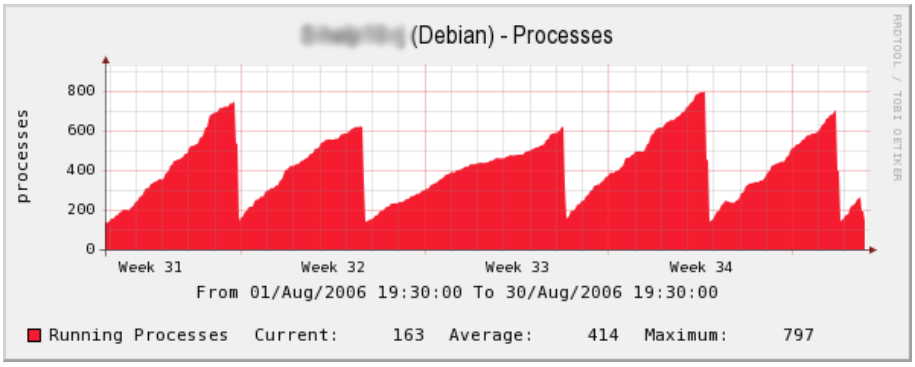

Figura 30 – Número de processos na estação NMS.

Através de consultas aos fóruns disponíveis do Cacti, foi sugerida uma alteração no parâmetro relacionado ao número de processos que o Cacti utiliza para fazer os pollings. Por padrão, o Cacti utiliza apenas um processo, mas esse número foi aumentado para dois. O problema continuou, então aumentando os números de processos para 3, 4, 5 e 6, notou-se que o tempo para reinicialização da estação apenas aumentava em alguns dias, porém a memória Swap continuava caindo até próximo de zero. Foi então adicionado uma quantidade maior de memória RAM passando de 512 MBytes originalmente para 1 GByte. Um número de processos menor com a quantidade de dispositivos utilizados poderia fazer com o tempo total do polling ultrapassasse os 5 minutos de intervalo entre cada polling. Caso isso ocorresse, o próprio suporte no site oficial do Cacti recomendou a utilização do *cactid*, que consiste em uma versão do mecanismo de poller da ferramenta desenvolvido em C e que tende ser mais rápido do que a utilização padrão. Entretanto, seguindo a mesma recomendação, não foi preciso tal utilização sabendo que o tempo total do poller ficava em torno de 47 segundos.

4.2.4 **Configuração dos roteadores**

O problema de rede diagnosticado com a utilização do PHP Weathermap no Cacti foi a inversão de rotas que fazia com que o tráfego originado de uma rede remota utilizasse o enlace redundante ao invés do enlace principal, onde a política de utilização do enlace redundante no ambiente é de uso somente em caso de pane do enlace principal. Nas figuras abaixo pode ser visto o problema identificado nas interfaces dos roteadores principal e redundante, onde antes não era possível ter conhecimento do problema.

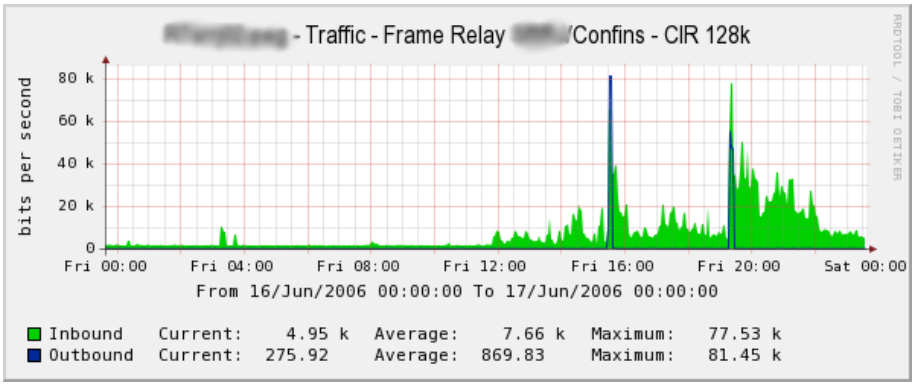

Figura 31 – Utilização do enlace redundante.

A figura 31 mostra o tráfego de entrada do roteador redundante no enlace de 128 Kbps com tráfego acima do esperado (em verde), enquanto o tráfego com destino à rede remota (em azul), é quase que totalmente identificado no enlace do roteador principal de 2 Mbps, como pode ser visto na figura 32.

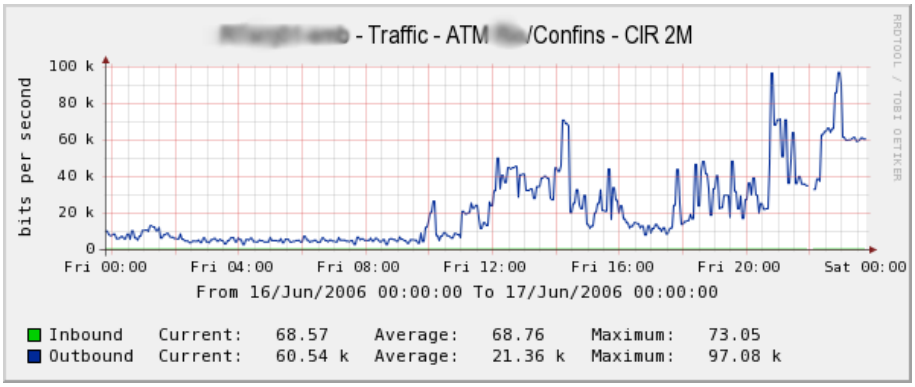

Figura 32 – Utilização do enlace principal.

O problema também pôde ser confirmado através da simples utilização do comando *Tracert* (Windows) ou *Traceroute* (Unix). Em uma estação localizada na rede principal, onde estão localizados os roteadores principal e redundante, a resposta do comando disparado e tendo como alvo uma estação remota, foi o roteador principal como primeiro gateway. Todavia, o mesmo comando executado em uma estação remota tendo como alvo uma estação da rede principal, mostrava como primeiro gateway, o roteador redundante da localidade, ou seja, um loop ocasionado pela inversão de rotas estáticas. A situação, depois de identificada com a ajuda do Cacti, foi corrigida e mantida segundo as normas de utilização vigentes da empresa.

4.2.5 **Configuração de tráfego do backbone**

Alguns dos Switches utilizados no ambiente possuem enlaces agregados de 2 Gbps, porém não havia até o momento alguma ferramenta que informasse o quão eficiente estavam esses enlaces especiais em si tratando de utilização e balanceamento de carga. Com a utilização do Cacti e do plugin PHP Weathermap, foi possível verificar que alguns desses enlaces não estavam trabalhando com o balanceamento de carga necessário. Foi evidente verificar esse detalhe nas figuras abaixo, onde os enlaces agregados entre três switches de grande porte estavam com folga de utilização, porém o balanceamento de carga não se mostrava presente. Na figura 33, o switch denominado Core01 possui duas interfaces ligadas ao um outro switch da mesma categoria, o Core02, sendo que as ligações entre esses equipamentos estão utilizando essas interfaces com diferentes cargas. Nos gráficos correspondentes, os momentos em que o tráfego é maior, não existe balanceamento de carga entre as interfaces. Nas figuras 33 e 34 é possível ver que no momento em que os mapas foram gerados, o Core01 tinha um tráfego de entrada e saída de 48,4 Mbps e 42,2 Mbps respectivamente pela interface 9 (Port 9), enquanto o tráfego detectado pela outra interface (Port 10) era de 1,3 Mbps de entrada e 18,2 Mbps de saída. Com essa análise foi possível rever alguns procedimentos de configuração desses equipamentos e melhorar a eficiência desse recurso suportado pelos equipamentos.

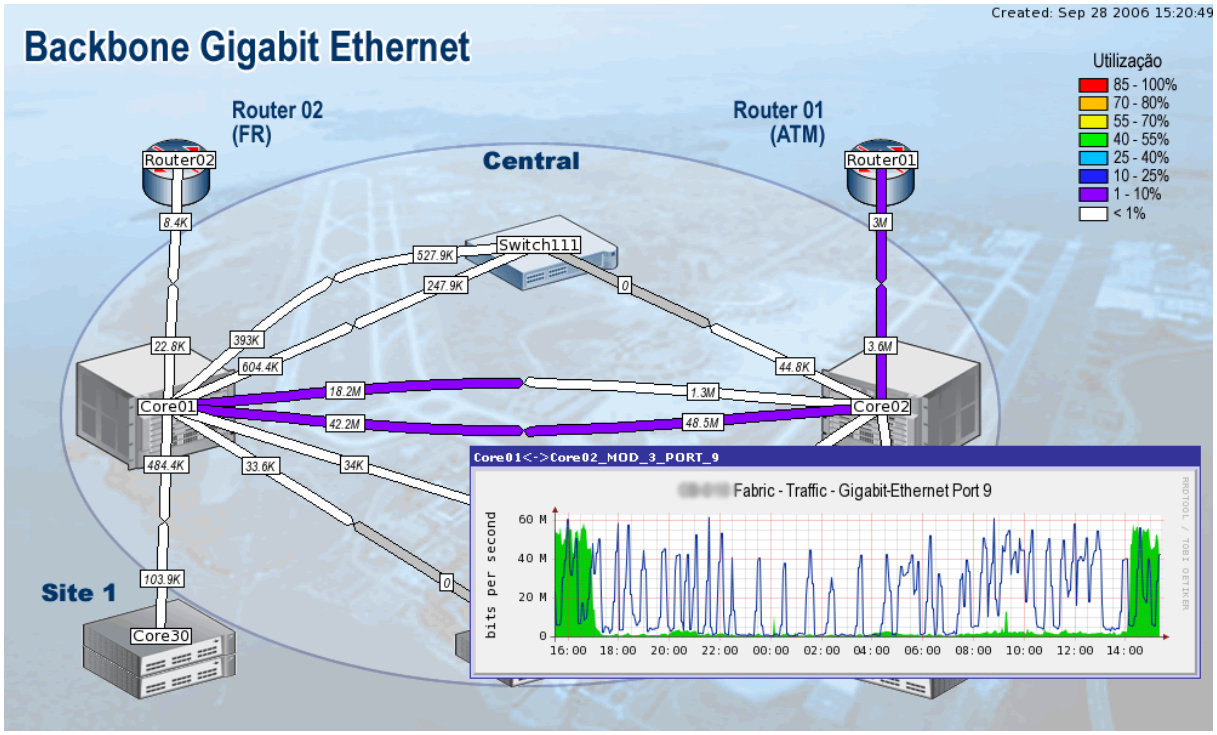

Figura 33 – Tráfego na interface Port 9.

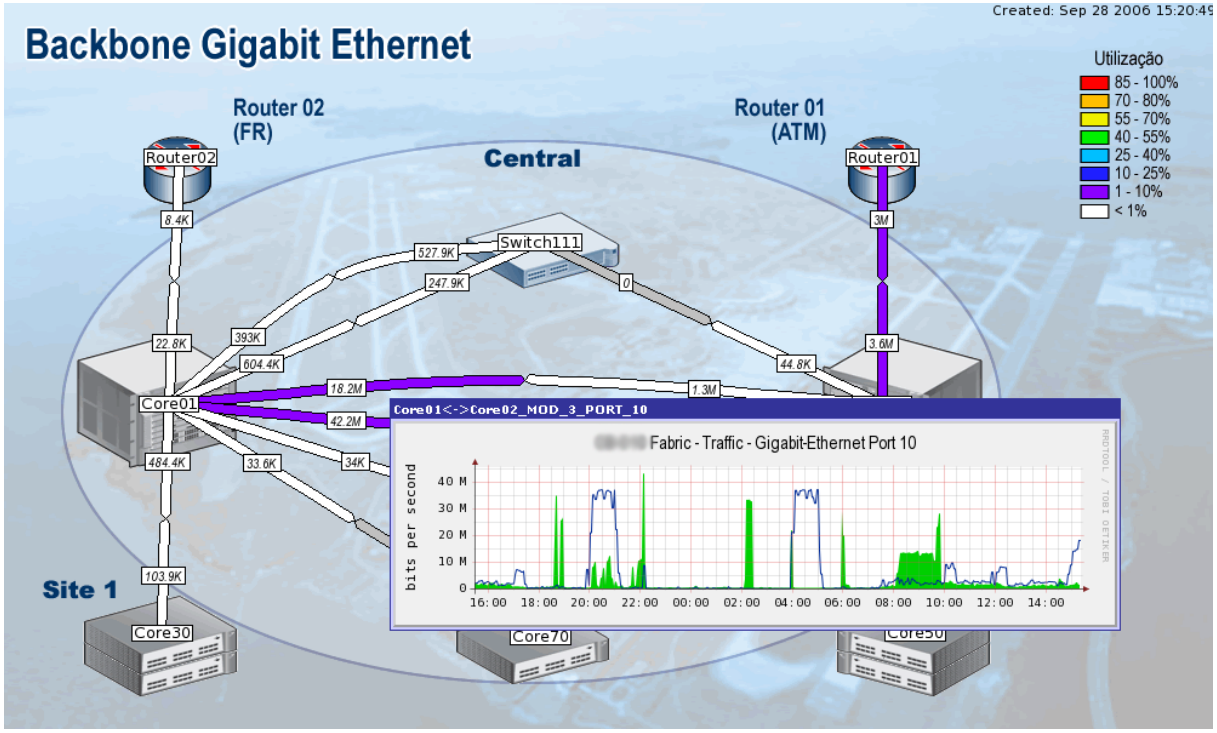

Figura 34 – Tráfego na interface Port 10.

4.3 A SOLUÇÃO SOB O FOCO DA GERÊNCIA DE CONTABILIDADE

Nesse âmbito em questão, as duas ferramentas não possuem recursos que pudessem ser utilizados para fins de tarifação ou contabilização com a mesma precisão e tratamento de ferramentas específicas. O Cacti é a ferramenta que mais se aproxima desse perfil. Nenhuma das ferramentas tem esse foco em questão, mas sim no gerenciamento dos recursos de rede, e nesse ponto, as ferramentas apresentaram recursos e funções de forma satisfatória.

4.4 A SOLUÇÃO SOB O FOCO DA GERÊNCIA DE DESEMPENHO

4.4.1 **Monitoria dos recursos de rede**

Nesta área o Cacti foi de grande importância. A ferramenta juntamente com o PHP Weathermap, foi decisiva em algumas tomadas de decisões e detectaram alguns problemas, que sem uma ferramenta com essas características, não seria possível. O Zabbix também oferece com a mesma eficiência, a possibilidade de gerar gráficos estatísticos com base nos alertas utilizados e nos dados obtidos através dos testes de conexão, do protocolo SNMP e das informações dos agentes instalados. Entre alguns gráficos e diversos tipos de relatórios gerados pelo Zabbix, alguns mostram os tempos de disponibilidade de serviços ou de dispositivos como mostrado na figura 35.

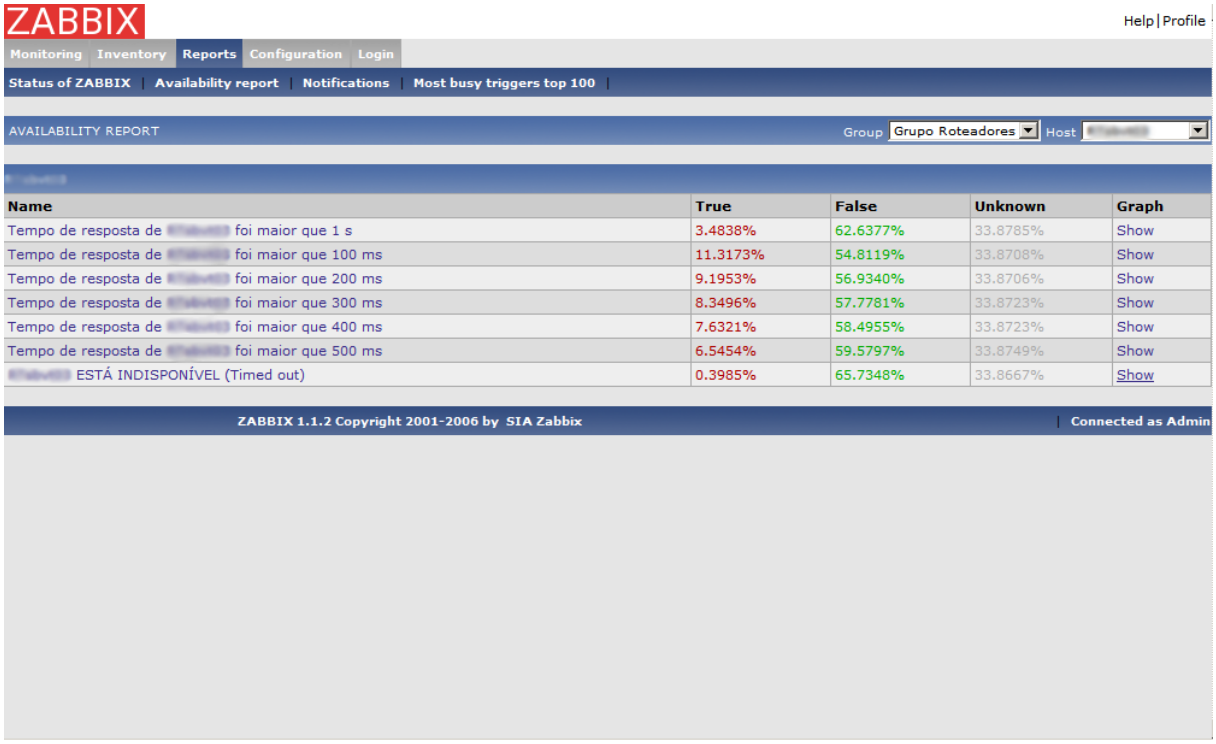

Figura 35 – Exemplo de relatório de disponibilidade de um roteador.

No Cacti, a possibilidade de armazenar as informações pelo período de um ano, fornece suporte aos gerentes que tratam da utilização de, além de fornecer dados estatísticos suficientes para se justificar a utilização ou substituição de algum recurso ou detectar a subutilização de outros.

4.4.2 **Demanda de utilização dos servidores**

Durante algum tempo levantou-se a questão da rede local estar com algum problema ou gargalo causando uma lentidão na utilização do servidor Lotus Notes. Levantou-se também a questão de alto processamento do processador ou dos discos. Com a utilização dos gráficos do Cacti, foi possível verificar o alto tráfego destinado a este servidor que alcançava níveis próximos a 70%. Esse processo de detecção possibilitou justificar a compra de uma interface Gigabit Ethernet para este servidor (Figura 36).

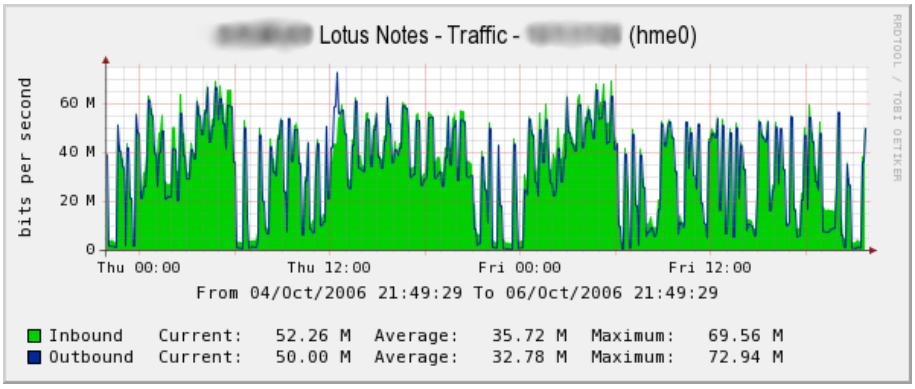

Figura 36 – Tráfego do servidor Lotus Notes.

Também foi possível identificar qual era o impacto gerado pela estação NMS na rede. Como é possível verificar na figura 37, em um primeiro momento, a estação gerava na rede um tráfego de aproximadamente 4 Kbps para monitorar cerca de 90 itens diferentes, incluindo dispositivos, interfaces e recursos de servidores. Em um segundo momento, por volta das 23 horas, foram adicionados aproximadamente de 40 a 50 itens diferentes, o que mostrou não aumentar expressivamente o tráfego gerado pela estação NMS.

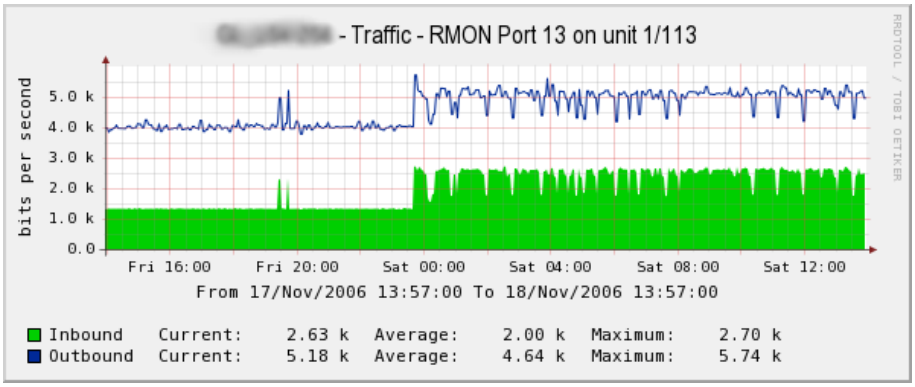

Figura 37 – Tráfego gerado pela estação NMS.

4.5 A SOLUÇÃO SOB O FOCO DA GERÊNCIA DE SEGURANÇA

O Zabbix foi a ferramenta que mostrou um tipo de recurso de auditoria mais completo. As ações dos usuários cadastrados e com permissão de acesso a ferramenta, ficam registradas e são acessíveis, por padrão, apenas pelo usuário administrador. Detalhes das ações são descritos, além do recurso e do momento do evento são registrados, como pode ser visto na figura 38.

ZABBIX

Help | Profile

Monitoring | Inventory | Reports | Configuration | Login

General | Users | Audit | Hosts | Items | Triggers | Actions | Maps | Graphs | Screens | IT services | Bulkloader

AUDIT LOG

<< Prev 100 | Next 100 >>

Time	User	Resource	Action	Details
2006.Nov.11 17:55:05	operador	Host	Deleted	Host [ ]
2006.Nov.11 17:54:57	operador	Host	Updated	Old status [0] New status [1]
2006.Nov.11 17:53:59	operador	Host	Added	Host [ ] IP [ ] Status [0]
2006.Nov.11 17:53:09	operador	Host	Added	Host [ ] IP [ ] Status [0]
2006.Nov.11 17:48:17	Admin	User	Added	User alias [operador] name [operador] surname [operador]]
2006.Nov.11 17:47:50	Admin	User	Deleted	User alias [fabio] name [fabio] surname [fabio]
2006.Nov.11 17:42:58	Admin	User	Added	User alias [fabio] name [fabio] surname [fabio]]
2006.Oct.14 16:20:20	Admin	User	Updated	User alias [Admin] name [Zabbix] surname [Administrator]]
2006.Oct.10 21:09:28	Admin	Host	Updated	Host [Cabbix] IP [127.0.0.1] Status [0]
2006.Oct.10 17:05:23	Admin	Configuration of ZABBIX	Updated	Image added
2006.Oct.10 17:05:11	Admin	Configuration of ZABBIX	Updated	Image added
2006.Oct.10 17:04:22	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:04:08	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:58	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:49	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:40	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:31	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:21	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.10 17:03:11	Admin	Configuration of ZABBIX	Updated	Image updated
2006.Oct.09 19:08:18	Admin	Host	Updated	Old status [1] New status [0]
2006.Oct.09 19:08:14	Admin	Host	Updated	Old status [0] New status [1]
2006.Oct.09 19:07:57	Admin	Host	Updated	Host [ ] IP [ ] Status [0]
2006.Oct.08 20:53:57	Admin	Configuration of ZABBIX	Updated	Image updated

Figura 38 – Auditoria de usuários no Zabbix.

Ambas as ferramentas possuem suporte às versões 1, 2 e 3 do protocolo SNMP. No ambiente não foram utilizadas as versões 2 e 3. Em alguns fóruns a respeito do Zabbix, usuários se queixaram quando utilizaram recursos com suporte ao SNMPv2, e disseram não receber as respostas às requisições do protocolo de forma correta.

O Cacti não possui um recurso de auditoria avançado, apenas alguns recursos para visualizar logs de eventos ocorridos, mas também mostrou compatibilidade com as versões posteriores do SNMP.

5 **CONCLUSÃO**

Durante alguns poucos meses que se seguiram com a implementação da solução, colocando a mesma em total funcionamento e integração com o ambiente de produção, diversos foram os pontos de vista que surgiram, principalmente nos momentos em que exigiram tempo além do esperado para solucionar algum tipo de problema, desde a escolha do perfil do ambiente a ser gerenciado, até problemas com a configuração de bibliotecas e outros relacionados ao sistema operacional ou ao hardware dedicado. Foram nesses momentos que a escolha de uma solução totalmente aberta, sem uso de qualquer tipo de licença pesou e ao mesmo tempo foram esses momentos que fortaleceram a idéia junto aos responsáveis pelo ambiente e solidificaram alguns conceitos, sobre tudo, no que se refere aos procedimentos e metodologias utilizadas pelos gerentes de rede em seu local de trabalho.

Todo o trabalho e tempo dispensados com a instalação e otimização da solução seriam em vão se, antes de todo o processo, alguns estudos e criação de procedimentos não fossem adotados. Foi necessário mapear todos os dispositivos que seriam monitorados, de que forma seriam monitorados, com qual periodicidade e qual o impacto da gerência em cada um deles. Não teria sentido fazer testes de portas TCP em roteadores na Wan, sabendo que estes possuem enlaces de baixa velocidade, ou que suportam uma carga de tráfego alta durante algum período do dia. Do mesmo modo que não fez sentido gerar gráficos com o tempo de resposta do protocolo ICMP, para dispositivos que estivessem na mesma rede local, e que utilizassem interfaces Fast ou Gigabit Ethernet. Com base nessas premissas, foram criados alguns perfis de gerência, sendo tipos de perfil para grupos de dispositivos comuns ou que tivessem características semelhantes na rede.

Como exemplo, pode-se dizer que os roteadores localizados em enlaces remotos e principalmente de baixa velocidade, somente eram monitorados através do protocolo ICMP e de forma otimizada com o SNMP. Sendo que o Zabbix realizava os testes e gerava avisos ou alarmes com base nos tempos de resposta. O Cacti por sua vez, gerava os gráficos com os tempos de resposta, a utilização de processador dos roteadores e a taxa de utilização dos respectivos enlaces Wan (com exceção de interfaces de rede local, interfaces de loopback e outros que não estivessem realmente sendo utilizados). O período de consulta a esses roteadores

também foi uma questão levantada. Ficou determinado em 1 minuto para testes via requisições ICMP e 5 minutos para as requisições via SNMP. Um tempo menor acarretaria um possível congestionamento nos enlaces utilizados por esses dispositivos.

Os servidores e equipamentos de rede (concentradores e switches de distribuição) também tiveram de ser estudados e colocados em perfis de gerenciamento, pois demandavam de necessidades e padrões de funcionamento distintos. No ambiente utilizado, apesar da necessidade dos servidores de funcionaram em período contínuo, com pouca ou nenhuma necessidade de reinicialização, a preocupação em saber com qual freqüência os equipamentos de rede eram reinicializados era maior. Não foi explanado com maiores detalhes anteriormente, mas pode-se dizer que além do cliente interno, diversos outros clientes externos utilizam os recursos de infra-estrutura física e lógica desta empresa e esta estrutura tende a ser a mais crítica e necessitar da maior disponibilidade possível.

Os servidores tiveram diversos perfis de gerência, dependendo da função e do grau de criticidade de suas funções. Não foi possível, por exemplo, utilizar o agente do Zabbix em diversos servidores, entre eles alguns com bancos de dados Oracle. Por regimento interno, não era possível utilizar componentes de software sem homologação, em servidores de produção. Nesses casos, somente o monitoramento com o ICMP, de serviços através do TCP, e em alguns casos com o uso do SNMP foi permitido. Em alguns servidores, menos críticos, foi possível a instalação do agente Zabbix. Desse modo foi possível fazer a análise desse recurso com a mesma confiabilidade que seria possível nos demais servidores, pois esses casos em questão foram tratados com a mesma importância dos servidores críticos, mesmo oferecendo uma forma de análise mais dinâmica.

As duas ferramentas se mostraram bastante flexíveis e permitiram um alto grau de adequação e otimização para o ambiente utilizado. Com os diversos tipos de plataformas disponíveis, foi possível com o Zabbix, monitorá-los de alguma forma e não existiram elementos na rede que não entrassem, de alguma maneira, na gerência através da solução aplicada. A forma como as ferramentas disponibilizam os gráficos de topologia, possibilitaram colocar a real situação dos elementos que compõe todo o sistema e adequaram seus mecanismos de monitoramento sem maiores problemas à realidade encontrada.

Em diversos pontos, foi possível notar como uma ferramenta pode complementar a outra. Os gráficos gerados através do PHP Weathermap no Cacti mostraram um ponto de vista de utilização dos enlaces que não é comum em muitas ferramentas, inclusive no Zabbix e principalmente em ferramentas comerciais, com exceção de algumas aplicações desenvolvidas especificadamente para uma família de produtos e que por isso, têm um custo elevado. Todavia, o Cacti com a instalação padrão e algumas otimizações, não mostrou a mesma facilidade que o Zabbix possui nativamente para monitorar serviços e sistemas principalmente em servidores, e ao mesmo tempo gerar relatórios e logs suficientemente explicativos.

Não foram encontradas grandes dificuldades com relação ao suporte às ferramentas e seus diversos componentes. Sem contar os diversos fóruns especializados sobre o assunto, grande parte do auxílio foi obtida a partir de artigos, sites de terceiros, fóruns de assuntos variados e diversos documentos também relacionados. Muitas ferramentas comerciais também oferecem serviços de helpdesk e suporte on-line, todavia a simples troca de mensagem entre profissionais através da Web se mostra muito mais rápida e eficaz na maioria dos casos.

As funcionalidades colocadas em prática nesta solução podem não resolver 100% dos problemas encontrados nos ambientes existentes, mas fornecem uma base sólida para o desenvolvimento e implementação de novos recursos e utilização de diversas funções disponíveis na Web, e tudo isso por uma fração do investimento aplicado em soluções comerciais ou específicas. Esse benefício foi possível através da flexibilidade e integração já inerentes às soluções abertas.

O desenvolvimento desta solução fez com que políticas e procedimentos de gerência fossem revistas e melhoradas, porém ficou evidente que o trabalho apenas começou e que novos desafios estão por vir. Com a base de conhecimento obtida no uso de ferramentas como as aqui mostradas, o trabalho de gerência e controle frente às mudanças rápidas de tecnologia, padrões e filosofias aplicadas à área de TI, é mais bem desenvolvido e adequado à realidade e os conhecimentos são aplicados com maior eficácia. O gerente ou equipe responsável nesse processo se especializa, expande o conhecimento técnico e gerencial e é reconhecido através do aumento da produtividade e da valorização profissional.

REFERÊNCIAS

[1] GNU Project. **GNU General Public Lisence**. Disponível em: <http://www.gnu.org/copyleft/gpl.html>. Acesso em: 15 de maio de 2006.

[2] PEREIRA, Mateus Casanova. **Administração e Gerência de Redes de Computadores**. Florianópolis, 2001. Disponível em: <http://www.lrg.ufsc.br/ine6404/Mateus.pdf>. Acesso em: 10 de Abril de 2006.

[3] TEIXEIRA, Suzana Ramos; MORAES, Luís Felipe M. de; JUNIOR, José Helvécio Teixeira. **Relatório Técnico "Gerência Baseada na Web"**. Rio de Janeiro. Disponível em: <http://www.ravel.ufrj.br/arquivosPublicacoes/suzana_rel_webmngt.pdf>. Acesso em: 10 de Abril de 2006.

[4] SANTOS, Carlos Eduardo Meyer dos. **Plataformas de Gerenciamento**. Rio de Janeiro, 2002. Disponível em: <http://www.gta.ufrj.br/~rezende/cursos/eel879/trabalhos/gerenciamento>. Acesso em: 10 de Abril de 2006.

[5] K. MCCLOGHRIE, M. ROSEREQUEST. **Request For Comments: 1066. Management Information Base for Network Management of TCP/IP-based internets**. Agosto de 1988. Disponível em: <ftp://ftp.rfc-editor.org/in-notes/rfc1066.txt>. Acesso em 28 de janeiro de 2007.

[6] K. MCCLOGHRIE, M. ROSE. **Request For Comments: 1213. Management Information Base for Network Management of TCP/IP-based internets: MIB-II**. Março de 1991. Disponível em: <ftp://ftp.rfc-editor.org/in-notes/rfc1213.txt>. Acesso em 28 de janeiro de 2007.

[7] OLIVEIRA, Décio Tostes. **Gerência de Redes de Computadores: Uma abordagem com o uso do SNMP**. Uberlândia, 2002. Disponível em: <http://www.computacao.unitri.edu.br/downloads/monografia/28211129405651>. Acesso em: 20 de outubro de 2006.

[8] DIAS, Beethovem Zanella; JR, Nilton Alves. **Protocolo de Gerenciamento SNMP**. Rio de Janeiro, 2002. Disponível em: <http://mesonpi.cat.cbpf.br/naj/snmp_color.pdf>. Acesso em 27 de outubro de 2006.

[9] OLIVEIRA, Mauro; FRANKLIN, Miguel; NASCIMENTO, Adriano; VIDAL, Marcelo. **Introdução à Gerência de Redes ATM**. XVI Simpósio Brasileiro de Redes de Computadores, Rio de Janeiro, 1998. Disponível em: <http://www.cefetce.br/Ensino/Professores/Publicacoes/livrogerencia.pdf>. Acesso em: 02 de outubro de 2006.

[10] PEIXOTO, João Carlos. **Gerência Estratégica SNMP**. Apostila utilizada como material didático no Curso de Pós-Graduação em Gerência de Redes – NCE/UFRJ. Rio de Janeiro, 2006.

[11] HILL, Benjamin Mako; HARRIS, David B.; VYAS, Jaldhar. **Debian GNU/Linux 3.1 Bible**. 1. ed. Indianápolis, IN: Wiley, 2005. 672 p.

[12] InfoWester. **Banco de dados PostgreSQL e MySQL**. Disponível em: <http://www.infowester.com/postgremysql.php>. Acesso em: 15 de maio de 2006.

[13] InfoWester. **Linguagem PHP**. Disponível em: <http://www.infowester.com/php.php>. Acesso em: 15 de maio de 2006.

[14] ZABBIX. **ZABBIX Manual v1.4**. Disponível em: <http://www.zabbix.com/downloads/ZABBIX%20Manual%20v1.4.pdf>. Acesso em: 28 de janeiro de 2007.

[15] WIKIPEDIA. **System Call**. Disponível em: <http://en.wikipedia.org/wiki/System_calls>. Acesso em: 28 de janeiro de 2007.

[16] MRTG. **Multi Router Traffic Grapher**. Disponível em: <http://oss.oetiker.ch/mrtg/>. Acesso em: 23 de junho de 2006.

[17] MRTG-RRD. **Integração com o RRDTOOL**. Disponível em: <http://www.mrtg.jp/en/pt/mrtg-rrd.html>. Acesso em: 28 de janeiro de 2007.

[18] MRTG. **Mrtg-rrd**. Disponível em: <http://oss.oetiker.ch/mrtg/doc/mrtg-rrd.en.html> . Acesso em: 10 de setembro de 2006.

[19] CACTI. **Cacti Users**. Disponível em: <http://cactiusers.org/>. Acesso em: 15 de maio de 2006.

[20] PHP NETWORK WEATHERMAP. **Howie's Stuff::Network Stuff::PHP Network Weathermap**. Disponível em: <http://wotsit.thingy.com/haj/cacti/php-weathermap/>. Acesso em: 19 de junho de 2006.

ANEXO A – PROCEDIMENTOS DE SEGURANÇA NA ESTAÇÃO NMS

A.1 **Arquivo Securetty**

No arquivo ***/etc/securetty***, na seção ***Standard Consoles***, foram comentadas todas as linhas, indicando que o usuário root não poderá acessar a estação através do teclado, mas somente através de usuário comum. Caso este deseje ter privilégio de root, deverá executar “- su” e colocar a senha de root.

A.2 **Arquivo Sshd.conf**

No arquivo ***/etc/ssh/sshd.conf***, na seção ***Authentication*** a linha ***PermitRootLogin*** foi colocada como **no**, assim caso um usuário queira acessar a estação, deverá entrar como um usuário comum e então se tornar um usuário privilegiado através do comando “su –“.

ANEXO B – Instalação e configuração do Cacti no Debian GNU/Linux 31r2

B.1 **Instalação e configuração inicial**

1.1 – Crie uma estrutura de diretórios semelhante a esta para auxiliar na instalação e configuração das ferramentas.

```
/downloads
/www
```

1.2 – Coloque os pacotes que serão utilizados em um lugar comum: **/downloads**. Os pacotes aqui mostrados eram as versões mais recentes até o fechamento deste documento:

```
httpd-2.2.2.tar.gz
php-4.4.2.tar.gz
cacti-0.8.6h.tar.gz
zlib-1.2.3.tar.gz
libpng-1.2.10.tar.gz
freetype-2.1.10.tar.gz
libart_lgpl-2.3.17
rrdtool-1.2.14.tar.gz
gd2-2.0.33.tar.gz
cacti-plugin-arch.tar.gz
php-weathermap-0.82.tar.gz
```

1.3 – Instale os seguintes pacotes disponíveis no Debian. A ferramenta de geração de dados em série e o servidor MySQL. Será vista no final do documento, a atualização da versão do 1.2.14, mais recente que a versão instalada por padrão pelo gerenciador de pacotes do Debian, porém, nada impede que a versão mais recente seja compilada antes, ao invés de se instalar a versão padrão:

```
# apt-get install rrdtool
# apt-get install mysql-server
```

1.4 – Instale as ferramentas SNMP que serão usadas pelo Cacti, contidos no seguinte pacote:

```
# apt-get install snmp
```

1.5 – Instale o SNMPD para que a estação Cacti também possa ser gerenciada. As configurações de snmp para os servidores e dispositivos gerenciados serão vistas mais à frente:

```
# apt-get install snmpd
```

1.6 – Vá para o diretório /downloads e instale o Apache 2 (httpd-2.2.2). Leve em consideração que o diretório raiz de trabalho do servidor Apache será **/www**:

```
# tar xvzf httpd...
# cd httpd...
```

1.7 – Compile o Apache. Sem parâmetros, o Apache será instalado em **/usr/local/apache2**:

```
# ./configure
# make
# make install
```

1.8 – Edite o arquivo /**usr/local/apache2/config/httpd.conf** e faça as seguintes alterações:

Linha original:

```
DocumentRoot "/usr/local/apache2/htdocs"
```

Linha após edição:

```
DocumentRoot “/www”
```

Linha original:

```
<Directory "/usr/local/apache2/htdocs">
```

Linha após edição:

```
<Directory “/www”>
```

Linhas originais:

```
<IfModule dir_module>
    DirectoryIndex index.html
</IfModule>
```

Linhas após edição:

```
<IfModule dir_module>
    DirectoryIndex index.html index.php
</IfModule>
```

1.9 – Crie os seguintes arquivos em **/usr/bin** com o comando:

```
# echo /usr/local/apache2/bin/apachectl –k start > usr/bin/apachestart
# echo /usr/local/apache2/bin/apachectl –k stop > /usr/bin/apachestop
```

1.10 – Torne-os executáveis

```
#chmod 774 /usr/bin/apachest*
```

1.11 – Crie links simbólicos em **/etc/rc2.d** para que o Apache seja inicializado no boot e fechado adequadamente no desligamento do sistema. Levando em consideração que o perfil do boot do Debian é o 2 (ver arquivo **/etc/inittabd** linha **id:2:initdefault:**):

```
ln –s /usr/bin/apachestart /etc/rc2.d/S98apache2
ln -s /usr/bin/apachestop /etc/rc2.d/K98apache2
```

1.12 – Voltando para o diretório /downloads, compile as seguintes bibliotecas necessárias para o Cacti e Weathermap:

```
# tar xvzf zlib-1.2.3.tar.gz
# cd zlib...
# ./configure
# make
# make install
# cd ..

# tar xvzf libpng-1.2.10.tar.gz
# cd libpng...
# ./configure
# make
# make install
# cd ..


# tar xvzf libart_lgpl-2.3.17
# cd libart...
# ./configure
# make
```

```
# make install
# cd ..

# tar xvzf freetype-2.1.10.tar.gz
# ./configure --prefix=/usr/include
# make
# make install
# cd ..

# tar xvzf gd2-2.0.33.tar.gz
# ./configure
# make
# make install
# cd ..
```

1.13 – Voltando para o diretório /downloads, compile o PHP4. Da forma descrita abaixo será instalado o php para o apache2, com suporte ao MySQL, e permitindo ao PHP a gerar imagens PNG com fontes personalizadas.

```
# tar xvzf php-4.4.4…
# cd php...
# ./configure --with-apxs2=/usr/local/apache2/bin/apxs --with-mysql \
 --with-gd --with-zlib-dir=/usr/include --with-freetype-
dir=/usr/include
# make
# make install
# cp php.ini-dist /usr/local/lib/php.ini
```

1.14 – Verifique no arquivo **/usr/local/apache2/config/httpd.conf** do Apache2 se foi criada a seguinte linha. Se não, adicione:

```
LoadModule php4_module modules/libphp4.so
```

1.15 – Adicionar também as seguintes linhas a este arquivo:

```
AddType application/x-httpd-php .php .phtml
AddType application/x-httpd-php-source .phps
```

1.16 – Volte ao diretório /downloads e instale o Cacti:

```
# tar xvzf cacti...
```

1.17 – Mova (renomeando) o diretório criado pelo tar (cacti-0.8.6h) para o diretório **/www/**:

```
# mv cacti-0.8.6h /www/cacti
```

1.18 – Crie o usuário cacti:

```
# adduser cacti
```

1.19 – Vá para o diretório /www/cacti e crie o banco de dados:

```
# mysqladmin --user=root create cacti
# mysql cacti < cacti.sql
# mysql --user=root mysql
mysql> grant all on cacti.* to cacti@localhost identified by '123456';
mysql> flush privileges;
mysql> exit;
```

1.20 – Edite as seguintes linhas do arquivo **/www/cacti/include/config.php**, que devem ficar no seguinte formato:

```
$database_type = "mysql";
$database_default = "cacti";
$database_hostname = "localhost";
$database_username = "cacti";
$database_password = "123456";
```

1.21 – Dê permissões ao usuário cacti:

```
# chown –R cacti rra/ log/
```

1.22 – Coloque no **/etc/crontab**:

```
*/5 * * * * cacti php /www/cacti/poller.php > /dev/null 2>&1
```

1.23 – Carregue no browser o endereço **http://localhost/cacti**. Se tudo funcionou corretamente, entrará a tela de configuração inicial do Cacti, como mostrado na figura 39.

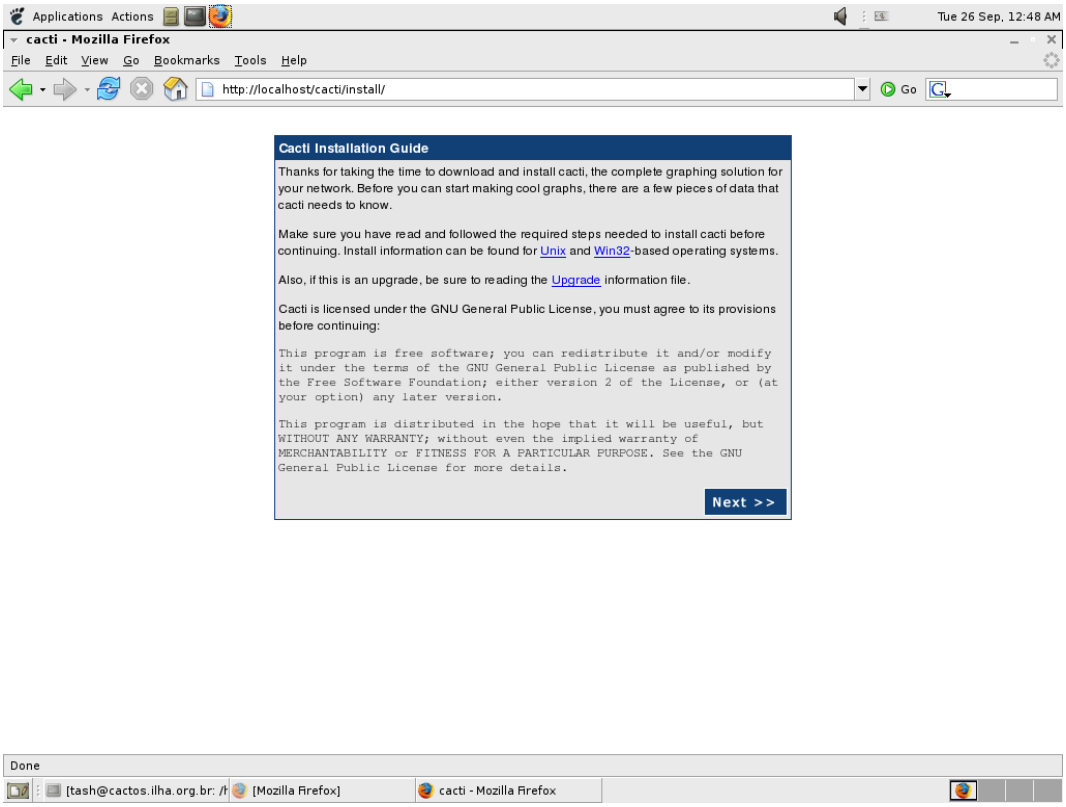

Figura 39 – Tela inicial do Cacti.

Atenção: Recomenda-se fazer a atualização (patches) que estão disponíveis no site do Cacti (www.cacti.net). Bastando apenas executar os comandos disponibilizados no site.

B.2 **Instalando plugins no Cacti**

Os procedimentos a seguir servem para qualquer plugin disponível para o Cacti. Neste caso será instalado apenas o Weathermap como plugin. Para mais informações sobre plugins para o Cacti, visite o site www.cactiusers.org.

2.1 – Antes de instalar qualquer plugin, faça um backup do diretório de trabalho do Cacti:

```
# tar cvf /home/cacti/backup-cacti.tar /www/cacti/.
```

2.2 – Dentro do diretório /downloads, instale o patch Plugin Architecture:

```
# tar xvzf cacti-plugin-arch...
# cd cacti-plugin…
```

2.3 – Dentro do diretório criado pelo tar existem vários diretórios relacionados à versão do Cacti utilizada. É necessário copiar os arquivos do diretório correspondente à versão do Cacti, para o diretório de trabalho do Cacti (neste caso estamos usando a versão **0.8.6h** do Cacti):

```
# cd files-0.8.6h
# cp –av * /www/cacti
```

2.4 – Edite o arquivo **/www/cacti/include/config.php**, e reconfigure o acesso ao banco de dados nas linhas iniciais, como no arquivo original.

Atenção: Não copie o arquivo config.php do backup realizado, apenas reedite as linhas iniciais referente ao acesso ao banco de dados.

2.5 – No mesmo arquivo config.php, edite a seguinte linha de forma que fique desta maneira:

```
$config[“url_path”] = ‘/cacti/’;
```

Atenção: mantenha as duas barras em ‘/cacti/’

2.6 – Recarregue a página do cacti. Se tudo der certo, o Cacti carregará a página da mesma forma como antes, sem nenhuma modificação visual.

2.7 – Vá para o diretório /downloads e instale o Weathermap:

```
# tar xvzf php-weathermap...
# mv php-weathermap... /www/cacti/plugins/weathermap
```

2.8 – Edite as seguintes linhas do arquivo **/www/cacti/include/config.php** da seguinte forma:

```
$plugins = array();
$plugins[] = 'weathermap';
;
```

Atenção: O nome do plugin deve ser exatamente o mesmo do diretório dentro de /www/cacti/plugins.

2.9 – Recarregue o Cacti. Acesse o menu ***User Management***, selecione o usuário que se queira ter acesso aos mapas. Dois ***realms*** aparecerão Configure ***Weathermaps*** e ***View Weathermaps***. Selecione os que forem de interesse para o usuário e clique em Save.

2.10 – Uma nova Aba – **Weathermap** – aparecerá no alto da tela ao lado de Graphs.

B.3 **Criando mapas no Weathermap**

3.1 – Dentro do diretório **/www/cacti/plugins/weathermap**, crie a seguinte estrutura de diretório que auxiliarão na criação e organização dos elementos dos mapas:

Tabela 7 – Estrutura de diretórios do Weathermap.

/www/cacti/plugins/weathermap/config/	diretório onde ficarão os arquivos de configuração dos mapas
/www/cacti/plugins/weathermap/config/bg/	diretório onde ficarão as imagens de fundo dos mapas
/www/cacti/plugins/weathermap/config/fontes/	diretório onde ficarão as fontes utilizadas nos mapas
/www/cacti/plugins/weathermap/config/imagens/	diretório onde ficarão os ícones utilizados nos mapas

3.2 – No diretório config, deverão ficar os arquivos de configuração dos mapas com a extensão **.conf**. Estes arquivos contêm os parâmetros que definirão como será o mapa. Outro arquivo obrigatório, que já se localiza nesse diretório por padrão, é o **overlib.js**. Este arquivo é utilizado para criar uma janela popup ao se passar com o mouse sobre um link, e assim mostrar um gráfico específico do Cacti. Este arquivo será comum para qualquer mapa criado que utilize esse recurso. Nos demais diretórios ficarão os arquivos auxiliares que ajudarão graficamente o mapa.

O mapa mostrado na figura 40 será utilizado como exemplo para se descrever os parâmetros utilizados no arquivo de configuração:

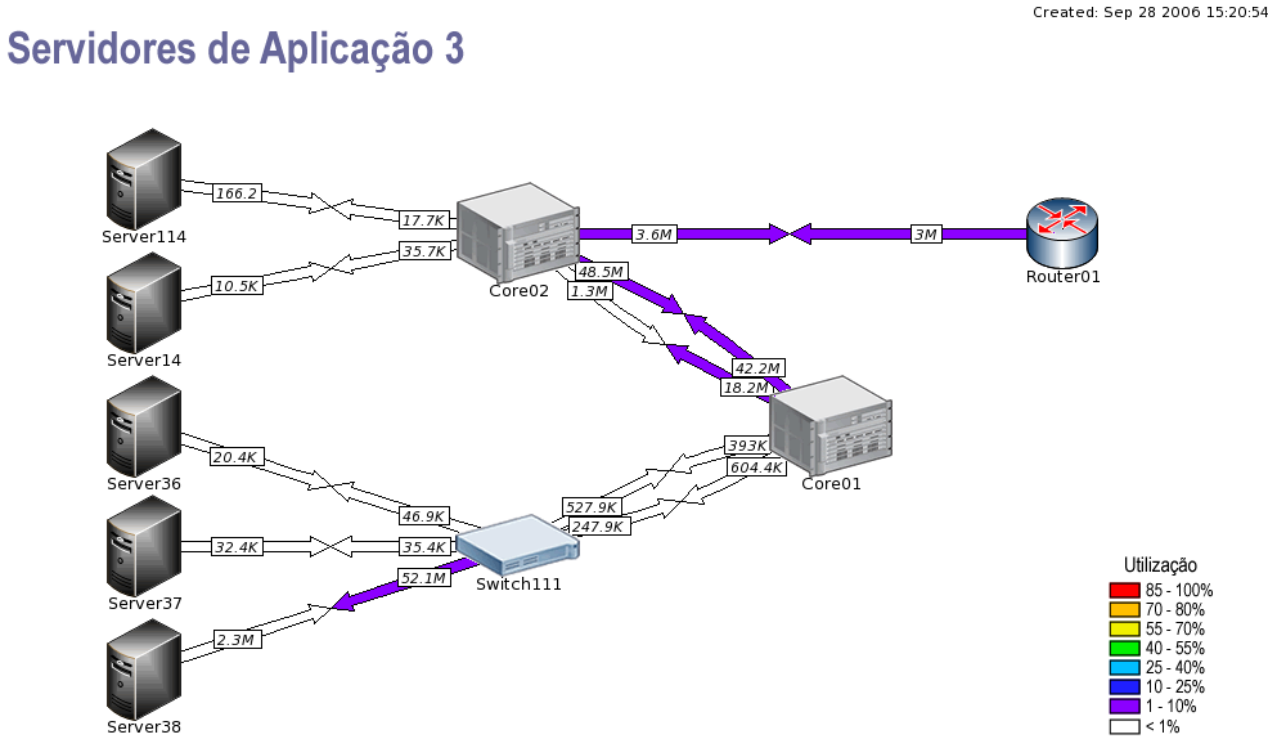

Figura 40 – Exemplo de mapa de topologia.

Para criar o mapa acima, são utilizados os seguintes arquivos, a partir do diretório config:

Tabela 8 – Arquivos utilizados na configuração de mapas.

aplic3.conf	arquivo de configuração
overlib.js	-
bg/aplic3.png	imagem de fundo com o título e a legenda
fontes/VeraIt.ttf	fonte utilizada o rótulo dos links
fontes/Vera.ttf	fonte utilizada no rótulo dos equipamentos e na data
imagens/switch-blue.png	imagem do switch menor
imagens/switch-gray.png	imagem do switch maior
imagens/server-black.png	imagem do servidor
imagens/router-blue.png	imagem do roteador

3.3 – Abaixo, é mostrado o conteúdo do arquivo **aplic3.conf** utilizado para criar o mapa, com a descrição dos parâmetros:

```
# Mapa dos Servidores de Aplicacoes

BACKGROUND /www/cacti/plugins/weathermap/configs/bg/aplic3.png ## imagem de fundo
TITLE Servidores de Aplicacao 3 ## titulo mostrado na lista de mapas do Cacti
HTMLSTYLE overlib

## Definicao das fontes
```

```
## Formato:
## FONTDEFINE <id> <caminho_do_arquivo_fonte> <tamanho_da_fonte>

FONTDEFINE 100 /www/cacti/plugins/weathermap/configs/fontes/VeraIt.ttf 8
FONTDEFINE 101 /www/cacti/plugins/weathermap/configs/fontes/Vera.ttf 10
FONTDEFINE 102 /www/cacti/plugins/weathermap/configs/fontes/Vera.ttf 8

## Definicao da fonte da data/hora do mapa

TIMEFONT 102

## Definicoes padrao para todos os NODES (dispositivos mostrados no mapa)

NODE DEFAULT
        LABELFONT 101

## Definicoes padrao para todos os LINKS (lingacoes entre os dispositivos)

LINK DEFAULT
        WIDTH 4
        BWFONT 100
        BWLABEL bits

## Definicoes especificas de cada NODE
## Formato:
## NODE <nome_do_node>
##      ICON <caminho_da_imagem_que_representara_o_dispositivo>
##      POSITION <pixels_a_partir_da_margem_esquerda> <pixels_a_partir_da_margem_superior>
##      LABEL <rotulo>
##      LABELOFFSET <posicao_do_rotulo_em_relacao_ao_node> -> valores S,N,W,E e demais pontos
##                  cardeais da língua inglesa

NODE Server36
        ICON /www/cacti/plugins/weathermap/configs/imagens/server-black.png
        POSITION 126 329
        LABEL Server36
        LABELOFFSET S

NODE Server37
        ICON /www/cacti/plugins/weathermap/configs/imagens/server-black.png
        POSITION 126 422
        LABEL Server37
        LABELOFFSET S

NODE Server38
        ICON /www/cacti/plugins/weathermap/configs/imagens/server-black.png
        POSITION 126 518
        LABEL Server38
        LABELOFFSET S

NODE Server14
        ICON /www/cacti/plugins/weathermap/configs/imagens/server-black.png
        POSITION 126 233
        LABEL Server14
        LABELOFFSET S

NODE Server114
        ICON /www/cacti/plugins/weathermap/configs/imagens/server-black.png
        POSITION 126 137
        LABEL Server114
        LABELOFFSET S

NODE Core01
        ICON /www/cacti/plugins/weathermap/configs/imagens/switch-gray.png
        POSITION 660 329
        LABEL Core01
        LABELOFFSET S

NODE Core02
        ICON /www/cacti/plugins/weathermap/configs/imagens/switch-gray.png
        POSITION 417 179
        LABEL Core02
        LABELOFFSET S

NODE Switch111
        ICON /www/cacti/plugins/weathermap/configs/imagens/switch-blue.png
        POSITION 417 422
```

```
        LABEL Switch111
        LABELOFFSET S

NODE Router01
        ICON /www/cacti/plugins/weathermap/configs/imagens/router-blue.png
        POSITION 840 179
        LABEL Router01
        LABELOFFSET S

## Definicoes especificas de cada LINK
## Formato:
## LINK <nome_do_link>
## NODES <node_1> <node_2>
##      IMPORTANTE!! - O grafico mostrado no popup durante a passagem do mouse sobre o link,
DEVE
##      ser o grafico referente ao primeiro dispositivo (node1), caso contrario, a link
mostrara
##      o sentido do trafego erroneamente.
## BANDWITH <velocidade_do_link> -> valores em K, M ou G. Os valores em bytes serao mostrados
##      sem letras referentes.
## TARGET <local_do_arquivo_.rrd> VER EXPLICACAO DEPOIS DO ARQUIVO!!
## OVERLIBGRAPH <url_da_imagem_mostrada_no_popup> VER EXPLICACAO DEPOIS DO ARQUIVO!!
## INFOURL <url_da_pagina_com_historico_do_node> VER EXPLICACAO DEPOIS DO ARQUIVO!!

LINK Core02<->Server114
        NODES Core02 Server114
        BANDWIDTH 100M
        TARGET /www/cacti/rra/cb020_modulo_6_traffic_in_112.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=109&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=109&rra_id=all

LINK Core02<->Server14
        NODES Core02 Server14
        BANDWIDTH 100M
        TARGET /www/cacti/rra/cb020_modulo_6_traffic_in_125.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=122&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=122&rra_id=all

LINK Switch111<->Server36
        NODES Switch111 Server36
        BANDWIDTH 1000M
        TARGET /www/cacti/rra/gl_ls4111_traffic_in_82.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=79&rra_id=1&view_type=tree&graph
_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=79&rra_id=all

LINK Switch111<->Server37
        NODES Switch111 Server37
        BANDWIDTH 1000M
        TARGET /www/cacti/rra/gl_ls4111_traffic_in_76.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=73&rra_id=1&view_type=tree&graph
_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=73&rra_id=all

LINK Switch111<->Server38
        NODES Switch111 Server38
        BANDWIDTH 1000M
        TARGET /www/cacti/rra/gl_ls4111_traffic_in_77.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=74&rra_id=1&view_type=tree&graph
_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=74&rra_id=all

LINK Core02<->Router01-emb
        NODES Core02 Router01
        BANDWIDTH 100M
        TARGET /www/cacti/rra/cb020_modulo_6_traffic_in_111.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=108&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=108&rra_id=all
```

```
## parametro VIA -> Utilizado quando se quer fazer um desvio do link.
## Neste caso, os dois dispositivos têm dois links entre eles e sem o parâmetro VIA,
## os dois links ficariam sobrepostos. Com isto, os links sao "desviados" e passam
## pela posicao definida no parametro.
## Formato:
## VIA <pixels_a_partir_da_margem_esquerda> <pixels_a_partir_da_margem_superior>

LINK Core01<->Core02_port1
        NODES Core01 Core02
        BANDWIDTH 1000M
        VIA 528 263
        TARGET /www/cacti/rra/cb010_fabric_traffic_in_105.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=102&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=102&rra_id=all

LINK Core01<->Core02_port2
        NODES Core01 Core02
        BANDWIDTH 1000M
        VIA 546 242
        TARGET /www/cacti/rra/cb010_fabric_traffic_in_104.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=101&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=101&rra_id=all

LINK Switch111<->Core01_PORT_1
        NODES Switch111 Core01
        BANDWIDTH 1000M
        VIA 543 360
        TARGET /www/cacti/rra/gl_ls4111_traffic_in_87.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=84&rra_id=1&view_type=tree&graph
_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=84&rra_id=all

LINK Switch111<->Core01_PORT_2
        NODES Switch111 Core01
        BANDWIDTH 1000M
        VIA 543 388
        TARGET /www/cacti/rra/gl_ls4111_traffic_in_88.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=85&rra_id=1&view_type=tree&graph
_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=85&rra_id=all
```

3.4 – Como configurar os parâmetros TARGET, OVERLIBGRAPH e INFOURL:

Utilizando o mapa especificado no exemplo, vamos supor que se queira mostrar o gráfico da porta do Core02, onde está ligado o servidor Server114, quando for passado o mouse sobre o link entre esses dois dispositivos, como no exemplo da figura 41:

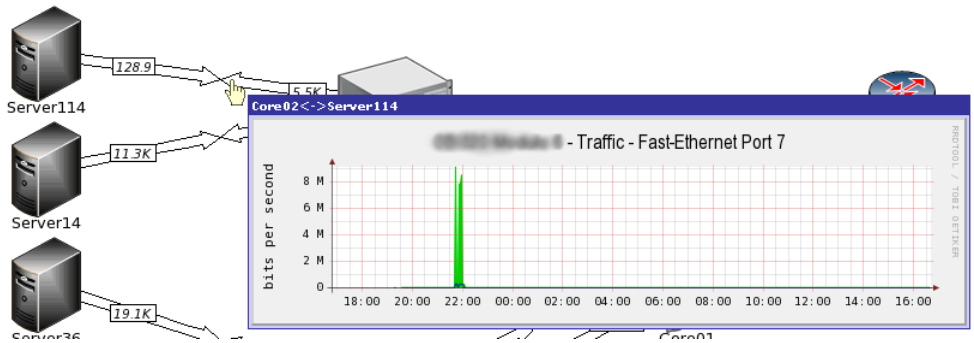

Figura 41 – Exemplo de link com gráfico.

3.5 – Vá até a opção **Graphs** do Cacti e visualize os gráficos do Core02. Clique sobre o gráfico. Uma página será carregada com o gráfico diário, semanal, mensal e anual em relação a esta porta (figura 42). A url mostrada no browser será o valor para o parâmetro **INFOURL**:

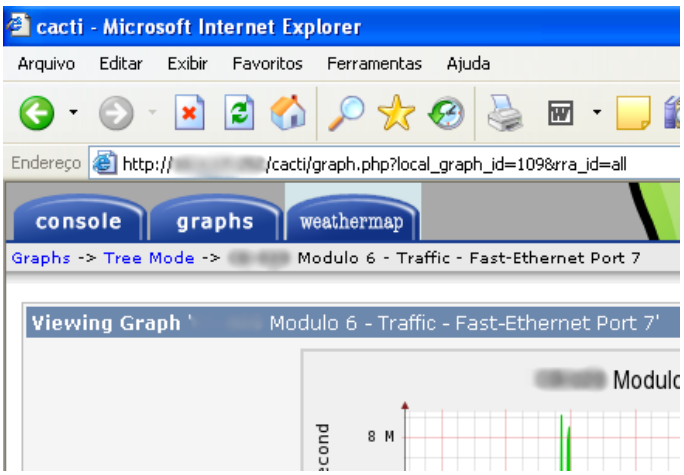

Figura 42 – Url utilizada como parâmetro de INFOURL.

3.6 – A url do primeiro gráfico, será o valor do parâmetro **OVERLIBGRAPH**. Para conseguir essa url, clique com o botão direito sobre o gráfico e depois em Propriedades. Copie todo o endereço em Endereço (URL), visto em destaque na figura 43.

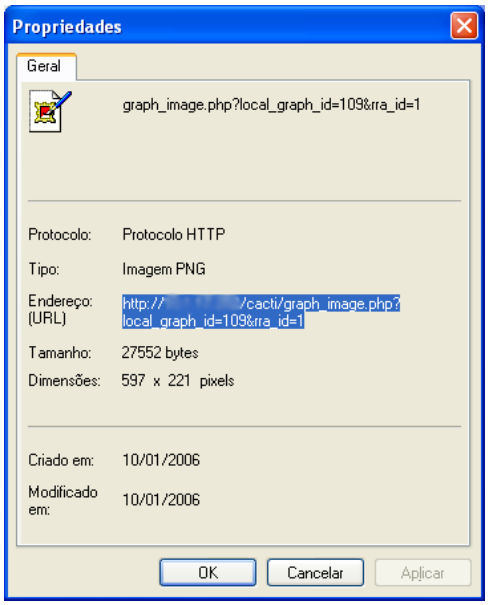

Figura 43 – Url utilizada como parâmetro de OVERLIBGRAPH.

3.7 – Para obter o valor do parâmetro **TARGET**, clique na chave que aparece no canto superior esquerdo do gráfico, figura 44:

Figura 44 – Ícone que informa parâmetros do gráfico.

3.8 – A página seguinte mostrará as propriedades do gráfico, como em um modo de depuração do gráfico. Entre elas uma das linhas mostradas é semelhante à linha abaixo:

```
DEF:a="/www/cacti/rra/cb020_modulo_6_traffic_in_112.rrd":traffic_in:AVERAGE \
```

3.9 – Esse é o caminho do arquivo de dados que o Cacti utiliza para montar os gráficos e também será o valor do parâmetro **TARGET**.

A definição do link então ficaria da seguinte forma:

```
LINK Core02<->Server114
        NODES Core02 Server114
```

```
        BANDWIDTH 100M
        TARGET /www/cacti/rra/cb020_modulo_6_traffic_in_112.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=109&rra_id=1
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=109&rra_id=all
```

3.10 – Para retirar a legenda e reduzir o tamanho do gráfico no popup, coloque o parâmetro **`&view_type=tree&graph_nolegend=true&graph_height=100&graph_width=500`** no final da linha do OVERLIBGRAPH.

O resultado da definição é concluído assim:

```
LINK Core02<->Server114
        NODES Core02 Server114
        BANDWIDTH 100M
        TARGET /www/cacti/rra/cb020_modulo_6_traffic_in_112.rrd
        OVERLIBGRAPH
http://<ip_server_cacti>/cacti/graph_image.php?local_graph_id=109&rra_id=1&view_type=tree&grap
h_nolegend=true&graph_height=100&graph_width=500
        INFOURL http://<ip_server_cacti>/cacti/graph.php?local_graph_id=109&rra_id=all
```

3.11 – Observações:

O parâmetro **KEYPOS**, que não foi mostrado no exemplo, cria uma legenda um pouco diferente e em inglês automaticamente. Porém, optou-se por criar a legenda como parte da imagem de fundo.

Na documentação do PHP Weathermap, existem parâmetros com efeitos adicionais e diferentes daqueles mostrados aqui e que podem facilitar o trabalho do usuário. A documentação recomendada está disponível em http://wotsit.thingy.com/haj/cacti/php-weathermap/manual/0.81/pages/config-reference.html.

B.4 **Backup da base de dados do Cacti**

4.1 – É mostrado aqui, um exemplo de procedimento para backup do diretório e do banco de dados do Cacti. Neste exemplo é criado um script que diariamente é executado através do crontab, e que copia esses dados automaticamente para outro servidor.

```
-- arquivo /home/cacti/backup-cacti --
#! /bin/sh
echo ..........................................
echo Iniciando backup da base de dados e do Cacti ..
echo ..........................................
echo

## Preparando os arquivos de backup ##
## Os arquivos terao a data e hora do backup no nome.
cd /www/cacti
NARQUIVO="backup-cacti-`date +%d-%m-%Y`"
NARQUIVOSQL="backup-cacti-DB-`date +%d-%m-%Y`"

## Export da base de dados do Cacti ##
mysqldump cacti > /home/cacti/$NARQUIVOSQL.sql
echo
echo Backup da base de dados realizada!
echo
sleep 1

## Compactando o diretório de trabalho do Cacti
tar cvf /home/cacti/$NARQUIVO.tar /www/cacti/.
gzip /home/cacti/$NARQUIVO.tar
echo

## Copiando os arquivos criados em /home/cacti para um servidor FTP
echo Copiando backup no Servidor Remoto...
echo $NARQUIVOSQL.sql
echo $NARQUIVO.tar.gz
echo
cd /home/cacti

## Substitua ftp_server pelo IP do servidor FTP
ftp -in ftp_server <<FIM user cacti cacti123 ## -> aqui, um usuário e senha válidos no
servidor FTP
bin
hash
put $NARQUIVOSQL.sql
put $NARQUIVO.tar.gz
bye
echo
echo Backup do diretorio Cacti realizado!
echo
echo Fim!
echo
```

4.2 – Além do script, crie uma entrada em ***/etc/crontab***. Na entrada abaixo, o procedimento de backup será realizado diariamente, às 6 horas:

```
0 6 * * * root /home/cacti/backup-cacti > /dev/null 2>&1
```

4.3 – Não se esqueça de tornar o arquivo com o script executável:

```
chmod 774 /home/cacti/backup-cacti
```

B.5 **Restauração da base de dados do Cacti**

5.1 – O procedimento de restauração do banco de dados do Cacti auxilia muito na tarefa de administrar a estação em caso de problemas com hardware ou outros em que a reinstalação de todo o sistema operacional seja inevitável. Para tal, é preciso que se tenha um backup do diretório de trabalho do Cacti e da base de dados mais recente, o arquivo .sql. Ver o procedimento de backup do Cacti descrito na seção anterior.

5.2 – Depois de executar todos os procedimentos de instalação do Cacti, não é necessário fazer nenhuma configuração na interface Web da ferramenta, apenas esteja certo de que todas as ferramentas (servidor Web, Mysql etc) e bibliotecas necessárias foram corretamente instaladas e que o Cacti esteja funcionando.

5.3 – Se foram seguidas as instruções de backup da seção anterior, é preciso ter disponível os dois arquivos criados. Um arquivo contendo diretório de trabalho do Cacti compactado (arquivo.tar.gz) e o arquivo de banco de dados salvo (arquivo.sql).

5.4 – Copie esses arquivos para o diretório **/home/cacti**. Dentro desse diretório, descompacte o arquivo tar.gz:

```
tar xvzf <arquivo>.tar.gz
```

5.5 – Provavelmente foi criado um diretório www/cacti. Copie o diretório cacti recém descompactado para o diretório original da ferramenta.

```
cp –R www/cacti /www
```

5.6 – Acesse o diretório /www/cacti

5.7 – Na reinstalação do Cacti, foi criado o banco de dados original, execute o comando abaixo para apagar esse banco:

```
# mysqladmin --user=root drop cacti
```

5.8 – Execute os comandos a seguir para recriar o banco com a base de dados recuperada. Esteja certo que o arquivo **.sql** que foi salvo no procedimento de backup esteja no diretório /home/cacti:

```
# mysqladmin --user=root create cacti
# mysql cacti < /home/cacti/arquivo.sql
mysql --user=root mysql
mysql> grant all on cacti.* to cacti@localhost identified by '123456';
mysql> flush privileges;
mysql> exit;
```

5.9 – Feche o browser e carregue novamente a página inicial do Cacti. Se tudo funcionou como deveria, a ferramenta será carregada com todas as configurações, dados, gráficos e mapas salvos antes da reinstalação do sistema.

B.6 **Atualizando o RRDTool para a versão 1.2.14**

6.1 – Talvez seja interessante atualizar a versão do RRDTool para a mais recente, permitindo que os gráficos gerados tenham uma aparência mais profissional sem os efeitos de serrilhado nos caracteres e nos gráficos (figura 45), possibilitando que fontes sejam utilizadas ao gosto do usuário (figura 46).

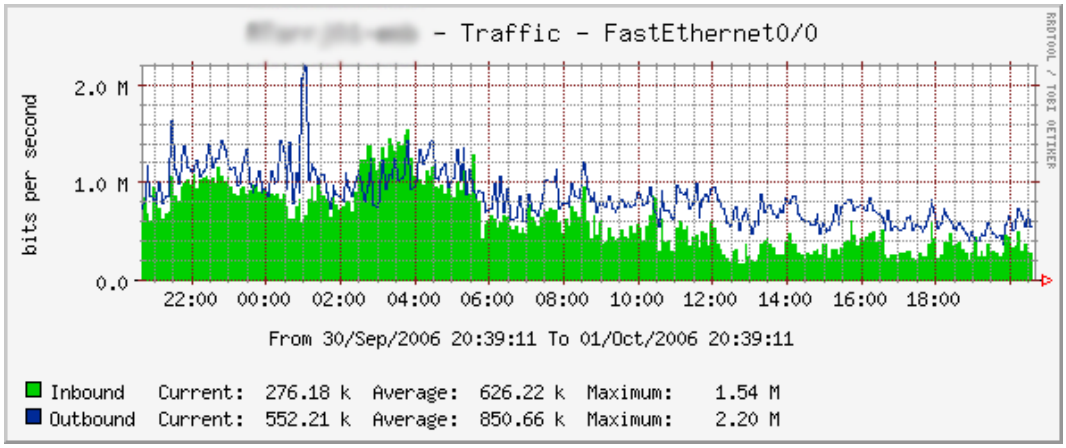

Figura 45 – Gráfico gerado com a versão 1.0.x do RRDTool.

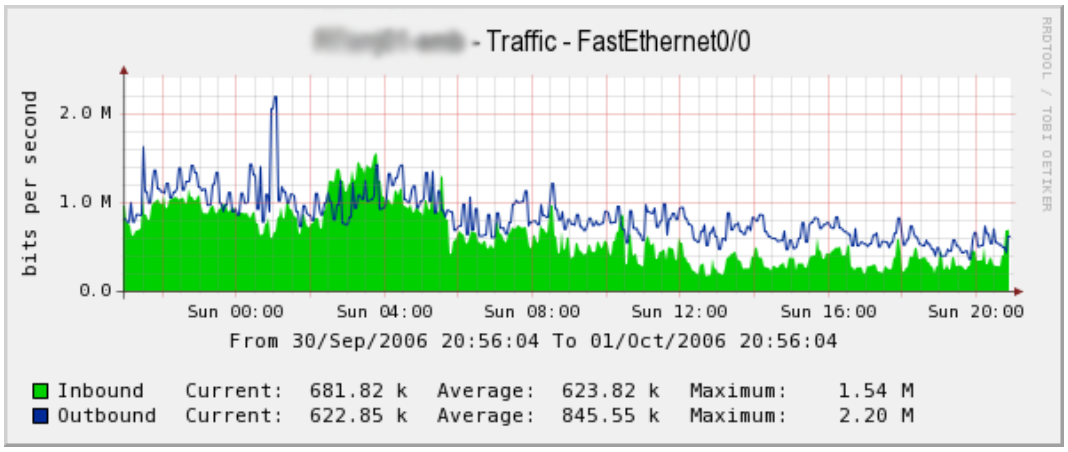

Figura 46 – Gráfico gerado com a versão 1.2.14 do RRDTool.

6.2 - Os procedimentos a seguir foram retirados do tutorial disponível no site oficial do RRDTool em http://oss.oetiker.ch/rrdtool/doc/rrdbuild.en.html

6.3 – Crie o diretório para compilar o rrdtool:

```
# mkdir /tmp/rrdbuild
```

6.4 – Execute os comandos abaixo. O diretório criado é referente à versão utilizada no momento:

```
# BUILD_DIR=/tmp/rrdbuild
# INSTALL_DIR=/usr/local/rrdtool-1.2.14

# IR=-I$BUILD_DIR/lb/include
# CPPFLAGS="$IR $IR/libart-2.0 $IR/freetype2 $IR/libpng"
# LDFLAGS="-L$BUILD_DIR/lb/lib"
# CFLAGS=-O3
# export CPPFLAGS LDFLAGS CFLAGS
```

6.5 – No diretório /downloads, copie a versão mais recente para o diretório utilizando a variável criada:

```
# cp rrdtool-1.2.14.tar.gz $BUILD_DIR
```

6.6 – Execute os comandos para compilar o rrdtool:

```
# cd $BUILD_DIR
# tar xvzf rrdtool-1.2.14.tar.gz
# cd rrdtool-1.2.14
# ./configure --prefix=$INSTALL_DIR && make && make install
```

6.7 – Se alguma mensagem de erro aparecer no final da compilação informando que alguma das bibliotecas está faltando, recompile as bibliotecas necessárias com alguns parâmetros mostrados logo abaixo. Antes, acesse o diretório onde deverão estar os pacotes: /downloads. Aqui, presuma-se que os pacotes já foram descompactados e que exista um diretório para cada uma das bibliotecas:

```
# cd zlib-1.2.3
# env CFLAGS="-O3 -fPIC" ./configure --prefix=$BUILD_DIR/lb
# make
# make install
# cd ..

# cd libpng-1.2.10
# env CPPFLAGS="-I$BUILD_DIR/lb/include" LDFLAGS="-L$BUILD_DIR/lb/lib" CFLAGS="-O3 -
fPIC" \
    ./configure --disable-shared --prefix=$BUILD_DIR/lb
```

```
# make
# make install
# cd ..

# cd freetype-2.1.10
# env CPPFLAGS="-I$BUILD_DIR/lb/include" LDFLAGS="-L$BUILD_DIR/lb/lib" CFLAGS="-O3 -
fPIC" \
    ./configure --disable-shared --prefix=$BUILD_DIR/lb
# make
# make install
# cd ..

# cd libart_lgpl-2.3.17
# env CFLAGS="-O3 -fPIC" ./configure --disable-shared --prefix=$BUILD_DIR/lb
# make
# make install
# cd ..
```

6.8 – Tente recompilar novamente o rrdtool:

```
cd $BUILD_DIR
tar xvzf rrdtool-1.2.14.tar.gz
cd rrdtool-1.2.14
./configure --prefix=$INSTALL_DIR && make && make install
```

6.9 – Acesse o Cacti como usuário admin e clique na aba Console. Selecione a opção de menu Settings e clique na aba menor à direita General. Mude a opção RRDTool Utility Version para 1.2.x e clique em Save.

6.10 – Clique na aba Paths e preencha os caminhos das seguintes opções, como mostrado na tabela abaixo:

Tabela 9 – Configurações da aba Paths.

RRDTool Binary Path	/usr/local/rrdtool-1.2.14/bin/rrdtool
RRDTool Default Font Path	/usr/local/rrdtool-1.2.14/share/rrdtool/fonts/DejaVuSansMono-Roman.ttf

6.11 – Clique em Save e verifique a aparência dos gráficos. Caso algum problema ocorra, basta voltar com a versão do rrdtool na aba General para 1.0.x e na aba Paths, colocar em RRDTool Binary Path o caminho do rrdtool original: **/usr/bin/rrdtool**. Não se esqueça de clicar em Save a cada operação realizada.

6.12 – Para detectar possíveis problemas com a nova versão do rrdtool, clique na opção de menu ***Graph Managment***. A lista de gráficos disponíveis será mostrada a direita. Clique em qualquer gráfico. Na página com as configurações do gráfico

clique na opção no canto superior direito ***Turn On Graph Debug Mode***. Logo abaixo serão mostrados os comandos utilizados pelo rrdtool para criar o gráfico e se algum problema for detectado, ele será informado logo abaixo da linha ***RRDTool Says***. Se o gráfico for gerado sem problemas, a mensagem ***OK*** será exibida.

6.13 – É possível também mudar as fontes dos títulos dos gráficos. Na aba Visual, coloque o caminho do arquivo de fonte escolhido na opção ***Title Font File***. Recomenda-se não mudar a fonte das demais opções, pois outros tipos de fontes diferentes da fonte padrão localizada em **/usr/local/rrdtool-1.2.14/share/rrdtool/fonts** podem fazer com que as informações nos gráficos (legenda, dados de eixos X e Y etc) fiquem desalinhadas.

ANEXO C – INSTALAÇÃO DO ZABBIX NO DEBIAN GNU/LINUX 31R2

C.1 **Instalação e configuração inicial**

1.1 – Crie uma estrutura de diretórios semelhante a esta para auxiliar na instalação e configuração das ferramentas.

```
zabbix-1.1.2.tar.gz
```

1.2 – Crie um usuário chamado zabbix:

```
# adduser --system --group zabbix
Adding system user `zabbix'...
Adding new group `zabbix' (112).
Adding new user `zabbix' (112) with group `zabbix'.
Creating home directory `/home/zabbix'.

# passwd zabbix
Enter new UNIX password:123456
Retype new UNIX password:123456
passwd: password updated successfully
```

1.3 – Instale os headers files do MySQL e snmp e o fping. Estes arquivos não são instalados no comando apt-get install snmp mysql-server, porém são necessários para o Zabbix:

```
# apt-get install libmysqlclient10-dev libsnmp5-dev fping
```

1.4 – Acesse o diretório /downloads, descompacte o pacote do Zabbix e mova os arquivos para o diretório home do usuário zabbix. Não se esqueça de acessar esse diretório:

```
# tar xzvf zabbix-1.1.2.tar.gz
# mv zabbix-1.1.2 /home/zabbix/zabbix-1.1.2
# cd  /home/zabbix/zabbix-1.1.2
```

1.5 – Crie o banco de dados:

```
# mysql
mysql> create database zabbix;
```

```
mysql> grant all on zabbix.* to zabbix@localhost identified by
'123456';
mysql> flush privileges;
mysql> quit;
```

1.6 – Preencha o banco de dados com os *schemas* fornecidos pelo Zabbix. Execute os comandos em **/home/zabbix/zabbix-1.1.2**:

```
# cd create/mysql/
# cat schema.sql |mysql zabbix
# cd ../data/
# cat data.sql |mysql zabbix
# cd ..
```

1.7 – Ainda no diretório **/home/zabbix/zabbix-1.1.2**, compile e instale o Zabbix:

```
# ./configure  --enable-server --enable-agent --with-mysql --with-net-
snmp
# make
# make install
```

1.8 - Edite o arquivo **/etc/services** e inclua as seguintes linhas:

```
zabbix_agent 10050/tcp
zabbix_trap 10051/tcp
```

1.9 – Crie o diretório **/etc/zabbix** e copie os arquivos de configuração do agente e do servidor zabbix para /etc/zabbix. Obs. Para as estações Linux/Unix que serão monitoradas, somente copie o arquivo **zabbix_agentd.conf**. O procedimento abaixo será utilizado no servidor que também será monitorado. Para estações W2000, WinXP e W2003, ver mais adiante:

```
# cp /home/zabbix/zabbix-1.1.2/misc/conf/zabbix_agentd.conf etc/zabbix
# cp /home/zabbix/zabbix-1.1.2/misc/conf/zabbix_server.conf etc/zabbix
```

1.10 – Edite esses arquivos e faça as mudanças básicas necessárias: Abaixo estão as linhas que inicialmente serão modificadas:

zabbix_agentd.conf:

```
...
Server=<ip_do_servidor_zabbix>
...
Hostname=<nome_da_estação_local>
```

```
...
```

zabbix_server.conf:

```
...
DBUser=zabbix
DBPassword=123456
...
```

1.11 – Edite o arquivo **frontends/php/include/db.inc.php** e deixe as linhas do arquivo como mostradas abaixo.

```
// DATABASE CONFIGURATION

//      $DB_TYPE        ="ORACLE";
//      $DB_TYPE        ="POSTGRESQL";
        $DB_TYPE        ="MYSQL";
        $DB_SERVER      ="localhost";
        $DB_DATABASE    ="zabbix";
        $DB_USER        ="zabbix";
        $DB_PASSWORD    ="123456";
```

1.12 – Crie o diretório onde ficarão os arquivos da interface Web do Zabbix:

```
# mkdir /www/zabbix
```

1.13 – Copie os arquivos para esse diretório:

```
# cp -R frontends/php/* /www/zabbix/
```

1.14 – Crie um link no **/etc/rc2.d** para carregar o servidor Zabbix no boot (somente para o servidor). No Debian, os scripts são carregados a partir do diretório /etc/rc2.d, caso no arquivo **/etc/inittab** esteja definido o parâmetro **id:2:initdefault:**

```
# ln -s /usr/local/bin/zabbix_server /etc/rc2.d/S98zabbix_server
```

1.15 – Carregue o servidor Zabbix pela primeira vez:

```
# /usr/local/bin/zabbix_server
```

C.2 **Instalando o agente do Zabbix em estações Linux/Unix**

2.1 – Crie um link no **/etc/rc2.d** para carregar o agente a cada boot (para todas as estações Linux/Unix monitoradas):

```
# ln –s /usr/local/bin/zabbix_agentd /etc/rc2.d/S98zabbix_agentd
```

2.2 – Carregue o agente pela primeira vez:

```
# /usr/local/bin/zabbix_agentd
```

C.3 **Instalando o agente do Zabbix em estações Windows**

3.1 – Copie o arquivo **/usr/local/bin/ZabbixW32.exe** (Windows 32 bits) ou **ZabbixW64.exe** (Windows 64 bits) e o arquivo **/etc/zabbix/zabbix_agentd.conf**, para uma estação Windows. De preferência, crie uma pasta no disco C com o nome c:\zabbix\ e copie os arquivos para essa pasta.

3.2 - No arquivo **zabbix_agentd.conf**, edite a seção Log file para o formato legível para o Windows:

Linha original:

```
LogFile=/tmp/zabbix_agentd.log
```

Linha após edição:

```
LogFile=c:\zabbix\zabbix_agentd.log
```

3.3 – Abra um prompt do DOS e execute o seguinte comando para instalar o agente como um serviço do Windows:

```
c:\zabbix\ZabbixW32.exe --config c:\zabbix\zabbix_agentd.conf install
```

3.4 – Carregue o agente:

```
c:\zabbix\ZabbixW32.exe start
```

C.4 **Backup da base de dados do Zabbix**

4.1 – É mostrado aqui, um exemplo de procedimento para backup do diretório e do banco de dados do Zabbix e dos arquivos principais. Neste exemplo é criado um script que é executado semanalmente através do crontab, e que copia esses dados automaticamente para outro servidor. Antes, crie um diretório dentro de /home/zabbix, com nome de **backup-zabbix-dir**, e dentro deste diretório crie outros dois, **usr/local/bin** e **etc/zabbix**, ficando dessa forma a estrutura de diretórios:

/home/zabbix/backup-zabbix-dir/usr/local/bin
/home/zabbix/backup-zabbix-dir/etc/zabbix

```
-- arquivo /home/zabbix/backup-zabbix --

#! /bin/sh
echo ..........................................
echo Iniciando backup da base de dados e do Zabbix ..
echo ..........................................
echo

cd /www/zabbix

## Preparando os arquivos de backup ##
## Os arquivos terao a data e hora do backup no nome.
NARQUIVO="backup-zabbix-`date +%d-%m-%Y`"
NARQUIVOSQL="backup-zabbix-DB-`date +%d-%m-%Y`"

## Export da base de dados do Zabbix ##
mysqldump zabbix > /home/zabbix/$NARQUIVOSQL.sql

echo
echo Backup da base de dados feita!
echo
sleep 1

## Copiando os arquivos de configuração e principais binários
## para diretórios dentro de /home/zabbix
cp /usr/local/bin/zabbix* /home/zabbix/backup-zabbix-dir/usr/local/bin
cp /etc/zabbix/* /home/zabbix/backup-zabbix-dir/etc/zabbix

## Empacotando e compactando todos os arquivos
tar cvf /home/zabbix/$NARQUIVO.tar /home/zabbix/backup-zabbix-dir/.
gzip /home/zabbix/$NARQUIVO.tar
echo

## Copiando os arquivos criados em /home/zabbix para um servidor FTP
echo Copiando backup no Servidor Remoto...
echo $NARQUIVOSQL.sql...
echo $NARQUIVO.tar.gz
echo
cd /home/zabbix

## Substitua ftp_server pelo IP do servidor FTP
```

```
ftp -in ftp_server <<FIM
user zabbix zabbix123
bin
hash
put $NARQUIVOSQL.sql
put $NARQUIVO.tar.gz
bye
echo
echo Backup do diretorio Zabbix feito!
echo
echo Fim!
Echo
-- fim do arquivo --
```

C.5 **Restauração da base de dados do Zabbix**

5.1 – O procedimento de restauração do banco de dados do Zabbix auxilia muito na tarefa de administrar a estação de gerenciamento em caso de problemas com hardware ou outros em que a reinstalação de todo o sistema operacional seja inevitável. Para tal, é preciso que se tenha um backup dos arquivos de trabalho do Zabbix e da base de dados mais recente, o arquivo **.sql**. Ver o procedimento de backup do Zabbix descrito na seção anterior.

5.2 – Depois de executar todos os procedimentos de instalação do Zabbix, não é necessário fazer nenhuma configuração na interface Web da ferramenta, apenas esteja certo de que todas as ferramentas (servidor Web, Mysql etc) e bibliotecas necessárias foram corretamente instaladas e que o Zabbix esteja funcionando.

5.3 – Se foram seguidas as instruções de backup da seção anterior, é preciso ter disponível os dois arquivos criados. Um arquivo contendo os arquivos de trabalho do Zabbix compactado (arquivo.tar.gz) e o arquivo de banco de dados salvo (arquivo.sql).

5.4 – Copie esses arquivos para o diretório /home/zabbix. Dentro desse diretório, descompacte o arquivo tar.gz:

```
tar xvzf <arquivo>.tar.gz –C /
```

5.5 – Esta operação irá criar um diretório chamado backup-zabbix-dir. Dentro dele também deverão estar os diretórios usr e etc.

5.6 – Copie os arquivos de trabalho para os locais corretos:

```
cp –avfR backup-zabbix-dir/etc /
cp –avfR backup-zabbix-dir/usr /
```

5.7 – Na reinstalação do Zabbix, foi criado o banco de dados original, execute o comando abaixo para apagar esse banco:

```
# mysqladmin --user=root drop zabbix
```

5.8 - Execute os comandos a seguir para recriar o banco com a base de dados recuperada. Esteja certo que o arquivo .sql que foi salvo no procedimento de backup esteja no diretório /home/zabbix:

```
# mysqladmin --user=root create zabbix
# mysql zabbix < /home/zabbix/arquivo.sql
mysql --user=root mysql
mysql> grant all on zabbix.* to zabbix@localhost identified by
‘123456’;
mysql> flush privileges;
mysql> exit;
```

5.9 – Feche o browser e carregue novamente a página inicial do Zabbix. Se tudo funcionou como deveria, a ferramenta será carregada com todas as configurações, dados, gráficos e mapas salvos antes da reinstalação do sistema.