NAGIOS

COMO SOLUÇÃO DE MONITORAMENTO DE REDE

HETTY ALVES DE ANDRADE

2006

HETTY ALVES DE ANDRADE

NAGIOS

COMO SOLUÇÃO DE MONITORAMENTO DE REDE

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras, como parte das exigências do curso de Pós-Graduação *Lato Sensu* em Administração de Redes Linux, para a obtenção do título de especialista em Administração de Redes Linux.

Orientador:

Prof.MSc. Joaquim Quinteiro.

LAVRAS
MINAS GERAIS - BRASIL
2006

HETTY ALVES DE ANDRADE

NAGIOS

COMO SOLUÇÃO DE MONITORAMENTO DE REDE

Monografia submetida à Comissão Examinadora designada pela Universidade Federal de Lavras para obtenção do título de Especialista em Administração de Redes Linux.

Aprovada em 29 de Setembro de 2006

_____

Prof. MSc. Herlon Ayres Camargo

_____

Prof. MSc. Denilson Vedoveto Martins

_____

Prof. MSc. Joaquim Quinteiro

(Orientador)

LAVRAS
MINAS GERAIS - BRASIL

AGRADECIMENTOS

A Deus

À minha família, pelo carinho e pela paciência que tiveram durante a execução deste trabalho.

Aos amigos e parentes, professores e todos que de alguma forma contribuíram para que este trabalho fosse concluído.

RESUMO

Este trabalho desenvolve um estudo do *software* livre Nagios, aplicativo que essencialmente monitora ativos e serviços de rede. Serão demonstrados seus recursos de forma prática, buscando auxiliar o administrador de rede no processo de configuração para a utilização desta ferramenta. Algumas das características do Nagios são: o monitoramento de serviços de rede *SMTP, POP3, HTTP, NNTP,* entre outros; monitoramento de recursos de servidores como *CPU*, memória, disco, e processos; capacidade de definir hierarquia da rede; notificações imediatas sobre problemas na rede, via *e-mail* e *pager*; capacidade de tomar contramedidas de acordo com o problema na rede; interface *web* com mapa da rede em 2D e 3D, relatórios, gráficos e históricos. É versátil, flexível e verifica constantemente a disponibilidade dos serviços e *hosts*.

Palavras-Chave: *plugin*, *software*, *hosts*, monitoramento, falhas de rede.

SUMÁRIO

LISTA DE FIGURAS

1 INTRODUÇÃO

Originalmente escrito sob o nome *Netsaint*, o Nagios[1] foi criado e ainda é mantido por Ethan Galstad e sua equipe de mais de 150 desenvolvedores espalhados por todo o mundo, dedicados a desenvolver *plugins*, corrigir *bugs*, desenvolver uma interface *web,* produzir e traduzir a vasta documentação, entre outras atividades. Este *software* de monitoramento de redes é distribuído livremente, através da lei de *copyleft GPL*. A habilidade em administrar ambientes com infra-estrutura de *WAN*, *LAN* e *MAN*, e a interface gráfica – *GUI* utilizada lhe garantem desempenho comparável a sistemas comerciais existentes, como WhatsUp e BigBrother, assim como o Angel Network Monitor, o PIKT, o Autostatus e outros[2].

Apesar de ser projetado para redes de grande porte, seu desempenho em pequenos ambientes é excelente. Isso se comprova seja alertando para a queda de serviços ou *hosts* vigiados nos arquivos de configuração, seja monitorando equipamentos com suporte a protocolos *SNMP*, este o principal agente de troca de informações entre o Nagios e seus *hosts*.

A eficácia do Nagios no monitoramento de uma rede depende de sua expansão através de *plugins*, complementos escritos em *CGI – Common Gateway Interface* – ou em qualquer outra linguagem interpretável, podendo ser desenvolvidos por diferentes programadores. Como complemento ao Nagios, o sítio *www.nagios.org* disponibiliza uma série de *plugins* oficiais. Apesar de ter sido desenvolvido originariamente para executar em qualquer plataforma *Linux*, este *software* trabalha também em variantes do *UNIX*, como *FreeBSD*, *OpenBSD* e *NetBSD*.

1 Em *www.nagios.org*.
2 Respectivamente, em www.ipswitch.com, *www.bb4.org, www.paganini.net*, *www.pikt.org e www.angio.net.*

Ao localizar um problema num *host* monitorado, através de *plugins* externos vigiados pelo *daemon*, o Nagios pode notificar ao administrador ou aos seus contatos determinados através de *e-mails*, mensagens instantâneas via celular ou *pager*, *SMS* ou outras alternativas que forem desenvolvidas. Este sistema de gestão pode também informar *status*, histórico de *logs*, e permitir que se definam previamente os usuários que terão acesso visual ao trabalho executado, via *web*. Além destes recursos e ferramentas, o Nagios disponibiliza também:

- Monitoramento de serviços de rede, como *HTTP, POP3, NNTP*, *SMTP*, *SSH, Telnet,* etc;
- Monitoramento dos recursos dos servidores (espaço em disco, utilização de memória, carga de processamento, etc.);
- Notificação de falhas, através de vários sistemas de comunicação, em tempo real;
- Interface *web*, que permite acompanhar o monitoramento e identificar mais facilmente os problemas da rede;
- Uso de tratadores de eventos para corrigir automaticamente um problema (por exemplo, reiniciar um servidor *web* que parou de responder);
- Facilidade em desenvolver *plugins* específicos, mesmo para verificadores de serviços em paralelo;
- Definição de hierarquia entre *hosts* de redes e de eventos a serem executados para solução pró-ativa de problemas;
- Rotatividade automática de *logs*.

Constituído por um módulo central que possibilita a adição de novas funcionalidades através de *plugins* (escritos em *C, Perl ou Shell)* para efetuarem a monitoração. O *software* é usado para acompanhamento em servidores, conferência de serviços e desempenho. Através da interface de gerência do

Nagios, pode-se acompanhar algumas opções como o estado do *link*, a quantidade de perda de pacotes, a latência, o índice de disponibilidade do *backbone*, dentre outros.

O presente trabalho irá fazer uma apresentação do *software* livre Nagios, aplicativo administrador de ativos e serviços de redes, bem com a análise de suas ferramentas e processos de configuração, com o objetivo de auxiliar na utilização de seus recursos. No Capítulo 2, será apresentada uma visão resumida da arquitetura do Nagios e as principais "habilidades" deste aplicativo: a utilização simplificada dos *plugins*, os diversos processos de monitoramento, a hierarquia interna da rede e as possíveis notificações de ocorrências, entre outros. No Capítulo 3 serão apresentados alguns dos *plugins* do Nagios e seu funcionamento, e no Capítulo 4 serão relatadas algumas aplicações específicas, como serviços, *softwares* de apoio, filtros e suportes para a base de dados. Por fim, no Capítulo 5 serão descritos os procedimentos de instalação e configuração deste aplicativo e no Capítulo 6 uma conclusão do trabalho.

2 APRESENTAÇÃO DO NAGIOS

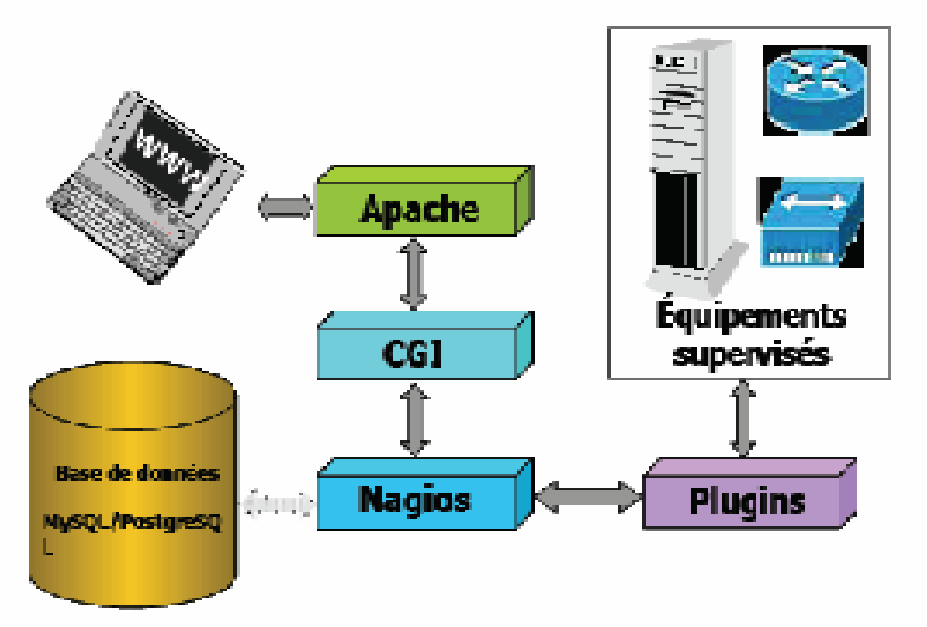

Figura 1 – Arquitetura do Nagios.
(*http://www.interlab.ait.ac.th)*

O Nagios foi construído em uma arquitetura[3] servidor/agentes e, usualmente em uma rede, executa em um servidor específico com seus *plugins* distribuídos nos servidores remotos que precisam ser monitorados, conforme Figura 1. Estes *plugins* enviam informações para o servidor onde se encontra o Nagios que então exibe-os em um *GUI (Graphical User Interface)*. Sua composição consiste de 3 partes:

- Um *scheduler* que é parte do servidor Nagios. Em intervalos regulares, ele verifica os *plugins* e de acordo com seus resultados executa ações;

- Um *GUI*. A interface do Nagios (com a configuração, os alertas, etc). Ele é exibido em páginas *web* geradas pelo *CGI* que podem ser botões de estado

3 Em http://*www.interlab.ait.ac.th*

(verde para normal, amarelo para situação de alerta e vermelho para erro), sons, gráficos *MRTG*, etc;

- Os *plugins*. São configurados pelo usuário e capazes de conferir um serviço e retornar um resultado para o Nagios.

Um estado *soft* é alcançado quando um *plugin* retorna um alerta ou um erro. Então no *GUI*, um botão verde torna-se vermelho e um som é emitido. Quando este estado *soft* é alcançado muitas vezes, o alerta torna-se *hard*, e o servidor Nagios envia as notificações pertinentes.

O objetivo da ferramenta é o de informar aos administradores rapidamente sobre condições questionáveis (*warning*) ou críticas (*critical*). O que é considerado "questionável" ou "crítico" é definido pelo administrador na configuração. Diferente das ferramentas de rede que mostram o tempo decorrido graficamente ou que registrem e meçam tráfego, o Nagios se utiliza de cores, como em um semáforo.

O Nagios diferencia entre verificações de servidores e serviços. A verificação de um servidor testa se um computador está alcançável, via de regra apenas um *ping* é utilizado. Esta é feita de forma irregular e apenas quando necessário. Seletivamente testa serviços de rede individuais tais como *HTTP*, *SMTP, DNS*, etc, mas também processos executando, carga de *CPU* ou arquivos de *log*. O teste mais simples para serviços de rede consiste em ver se a porta de destino está escutando, e se o serviço está ativo.

Um aspecto especialmente interessante do Nagios é o fato de poder considerar dependências na topologia de rede. Se o sistema de destino só pode ser alcançado por um roteador específico que acabou de cair, então o Nagios reporta que o sistema está inatingível, e não irá mais bombardeá-lo com novas verificações.

A ferramenta permite ao administrador poder detectar rapidamente a causa real do problema e corrigir a situação. Uma vantagem do Nagios reside em sua estrutura modular: o seu núcleo não contém um único teste. Ao contrário, ele usa programas externos, conhecidos como *plugins*, para verificações de serviços e servidores. O pacote básico já contém uma quantidade padrão de *plugins* para as aplicações mais conhecidas. Um *plugin* é um programa simples - normalmente apenas um *shell script* (*Bash, Perl*, etc) - que fornece uma das quatro possíveis condições: *ok, warning, critical, unknown*. Isto significa que, a princípio, ele pode testar quase tudo que possa ser medido ou contado eletronicamente: a temperatura e umidade na sala do servidor ou a presença de pessoas em determinada hora e lugar. Não existem limites, considerando que se possa encontrar um meio de prover dados ou eventos como informação para ser avaliada por computador.

O Nagios possui um sofisticado sistema de notificação. No lado do emissor (ou seja, com a verificação de servidor ou serviço) pode-se configurar quando cada grupo de pessoas - os conhecidos 'grupos de contato' - são informados sobre quais condições ou eventos (falhas, recuperação, advertências, etc). No lado do receptor pode-se também definir em múltiplos níveis o que deve ser feito com uma mensagem correspondente - por exemplo quando o sistema deve passá-la adiante, dependendo da hora do dia, ou descartar a mensagem.

Com sua interface *web,* ele provê ao administrador uma grande variedade de informações, claramente organizadas de acordo com os assuntos envolvidos. Fornece uma página de informação individualmente estruturada para praticamente todo propósito caso este necessite de um resumo de toda situação, uma visualização de serviços problemáticos e servidores que provoquem indisponibilidade da rede, ou a situação de todos os grupos de servidores ou serviços. Informação já obtida pode ser salva como comentário,

assim como paradas programadas: o Nagios ainda previne que falsos alarmes sejam emitidos nesses períodos.

O *software* livre Nagios é aplicável aos mais diferentes segmentos comerciais. Diversas empresas o utilizam não só para monitoramento de conectividade de usuários, mas também para controle de pontos de acesso de antenas *wireless* em provedores de *internet*, em servidores e estações de trabalho, monitoramento de clientes à distância e previsão de possíveis falhas nos sistemas. Enfim, cada empresa pode utilizar o Nagios de acordo com suas necessidades.

A checagem de serviços pode ser: indireta - usa um agente remoto para colher informações; passiva – os resultados são enviados ao Nagios através do uso de arquivo de comando externo; e paralela – todas as checagens entram em uma fila de evento. Através do *NSCA* (*Nagios Service Check Acceptor*) um computador remoto pode escrever o resultado da checagem passiva no arquivo de comando externo do servidor Nagios.

O Nagios também pode fazer uso de programas externos para notificações livremente configuráveis, para que se possa integrar qualquer sistema que se deseje: *e-mail*, SMS, servidor de recados que o administrador chama pelo telefone e recebe uma mensagem de voz referente ao erro. O contrário também é possível onde, através de uma interface separada, programas independentes podem enviar informação de estado e comandos para o Nagios. A interface *web* faz largo uso dessa possibilidade pois permite ao administrador o envio de comandos interativos.

Permite monitoração distribuída. Isto significa várias instalações descentralizadas, enviando os resultados de seus testes para uma instância central, que então ajuda a manter uma visão geral da situação a partir de um ponto único. Reduz a carga no servidor de monitoramento com envio de

resultados para o servidor central e uso de checagem passiva . Monitoramento redundante também é possível em um ambiente onde teriam dois ou mais Nagios monitorando os mesmos recursos, sendo que um envia notificações e o outro assume esta tarefa no caso de falha do primeiro. Ainda é possível monitorar *cluster* de máquinas ou serviços.

Pela revisão de eventos passados, a interface *web* pode revelar quais problemas ocorreram em um intervalo de tempo selecionado, quem foi informado, qual situação estava prejudicando a disponibilidade de um servidor e/ou serviços durante um período de tempo particular. A opção chamada de *state stalking* registra, em arquivo de *log,* alterações ocorridas na saída do *plugin* de checagem, mesmo que o estado do serviço não se altere.

Pode-se citar ainda: dados de performance - dados detalhados sobre a monitoração de um determinado serviço ou máquina; paradas agendadas; monitoramento adaptativo - mudar alguns parâmetros de monitoramento sem que seja necessário reiniciar o Nagios; herança de definições de objetos - reduzir o tempo de configuração do sistema e facilitar suas alterações; o estado *flapping* - quando um serviço muda freqüentemente de estado, evita avalanches de notificações e alertas; escalamento de notificações - permite criar hierarquia de notificações, todos os contatos inferiores recebem cópias das notificações enviadas aos superiores; tratadores de eventos - comandos opcionais executados quando há mudança no estado do serviço; *freshness* - certifica que resultados de checagens passivas estão sendo recebidos regularmente; dependências - notificações e execuções de checagens podem depender de algo para serem realizadas.

2.1 O Nagios no mundo

Conforme dados de 06/12/2004, apresentados em seu sítio oficial, o Nagios é utilizado por 1.112 empresas em todo o mundo (usuários registrados

apenas), num total de mais de 163 mil redes monitoradas. No Brasil existem 53 usuários, conforme apresentado na Figura 2.

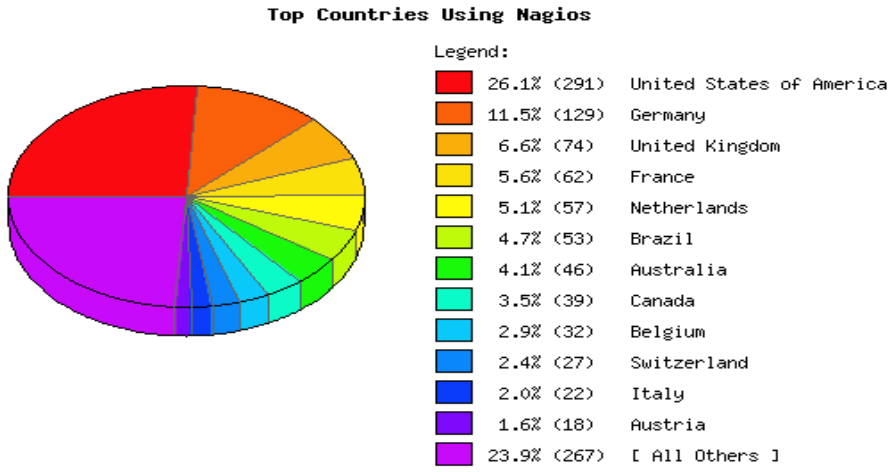

Figura 2 – Principais Países Usuários do Nagios.
(*http://www.nagios.org*)

Também é possível identificar uma maior utilização do Nagios por empresas provedoras de acesso à *internet*, com 18,0% dos usuários registrados, conforme a Figura 3.

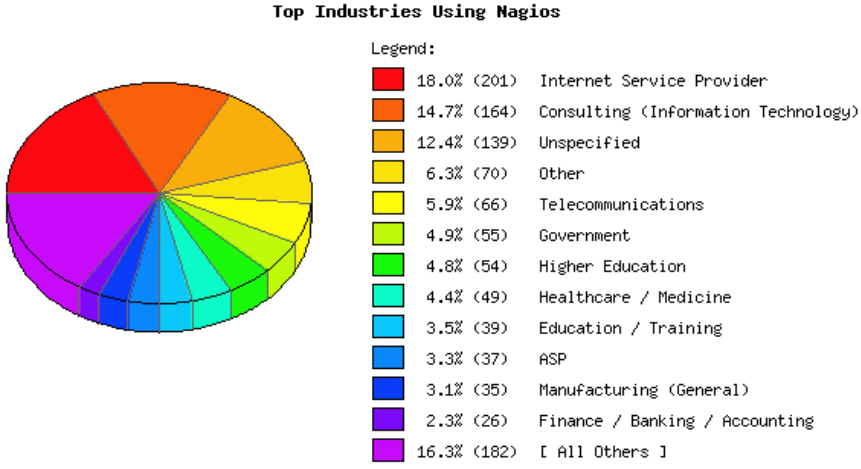

Figura 3 – Principais Áreas de Atuação das Empresas Usuárias do Nagios.
(*http://www.nagios.org*)

Confirmando a maturidade deste sistema, distribuições importantes do *Linux* como AltLinux, Debian, Gentoo, Mandrake (Mandriva), Sentinix e SkolLinux já incorporam em seu pacote o Nagios como aplicativo principal para a configuração de servidor de monitoramento. No Brasil, alguns dos usuários registrados são AT&T Latin America, Infolink Teleinformática Ltda, Banco Mercantil do Brasil Interneting, Camargo Corrêa S/A, ITI e Dataprev, PUC/PR, SEFAZ-MT (Secretaria da Fazenda do Estado de Mato Grosso) e Telemig Celular.

A preferência pela gestão com o Nagios vem aumentando a cada dia, e nele destacam-se as ferramentas de monitoramento de redes e serviços, organização de *plugins*, monitoramento dos recursos de clientes, monitoramento de fatores ambientais, notificação de resultados, definição de hierarquia de redes, entre outras. A seguir, serão descritas cada uma destas ferramentas.

2.2 Monitoramento de redes e serviços

O Nagios monitora, desde que definido pelo administrador da rede, serviços como *HTTP*, *SMTP*, *POP3* e *NNTP*. Esses serviços, em caso de imprevistos, precisam permanecer o menor tempo possível fora do ar, a fim de evitar o comprometimento de atividades essenciais à empresa. Desta forma, o Nagios permite o monitoramento da conectividade de maneira a perceber ou não a existência de um *host* ou serviço na rede.

Na Figura 4 está representado um modelo de tela do Nagios com serviços de rede a serem monitorados.

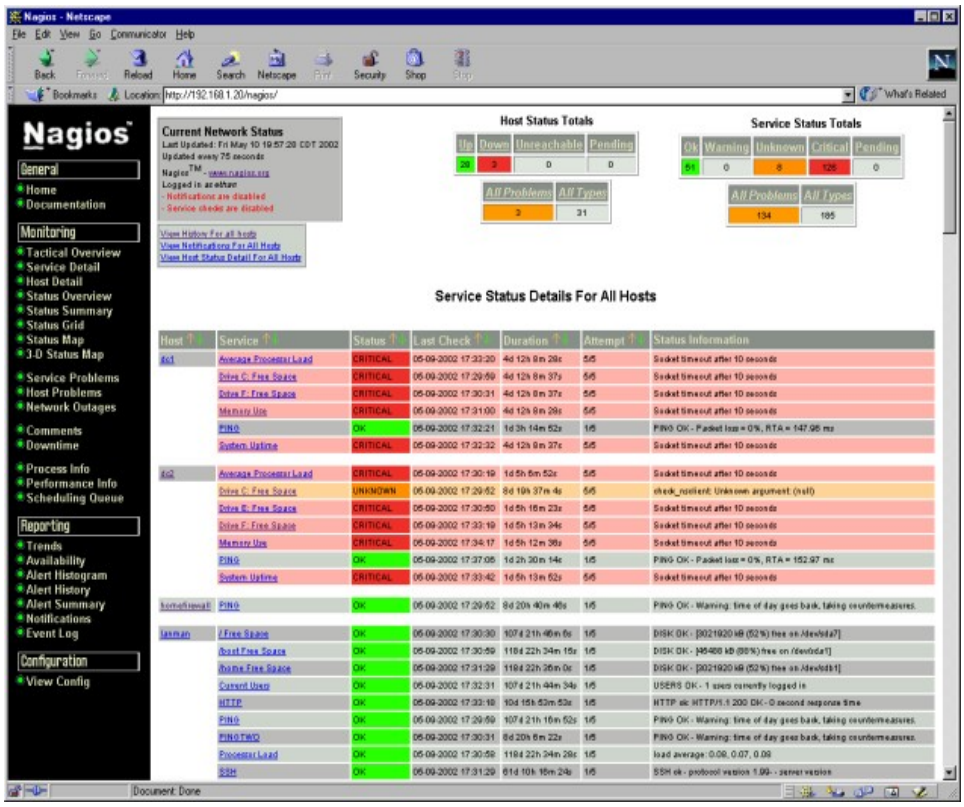

Figura 4– Tela Padrão de Detalhamentos de Serviços

(*http://freshmeat.net*)

2.3 Organização de *plugins* de forma simplificada

Os serviços de checagem no Nagios são exercidos por *plugins* no formato *CGI*, e podem ser desenvolvidos pelo próprio usuário. Estes *CGI*s são armazenados em uma única pasta e carregados pelo *browser*, quando solicitado.

2.4 Monitoramento dos recursos de clientes

Os computadores "clientes" podem ter seus *hardwares* monitorados plenamente, com o intuito de obter estatísticas em tempo real da utilização de *hosts* específicos e conseqüentemente balancear a carga entre servidor (ou servidores) e estações de trabalho. Dentre os recursos monitorados, pode-se

destacar: processos em execução, uso de disco rígido, carga de trabalho do processador e uso de memória *RAM*.

2.5 Monitoramento de fatores ambientais

O controle de temperatura ambiente também pode ser efetuado através do Nagios, mediante a aquisição do *Esensor*, aparelho disponível no sítio oficial do *software*, onde também é oferecida uma interface *web* interna que possibilita a alteração dos valores mínimos e máximos pré-estabelecidos. Este equipamento faz a leitura e repassa as informações ao aplicativo, para arquivo em *log,* e pode-se obter o resultado do *status* através de consultas diretas ao *host* responsável pelo monitoramento.

Alguns modelos do *Esensor* são diretamente acoplados ao *hub* ou ao *switch* e possuem IP próprio, o que permite que os dados sejam transmitidos pela rede interna de dados, e possibilitam o diagnóstico da temperatura ambiente, da iluminação e da umidade relativa do ar, por exemplo, no local onde se concentram os servidores de uma empresa. Como nos demais casos, ao localizar uma anomalia, o Nagios irá informar ao administrador através dos meios de comunicação determinados.

2.6 Notificação de resultados

O Nagios dá ao administrador a opção de programá-lo para, em caso de irregularidade (falhas) e dependendo do tipo desta ocorrência, informar a um ou mais grupos de contato cadastrados, seja através de *e-mail*, *SMS*, *pager* ou outros métodos definidos. Além disso, o Nagios pode ser programado para reagir e solucionar alguns dos eventuais problemas, obviamente informando novamente ao administrador da solução o *status* atual da rede.

Na Figura 5 está representado um dos meios que o Nagios se utiliza para envio de notificações.

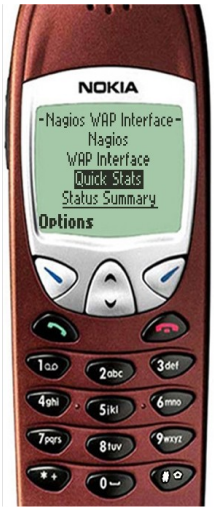

Figura 5 – Interface *WAP*
(*http://nagios.sourceforge.net*)

2.7 Definição de hierarquia de redes

É possível, através de um sistema de hierarquização, definir *hosts* pais e filhos dentro de uma rede e diferenciar clientes desativados de clientes inalcançáveis. A partir do servidor de monitoramento, pode-se construir uma árvore hierárquica onde o servidor fica no topo, enquanto que os *hosts* seguintes são posicionados ao longo das ramificações.

2.8 Outras ferramentas

O Nagios possui e permite a utilização de inúmeras outras ferramentas; porém, para algumas delas se faz necessário um trabalho prévio de autorização em operadoras de telefonia celular, serviços de *pager*, etc. Ainda é possível gerar gráficos 2D e 3D dos *hosts* monitorados e distinguí-los quanto ao sistema operacional usado e tipo de *host* (roteador, *proxy*, servidor, estação de trabalho, etc.).

3 INTERFACE *WEB* DO NAGIOS

A interface *web* de administração do Nagios é composta por um conjunto de *scripts CGI*. Provida dos direitos de acesso adequados, esta interface permite muito mais que uma simples observação das informações fornecidas pelo Nagios. Pode-se executar uma série de comandos e controles de forma ativa: desde modificar um comando único, alternar o estado de mensagem entre ligado ou desligado, até reiniciar o servidor. Será descrito aqui o conceito básico em que os programas *CGI* foram construídos, de forma a dar uma noção do grande alcance de opções disponíveis que eles possuem. A seguir uma visão geral de todos os programas *CGI* incluídos no pacote.

status.cgi – exibe as em várias formas de estado, sendo o programa *CGI* mais importante.

statusmap.cgi – representação da topologia dos *hosts* monitorados.

statuswrl.cgi – representação da topologia em formato 3D; requer um *browser* com capacidade *VRML* (*Virtual Reality Markup Language*) e permite navegação interativa em um espaço virtual.

statuswml.cgi – página de estado simples para dispositivos WAP (telefonia celular)

extinfo.cgi – informação adicional em um *host* ou serviço, com a possibilidade de executar comandos.

cmd.cgi – execução de comandos.

tac.cgi – visão geral de todos os serviços e *hosts* monitorados, a Visão Geral Tática.

outages.cgi – nós de rede que causam falhas de rede parciais.

avail.cgi – relatório de disponibilidade.

config.cgi – exibe as definições de objetos do Nagios.

histogram.cgi – histograma do número de eventos ocorridos.

history.cgi – exibe todos os eventos que já ocorreram.

notifications.cgi – visão geral de todas as notificações enviadas.

showlog.cgi – exibe todas as entradas do arquivo de *logs.*

summary.cgi – relatório de eventos, que podem ser agrupados por *host,* serviço, categoria de erro e período de tempo.

trends.cgi – coordenadas de tempo gravando os estados que ocorreram.

Todos estes *CGI*s verificam se a pessoa que está executando as ações solicitadas possui permissão para isto. Normalmente um usuário pode somente acessar os *hosts* e serviços para os quais ele estiver definido como um contato, mas há a possibilidade de definir usuários específicos com direitos compreensivos, assim eles estão basicamente habilitados a exibir todas as opções disponíveis.

3.1 *status.cgi*

Exibição das variações de estado. Este *CGI* permite a visualização do estado de todos os clientes e serviços monitorados pela rede. O que será exibido é determinado por três grupos de parâmetros:

- O primeiro define se a página *web* gerada exibe *host*, grupo de *hosts* ou grupo de serviços. Adicionando "*all*" às opções citadas indica que se deseja todas as ocorrências da opção solicitada e não um específico.

- O segundo provê cinco possibilidades do estilo de saída: *overview* – representa os *hosts* em uma tabela, mas resume os serviços de acordo com os estados; *summary* – comprime a saída do *overview,* somente exibe um grupo de *host* para cada linha; *grid* – provê um resumo extremamente atrativo em que pode-se ver o estado de cada serviço de forma individual por meio de cores com as quais são destacados; *detail* – mostra cada serviço, detalhadamente, em uma linha separada; *hostdetail* – é limitado somente a informação de *host,* provendo informação detalhada com uma linha para cada um.

– O terceiro e último grupo de parâmetros permite determinar, através de *selectors* (seletores) quais estados e propriedades serão mostrados, tais como: todos os serviços em um estado de erro para os quais nenhum *acknowledgements* ainda tenha sido indicado por um administrador.

Na Figura 6 está representado uma das telas de estado.

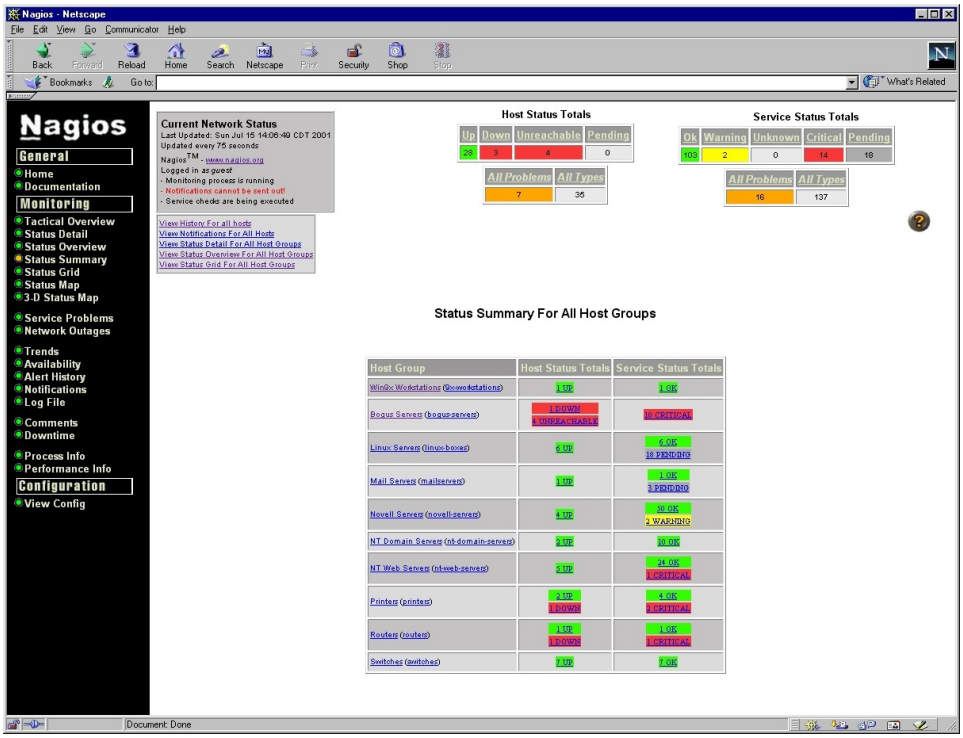

Figura 6 – Tela Padrão para Resumo do Estado
(*http://nagios.sourceforge.net*)

3.2 *statusmap.cgi*

Mapa da topologia da rede. Este CGI provê uma visão de dependência entre *hosts* monitorados. Iniciando a partir de um servidor central Nagios no meio da tela, linhas conectam todos os *hosts* que o servidor alcança diretamente – e que definições de *host* não precisam do parâmetro *parent* (pai) especificado.

Os gráficos também revelam os *hosts* para os quais o Nagios tem somente acesso indireto através de outros *hosts*.

A maneira como o Nagios arranja os *hosts* no gráfico é definido no arquivo de configuração *cgi.cfg* ou através da interface *web*. As coordenadas e ícones são definidos no arquivo *hostextinfo.cfg*. A Figura 7 representa a utilização do *CGI* de mapa de estado.

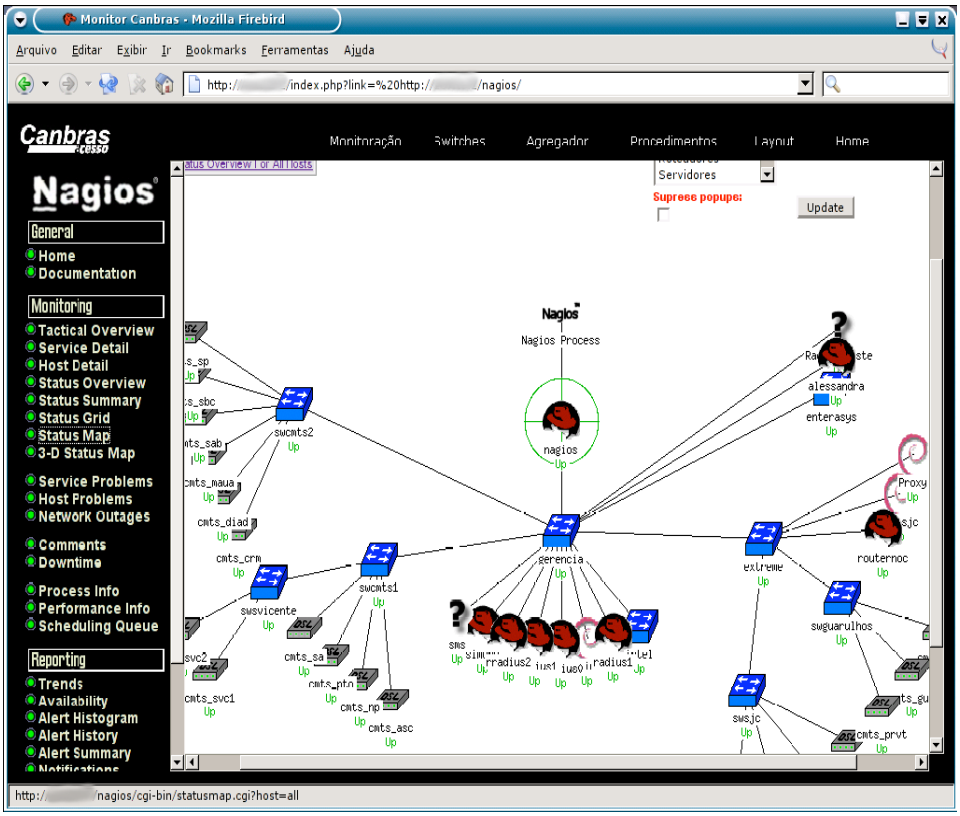

Figura 7 – Tela Padrão para Visualização de Rede 2D

(*http://eng.registro.br*)

3.3 *statuswrl.cgi*

Este *CGI* permite ao Nagios utilizar-se de uma representação 3D para mostrar a topologia da rede. Nesta opção pode-se aplicar *zoom* em *hosts*, ter uma visão geral, modificar o posicionamento da figura exibida, etc. Um

browser com capacidade *VRML* (*Virtual Reality Markup Language*) é necessário para esta exibição. Na Figura 8 temos um modelo do gráfico gerado por este *CGI.*

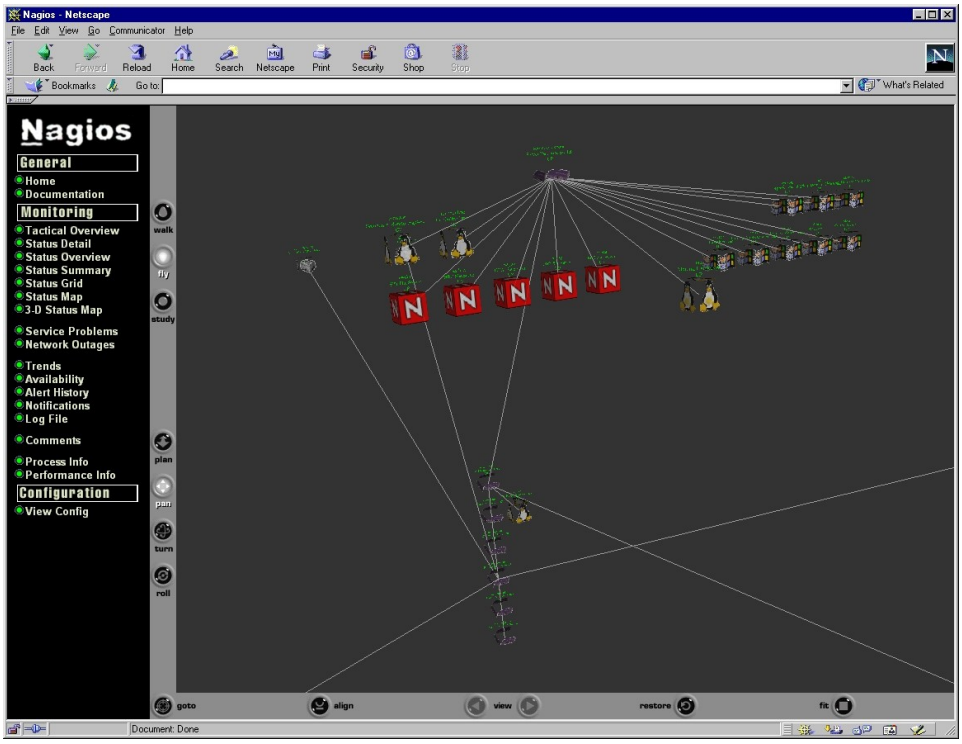

Figura 8 – Tela Padrão de Visão 3D

(http://nagios.sourceforge.net)

3.4 *statuswml.cgi*

Consultas de estado com um telefone celular. De maneira a tornar a informação fornecida pelo Nagios acessível através de WAP (*Wireless Access Protocol*) – capacitar dispositivos sem uma completa funcionalidade de *browser*, este *CGI* gera uma página *web* em um formato WML (*Wireless Markup Language*), que pode ser exibida em um telefone celular – caso o servidor *web* esteja acessível pela *internet*. Além das consultas de estado para *hosts* e serviços, ele também permite que o programa CGI desligue testes e notificações, e confirme problemas existentes com *acknowledgements*.

A disponibilização deste acesso pela *internet* deve ser estudada com cuidado no que tange à segurança. Nagios disponibiliza muitos dados sensíveis que podem ser mal utilizados por *hackers*.

3.5 *extinfo.cgi*

Informação adicional e centro de controle. Se executado com o parâmetro *host* ou *service*, este CGI não somente provê informação detalhada sobre um *host* específico ou serviço, como também serve como um centro de controle para *hosts* e serviços (parâmetro *hostgroup*) e para grupos de serviço (*servicegroup*). Dependendo da classe do objeto para o qual ele é chamado pode-se executar vários comandos a partir dele.

De acordo com o valor especificado como parâmetro, este CGI pode mostrar dados tais como: identificação de processo, horário de início, comandos e informações em *host* e serviço, comentários disponíveis em uma única página, desempenho do Nagios, verificações ativas e passivas, centro de comandos para o *hostgroup* e *servicegroup*, períodos de manutenção planejados, visão geral de todos os testes planejados classificados pelo próximo horário de implementação, hora da última verificação, entre outros.

3.6 *cmd.cgi*

Interface para comandos externos. Muito versátil, com 100 funções, cobre quase todas as possibilidades que a interface provê para comandos externos. Para se obter a descrição de um único serviço , deve-se especificar o *host* e o nome do serviço. Caso o CGI seja executado manualmente , a forma *web* mostra consultas desses valores, e se ele foi iniciado por outro programa CGI, os dados requeridos são passados através de parâmetros CGI. O arquivo de código fonte *include/common.h* contém uma lista completa de todos os

possíveis valores, incluindo aqueles que foram previstos mas ainda não foram implementados.

3.7 *tac.cgi*

As coisas mais importantes para uma visualização rápida. Como uma "Visão Geral Tática", este CGI provê informação sobre a "saúde" da rede em uma única página *web*, exibida em um sumário, como representado na Figura 9. Na página são mostrados, em ordem de prioridade, primeiro a falha de toda a rede alcançavel, seguido pelo estado de *hosts* e serviços, e lista se características de monitoração individuais, tais como notificações e manipuladores de eventos, estão ativos. Tudo está concentrado em exibir problemas. Para todos os problemas exibidos que venham a ocorrer pode-se ter uma única visão geral específica mostrando *hosts* e serviços em questão.

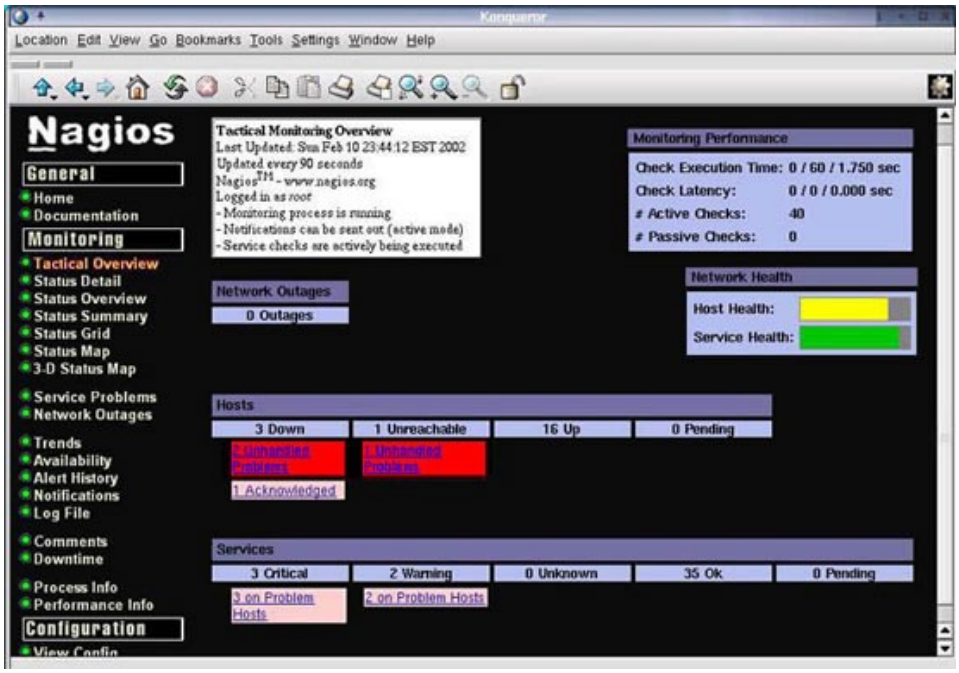

Figura 9 – Tela Padrão de Visão Geral Tática

(*http://www.onlamp.com)*

3.8 *outages.cgi*

Analisando redes parcialmente corrompidas. O *CGI* de informações estendidas somente mostra a falha parcial da rede; ao contrário da visão geral de estado, este CGI especifica quantos serviços e *hosts* estão sendo afetados em cada caso. Com os ícones na coluna que representa "Ações" pode-se chamar outros programas CGI que seletivamente filtram informações dos *hosts* ali representados. O termo *outage* deve ser entendido aqui como um "apagão", uma falha geral da rede, causando uma situação em que nenhum computador-cliente responda às solicitações enviadas pelos *plugins* do Nagios.

3.9 *avail.cgi*

Estatísticas de disponibilidade. Caso esteja monitorando sistemas, então pode-se ter interesse em sua disponibilidade. Este *CGI* primeiro pergunta se há interesse em *hosts,* serviços, *hostgroups* e *servicegroups*. Depois que o período de tempo foi selecionado, será apresentada uma visão geral. Os *hosts* envolvidos são apresentados separadamente dos serviços. A disponibilidade é apresentada duas vezes em cada caso: primeiro com um valor absoluto para o período avaliado, e então, entre parênteses, com respeito ao tempo durante o qual os dados realmente estavam disponíveis.

3.10 *config.cgi*

Consultando as definições dos objetos. Este CGI mostra uma visão tabular da definição de todos os objetos para um tipo que pode ser especificado – o tipo do objeto envolvido pode ser definido no campo de seleção disponível na tela. Este programa não provê nenhum meio para alterar qualquer uma das definições existentes na configuração. Adicionalmente, somente os usuários autorizados terão acesso a esta visualização. Na Figura 10 está representado um modelo de tela onde se visualiza a configuração

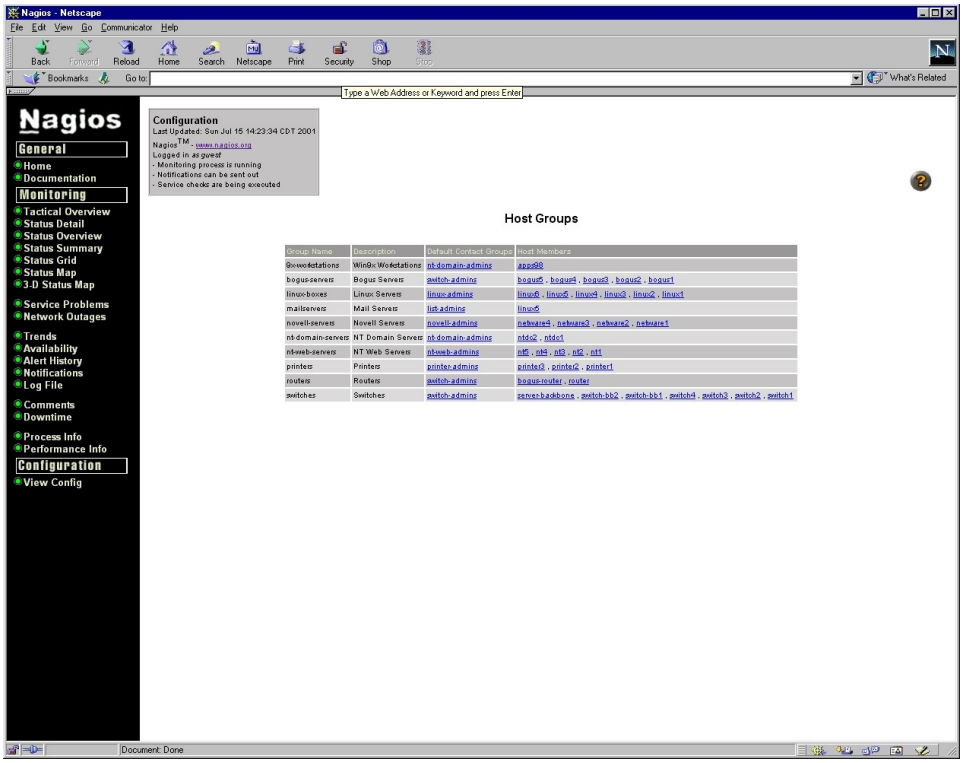

Figura 10 – Tela Padrão de Visão de Configuração

(*http://nagios.sourceforge.net)*

3.11 *history.cgi*

Filtrando entradas no *log* depois de estados específicos. Este *script* CGI permite que os estados de um determinado tipo (*soft* ou *hard*) sejam identificados seletivamente do arquivo de *log* usando o campo de seleção, para então extrair os eventos específicos. As entradas a serem mostradas podem ser restritas através de parâmetros de *hosts* individuais, serviços, ou grupo de *hosts* ou serviço quando o programa CGI é chamado. O período que este programa mostra depende do intervalo de arquivamento. O *script* sempre se refere ao conteúdo de um arquivo de *archive*.

3.12 *notifications.cgi*

Quem disse, o quê, e quando? Outra forma de filtro do arquivo de *log* é oferecida por este CGI. Ele mostra todas as mensagens enviadas. Aqui a exibição pode ser restrita para um grupo de mensagem específica, através do campo de seleção: para todas as notificações envolvendo *hosts*, para todas as que são sobre serviços em um estado crítico, e assim por diante. Mostra as notificações de serviços e clientes enviadas a vários contatos, conforme pesquisa filtrada pelo administrador, e ilustrado na Figura 11.

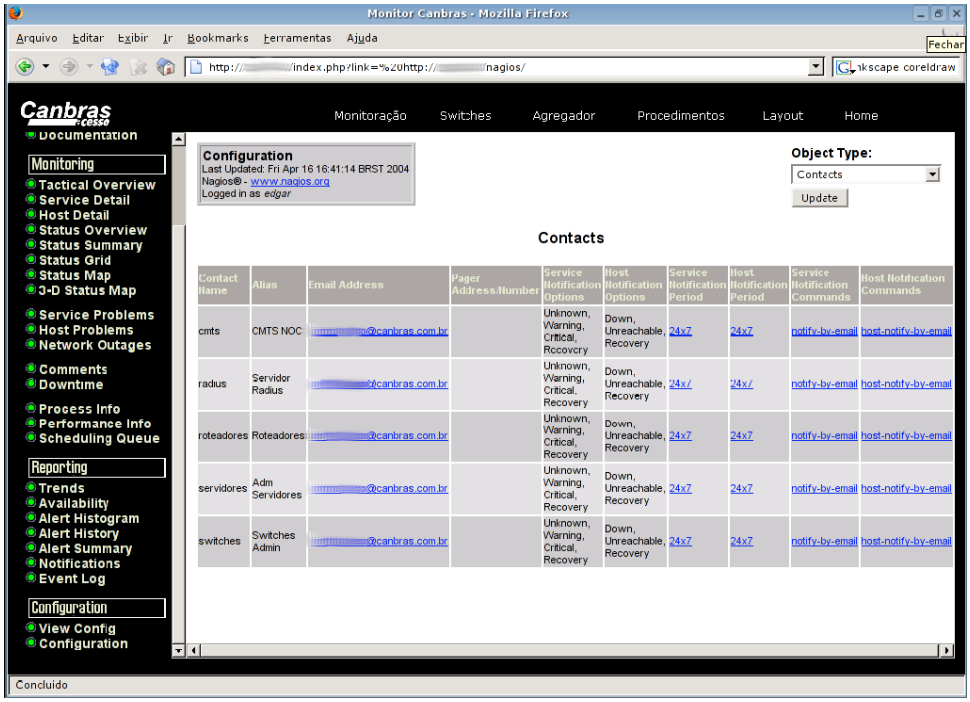

Figura 11 – Tela Padrão para Grupos de Contato e Notificações
(*http://eng.registro.br)*

3.13 *histogram.cgi*

Quais eventos ocorrem e com qual frequência. Se o estado de um *host* ou serviço muda, é chamado de um evento. Este programa CGI mostra a frequência destas mudanças em visões diferentes. De acordo com a

configuração efetuada este programa pode mostrar em que dia da semana ocorreram a maioria dos eventos, ajustar o período do relatório, contabilizar entre resultados *soft* e *hard*, assumir que o estado depois da inicialização do sistema é idêntico ao que existia antes, entre outras possibilidades.

3.14 *showlog.cgi*

Mostrando todas as entradas do arquivo de *log*. Este programa CGI mostra o arquivo de *log* como ele é, com um pouco de ícones coloridos adicionados para ajudar a encontrar o que se deseja: um botão vermelho marca estados de serviço ou *host* com problemas, um botão amarelo para estado de alerta e um verde para situações de normalidade. Outros botões referem-se a entradas de informação ou reinicialização do Nagios. A única opção disponível é a ordem cronológica. Normalmente mostra as entradas mais recentes primeiro. O período representado também depende do método de arquivamento.

3.15 *summary.cgi*

Avaliando o que se quer obter. Se as opções de tela e seleção introduzidas não forem suficientes para o que se deseja verificar, pode-se criar um relatório personalizado. Provê um resumo rápido em que somente um tipo de relatório fixo pode ser selecionado. Pode-se citar alguns relatórios como lista dos últimos eventos de um tipo específico, total de eventos ocorridos num determinado período, *hosts* ou serviços que geraram o maior número de eventos, entre outros.

3.16 *trends.cgi*

Acompanhando estados graficamente ao longo do tempo. Uma rápida visão geral de qual estado ocorreu, e quando, para um *host* ou serviço particular é provido pela saída deste CGI. Depois de selecionar um serviço ou *host*

específico, pode-se definir um período. Os estados são coloridos, o que facilita a navegação neste tipo de exibição. A função *zoom* do programa *CGI* é um detalhe interessante: após selecionar a área colorida, pode-se expandir ou reduzir o período de tempo mostrado. Na Figura 12 está uma representação deste *CGI.*

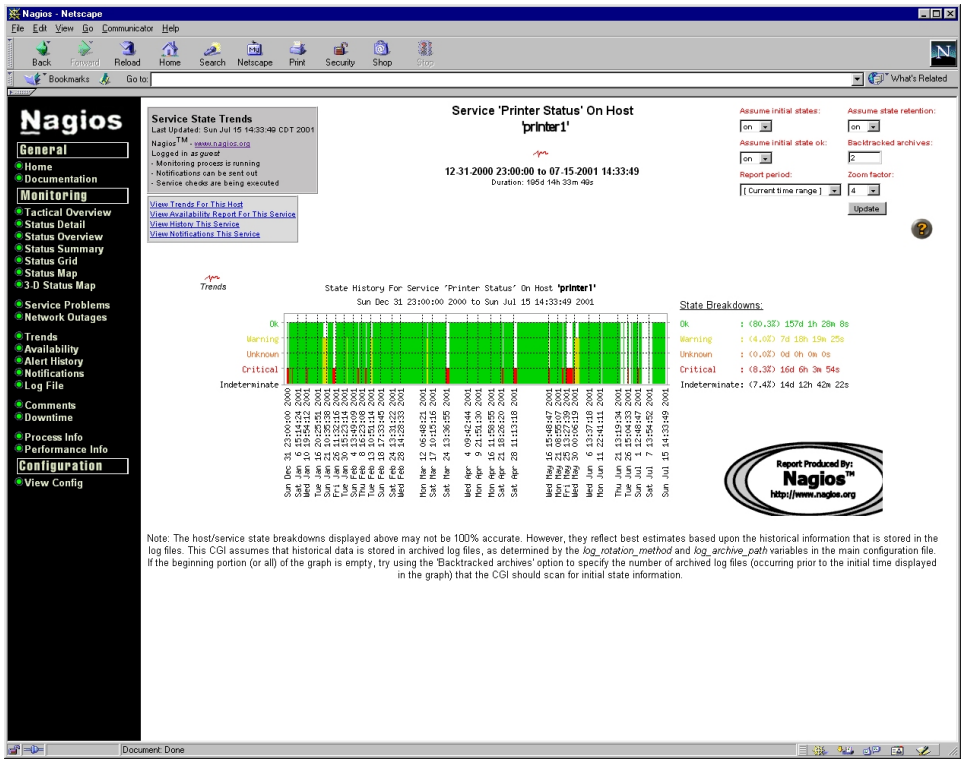

Figura 12 – Tela Padrão para *Trends*
(*http://nagios.sourceforge.net*)

4 APLICAÇÕES ESPECÍFICAS DO NAGIOS

4.1 Verificações de *hosts*

O Nagios executa as verificações de *hosts* apenas quando necessário. Embora haja um parâmetro que provê um meio de forçar uma verificação de *host,* não existe um motivo real para fazê-lo. Entretanto, existe sim um motivo para não fazê-lo: uma verificação contínua provoca uma considerável influência no desempenho do Nagios.

Sempre que o Nagios se deparar com um serviço "indisponível", é efetuada uma verificação do *host* ao qual se destina; se esta verificação também resultar em estado de indisponibilidade, o *host* será classificado como "inoperante *soft*", e uma rotina ininterrupta de verificações começará, até que o *host* responda como "operacional". Caso isso não ocorra, e após executar um número máximo de verificações, o Nagios irá classificar este *host* como "inoperante *hard*". O estado *soft* é encontrado tanto nos processos de verificação quanto de recuperação. O estado *hard* é aplicado a um *host* quando são recebidas, de forma consecutiva, várias mensagens de inoperância. A ocorrência deste estado durante um processo de recuperação se dá quando o *host* estiver em condição de erro.

De qualquer maneira, caso seja preciso verificar regularmente se o *host* está alcançável é preferível utilizar uma verificação de serviço baseada em *ping*. Ao mesmo tempo, pode-se obter informações adicionais tais como tempo de resposta ou possível perda de pacotes, as quais dão pistas indiretas sobre a carga da rede ou possíveis problemas na rede. Uma verificação de *host*, por outro lado, também emite um OK mesmo que haja uma alta taxa de perda de pacotes e um nível de desempenho catastrófico. O que está envolvido aqui, como o próprio nome implica, é apenas a acessabilidade como princípio e não a qualidade da conexão.

4.2 Verificações de serviços

Para testar serviços o Nagios faz uso de programas externos chamados *plugins*. No caso mais simples isto envolve testar um serviço de *internet*. Aqui o serviço pode ser chamado diretamente na rede, pois é suficiente chamar um programa local no servidor Nagios que teste o serviço específico no servidor remoto.

Nem tudo o que se queira testar pode ser alcançado tão facilmente na rede. Entretanto, não existe um protocolo de rede para verificação da área disponível em disco, por exemplo. Neste caso é necessário iniciar um *plugin* num *host* remoto através de um *shell* remoto (mas primeiro ele precisa ser instalado no computador remoto), ou então usa-se de outros métodos tal como o protocolo SNMP (*Simple Network Management Protocol*) para verificar a capacidade do disco.

Os quatro métodos de verificação de serviço de forma "ativa", onde o Nagios toma a iniciativa e dispara o teste, são: executar diretamente um *plugin* no seu próprio servidor; executar um *plugin* que por sua vez vai chamar um outro no *host* remoto; executar um *plugin* também diretamente na máquina cliente, mas utilizando o serviço *NRPE - Nagios Remote Plugin Executer* - criado especificamente para este propósito; e, por último, utilizar uma consulta via *SNMP* - para isto o cliente precisa de um agente *SNMP* disponível. Existem vários *plugins* disponíveis para consulta de dados através de protocolo.

Outro tipo de verificação é feita de forma passiva: o Nagios apenas aguarda pela informação a ser enviada pelo cliente através do programa *NSCA (Nagios Service Check Acceptor)*. No próprio servidor Nagios o NSCA executa como um *daemon* que recebe os resultados enviados e os repassa à interface para comandos externos.

Existem outras maneiras de efetuar verificações além dessas. Usualmente, um serviço separado é instalado no cliente, que então é consultado pelo servidor Nagios através de um *plugin* especializado. Um típico exemplo é o *NSClient/NC_Net*, que pode ser utilizado para monitorar servidores Windows.

O Nagios pode chamar uma grande variedade de *plugins*, cada um específico para um serviço particular. Tal programa especializado possui vantagens sobre um genérico. Um *plugin* genérico testa apenas se a porta TCP ou UDP está aberta e se o serviço está "escutando", mas ele não determina se o serviço correto está na porta e se está ativo. *Plugins* especializados utilizam protocolos de rede e testam se o serviço na porta em questão se comporta da maneira esperada.

O pacote com os *plugins* do Nagios, que são instalados separadamente, incluem *plugins* especializados nos serviços de rede mais importantes. Caso algum esteja faltando para um serviço específico, vale a pena dar uma olhada no sítio no Nagios, ou no sítio *www.nagiosexchange.org*. Se não for encontrado nenhum *plugin* que se adeque, pode-se usar *plugins* genéricos, que apesar de simplesmente testarem uma porta, podem também enviar dados à porta de destino e avaliar a resposta (mas isto só faz sentido, na maioria dos casos, se um protocolo baseado em ASCII estiver envolvido).

O processo de definição das verificações dos serviços em *hosts* busca equilibrar o processamento interno do Nagios e, conseqüentemente, diminuir a carga sobre os *hosts* remotos. Para realizar a primeira série de verificações, o Nagios distribui cada uma destas verificações ao longo do intervalo de tempo necessário para conferir todos os serviços. O Nagios tenta, ao menos na fase inicial, manter as verificações relativamente eqüidistantes.

O Nagios possui um sistema interno de controle que permite a atualização dos resultados relativos à verificação dos serviços. Se esta opção

for ativada pelo administrador, assim como for definido o intervalo máximo entre cada verificação, o Nagios irá forçar uma verificação de serviços ativos sempre que este intervalo for atingido.

4.3 Tratadores de Eventos

Dois tipos de tratadores de eventos (comandos opcionais, executados sempre que ocorrer mudança de estado do *host* ou do servidor) são utilizados pelo Nagios: os tratadores de eventos locais, executados somente quando há mudança de estado específico para o *host* local; e os tratadores de eventos globais, executados quando há uma mudança de estado de qualquer *host* ou servidor.

Os tratadores de eventos dependem de permissões, e, se o serviço ou *host* tiver um nível de acessibilidade mais alto, estes "tratamentos" não serão executados. O arquivo de comandos será previamente verificado, em um período definido pelo administrador durante a configuração, e também imediatamente após a execução de um tratador de eventos.

O Nagios permite, também, receber comandos externos oriundos de outros *softwares*, permitindo assim determinadas ações como a parada de todas as verificações ou a adição de novos *hosts*.

4.4 *Softwares* de apoio *NRPE* e *NSCA*

Um método alternativo de executar *plugins* instalados em um computador remoto através de um *shell* seguro é representado pelo *NRPE*. O *NRPE* é instalado em um *host* remoto e iniciado pelo *daemon inetd*, que deve ser configurado apropriadamente. Se o *NRPE* recebe uma consulta do servidor Nagios através de determinada porta TCP, ele irá executar uma consulta semelhante a esta. Como o método usando *shell* seguro, o *plugin* que executará

o teste deve estar instalado no servidor remoto. Este método requer uma conta em um *shell* local, permitindo assim que qualquer comando seja executado, mas o *NRPE* fica restrito aos comandos que lhe foram configurados.

O *NSCA* executa como um *daemon* no servidor Nagios e aguarda a chegada de resultados dos testes, por esta característica é conhecido como passivo pois o próprio Nagios não toma iniciativa. O *NSCA* se utiliza da interface para comandos externos usada pelos *scripts CGI*, entre outros, para enviar comandos para o Nagios. Esta interface consiste de um *buffer* para o qual os processos gravam e do qual eles lêem os dados. É importante considerar os aspectos de segurança para a implementação do *NSCA*.

4.5 Monitoramentos

O monitoramento distribuído é o resultado da ação de vários servidores Nagios espalhados pela rede, que fazem suas checagens individuais e repassam seus resultados a um servidor central. Os servidores Nagios distribuídos, talvez em diferentes filiais de uma companhia, trabalham como instâncias autônomas, exceto por eles enviarem os resultados para o escritório principal. Isto não verifica as redes descentralizadas de forma ativa, mas processa informações enviadas das filiais em uma maneira puramente passiva. Este tipo de monitoramento permite o balanceamento da carga de memória, da *CPU*, etc. Os servidores secundários podem conter apenas o "esqueleto" do Nagios e o aplicativo *NSCA*.

O Nagios efetua uma verificação ativa de um serviço em um *host* remoto. Assim que a resposta é recebida, o resultado é enviado de forma automática para o cliente *NSCA.* O cliente *NSCA* envia o resultado para o *daemon NSCA* localizado no servidor central que escreve um comando de processamento de resultado no arquivo de comandos externos, informando ao Nagios central as ocorrências. Este por sua vez faz uma leitura periódica do

arquivo de comandos externos e, a seguir, executa o comando de processamento de resultado, recebendo assim o resultado da verificação.

O monitoramento redundante, diretamente ligado ao monitoramento distribuído, representa os servidores paralelos, responsáveis pela coleta de dados. No caso do servidor Nagios central apresentar alguma falha e, em conseqüência disso, se desconectar da rede, um destes servidores paralelos pode assumir o lugar do Nagios central. Há a hipótese, ainda, de um *host* cliente deixar de responder a solicitações de *ping* de forma eventual, estado conhecido como *flapping*. Este caso, de acordo com os criadores do Nagios, é um evento normalmente associado a defeitos de configuração, gerando uma corrente de notificações de erro e recuperação.

O Nagios permite o monitoramento de *clusters,* tanto de computadores , oferecendo diversos serviços, quanto dos próprios serviços, como se estes fossem um único computador ou um único serviço.

4.6 Notificações e filtros

Uma das preocupações a se ter para um sistema de notificação é a configuração correta de quem deverá ser avisado no momento em que falhas ocorrerem. Dificilmente algum administrador de sistema ou rede teria a possibilidade de visualizar continuamente a interface *web* do Nagios e aguardar a ocorrência de mudanças de estado. Um sistema prático deve informar ao administrador de forma ativa para que ele possa intervir somente quando o Nagios alcançar um estado de alarme. Outro ponto a ser analisado é o excesso de mensagens, por vezes errôneas, que dificulta a identificação de problemas reais.

As notificações são um meio de se fornecer informações, em tempo real, relativas ao estado de *hosts* ou serviços*,* como representado na Figura 13.

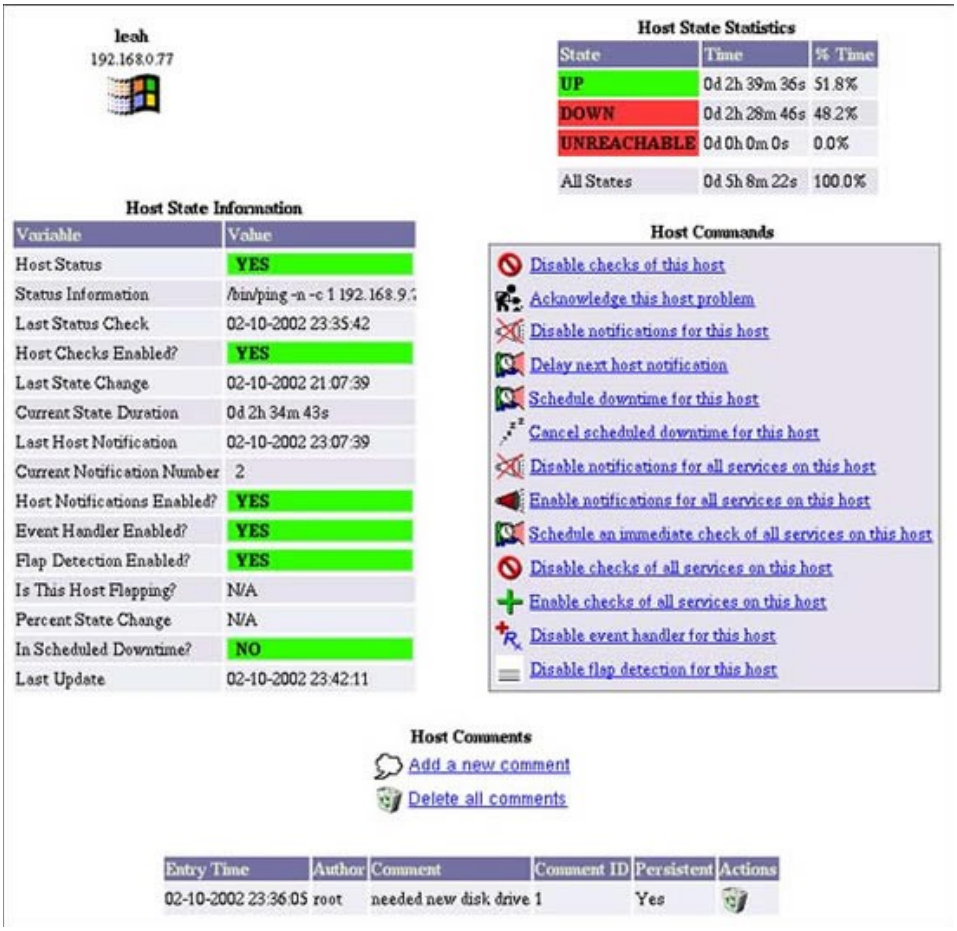

Figura 13 – Tela Padrão de *Status* de *Host*
(*http://www.onlamp.com*)

O Nagios provê um sistema de notificação sofisticado permitindo com que o meio ambiente seja finamente ajustado às necessidades de cada instalação. Esforços para mantê-lo pequeno e modular também se aplicam ao sistema de notificação. O envio de mensagens é novamente deixado para programas externos: de um simples *e-mail* a *SMS*, são inúmeras possibilidades.

Na intenção de fazer com que o Nagios envie mensagens significativas, o administrador deve ter em mente o momento em que o sistema deverá gerar a

mensagem, quando ela deverá ser enviada, quem será informado e de qual modo será enviada.

As verificações de *host* e serviço geram a mensagem, esta passa através de filtros que usualmente referem-se a tempo, e, no caso de aprovada, é utilizado um programa externo que a envia ao respectivo contato. Filtros evitam uma mensagem de ser criada ao invés de filtrar as já geradas mas, para manter as coisas de forma simples, entretanto, digamos que o Nagios crie uma mensagem que será descartada por um filtro correspondente.

Cada mensagem é precedida por uma verificação de *host* e serviço, que determina o estado corrente. A mensagem é gerada quando um estado *hard* muda para outro estado *hard* e quando um computador ou serviço permanece em um estado de erro *hard*, confirmando a existência de um problema. Enquanto não for atingido o número especificado de repetições de eventos para o estado *soft,* o administrador não será informado a menos que verifique na interface *web* ou no arquivo de *log*. Assim o administrador somente será informado quando o problema não tiver sido resolvido. Por outro lado, para se obter disponibilidade como tal, normalmente tem importância se um serviço não está disponível durante minutos no final, motivo pelo qual os estados *soft* também são considerados na avaliação.

Para um *host* qualquer ser definido como o causador da falha, primeiro tem de estar hierarquicamente abaixo ou inacessível, e pelo menos um de seus *hosts* pais estar operacional; segundo, todos os seus *hosts* filhos têm de estar abaixo ou inacessíveis e não haver nenhum *host* pai operacional.

Os filtros do programa fazem uma verificação geral do Nagios, no que concerne às notificações que precisam ser enviadas, não fazendo distinção, ou seja barra todas as notificações, ou não barra nenhuma. Os filtros de serviço e de *hosts* selecionam notificações de forma mais completa, podendo reter

alguma , segundo alguns critérios. Já os filtros de contatos referem-se aos critérios específicos de cada contato, onde qualquer notificação discordante das opções incluídas na definição do contato serão filtradas.

4.7 Suporte para Bancos de Dados

Nagios provê três *plugins* para monitoramento de banco de dados: *check_pgsql* para o PostgreSQL, *check_mysql* para o MySQL e *check_oracle* para o Oracle. Eles todos têm em comum o fato de que podem ser usados tanto localmente como através da rede. O último tem a vantagem de que o *plugin* em questão não precisa estar instalado no servidor de banco de dados. A desvantagem é que será necessário um envolvimento maior com a autenticação porque configurar um acesso local seguro ao banco de dados é algo mais simples.

Para sistemas menos críticos, o acesso à rede por *plugins* pode ser feito sem a utilização de senha. Para fazer isto, o usuário *nagios* é configurado com sua própria base de dados no sistema de gerenciamento a ser testado e que não contenha dados importantes. Os *plugins* citados aqui possuem somente acesso de leitura no banco de dados. O *check_mysql* adicionalmente permite uma verificação de conexão pura, sem acesso de leitura. Um acesso de gravação para o banco de dados não está disponível em nenhum dos *plugins* mencionados. Para o Oracle há um *plugin* no sítio *www.nagiosexchange.org* chamado *check_oracle_writeacess.sh* que testa a capacidade de gravação no banco de dados.

5 CONFIGURAÇÃO E INSTALAÇÃO DO NAGIOS

Embora a configuração do Nagios pode se tornar demorada, existe a opção de somente modificar uma pequena parte dela e conseguir deixar o sistema ativo e executando. Por sorte, muitos parâmetros no Nagios já estão definidos com valores de padrão aceitáveis.

Nagios inclui uma documentação extensa própria, uma vez instalado no diretório */usr/local/nagios/share/docs*, que pode também ser acessado pela interface *web*. Esta é sempre recomendada como uma fonte útil para maiores informações.

Os arquivos de referência da configuração do diretório */etc/nagios* terminam com *-sample* e são os que serão modificados em uma atualização do produto, ficando assim preservados os arquivos já customizados para produção.

Todos os trabalhos subsequentes devem ser carregados como o usuário *nagios*. Na edição de um arquivo como um usuário privilegiado, deve-se assegurar que os conteúdos do diretório */etc/nagios* mais tarde pertençam ao usuário *nagios*. Com a exceção do arquivo *resource.cfg*, que poderá ter acesso permitido apenas ao usuário *nagios*, todos os outros podem ser liberados para leitura.

A configuração central é feita no arquivo *nagios.cfg* que contém as referências aos outros arquivos de configuração.

Aqueles que compilam e instalam o Nagios têm a vantagem de não precisarem ajustar o *nagios.cfg*, desde que todos os caminhos estejam corretamente configurados. E isto seria tudo o que necessitaria ser feito. Não obstante uma pequena modificação seja recomendada, pois ajuda a manter uma

apresentação clara e uma configuração consideravelmente simplificada onde redes maiores estão envolvidas.

Os objetos do Nagios descrevem uma unidade específica: um *host*, um serviço, um contato, mas também os grupos a que eles pertencem. Até mesmo comandos são definidos como objetos. Esta definição não é por acaso: Nagios é também capaz de herdar características.

Host

O objeto *host* descreve um dos nós de rede que está sendo monitorado. Nagios espera o endereço *IP* como um parâmetro (ou um nome de domínio completo) e o comando para identificar se o *host* está ativo. A definição de *host* é novamente referenciada na no arquivo que trata do serviço.

Hostgroup

Alguns *hosts* podem ser combinados em um grupo. Esta configuração simplifica, pois grupos de *hosts* podem ser especificados ao invés de *hosts* únicos quando definir os serviços (o serviço vai então existir para cada membro do grupo). Em adição, Nagios representa os *hosts* de um grupo de *host* juntos em uma tabela na apresentação *web*, que também ajuda para tornar mais clara a visualização.

Service

Os serviços individuais a serem monitorados são definidos como objetos de serviço. Um serviço nunca existe independente de um *host*. Assim é possível ter alguns serviços com o mesmo nome, contanto que eles pertençam a *hosts* diferentes. Na linguagem do Nagios, um serviço é sempre um par *host*-serviço.

Servicegroup

Da mesma forma que em *hostgroups*, Nagios também combina alguns serviços para representá-los na apresentação *web* como uma unidade com sua própria tabela. Grupos de serviços não são absolutamente essenciais, mas ajudam a melhorar a visualização e também são usados nos relatórios.

Contactgroup

Notificação de eventos em *hosts* e serviços são feitas para um grupo de contato. Uma ligação direta entre *host*/serviço e uma pessoa de contato não é possível.

Timeperiod

Descreve um período de tempo dentro do qual o Nagios deve informar os grupos de contato. Fora destes intervalos, o sistema não enviará qualquer mensagem. A cadeia de mensagens pode ser ajustada para vários períodos de tempo, dependendo do *host*/serviço e contato/grupos de contato.

Command

Nagios sempre chama programas externos pelos objetos de comando. Além dos *plugins*, programas de mensagens incluem envio de *e-mails* ou *SMS*.

Servicedependency

Este tipo de objeto descreve dependências entre serviços. Se, por exemplo, uma aplicação não funciona sem um banco de dados, um objeto de dependência correspondente irá assegurar que o Nagios vai representar uma falha no banco de dados como um problema primário ao invés de somente anunciar o não funcionamento da aplicação.

Serviceescalation

Usado para definir um gerenciamento de escalamento próprio: se um serviço não está disponível depois de um período de tempo específico, Nagios informa um círculo de pessoas adicional. Este pode ser configurado em múltiplos níveis, em qualquer modo que se deseje.

Hostdependency

O mesmo que o *servicedependency* só que para *hosts.*

Hostescalation

O mesmo que o *serviceescalation* só que para *hosts.*

Hostextinfo

Objetos de informação estendida de *host* são opcionais e definem um gráfico específico e /ou *URL*, que o Nagios adicionalmente inclui em seus gráficos de saída. A *URL* pode se referir à página *web* que provê informação adicional no *host*.

Serviceextinfo

Informação estendida de serviço, o mesmo que *hostextinfo* só que para serviço.

Quando se inicia o uso do Nagios, é recomendado que se restrinja à utilização de uma configuração mínima, somente com um ou dois objetos por tipo de objeto, de forma a minimizar fontes potenciais de erro e obter um sistema executando o mais rápido possível.

O método mais simples de instalação é utilizar os pacotes fornecidos com a distribuição do sistema operacional existente. Pode-se optar por compilar

manualmente, assim têm-se uma influência nas estruturas de diretório e em alguns parâmetros. Uma compilação desta forma promove um arquivo de configuração principal quase completo, no qual inicialmente nada tem que ser mudado, mas pode envolver uma extensa pesquisa dos pacotes de desenvolvimento necessários dependendo do que já estiver instalado no computador. Aqui não foram utilizadas configurações para o banco de dados.

5.1 Pré-requisitos para instalação

Com base no que foi escrito pelo autor Anderson Leite (2005) na secforum[4], primeiramente deve-se instalar os aplicativos que o Nagios poderá utilizar, como, por exemplo, o *Openssl libssl.0.9.7 (*sistema de criptografia através de chaves de segurança) para garantir o tráfego de informações para o servidor *web* Apache de forma segura, e a biblioteca gráfica *libgd-dev* (biblioteca gráfica de desenvolvimento), tais pacotes disponibilizam recursos para que o Nagios seja utilizado de forma gráfica. O pacote *MySQL mysql-server-4.5* também poderá ser instalado para armazenamento de informações, os *log*s podem ser substituídos por um *SGBD*. Os pacotes utilizados estão disponíveis em *www.nagios.org* e são: *nagios-mysql* - biblioteca que permite comunicação ao *SGBD mysql*; *nagios-common*- aplicativo do Nagios; *nagios-nrpe-plugin* - plugin que permite receber dados do *NRPE; nagios-nrpe-server* - servidor do *NRPE*; *nagios-plugins* - conjunto de *plugins* utilizados para monitoramento.

Para a instalação dos pacotes do Nagios, o administrador deverá primeiramente possuir uma *source.list* (arquivo que contém, listado, o endereço de vários servidores que possuem pacotes para instalação da distribuição escolhida) atualizada. Após sua atualização, o mesmo deverá digitar dois comandos descritos a seguir, para distribuições baseadas em *Debian*:

4 Em *http://www.secforum.com.br*

▪ *apt-get update*: este comando copiará a lista de aplicativos dos *hosts,* armazenando em *cache* local;
▪ *apt-get install nagios-common nagios-nrpe-plugin nagios-nrpeserver nagios-plugins*: este comando fará a baixa e instalação dos arquivos.

Durante o processo de instalação, o Nagios cria um usuário que será o responsável pelos serviços e, por padrão, esse usuário é conhecido como *nagios*.

5.2 Modelo para instalação do Nagios com compilação

O pacote atualizado do Nagios e de seus *plugins* podem ser obtidos no sítio do projeto Nagios, *www.nagios.org*. O nome do pacote Nagios e do diretório a ser criado irão depender da versão escolhida para instalação. Na sequência deve-se descompactar o arquivo, criar o diretório de instalação e adicionar usuário e grupo para o Nagios:

tar xzvf <nome do pacote do nagios>
mkdir -p /usr/local/nagios
adduser nagios / useradd -g nagios -d /usr/local/nagios -s /bin/false nagios
groupadd nagios

Em alguns sistemas o comando *adduser* irá criar o grupo correspondente; em outros poderá ser necessário a edição do arquivo */etc/group* para adição do usuário manualmente.

Antes de compilar o Nagios, verifique se a biblioteca *gd-lib* está instalada no seu *Linux*, pois caso contrário não será possível ver o s*tatusmap*, que é o mapa da rede. No *Debian* use *apt-get install libgd-dev*. Caso as bibliotecas *GD* já estiverem instaladas mas a compilação não estiver encontrando-as, acrescente ao comando *configure* as opções *--with-gd-lib* e *--with-gd-inc* para especificar os diretórios onde os arquivos estão localizados.

./configure --prefix=/usr/local/nagios –with-nagios-user=nagios –with-nagios-grp=nagios --with-cgiurl=/nagios/cgi-bin –withhtmurl=/nagios
(parâmetros para compilação) onde:
--prefix=/usr/local/nagios : Local dos arquivos de configuração.
--with-nagios-user=nagios : Usuário padrão
--with-nagios-grp=nagios : Grupo padrão
--with-cgiurl=/nagios/cgi-bin : Diretório onde estão os arquivos *CGI*.
--with-htmurl=/nagios : Diretório onde estão os arquivos *HTML*.

make all -s (criação de binários)
make install -s (instalação de binários)
make install -init -s (instalação de *script* de inicialização)
make install-commandmode (configura as permissões de pasta para aplicativos externos)

No *Slackware* 8.0 é instalado um *script* do Nagios no diretório */etc/rc.d*. Dependendo da distribuição este arquivo deverá também ser instalado no diretório */etc/rc.d/init.d/*. No *FreeBSD,* o arquivo deverá ficar no diretório */usr/local/etc/rc.d* e ser renomeado para *nagios.sh* para que funcione propriamente.

make install-config -s (instalação de exemplos de configuração no diretório */usr/local/nagios/etc*)

O diretório */usr/local/nagios*, neste momento, contém cinco diretórios a saber: *bin* – com arquivos binários, responsáveis pela monitoração; *sbin* – contém os *scripts CGI* que serão usados pela interface *web*; *share* – onde se encontram os arquivos *html* e a documentação do Nagios; *var* – onde o Nagios armazena suas informações uma vez que esteja executando, ou seja, os *logs*; e por fim *etc* – com arquivos de configuração.

5.3 Instalação de *Plugins* do Nagios

Eles devem ser baixados separadamente do sítio *www.nagios.org* e instalados. Embora os *plugins* sejam distribuídos em um mesmo pacote, são

independentes entre si, assim pode-se substituir uma versão de um *plugin* de forma individual a qualquer tempo, inclusive como um *plugin* de construção própria. Antes de iniciar a instalação é recomendável a leitura do arquivo *requirements*, para verificar quais os requisitos dos *plugins*. Em caso de faltarem programas ou módulos *Perl* enquanto a configuração estiver executando será emitida uma notificação. Para sua instalação deve-se descompactar o arquivo com o código-fonte e compilá-los:

tar xzvf <nome do arquivo>
./configure --prefix=/usr/local/nagios –with-nagios-user=nagios –with-nagios-grp=nagios

make all -s (criação de binários)
make install -s (instalação de binários)

As bibliotecas serão instaladas no diretório */usr/local/nagios/libexec/* e, se não houver conteúdo nele, deve-se criá-lo e copiar os arquivos do diretório */usr/lib/nagios/plugins/* para lá. Caso apresente erro da *libcrypto*, crie o seguinte simbólico:

ln -s /lib/libcrypto.so.0.9.7a /lib/libcrypto.so.4

5.4 Configuração pós-instalação do Nagios

Após a instalação do Nagios e de seus *plugins* é necessário fazer a configuração apropriada ao ambiente que se deseja monitorar. No diretório */usr/local/nagios/etc* podemos encontrar os arquivos modelo para configuração. É interessante criar um diretório para mantê-los como fonte de consulta mas, para que o Nagios possa executar os arquivos devem ser renomeados para **.cfg*. O processo de instalação do Nagios assegura que os caminhos usados no arquivo de configuração, *nagios.cfg,* estejam corretos. Depois serão copiados, para customização, somente os arquivos necessários para o funcionamento do Nagios:

```
cd /usr/local/nagios/etc
mkdir sample
for i in *cfg-sample; do mv $i `echo $i | sed -e s/cfg-sample/cfg/`; done;
cd /usr/local/nagios/etc/sample
cp cgi.cfg checkcommands.cfg misccommands.cfg nagios.cfg
resource.cfg timeperiods.cfg ../
```

Ainda será necessário criar os seguintes arquivos e diretório:

```
touch /usr/local/nagios/etc/dependencies.cfg
touch /usr/local/nagios/etc/escalations.cfg
mkdir -p /usr/local/nagios/var/rw
chown nagios:nagios -R /usr/local/nagios/var/rw
```

Descrição dos arquivos de configuração:

nagios.cfg : responsável por iniciar os serviços de monitoramento.
cgi.cfg : programas *CGI*s localizados na pasta *sbin*.
hosts.cfg : informações dos *hosts*.
hostgroups.cfg : informações dos *hosts* por grupos.
contacts.cfg : contatos que deverão ser notificados em caso de problema.
contactgroups.cfg : contatos divididos em grupos.
service.cfg : serviços que deverão ser monitorados
hostextinfo.cfg: definição das imagens apresentadas no *statusmap*.
dependencies.cfg : informações de dependências de serviços.
timeperiods.cfg : informações de diferentes períodos de monitoramento.
checkcommands.cfg : definição dos comandos que podem ser executados
resource.cfg : macros definidas pelos usuários.

5.5 Instalando o servidor *web* Apache

Baixar o pacote de instalação do Apache do sítio *www.apache.org*, descompactar, compilar e instalar.

```
 tar xzvf <nome do pacote>
./configure
make
make install
```

Editar o arquivo de configuração do servidor Apache, *httpd.conf*, podendo-se utilizar o comando *vi /etc/httpd/conf/httpd.conf ,* e adicionar a seguinte cláusula no final do arquivo: *Include /etc/httpd/conf/nagios.conf.*

Criar o arquivo de configuração do Nagios, podendo-se também utilizar o comando *vi /etc/httpd/conf/nagios.conf ,* e adicionar a seguinte configuração:

```
ScriptAlias /nagios/cgi-bin /usr/local/nagios/sbin/
<Directory "/usr/local/nagios/sbin/">
AllowOverride AuthConfig
Options ExecCGI
Order allow,deny
Allow from all
</Directory>
Alias /nagios /usr/local/nagios/share/
<Directory "/usr/local/nagios/share">
Options None
AlowOverride AuthConfig
Order allow,deny
Allow from all
</Directory>
```

Depois é necessário criar o arquivo com usuário e senha, sendo que a opção *-c* no comando *htpasswd* só deverá ser usada na primeira vez quando da criação do arquivo.

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users usuario
chown apache:apache /usr/local/nagios/etc/htpasswd.users
```

Criar o arquivo *.htaccess* e adicionar os seguintes conteúdos, nos diretórios */usr/local/nagios/share/* e */usr/local/nagios/sbin*/:

```
#touch /usr/local/nagios/share/.htaccess
#touch /usr/local/nagios/sbin/.htaccess
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
require valid-user
```

Criar usuário e senha para logar na interface do Nagios e torná-lo capaz de executar *HTTP*. Após esta configuração deve-se reiniciar o servidor *web.*

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
ls -l /usr/local/nagios/etc/htpasswd.users
chmod o+r /usr/local/nagios/etc/htpasswd.users
service httpd restart
```

5.6 Configurações Avançadas do Nagios

O parâmetro *check_external_commands,* no arquivo *nagios.cfg,* deve ser alterado para o valor 1. As seguintes mudanças serão necessárias no arquivo *cgi.cfg*:

```
use_authentication=1
authorized_for_system_information=nagiosadmin
authorized_for_configuration_information=nagiosadmin
authorized_for_system_commands=nagiosadmin
authorized_for_all_services=nagiosadmin
authorized_for_all_hosts=nagiosadmin
authorized_for_all_service_commands=nagiosadmin
authorized_for_all_host_commands=nagiosadmin
```

Adicionar servidores a serem monitorados

Fazer as mudanças específicas para os *hosts* no arquivo *hosts.cfg.*

```
Generic host definition template
define host{
      name                    generic-host
      notifications_enabled       1
            max_check_attempts 5
            notification_interval 20
            notification_options d,u,r
            notification_period 24x7
              event_handler_enabled       1
      flap_detection_enabled       1
      process_perf_data          1
      retain_status_information     1
      retain_nonstatus_information    1
      register                 0
      }
```

```
define host{
        use                     generic-host            ; Name of host template to use
        host_name               <FQDN of server>
        alias                   <Any alias name for the host>
        address                 <IP of server>
        check_command           check-host-alive
        max_check_attempts      10
        notification_interval   120
        parents
        contact_groups
        notification_period     24x7
        notification_options    d,u,r
        }
```

Repetir para todos os *hosts* a serem monitorados.

Agrupá-los no arquivo *hostgroups.cfg*

Na opção *members*, os membros listados devem existir no arquivo de configuração de *hosts* (*hosts.cfg*), bem como as pessoas a serem notificadas devem estar definidos no arquivo de configuração de grupos de contatos. Vale ressaltar que, caso se deseje adicionar mais de um membro ao grupo, basta separar o nome dos membros por vírgula.

```
Define hostgroup{
        hostgroup_name  <Host group name>
        alias           <any alias name>
        contact_groups  <contact group name>
        members         <<host1>,<host2>,..>
}
```

Repetir para todos os grupos de *hosts*.

Especificar os serviços que se deseja monitorar

Editar os detalhes de todos os serviços que se deseja que o Nagios verifique no arquivo *services.cfg*.

```
Generic service definition template
 define service{
 name                           generic-service
host_name host_name
```

```
service_description service_description
servicegroups servicegroup_names
is_volatile [0/1]
check_command command_name
max_check_attempts #
normal_check_interval #
retry_check_interval #
active_checks_enabled [0/1]
passive_checks_enabled [0/1]
check_period timeperiod_name
parallelize_check [0/1]
obsess_over_service [0/1]
check_freshness [0/1]
freshness_threshold #
event_handler command_name
event_handler_enabled [0/1]
low_flap_threshold #
high_flap_threshold #
flap_detection_enabled [0/1]
process_perf_data [0/1]
retain_status_information [0/1]
retain_nonstatus_information [0/1]
notification_interval #
notification_period timeperiod_name
notification_options [w,u,c,r,f]
notifications_enabled [0/1]
contact_groups contact_groups
stalking_options [o,w,u,c]
}

define service{
        use                     generic-service
        host_name               <FQDN of server to check the service>
        service_description     <Service name>
        is_volatile             0
        check_period            24x7
        max_check_attempts      3
        normal_check_interval   3
        retry_check_interval    1
        contact_groups          <contact group name for alerts to
                                send>
        notification_interval   120
        notification_period     24x7
        notification_options    w,u,c,r
```

check_command *<check command for the service>*
}

Repetir para cada serviço em cada *host*.

Agrupá-los no arquivo *servicegroups.cfg*

Neste arquivo agrega-se os serviços entre os vários *hosts.*

define servicegroup{
servicegroup_name servicegroup_name
alias alias
members members
}

Definir informações adicionais aos *hosts*

O diretório padrão das imagens é */usr/local/nagios/share/images/logos/*, no entanto, poderão ser utilizadas imagens de um outro pacote disponível no sítio *prdownloads.sourceforge.net*.

No caso do *Debian* o diretório é */usr/share/nagios/htdocs/images/logos*. Para que o Nagios acesse as imagens é preciso adicionar uma linha no arquivo *cgi.cfg*:

xedtemplate_config_file=/etc/nagios/hostextinfo.cfg

As linhas abaixo representam o arquivo *hostextinfo.cfg:*

define hostextinfo{
name email
host_name
2d_coords
3d_coords
icon_image
icon_image_alt
vrml_image
statusmap_image
gd2_image
register 0
}

Um outro arquivo também pode ser utilizado para adicionar imagens aos serviços do Nagios, o *serviceextinfo.cfg*.

Especificar os contatos para mensagens de alerta

Na intenção de especificar os detalhes dos contatos para alertas, é necessário editar o arquivo *contacts.cfg*. Métodos de notificação como *SMS* para celulares, podem ser utilizados mas, para isto, em alguns estados, é necessário contratar os serviços da operadora de telefonia celular, que disponibiliza um *gateway* por onde as mensagens podem ser enviadas.

```
define contact{
contact_name contact_name
alias alias
contactgroups contactgroup_names
host_notification_period timeperiod_name
service_notification_period timeperiod_name
host_notification_commands command_name
service_notification_commands command_name
email email_address
pager pager_number or pager_email_gateway
addressx additional_contact_address
service_notification_options w,u,c,r ( w=warning / u=unknown /
c=critical / r=recoveries / n=none)
host_notification_options d,u,r ( d=down / u=notify / r=recoveries /
n=none )
}

 'nagios' contact definition
define contact{
        contact_name                    <contact name>
        alias                           <some nickname>
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r
        host_notification_options       d,u,r
        service_notification_commands  notify-by-email,notify-by-epager
        host_notification_commands
         host-notify-by-email,host-notify-by-epager
        email                           <email address of the contact>
```

```
        pager                   <pager or mobile no.(optional)>
        }
```
Repetir para todos os contatos.

Agrupá-los no arquivo *contactgroups.cfg*

```
define contactgroup{
        contactgroup_name       <contact group name>
        alias                   <any nickname for the group>
        members                 <members (comma seperated)>
        }
```
Repetir para todos os grupos que incluem todos os contatos.

Escalando com Nagios

Se for necessário enviar alertas de forma seletiva, por exemplo quando alertas críticos ocorrerem, é necessário configurar o arquivo *escalations.cfg*.

```
define serviceescalation{
        host_name               <FQDN of the server>
        service_description      < name>
        first_notification       2
        last_notification        6
        contact_groups           < groups (comma seperated)>
        notification_interval    0
        }
```
Repita para todas as escalas.

Algumas definições de parâmetros:

host_name: nome do micro na rede;
alias: uma descrição para o computador;
address: o endereço IP do *host*;
check_command: comando de checagem do *host* a ser executado, definido em *checkcommand.cfg*;
max_ckeck_attempts: quantidade de tentativas de checagem antes de reportar erro/indisponibilidade;
notification_interval: espaço de tempo (em minutos) em que deve ser enviada a notificação de erro/indisponibilidade dos serviços ao usuário responsável;
notification_period: intervalo de tempo em que o serviço está ativo (intervalos de tempo podem ser definidos no arquivo *timeperiods.cfg*);

notification_options: tipos de erros que devem ser notificados para este *host*, onde: **d** - o serviço está inativo (*down*); **u** - o serviço não pode ser encontrado (*unreachable*); **r** - o serviço voltou a funcionar (*recovery*).
parents: utilizada normalmente caso a máquina seja ligada a outra, desta forma a organização/administração fica mais eficiente.

5.7 Verificação do processo de instalação

O administrador, após efetuar todo o processo de configuração, poderá averiguar se o Nagios apresenta alguma falha em sua configuração. O Nagios não irá iniciar se existir algum erro, já com alertas ele inicia, mas é conveniente verificar a causa dos mesmos. Pode-se checar se a configuração está correta e, em caso positivo obter erros e alertas além de qual arquivo está gerando o problema. Para isso, basta digitar o seguinte comando:

/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Pode-se ainda acrescentar o Nagios para os serviços do sistema com o comando *chkconfig --add nagios.* Para habilitá-lo em nível de serviço para que inicialize junto com o sistema operacional use o comando: *chkconfig nagios on.* Caso deseje pode criar uma entrada no arquivo *cron.daily* para que o serviço seja reiniciado diariamente, por exemplo:

vi etc/cron.daily/nagios-restart.cron
#!/bin/sh
/sbin/service nagios restart >/dev/null 2>&1
chmod +x etc/cron.daily/nagios-restart.cron

5.8 Utilizando o Nagios pela primeira vez

Para se iniciar o Nagios podemos utilizar dois métodos:

/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
ou
etc/init.d/nagios start

Para acessar o Nagios através de qualquer interface *web* na rede, basta digitar no *browser* o endereço *http://localhost/nagios*. Para que o Nagios seja usado na *internet*, é necessário que o servidor tenha um *IP* válido, e que permita que sítios sejam publicados.

5.9 Instalação do *plugin NRPE*

A instalação desse *plugin* permite que o Nagios monitore *hosts* e serviços de forma passiva. Nesse caso, existem duas versões de *NRPE*, uma voltada para clientes *Linux,* e outra voltada para clientes *Microsoft Windows* (98, ME, 2000, XP, 2003 ou versões mais atualizadas). Neste tópico será descrito seu processo de instalação para o segundo sistema operacional.

Primeiramente, o administrador deverá fazer o *download* do arquivo *NRPE_NT*, disponível no sítio *prdownloads.sourceforge.net*. Em seguida, deverá ser descompactado no diretório *c:/nrpe_nt*. Então, faz-se necessário configurar o arquivo *nrpe.cfg* e, conseqüentemente, será necessário instalar o servidor do *NRPE_NT* no próprio Windows. Para isso deverão ser utilizadas as duas linhas de comando abaixo:

```
c:/nrpe_nt/bin>nrpe_nt -c nrpe.cfg -i
c:/nrpe_nt/bin>net start nrpe_nt
```

A primeira linha é para a instalação do serviço no Windows; a segunda linha é para executar o serviço.

6 CONCLUSÃO

A instalação e configuração do Nagios é bastante facilitada pela quantidade de listas de discussões na *internet*. Além disso, o próprio sítio oficial do Nagios disponibiliza formas de contato bastante ágeis entre os usuários e os desenvolvedores do sistema. O presente estudo permitiu, através do *software* de gestão Nagios, a avaliação de diversos aspectos da gestão e monitoramento de redes de computadores.

Alguns itens podem ser implementados para aprimorar o Nagios: desenvolvimento de um *front-end* de configuração do Nagios, com o intuito de facilitar e concentrar o meio de configuração dos seus arquivos *cfg;* ampliação da capacidade de monitoramento do Nagios, para abranger arquivos de servidor *web*, evitando, assim, a ação de *hackers;* desenvolvimento de *plugins* para o Nagios com objetivos específicos, voltados para o monitoramento de algum equipamento em particular, como controladores de temperatura, umidade, volume de água, etc.

Com os recursos humanos tornando-se cada vez mais escassos, nenhum departamento de TI pode se dar ao luxo de ter seus sistemas manualmente verificados. Redes estão se tornando mais complexas e demandam especialmente a necessidade de serem informadas, o quanto antes, sobre quedas que aconteceram ou por problemas que estão por acontecer. O Nagios, uma ferramenta de código aberto para monitoração de sistemas e redes, ajuda o administrador a detectar problemas antes que o telefone comece a tocar.

Devido à eficiência do *daemon* de monitoração, ele não sobrecarrega o servidor nem os dispositivos de rede, outras aplicações podem compartilhar o dado SNMP, o teste de dispositivos é feito de forma rápida, gera relatórios identificando imprecisão na monitoração existente e possui dados de configuração unificados. Sempre tem algo mais no Nagios a se implementar.

Grupos de serviços, notificações, dependências e escalamento vão refinar ainda mais a maneira que ele trabalha. Ferramentas baseadas no Nagios possuem potencial para monitorar a rede pró-ativamente, ativando agentes remotos para executarem procedimentos corretivos de recuperação de falhas.

O Nagios provê uma visão do essencial de performance e disponibilidade e é mais um exemplo de como a Comunidade de Código Aberto pode ajudar no gerenciamento da rede. Com relação à utilização de *softwares* de código livre, esta parece ser uma alternativa cada vez mais usada pelas grandes corporações. No entender de muitos administradores, este fato deve-se à maturidade que aplicativos como o Nagios vêm adquirindo, e à solidez do próprio Linux, principalmente no que tange ao seu conceito no mercado mundial de *softwares*. O esforço despendido na configuração do Nagios é gratificado com o resultado obtido após sua implementação.

Finalmente, gostaria de citar, de forma breve, um caso de sucesso do qual tive a satisfação de participar, mas que, por questões profissionais, não foi utilizado como base neste trabalho. Iniciamos a avaliação do aplicativo Nagios, na empresa em que trabalho, devido ao fato dos órgãos públicos estarem com a diretiva de utilização de *software* livre. A implantação começou de forma modesta, apenas com o cadastramento de *switches* e roteadores, mas a cada implementação ia se tornando mais robusto e hoje comporta servidores, serviços e tudo que é considerado como produtivo e de responsabilidade do CPD.

A equipe, antes conhecida como operação, tem atualmente o *status* de monitoração e é responsável pelo acompanhamento ininterrupto do que podemos considerar como a "saúde" da rede. Há uma console ligada ao Nagios com acompanhamento de 24 horas, configurada com alarmes de cores, sonoros e envio de mensagens para os responsáveis.

Foi implementada uma rotina de atendimento para ações pró-ativas onde a intenção é identificar o problema antes da reclamação do usuário ou pelo menos minimizar as consequências do problema. No caso falhas em equipamentos, a equipe de monitoração aciona o Serviço de Instalação e Manutenção de Equipamentos para que seja feita uma verificação no local. No caso de problemas nos serviços, o pessoal de suporte responsável pelo serviço é acionado imediatamente. Em ambos os casos é verificada a necessidade de se informar ao Serviço de Atendimento ao Usuário para que, ciente do fato, possa comunicar aos usuários a ocorrência do problema para que tomem ciência de que já estão sendo tomadas as providências, tranquilizando-os e evitando a abertura de vários chamados.

Estamos instalando os equipamentos para a nossa central de monitoramento, NOC (*Network Operationl Control*), e, para satisfação da equipe de rede, o primeiro *software* a ser migrado será o Nagios, nosso "semáforo", que de um simples teste passou para um serviço que, pelo menos atualmente, está sendo considerado como essencial para a produção. Como complemento do Nagios temos implementado o MRTG, mas agora iniciamos a utilização do Cacti também.

REFERÊNCIAS BIBLIOGRÁFICAS

BARTH, Wolfgang. **Nagios System and Network Monitoring** 1st ed. São Francisco: No Starch, 2006

NAGIOS. **Nagios Official Website,** em <http://www.nagios.org>

NETSAINT. **Netsaint Official Website,** em <http://www.netsaint. org>

NAGIOS PLUGINS, em <http://www.nagiosplug.sourgeforge.net>

NAGIOS SCREENSHOTS, em <http://www.nagios.org>

Procedure for the installation of the Nagios Network Monitoring Program, em <http://nagios.sourceforge.net/download/contrib/documentation/misc/InstallingNagios-r2.pdf>.

Nagios Version 1.0 Documentation, em <http://nagios.sourceforge.net/download/contrib/documentation/english/Nagios_1_0_Docs.pdf>.

Nagios in High Availability Environments. em <http://nagios.sourceforge.net/download/contrib/documentation/misc/HighAvailability/NagiosHA_EN.pdf>.

Estudo de uma Ferramenta de Gestão de Redes, em <http://nagios.sourcefor.ge.net/download/contrib/documentation/misc/Nagios_Portuguese.pdf>.

GLOSSÁRIO

Acknowledgement: no contexto do Nagios, mais que um simples comentário, é uma sinalização ao administrador que alguém já está cuidando do problema.

Browser: o mesmo que navegador *WWW*, *browser WWW* ou *web browser*. Programa utilizado para visualizar as páginas armazenadas em servidores da *World Wide Web*. Utilizando uma definição um pouco mais técnica, um *browser* é um programa cliente que permite acessar, geralmente por meio de uma interface gráfica, informações diversas em formato hipertexto (na linguagem *HTML*) armazenadas em servidores locais ou remotos.

Bug: um *bug* em um computador é uma falha. *Softwares* "*bugados"* costumam travar mais.

CGI: *Common Gateway Interface* é um padrão para programas externos se comunicarem com servidores de informação tais como um servidor *HTTP*.

Cluster: é um conjunto de máquinas (no caso de *cluster Linux*, especificamente, PC's) interligadas via rede que trabalham em conjunto trocando informações entre si.

Copyleft: tipo de programa ou serviço derivado de um código livre, que deve obrigatoriamente permanecer com uma licença livre.

Daemon: programa que é executado em computadores servidores cuja função é estar constantemente aguardando solicitações de outros programas para, então, executar uma determinada ação e retornar a resposta adequada.

DNS: acrônimo de *Domain Name System* (Sistema de Nome de Domínio). É o sistema responsável por traduzir nomes de domínio em endereços numéricos (endereços IP) de servidores *Internet* e vice-versa.

E-mail: o mesmo que correio eletrônico.

Firewall: em português, "parede corta-fogo". Conjunto de equipamentos e softwares que impedem o acesso não autorizado a redes de computadores privadas.

Flapping: situação em que um serviço muda freqüentemente de estado.

FTP: o *File Transfer Protocol* (Protocolo para Transferência de Arquivos) é o protocolo padrão utilizado para a transferência de arquivos de um computador remoto para um computador local e vice-versa.

GUI: *Graphical User Interface* (Interface Gráfica do Usuário) é uma interface de usuário para interação com um computador que emprega imagens gráficas em adição ao texto que representa as informações e ações disponíveis para o usuário.

Hacker: uma pessoa que sente prazer em conhecer mais profundamente o funcionamento de um sistema, de um computador e de redes de computadores, em particular. O termo tem sido usado equivocadamente como sinônimo de *cracker*.

Hardware: é toda a parte física do computador que se pode tocar. Essa parte física executa as instruções do *software* para gerar a saída ou entrada de informações de dados.

Host: tecnicamente, *host* é um computador principal num ambiente de processamento distribuído ou o computador central que controla uma rede. Na *Internet* é qualquer computador ligado à rede, não necessariamente um servidor.

HTTP: o *Hyper Text Transport Protocol* (Protocolo de Transporte de Hipertextos) é o protocolo padrão da *WWW* (*World Wide Web*), utilizado para a transferência de arquivos do tipo hipertexto na *Internet*.

Hub: é o objeto da rede que repassa dados adiante, seja para todos os dispositivos conectados, como na *Ethernet,* seja para apenas um deles como nas redes *Token Ring*.

Interface: parte do sistema computacional com o qual o usuário entra em contato físico e perceptivo.

Internet: é a conexão de várias redes. Cada provedor de *internet* pode ser uma 'rede'. Quando se tenta acessar um computador fora da rede do seu provedor, será preciso um roteador para que o mesmo indique o caminho até a rede que se queira chegar.

Layout: é o desenho, a criação bidimensional de um sistema qualquer.

Log: arquivo de armazenamento de informações referente aos resultados de falhas, operações bem sucedidas e etc.

NNTP: acrônimo de *Network News Transfer Protocol*. Protocolo que permite a assinatura, leitura e remessa de mensagens armazenadas em um servidor de notícias.

Ping: comando para checagem de resposta de um determinado cliente.

Plugin: extensão para *browsers WWW* que fornece recursos adicionais de multimídia, facilitando a visualização de textos, som, vídeo, etc.

POP3: o protocolo *POP* (*Post Office Protocol)* é utilizado por aplicações-cliente (programas) de correio eletrônico para a manipulação (leitura, remoção)

das mensagens armazenadas numa caixa de correio eletrônico. O protocolo *POP* está em sua terceira versão, conhecida como *POP3*.

Proxy: um *proxy* é um servidor que serve de "ponte". Uma conexão feita através de *proxy* passa primeiro por ele antes de chegar ao seu destino. Desse modo, se o *proxy* não estiver disponível, a conexão não pode ser efetuada. Como todos os dados trafegam pelo *proxy* antes de chegar à *Internet*, eles são usados largamente em redes empresariais para que diversos computadores tenham conexão limitada e controlada.

Roteador: dispositivo responsável pelo encaminhamento de pacotes de dados dentro de uma rede ou entre redes. Os roteadores são programados para saberem por quais rotas de rede enviar um pacote baseado em seu endereço de destino.

Scheduler: Um escalonador que organiza a realização de processos ou procedimentos dada uma lista de eventos e períodos em que devem ser realizados.

Site: uma localidade onde computadores são instalados e operados. Também é chamado *site* o conjunto de informações disponibilizados por uma determinada instituição através de um ou mais métodos de obtenção dessas informações (como *FTP* e *WWW*) e acessíveis a partir de um endereço único na *Internet*. Assim, uma determinada instituição pode oferecer um acervo de imagens, por exemplo, armazenando-as em seu "*site FTP*" ou em seu "*site www*".

SMTP: o *Simple Mail Transfer Protocol* é um protocolo que faz parte do ciclo das mensagens eletrônicas. Como o próprio nome diz, ele é um protocolo muito simples se comparado a outros, inclusive ao seu colega de trabalho, o *POP*.

Software: *software* é a parte lógica do computador, os programas.

Status: estado; o momento, a condição em que se encontra um computador ou serviço.

Switch: *switches* são dispositivos que filtram e encaminham pacotes entre segmentos de redes locais.

VRML: *Virtual Reality Modeling Language* - linguagem de programação em 3D.

WAP: *Wireless Application Protocol* - protocolo de aplicação sem fio.

WEB: é considerada a rede mundial de computadores que trocam informações através do *HTTP*. Para que isso ficasse bem claro, foi inventado o subdomínio *WWW*, que é a abreviação de *World Wide Web*. A *web* é o serviço mais usado na *Internet.* Outros serviços comuns que a *Internet* proporciona são o *e-mail* e *IRC (Internet Relay Chat)*. Dependendo do contexto, "*web*" pode significar o mesmo que *internet.*

Zoom: Aumento da visualização de parte de um objeto qualquer, seja um texto, figura, etc.