Faculdade de Engenharia da Universidade do Porto

FEUP

# Monitorização de Sistemas de Informação Críticos

Vladimiro Macedo

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Informática e Computação

Orientador: Prof. Dr. João Neves

# Monitorização de Sistemas de Informação Críticos

**Vladimiro Macedo**

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Prof. Luís Paulo Reis

Vogal Externo: Prof. José Manuel Torres

Orientador: Prof. João Neves

17 de Junho de 2011

# Resumo

Uma rede de computadores com os seus equipamentos, servidores e serviços nela disponibilizados pode ser, hoje em dia, o activo mais importante de uma empresa ou instituição. De facto, cada vez mais, nos momentos em que a rede se encontra inoperacional, um número significativo de operações do quotidiano da empresa ou instituição deixa de ser possível. Os prejuízos que esses tempos acarretam tornaram claro que o cuidado com a monitorização da rede e a tentativa de previsão do surgimento de problemas podem ser um dos melhores investimentos feitos por uma companhia.

Este documento apresenta o estudo efectuado sobre as várias possibilidades de monitorização de uma rede informática, de forma a que fosse feita uma escolha das ferramentas mais adequadas à Monitorização de Sistemas de Informação Críticos. Ao longo da dissertação foram testadas várias plataformas de monitorização, de carácter livre, sendo avaliadas quais as que melhor correspondiam, em termos de resposta, aos requisitos da instituição.

# Abstract

*A computer network comprising its equipment, servers and available services can, nowadays, be considered one of the most important assests of a company or institution. In fact, more and more, whenever a network has a downtime, a significant number of regular operations from the company or institution stop happening. The losses the company incurs due to those periods have made clear that the care in the computer network monitoring and predicting network problems may very well be some of the company's best investments.*

*This document presents a study done on several possibilities of computer network monitoring, in order to choose the best tools for Critical Information Systems Monitoring. Along this dissertation several open-source monitoring platforms were evaluated to check which ones better responded to the needs of the institution.*

# Agradecimentos

- Ao Prof. Doutor João Neves, por saber apontar caminhos e ao mesmo tempo conferir toda a liberdade.
- À Engª. Susana Gaio, pela simpatia, disponibilidade e paciência.
- Ao Eng. Tito Vieira, pela simpatia e disponibilidade.
- Ao Jorge Montero, da OP5, pela simpatia e disponibilidade.
- À Amanda Mailer, da Icinga, pela simpatia, vontade de ajudar e entusiasmo contagiante
- Aos meus colegas destes anos de faculdade. Por tudo. Quantas vidas não levo eu daqui convosco?
- À Mariana, pelo amor, carinho e compreensão.
- À minha mãe porque é, obviamente, a melhor do mundo.

# Índice

# Lista de figuras

# Lista de tabelas

# Abreviaturas e Acrónimos

| | |
|---|---|
| AJAX | *Asynchronous JavaScript and XML* |
| API | *Application Programming Interface* |
| CPU | *Central Processing Unit* |
| ICMP | *Internet Control Message Protocol* |
| FEUP | Faculdade de Engenharia da Universidade do Porto |
| MIB | *Management Information Base* |
| MTTR | *Mean Time To Repair* |
| QoS | *Quality of Service* |
| RMON | *Remote Network Monitoring* |
| SiFEUP | Sistema de Informação da FEUP |
| SLA | *Service Level Agreement* |
| SNMP | *Simple Network Management Protocol* |
| TCP | *Transmission Control Protocol* |
| UDP | *User Datagram Protocol* |
| VoIP | *Voice over Internet Protocol* |
| VPN | *Virtual Private Network* |

# Capítulo 1

# Introdução

Esta secção faz uma breve apresentação do contexto, motivação e objectivos do projecto sobre o qual foi efectuada a presente dissertação. Seguem-se uma apresentação do estado da arte no segundo capítulo, uma apresentação de propostas no terceiro, servindo o quarto capítulo para avaliação das soluções testadas. Esta dissertação termina no quinto capítulo com as conclusões alcançadas e propostas de trabalho futuro.

## 1.1 - Contexto

As redes informáticas tornaram-se, na actualidade, um dos componentes mais vitais de qualquer instituição de média ou grande dimensão. Com efeito, as redes informáticas cresceram não só em tamanho e complexidade, mas também em significado e valor para a operação das companhias ou instituições. A partir de certa altura, as redes informáticas deixaram de ser uma mera ferramenta da companhia e passaram a ser a própria companhia devido à forma ubíqua com que se integraram na mesma. Esta conclusão é óbvia quando se consideram companhias de comércio electrónico ou outras cujos proveitos venham do seu *website.*

No entanto, do ponto de vista estratégico, a rede é aquilo que faz a companhia ou instituição crescer com um mínimo de esforço, pois facilita a colaboração, comunicação e transacção. É onde está guardada a informação dos funcionários, dos clientes, dos alunos e onde se efectua o acesso às operações diárias de vendas, facturação, tesouraria, e-mail e fichas de alunos e pessoal, variando consoante a natureza da actividade. Alguns dos serviços existentes na rede conseguem ser críticos ao ponto de paralisar toda a instituição, bastando para isso pensar no serviço de autenticação, sem o qual os utilizadores ficam sem acesso à rede e impedidos de utilizar todos os restantes.

Existe assim um determinado conjunto de serviços de rede que podem ser denominados de serviços críticos, tendo em conta a importância do seu funcionamento para a empresa ou instituição.

Um recurso tão valioso como este exige que se assegure a sua disponibilidade, sendo que a mesma pode ser afectada tanto por falhas ao nível dos equipamentos, como pelo subdimensionamento da capacidade dos mesmos, ou mesmo por tentativas de ataques ilícitos à rede. Os prejuízos causados pelo tempo de inoperacionalidade ou pela perda de dados poderão tornar-se avultados.

Mesmo o mais experimentado dos administradores de rede pouco poderá fazer se não tiver o conhecimento do que se passa a cada momento na sua rede, sabendo assim onde e como actuar; a única forma de se criar esse nível de conhecimento da rede é fazendo uma monitorização contínua da mesma.

A complexidade de uma rede actual, com todos os equipamentos que a compõem, bem como com todos os serviços e elementos de software instalados, sendo utilizados quer por utilizadores locais ou remotos via *Virtual Private Network* (VPN), torna por vezes bastante difícil obter uma percepção de qual a verdadeira utilização da rede, quer em volume de tráfego, quer na utilização de serviços.

Perceber a composição e complexidade da rede, e ter a capacidade de, a cada momento, estar informado sobre a disponibilidade de cada um dos elementos que a compõe pode ser um factor de sucesso da integridade e disponibilidade da rede informática e consequentemente factor contributivo para o sucesso da empresa ou instituição.

## 1.2 - Motivação

A monitorização de redes de computadores é actualmente umas das áreas de maior importância na área das tecnologias de informação em qualquer empresa de média ou grande dimensão. Cada vez mais as empresas estão conscientes do prejuízo económico que acarreta cada minuto em que a sua rede informática se encontre inoperacional.

Assim, quando consciente da importância da monitorização de rede para o bom funcionamento da mesma e sendo confrontado com a hipótese de poder efectuar uma dissertação na área, principalmente na realização de um estudo de proposta para uma rede de elevado tráfego como a da FEUP, foi considerado que era uma hipótese excelente de aquisição de conhecimentos sobre monitorização de redes. Isto devido a tratar-se de uma rede constituída por equipamentos não só actuais, como também de grande capacidade, aliado a um ambiente académico, que encoraja e potencia a aquisição de conhecimento, tornando-a numa oportunidade dificilmente replicável.

## 1.3 - Objectivos

Os objectivos propostos para a dissertação são baseados no estudo das ferramentas de monitorização preventiva de software e hardware disponíveis no mercado, como por exemplo o Oracle Grid Control, Nagios, Zabbix e equivalentes. Mediante esse estudo será criada uma pequena infra-estrutura de testes que permita comparar as soluções tidas como mais indicadas para o ambiente da FEUP, pretendendo-se que no final deste trabalho seja apresentada uma proposta de sistema de monitorização preventiva que contribua para o reforço do nível de segurança da rede informática da instituição.

Tendo em conta a criticidade do núcleo central da rede onde se concentram os serviços, torna-se necessário adoptar uma estratégia de monitorização que permita não só detectar falhas, como também eventuais sobrecargas que permitam que sejam tomadas acções correctivas antes da ocorrência de uma falha total. Com o sistema proposto será possível ao administrador de sistema ser avisado em tempo útil da probabilidade de ocorrência de uma falha em qualquer um dos serviços críticos da rede.

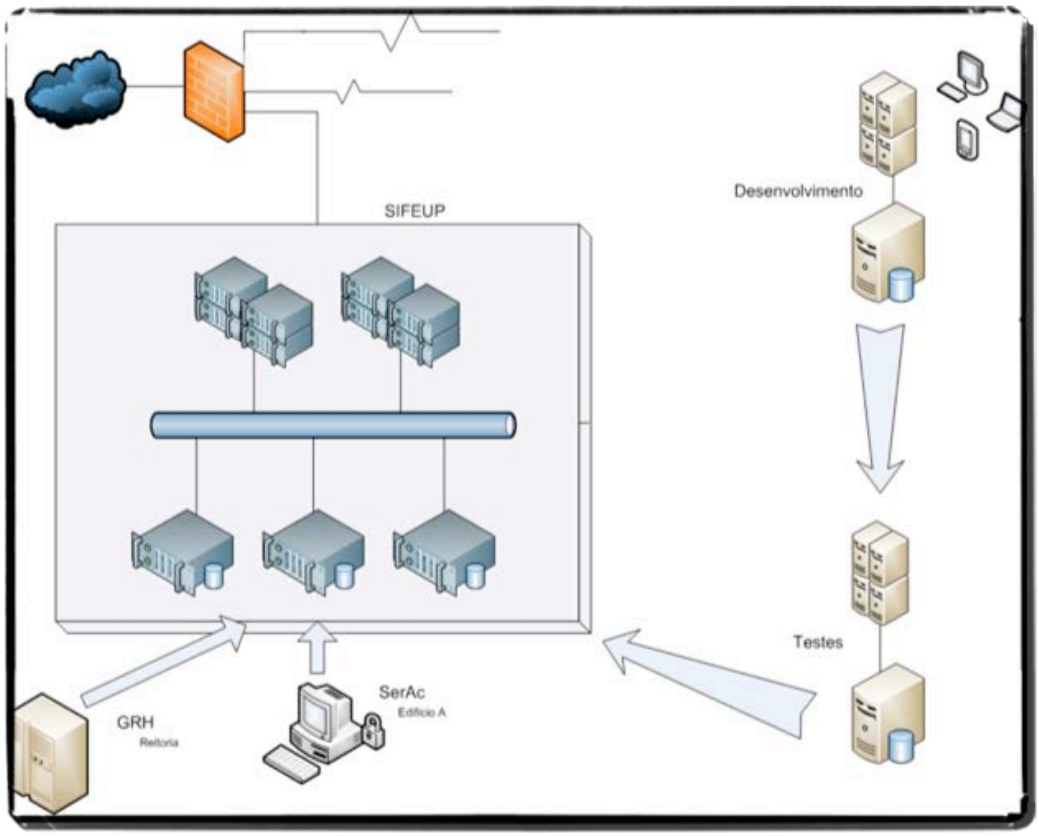



Figura 1.1 – Arquitectura do sistema central a monitorizar, com 2 grupos de servidores de aplicações balanceados e 3 servidores de base de dados Oracle igualmente balanceados

| O Quê | Porquê |
|---|---|
| Disponibilidade dos dispositivos de rede (routers, switches, servidores) | São o que garante o fluxo de dados na rede. |
| Disponibilidade dos serviços críticos | Basta que um serviço crítico esteja em baixo para existir um impacto negativo nos resultados. |
| Espaço em disco em servidores chave | Vários serviços utilizam espaço em disco. Um crescimento anómalo da ocupação pode ser um indicador de alguma configuração errada ou de aplicações imprevistas ou não autorizadas. |
| Percentagem da média da taxa de transferência máxima dos routers | Uma percentagem elevada constante poderá ser um indicador da necessidade de reforço. |
| Carga média da ocupação de memória e CPU em servidores chave | Sem memória os programas não correm e os utilizadores não são servidos. |
| Verificação da disponibilidade de *Firewalls* e outros dispositivos de segurança | Um dispositivo de segurança só é útil se estiver a funcionar. |
| Tráfego de entrada/saída nos routers | A identificação de períodos de pico permite a realização de um bom desenho/gestão de rede. |
| Disponibilidade dos clientes de rede | Estações de trabalho, impressoras e servidores secundários que devam estar sempre ligados. |
| Logs / Registos | A análise de registos pode ser uma excelente fonte de informação de problemas na rede. |
| SNMP Traps | Todo o tipo de informação útil enviada por alertas SNMP, desde falta de papel na impressora até temperatura alta no *datacenter*. |

Tabela 1.1 — Pontos Base a Monitorizar.

O sistema proposto no final deste projecto deverá, em resumo, permitir saber o que se passa a cada momento na rede, para que, baseado nesses dados constantemente actualizados e recebidos em tempo útil, seja possível diagnosticar rapidamente a ocorrência de problemas, seja fácil de ilustrar e mostrar a terceiros o que se passa na rede monitorizada, saber quando aplicar as soluções de *disaster-recovery*, assegurar a conformidade de funcionamento de todos os serviços alvo e garantir as condições de manutenção do tempo de actividade.
Através das capacidades anteriores o sistema deverá contribuir para uma redução de custos, evitando quebras de produtividade na instituição, bem como ser uma ajuda ao planeamento de actualizações e alterações da rede.

Para levar a efeito os objectivos serão testadas várias soluções de monitorização, constantes de uma lista de pré-selecção, elaborada com base nas capacidades da plataformas, bem como a sua popularidade, de forma a garantir uma elevada utilização, que favoreça a descoberta de defeitos na mesma. A frequência de actualizações e posicionamento em inquéritos de satisfação serão factores igualmente a ser levados em conta, junto com o custo, facilidade de implementação e manutenção, e ainda capacidade de adaptação a alterações na rede com um mínimo de reconfiguração necessária.

# Capítulo 2

# Estado da Arte

A monitorização das redes informáticas, tal como previamente discutido, afirma-se hoje de uma importância vital para a vasta maioria das empresas ou organizações. No entanto, com o difundir e aumentar da complexidade das mesmas constata-se que a escalabilidade das redes IP foi conseguida às custas da observabilidade [1], o que por sua vez causou o surgimento de novas e diferentes formas de monitorização de redes que pudessem ser utilizadas como ferramenta de apoio para as funções de controlo e gestão.

A maioria dos routers actuais contêm funções de monitorização embebidas, normalmente acessíveis através de mecanismos como o *Simple Network Management Protocol* (SNMP), *Remote Network Monitoring* (RMON), ou NetFlow. No entanto, estas funções internas são, por norma, ou específicas demais em relação às funcionalidades do fabricante ou então muito generalistas procurando ser o mais interoperáveis possível, não fornecendo então muitos dos parâmetros requeridos a uma monitorização moderna, com a granularidade que a mesma muitas vezes exige.

As necessidades dos utilizadores variam de acordo com o tempo e é difícil de antecipar ou prever as necessidades futuras aquando do desenho de um componente de hardware. Aliado a esse problema, existem ainda necessidades muito específicas de minorias de utilizadores que simplesmente não são comercialmente viáveis de introduzir nos equipamentos devido ao custo extra que representariam.

É neste contexto de falta de flexibilidade e desenvolvimento atempado de novas características embebidas no hardware que surgem as soluções de monitorização baseadas em ferramentas de software, com ciclos de desenvolvimento muito mais curtos, tanto das ferramentas em si, como de futuros *plugins* rapidamente desenvolvidos para dar resposta a novas solicitações do mercado.

Dentro deste paradigma de monitorização com base em software surgiram duas grandes áreas, grosso modo classificadas como monitorização activa e monitorização passiva.

## 2.1 – Monitorização Activa

A monitorização activa é uma metodologia que se caracteriza pelo recurso à introdução de tráfego na rede, injectado com o objectivo de efectuar uma medição específica. Desta forma podem-se criar vários cenários de medição possíveis, de acordo com os objectivos pretendidos. Esta metodologia é principalmente indicada nos casos em que se pretendam efectuar medições extremo-a-extremo de forma a verificar parâmetros de qualidade de serviço (QoS) e desempenho geral da rede[27]. Tal acontece porque os pacotes sonda, ao serem injectados na rede, contêm um número de sequência e uma datação (*timestamp*) que permite verificar qual o tempo que o pacote demorou a percorrer o troço de rede em análise, bem como se existiram quaisquer perdas. No entanto, enquanto esta estimativa é simples de fazer para um troço de rede, torna-se difícil combinar a totalidade dos resultados quando se efectuam análises de redes complexas, podendo mesmo deixar de ser um método eficiente, caso não sejam tomadas precauções na escolha criteriosa dos extremos.[12]

A análise é realizada com a monitorização em curso, ou seja, à medida que vão sendo recebidos os resultados das sondas, os mesmos são exibidos no monitor do administrador de sistema que tem uma visão do estado da rede a cada momento.

A monitorização activa é assim particularmente indicada para a disponibilização de dados relativos a tráfego e disponibilidade de serviços no curto e no médio prazo, ou seja, os seus resultados servirão para alimentar mecanismos reactivos, que permitam resolver situações anómalas surgidas na rede.

É uma metodologia que possui uma grande flexibilidade, quer por ser interinsecamente apropriada para métricas de extremo-a-extremo, quer pela liberdade conferida por não ter de aceder a todos os equipamentos de rede do troço a analisar, bastando aceder ao equipamento que faz a introdução do tráfego de monitorização na rede e ao que o recebe. Desta forma é um tipo de medição que escala particularmente bem, permitindo que se façam medições em troços de rede longos e complexos, ou mesmo entre redes, sem que daí advenha uma um aumento de complexidade às medições individuais.

Os pacotes de monitorização podem ser encapsulados nos protocolos existentes como o ICMP[14], UDP[15] ou TCP[16]. Como exemplo de ferramentas encapsuladas nos anteriores exitem o ping, o traceroute e o *One-Way Delay Protocol* (IPPM/OWDP)[17]. Os protocolos

utilizados para encapsular estas formas de medida não foram originalmente desenvolvidos com o objectivo da monitorização, assim não é de estranhar que existam limitações provenientes das especificidades de cada protocolo[11].
Existem várias ferramentas simples que podem ser usadas para a realização da monitorização activa, tais como o clink[18], netest[19], pathchar[20], pchar[21] ou pipechar[21]. No entanto, e como cada uma destas ferramentas simples procurava normalmente responder apenas a uma medição de um factor específico, foram criados pacotes de software que, juntando as funcionalidades oferecidas por estas e outras ferramentas procuram oferecer uma solução integrada e completa em termos de monitorização. Serão estas ferramentas integradas aquelas que serão objecto de estudo ao longo desta dissertação.

A monitorização activa é então utilizada quando se pretende uma informação a cada momento do estado da rede, na qual seja possível monitorizar um conjunto bastante heterogéneo de serviços ou equipamentos, pois existem vários tipos de testes normalizados a que qualquer equipamento deverá responder independentemente das suas especificidades. Para informação mais completa poderão ser utilizadas soluções baseadas em agentes, as quais são discutidas mais adiante neste capítulo. A monitorização activa é ainda uma técnica com boa escalabilidade: devido ao aumento do poder computacional nos últimos anos, as estações de monitorização conseguem processar sem dificuldades de maior as informações recebidas por parte das sondas ou agentes a cada momento. O principal factor que afecta a escalabilidade é a capacidade da própria rede, pois um aumento da complexidade da rede, quer em número de equipamentos ou de tráfego irá por sua vez aumentar o tráfego de monitorização, o que poderá contribuir para o degradar do desempenho da rede no seu todo.[3] É, no entanto, possível efectuar uma degradação graciosa do tráfego de monitorização, de forma a que, quando a estação de monitorização determina a influência perniciosa do seu próprio tráfego na rede, efectue uma diminuição do mesmo, passando a inferir o estado da rede através de menos dados até que seja de novo possível voltar à granularidade de informação inicial[31]. Neste sentido é uma tecnologia bastante adaptável às condições do estado da rede.
As estações de monitorização deverão ser alvo, no entanto, de uma colocação criteriosa[12] dentro da rede de forma a que possam ter acesso a todos os equipamentos a monitorizar sob pena de não conseguir enviar ou receber tráfego para ou de determinadas partes da rede, impedindo assim a colecção de dados. Quando a centralização da monitorização não for possível por constrangimentos físicos ou políticas de rede, a estratégia a utilizar deverá passar por uma divisão hierárquica da monitorização, com sub-

-estações de monitorização, também chamados nós de monitorização, com acesso à sua rede e que depois, por sua vez reportem a uma estação central coordenadora que agregará os resultados obtidos em cada uma delas[24].

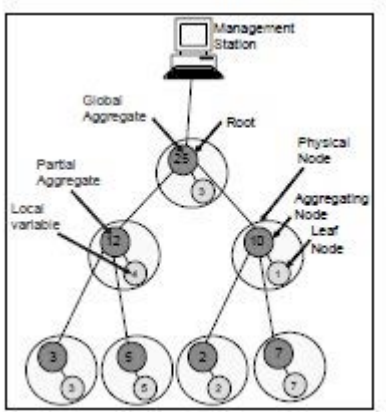



Figura 2.1 – Arquitectura de um sistema de monitorização activa baseado em agentes onde a estação de monitorização não pode contactar todos os agentes, recorrendo a nós de monitorização.[24]

Esta estratégia tinha sido já previamente defendida para a monitorização de redes complexas numa altura em que o poder de computação não era tão barato como na actualidade, dividindo a rede em domínios virtuais independentes, sendo a análise feita de forma isolada em cada um dos domínios virtuais e posteriormente apenas agregados e processados os resultados com recurso a computação paralela, garantindo assim um tempo tão reduzido quanto possível, especialmente em redes nas quais o tempo de obtenção de resultados seja um factor crítico[42]. Com os resultados encorajadores dessa experiência há quem defenda que poderá ser uma estratégia a revisitar devido ao disparo previsto de utilização de largura de banda nos próximos anos com aumentos entre 100 a 131%[41], acima daquilo que é expectável que seja o aumento da capacidade de processamento.

Com a vulgarização de requisitos de QoS e *Service Level Agreement* (SLA) no decorrer dos últimos anos a monitorização online tem assumido um papel de importância no fornecer de informação em tempo útil em relação ao estado da rede quer a clientes, quer a fornecedores de serviço[9].

Neste momento começam surgir novas propostas de visualização do tráfego na rede, aproveitando as capacidades de desenho tridimensional das actuais placas gráficas, bem como do poder de computação associado às estações de trabalho que funcionam como estação de monitorização[7]. Por outro lado estes mesmos desenvolvimentos ao nivel da

disponibilidade do poder computacional permitem que se possam começar a propor soluções de monitorização activa com uma componente passiva significativa. Esta componente pode estar sempre presente[10][14] ou pode ser despoletada pela componente activa, quando a mesma detecte qualquer comportamento suspeito e entenda ser aconselhável a recolha do tráfego em circulação para posterior análise offline[11].

Tanto no caso da monitorização com recurso a agentes, como nos casos em que o tráfego de monitorização gerado é feito com recurso a protocolos de monitorização existentes, uma das principais preocupações a ser levada em linha de conta é a interferência que o tráfego de monitorização causa na rede alvo da vigilância e análise. Com efeito, um tráfego de monitorização excessivo pode degradar o desempenho das funções normais da rede. Assim, torna-se necessário encontrar um equilíbrio entre a necessidade de actualização constante dos dados de monitorização e a minimização da interferência desse tráfego em relação ao restante, de forma a que o impacto da monitorização seja o mais desprezável possível.

Num cenário ideal, a plataforma de monitorização não estaria limitada nem em termos de restrições de utilização de largura de banda nem de capacidade de processamento. No entanto, num cenário real, há que considerar a existência de ambas. Assim, se assumirmos que a plataforma de monitorização apenas consegue processar um determinado número de respostas provenientes das sondas a cada momento, passamos a poder conhecer qual o tempo máximo necessário para o processamento das respostas de todas as sondas em circulação. Esse tempo é o tempo necessário para se poder enviar uma nova ronda de sondas e começar a processar as suas respostas. Podemos então definir esse valor como $T_{delay}$ (tempo entre dois contactos sucessivos com o mesmo equipamento), obedecendo à fórmula $T_{delay} \geq (\Delta \times N)$ , em que $\Delta$ é o tempo necessário à recepção e processamento dos dados de uma sonda e $N$ o número de sondas.

Por outro lado sabemos também que o tempo entre cada leitura da sonda enviada a determinado equipamento, para que a amostragem dos eventos de rede seja garantidamente completa, deverá ser superior ao dobro da frequência máxima de variação do valor que estamos a monitorizar, isto pela generalização que se pode fazer do teorema da amostragem de Nyquist-Shannon $T_{Sample} = \frac{1}{2 \times f_{max}}$.

A essa consideração temos então de introduzir a limitação ao nível de largura de banda que estará reservada para tráfego de monitorização. Assim, podemos calcular qual o tempo mínimo que será necessário para que todas as sondas consigam retornar informação à estação de monitorização, consoante a largura de banda que tiver sido alocada ao tráfego de monitorização. Esse tempo mínimo pode ser calculado através da expressão $T_{Traffic} = \frac{\sum(Tráfego\ gerado\ por\ cada\ sonda)}{Largura\ de\ banda\ alocada}$.

Juntando as expressões anteriores, define-se o intervalo de valores dentro do qual deve estar definido o tempo de envio das sondas para a rede, de forma a que se obtenha uma representação fiel da rede, definido como $T_{Polling}$ e estando o valor contido em $T_{Sample} \geq T_{Polling} \geq Max\ (T_{Delay}, T_{Traffic})$ em que $Max\ (T_{Delay}, T_{Traffic})$ representa o maior dos valores entre $T_{Delay}$ e $T_{Traffic}$[32].

Utilizando agentes dinâmicos é possível executar alterações a estes valores via negociação entre os agentes e a plataforma de monitorização permitindo utilizar maior percentagem da largura de banda alocada para monitorização aos agentes ligados a equipamentos com maior volume de tráfego e maior frequência de variação dos volumes monitorizados, permitindo que agentes ligados a equipamentos com valores mais estáticos comuniquem apenas mais esporadicamente com a estação de monitorização, tal como descrito na secção seguinte.

Sempre que o tráfego gerado pela monitorização online através do recurso a agentes, sondas que impliquem tráfego extra ou outras formas de injecção de tráfego, cause alguma degradação na taxa de transferência da rede disponível aos utilizadores que não seja desprezável, é aconselhada a utilização da monitorização offline, retirando assim o tráfego de monitorização da rede controlada. Na monitorização offline todo o tráfego de monitorização é trocado entre os equipamentos e a estação de monitorização através de um caminho alternativo reservado a esse tráfego específico. Mesmo quando a influência do tráfego de monitorização é irrelevante é usual utilizar-se uma rede reservada, normalmente com recurso a uma VPN, para garantir a privacidade e segurança dos dados de monitorização, evitando assim que sejam interceptados por terceiros, ficando as informações recolhidas apenas disponíveis à estação de monitorização.

### 2.1.1 – Monitorização Activa com Recurso a Agentes

Um agente é um componente de software, tipicamente uma aplicação pequena, que se encontra residente no equipamento a monitorizar e que recolhe dados. Os dados recolhidos são então enviados para a estação de monitorização, quer em intervalos regulares por iniciativa do agente, ou então sempre que os mesmos são requisitados pelo software de coordenação a partir da estação de monitorização. Os agentes que não são meramente reactivos, ou seja, aqueles que possuem a capacidade de realizarem acções sozinhos são mais flexíveis, mas por outro lado exigem uma maior capacidade de processamento dedicada por parte do equipamento no qual estão colocados, o que pode

resultar numa diminuição do desempenho do equipamento ou serviço que estão a monitorizar. Esta solução baseada em agentes activos ou dinâmicos é escolhida por várias plataformas bem conhecidas existentes no mercado como a IBM Tivoli, HP Openview, BMC, Patrol e CA Unicenter.

Um agente reactivo, necessariamente estático pois não possui a capacidade de se alterar sob pena de alterar o seu comportamento para um não previsto pela plataforma de monitorização, é bastante mais leve em termos de processamento no equipamento cliente a ser monitorizado. A perda de flexibilidade de um agente dinâmico para um agente reactivo pode ser justificável sempre que exista impacto do processamento requerido pelo agente no desempenho do equipamento.

Numa solução típica baseada em agentes, o agente comunica com a estação de monitorização em intervalos pré-definidos, enviando os dados que recolhe a cada momento. No caso de algum dos valores não se encontrar dentro dos parâmetros determinados, é gerado um alerta pela plataforma de monitorização.

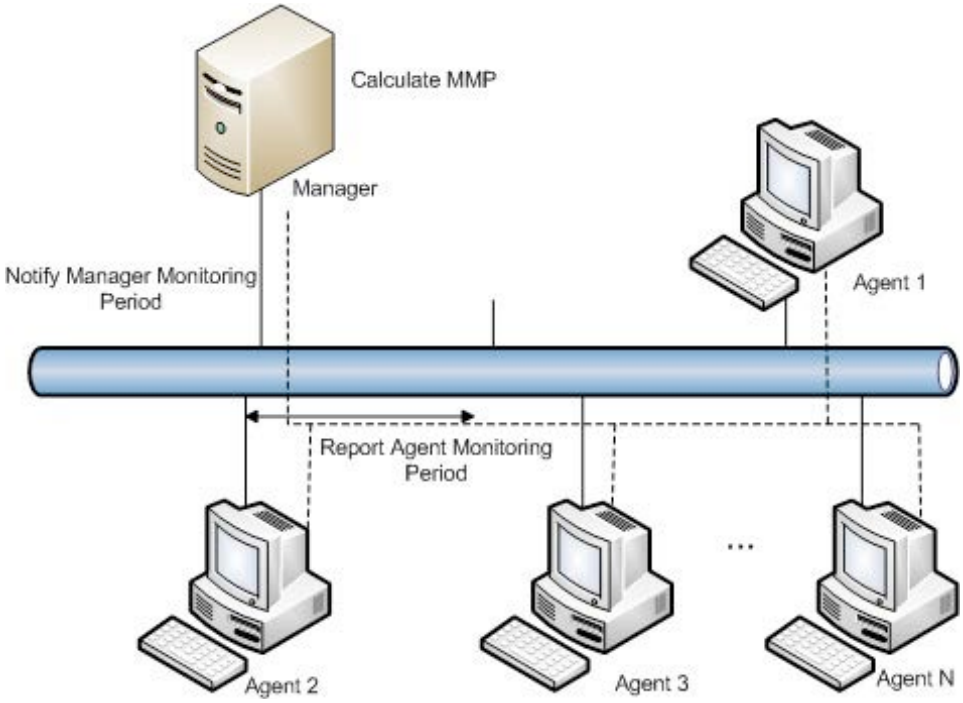



Figura 2.2 – Arquitectura de um sistema de monitorização activa baseado em agentes que negoceiam a gestão da monitorização com a plataforma localizada na estação de gestão.

Uma das maiores vantagens de se utilizar uma solução baseada em agentes é a granularidade dos dados que é possível de obter. Desta forma a plataforma de monitorização consegue estar informada sobre as condições dos servidores, do hardware presente na rede, dos processos a correr em cada um deles e da carga de processamento, memória ou espaço em disco. Isto traduz-se numa capacidade de baixar o tempo médio para reparação (MTTR) devido à informação atempada, aliada a uma melhor capacidade de planeamento da rede, melhor compreensão do comportamento dos sistemas e melhor afinação de desempenho[26].

No limite, os agentes podem ser complexos o suficiente para efectuarem acções no equipamento que estão a monitorizar, podendo por decisão autónoma, ou por indicação proveniente da estação de monitorização matar processos em curso ou apagar ficheiros antigos ou temporários em caso de escassez de recursos de processamento ou de espaço, por exemplo.

Um agente dinâmico pode também negociar com a plataforma de monitorização qual a frequência de actualização de dados por ter chegado à conclusão de que se encontra num sistema cujos dados tendem a ser constantes ao longo do tempo ou por se encontrar num ponto de rede com pouco tráfego, por exemplo. A plataforma de monitorização poderá então conceder ao agente autorização para mudar o seu comportamento.

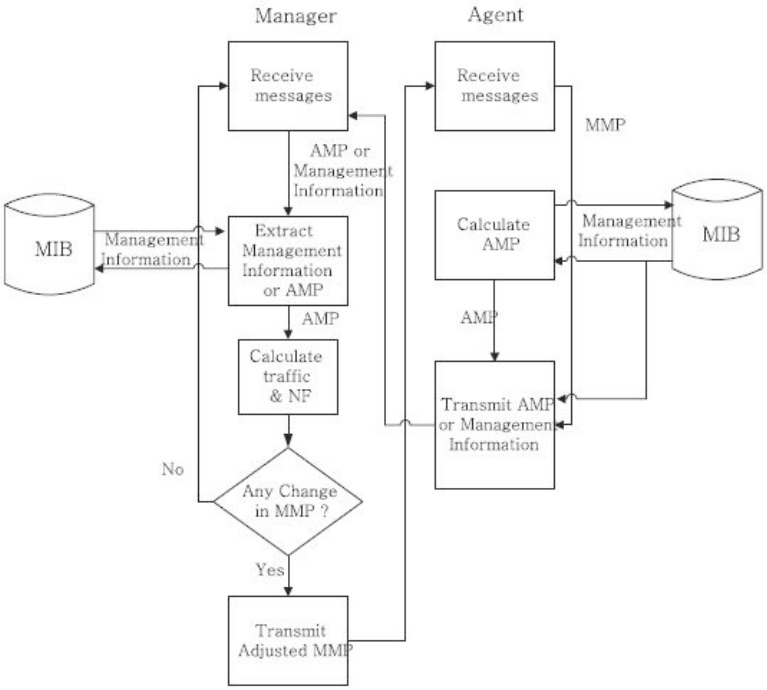



Figura 2.3 – Sequência de negociação entre os agentes e a plataforma de monitorização, dependendo da quantidade de tráfego junto a cada agente [32]

A utilização de agentes, no entanto, força à instalação do agente em todos os equipamentos a monitorizar, o que pode ser uma tarefa fastidiosa ou mesmo impossível quando na rede existam equipamentos sem implementação de SNMP e que igualmente não permitam a instalação de agentes próprios do software de monitorização. De igual forma parte-se do pressuposto que existirão agentes adequados ou compatíveis com todos os equipamentos a monitorizar, o que não é necessariamente verdade e, mesmo existindo, poderá não ser sempre possível obter autorização para a instalação de agentes em equipamentos considerados de mais alta segurança mesmo quando se utilizem protocolos de comunicação cifrados entre os agentes e a estação de monitorização.
Do ponto de vista financeiro, o tempo e complexidade exigidos por uma instalação e configuração de uma solução baseada em agentes podem por vezes ser um impeditivo à sua realização, dependendo de quais as necessidades em termos de granularidade de informação e tipos de dados exigidos pela empresa ou instituição.

### 2.1.2 – Monitorização Activa sem Recurso a Agentes

A monitorização activa sem recurso a agentes pode seguir uma aproximação baseada na *Application Programming Interface* (API) do serviço ou equipamento a ser monitorizado, ou através da análise a cada momento dos pacotes IP que circulam na rede[13], numa aproximação online à metodologia da monitorização passiva, apresentada mais adiante nesta secção. A análise aos pacotes que circulam numa rede é uma funcionalidade que costuma estar presente nas plataformas de monitorização activa, seja com ou sem recurso a agentes. Este tipo de análise não fornece indicações ou métricas detalhadas sobre os servidores ou equipamentos que estejam a fornecer os serviços, mas podem inferir-se disponibilidades e desempenhos, bem como análise da actividade dos utilizadores da rede.

A forma mais comum de monitorização sem recurso a agentes é através do uso de funcionalidades pré-instaladas pelos fabricantes dos equipamentos, e através das quais, usando um protocolo pré-definido, esses equipamentos podem ser inquiridos sem a necessidade de neles se instalar um qualquer agente externo da plataforma de monitorização. Os protocolos típicos para este tipo de implementação são o SNMP, RMON

ou NetFlow, instalados de origem em diverso equipamento de rede. Dos três referidos o mais comummente encontrado numa maior variedade de equipamentos é o SNMP. Apesar do SNMP possuir alguma variedade de dados sobre o qual pode fornecer informação, serão sempre em muito menor número que aqueles que poderiam ser obtidos numa solução baseada em agentes, que não está limitada por uma preocupação de uniformização entre equipamentos como é o caso do SNMP, embora essa limitação possa ser em parte mitigada pela implementação de campos específicos de cada fabricante no seu ramo da árvore *Management Information Base* (MIB), utilizada pelo SNMP. No entanto, a velocidade de desenvolvimento dos agentes de software é muito mais rápida que a actualização das APIs dos fabricantes.

A ausência de agentes externos, no entanto, implica que não são utilizados recursos extra do equipamento, sendo esta uma monitorização mais leve em termos de gasto de recursos. Existe assim um compromisso entre as funcionalidades disponíveis, o tipo e granularidade de informação recolhida, a interacção possível entre a estação de monitorização e os equipamentos de rede, e os recursos gastos pelos mesmos para as funções de monitorização da rede. A escolha da API a utilizar numa solução sem recurso a agentes deverá igualmente ter em conta preocupações de segurança no que toca à circulação de dados de monitorização sem cifra na mesma rede que o restante tráfego, sendo de considerar a criação de uma VLAN reservada ao tráfego de monitorização, utilizando o paradigma de monitorização offline anteriormente referido.

## 2.2 – Monitorização Passiva

Ao contrário da monitorização activa, a estratégia passiva distingue-se pela não introdução de qualquer tráfego de controlo na rede, evitando assim qualquer interferência com a mesma, o que poderia adulterar o cenário a ser avaliado, podendo mesmo no limite degradá-lo de forma a que um elevado volume de tráfego de monitorização possa causar uma diminuição sensível ao desempenho da rede a monitorizar.
Assim, a monitorização passiva recorre à amostragem de pacotes que circulam na rede e sua posterior análise, normalmente em offline, procurando assim ganhar um melhor entendimento do que se passa na troço em análise, em termos de utilização de serviços,

identificação de fluxos que circulam na rede, ou pacotes com conteúdo potencialmente malicioso ou ilegal[25][29].

Uma conclusão comum a vários estudos de monitorização [8] é a de que uma pequena percentagem de fluxos é responsável por uma elevada percentagem do total do tráfego[30]. Nomeadamente que 9% dos fluxos são responsáveis por 90% do tráfego circulante.

Os fluxos de tráfego são identificados por análise do cabeçalho dos pacotes, permitindo diferenciar quais os pacotes pertencentes a cada serviço computacional destrinçando assim cada fluxo de todos os restantes. Um fluxo é considerado relevante sempre que os pacotes IP ligados a ele ocupem uma determinada capacidade da largura de banda disponível durante mais do que determinado tempo. Estes valores são arbitrados de acordo com as especificidades de cada rede e das políticas de cada empresa ou instituição. Um fluxo pode ser considerado relevante se ocupar mais de 1% da largura de banda durante mais que 1 segundo, por exemplo, caso fosse esse o valor arbitrado para o conceito de relevante.

A monitorização passiva apresenta como principal vantagem o seu potencial de completude, ou seja, a análise pode ser tão completa consoante a frequência de amostragem de pacotes escolhida [1]. No limite, poderá efectuar-se uma amostragem total recolhendo todo o tráfego que passa na rede para posterior análise, garantindo assim um estudo completo e fiável dos fluxos e da utilização de serviços por parte dos utilizadores[5]. Este tipo de estudo permite detectar atempadamente tendências de utilização e crescimento de serviços, o que oferece a hipótese de reforçar o hardware ou software desses mesmos serviços antes dos mesmos sofrerem degradação de desempenho notória para os utilizadores. Por outro lado a recolha total de tráfego de uma rede facilmente se torna inexequível devido ao espaço de armazenamento requerido para o posterior processamento offline. Assim, uma das decisões mais importantes aquando da configuração de um sistema de monitorização passivo é a frequência de amostragem a ser utilizada. Por um lado, tem de ser uma frequência pequena o suficiente para manter o tamanho do conjunto de amostras abaixo da capacidade de armazenamento do sistema, bem como da própria capacidade de processamento do mesmo, visto que os resultados deverão ser conhecidos em tempo útil para a detecção de eventuais correcções a efectuar na rede. Por outro lado, deverá ser garantido que o número de amostras é suficiente para alcançar uma significância estatística relevante e que corresponde a uma imagem do que de facto se passa na rede.

Estando em posse de dados em quantidade relevante as ferramentas de monitorização passiva são especialmente apropriadas para análises comportamentais através do reconhecimento de padrões ou sequências de acções. Este tipo de análise, normalmente

implementada à custa de autómatos finitos deterministas pela sua simplicidade, mostra-se muitas vezes como um excelente aliado na detecção de comportamentos ilícitos, pois os mesmos normalmente seguem uma sequência conhecida de acções.

O facto de a análise ser feita a partir de dados armazenados faz com que as análises sejam repetíveis pois os dados estão guardados, logo mantêm-se inalterados permitindo que inclusivamente possam ser sujeitos a diferentes tipos de análise.

O armazenamento de dados é, por sua vez, a principal razão da má escalabilidade das soluções de monitorização passivas, pois quanto maior for o tráfego da rede a analisar maior será o número de amostras a recolher para manter a significância estatística da análise. O facto da possibilidade de armazenamento de todo o tráfego circulante na rede para análise tem igualmente levantado algumas reservas em relação à privacidade dos dados dos utilizadores, pois as amostras não fazem distinção entre conteúdos pessoais ou quaisquer outros.

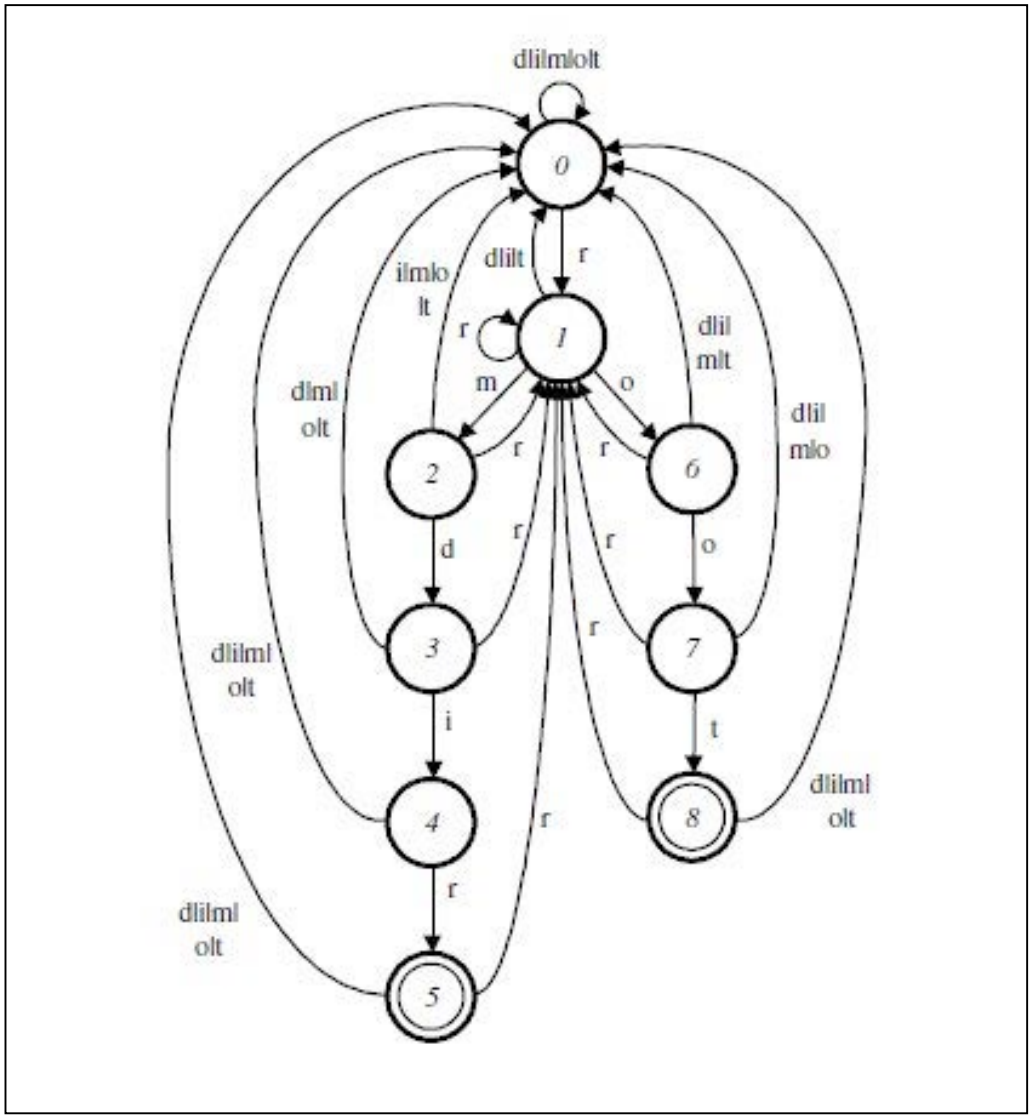



Figura 2.4 – Autómato Finito Determinista que efectua o reconhecimento das expressões "root" e "rmdir" [28]

As soluções de monitorização passiva enfermam ainda do facto de capturarem apenas dados de tráfego, não obtendo por isso acesso a informações do hardware de rede, como cargas de processadores, ocupação de espaço em disco ou qualquer outra informação que apenas se possa obter através de inquisição directa ao equipamento em questão.

# Capítulo 3

# Proposta de solução

Como proposta de solução serão estudadas e testadas várias plataformas de monitorização de redes informáticas, tendo em conta a sua lista de funcionalidades, desde que relevantes para os objectivos propostos, a actualização periódica da plataforma, como indicador de provável desenvolvimento continuado no futuro. Será também avaliado parque instalado ao nível de utilizadores, sendo que uma solução popular mais facilmente encontrará situações menos comuns devido à quantidade de situações diferentes que encontrará nos vários clientes, e da mesma forma, um número elevado de clientes terá uma maior probabilidades de encontrar falhas na aplicação que uma com um número reduzido de utilizadores.

A comparação incidirá igualmente sobre a facilidade de implementação e utilização diária da plataforma e do tempo dispendido a efectuar alterações à instalação ou configurações aquando de mudanças efectuadas na estrutura da rede, sendo dada preferência a uma plataforma que se revele mais flexível, garantindo assim uma maior probabilidade de que a mesma seja mantida actualizada ao longo da sua utilização futura.

Tendo em conta a actual composição a nível dos equipamentos a monitorizar, baseada em sistema operativo Unix e Suse Linux, a proposta a apresentar será uma capaz de correr nesse tipo de sistema, sendo assim descartadas à partida as soluções baseadas em Microsoft Windows, quer por não serem as mais apropriadas visto estarem essencialmente vocacionadas para a monitorização de servidores do mesmo sistema operativo, quer pelo custo associado às mesmas, existindo apenas soluções de base comercial.

As soluções preferidas serão assim baseadas em software de código aberto, nas suas versões "core", terminologia com que são referidas as versões gratuitas, sem pré configuração associada. Na actualidade, as soluções de código aberto mais populares

possuem também versões *Premium*, que são fornecidas já pré-configuradas de acordo com as necessidades do cliente, sendo cobrado por essa configuração um valor na linha do software comercial. Fará parte dos objectivos desta dissertação a configuração de uma solução "core" que responda às necessidades de monitorização do Sistema de Informação da FEUP (SiFEUP), constituído por 2 grupos de servidores de aplicações redundantes e balanceados que fornecem todos os serviços disponíveis aos utilizadores e três servidores de base de dados Oracle igualmente redundantes e balanceados.

Das várias propostas existentes no mercado, foram seleccionadas as mais populares para uma primeira avaliação, sendo o objectivo a criação de uma lista de pré-selecção constituída apenas pelas plataformas cuja lista de funcionalidades incluisse todas as necessárias que permitissem responder ao problema, sendo essas então incluídas no grupo a transitar para a fase de dissertação e instaladas na infra-estrutura de testes a ser criada para avaliação das mesmas.

Assim ficou a pré-selecção constituída pelas ferramentas Nagios, Zabbix, Zenoss e as ferramentas derivadas de Nagios, que expandem a sua eficiência ou funcionalidades, Shinken, Icinga, MK_Check, Op5Monitor e a sua versão de desenvolvimento Merlin. Da lista inicial considerada as ferramentas Ganglia, Spiceworks, Munin, OpenNMS e OPSView revelaram falhas ao nível das funcionalidades disponibilizadas ou da frequência de desenvolvimento, sendo essa por vezes a causa do primeiro item. A ferramenta cacti é bastante completa, mas é primordialmente um interface para a ferramenta RRDTool[33], sendo ainda muitas das operações forçosamente realizadas pelo operador directamente na base de dados RRDTool, com uma sintaxe complexa. Atendendo ao facto de que outras ferramentas como o Zenoss ou o Nagios são compatíveis com o formato RRDTool, podendo usar as vantagens da ferramenta, mas realizando as operações de forma transparente para o utilizador, foi dada preferência a estas últimas, que integram a ferramenta no seu interface, tanto a nível de base de dados, como de desenho de tráfego de rede. A ferramenta RRDTool apresenta-se como a sucessora mais completa do bem conhecido MRTG, sendo produzida pelo mesmo autor.

A ferramenta Nagios é a plataforma de monitorização mais difundida na actualidade com uma enorme comunidade de utilizadores e com uma panóplia excepcionalmente completa de funcionalidades disponível através da utilização de *plugins* desenvolvidos pelos utilizadores. Encontra-se disponível na versão "core" ou pré-configurada com o nome Nagios XI, sendo esta versão paga. É a plataforma de monitorização que invariavelmente se situa na posição cimeira dos inquéritos de satisfação realizados em vários *websites* da

área, sendo no entanto um mero indicador visto os inquéritos deste tipo carecerem de validade estatística ou científica.

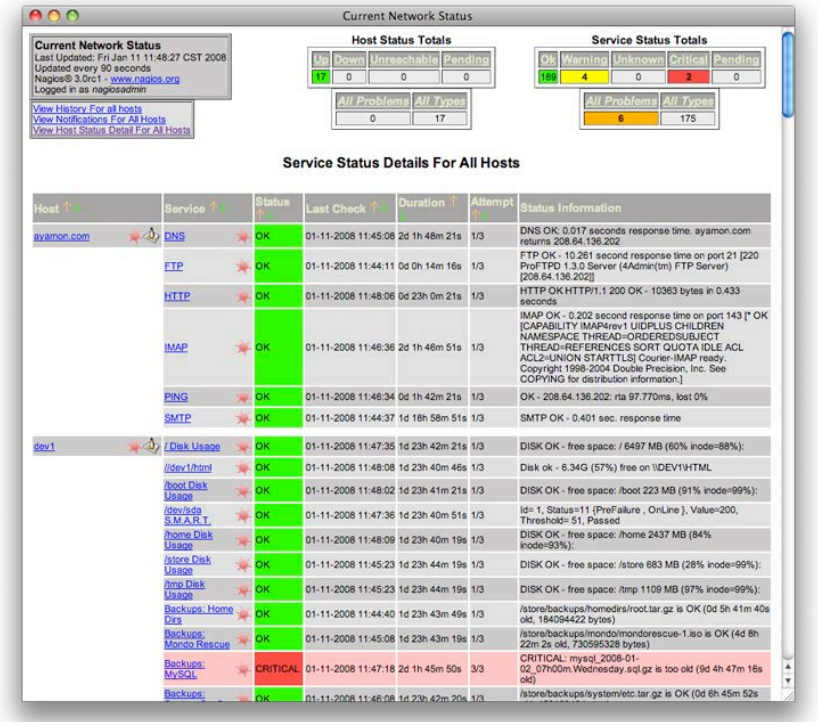



Figura 3.1 – Ecrã de monitorização de serviços da plataforma Nagios

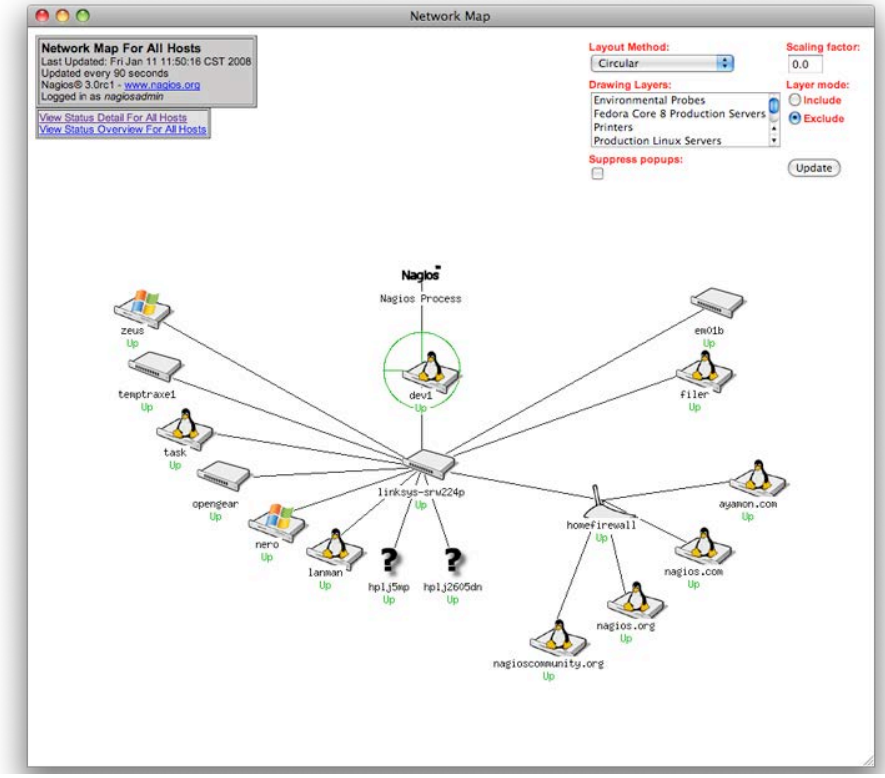



Figura 3.2 – Mapa de rede na plataforma Nagios [35]

Tal como o Nagios, a ferramenta Zabbix é bastante completa, mas eventualmente menos flexível na forma de configuração dos equipamentos a monitorizar. Não possui uma comunidade tão grande nem um desenvolvimento de funcionalidades por terceiros comparável ao do Nagios. No entanto, possui incorporadas várias das funcionalidades de *plugins* do Nagios, sendo uma ferramenta muito completa logo de origem.

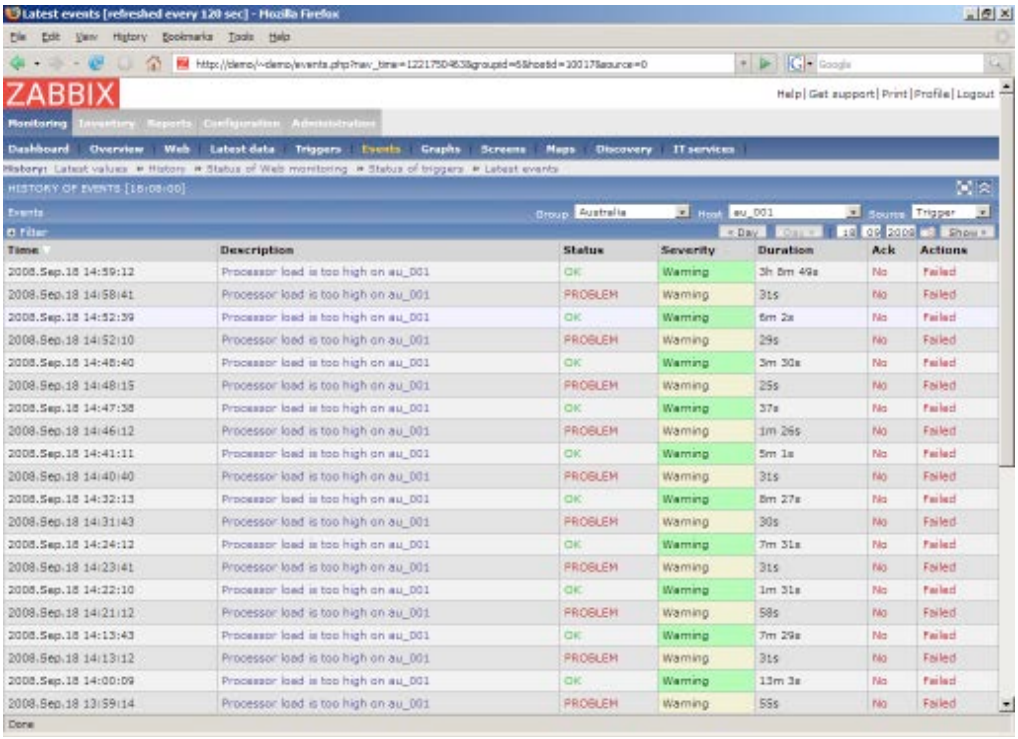



Figura 3.3 – Lista de ocorrências actuais na plataforma Zabbix

O Zenoss encontra-se nesta lista final de soluções a testar pois trata-se de uma ferramenta completa, de configuração fácil e visualmente agradável. Como desvantagem, não possui uma monitorização baseada em agentes, mas sim exclusivamente na utilização do protocolo SNMP. No entanto, o anúncio da versão 3 no *website* do Zenoss para uma data compreendida na duração desta dissertação fez com que a ferramenta continuasse nesta lista.

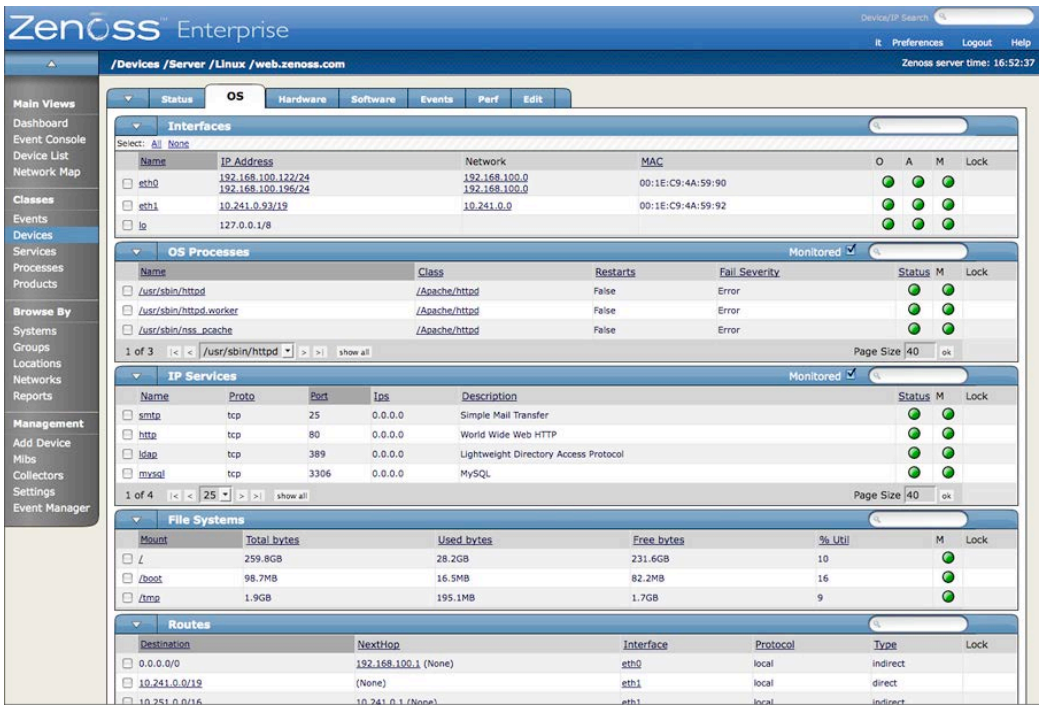



Figura 3.4 – Vista geral dos equipamentos na plataforma Zenoss

O Shinken mantém-se nesta lista de ferramentas a ser incorporada na rede de testes por interesse maioritariamente académico. Esta ferramenta é o Nagios reescrito em linguagem Python por um dos programadores do Nagios. Afirma-se como totalmente compatível com os ficheiro de configuração do Nagios e os seus *plugins*, mas detentor de um desempenho bastante superior. Desta forma a configuração do Shinken não deverá demorar grande tempo, sendo meramente replicadas as configurações que vierem a ser produzidas para o Nagios. Como impedimento de ser considerada como solução final está logo à partida o reconhecimento por parte do seu autor de que as falhas presentes no código não permitem que o Shinken seja usado ainda em ambientes de produção.

A ferramenta Check_mk, um acrescento ou *addon* na terminologia Nagios que altera o seu comportamento normal, utiliza agentes próprios mantendo-se, no entanto, retrocompatível com os agentes via NRPE nativos do Nagios. O Check_mk utiliza um paradigma de monitorização diverso do Nagios pois a estação de monitorização não invoca cada um dos agentes separadamente. Ao invés, o agente instalado no equipamento a monitorizar recolhe toda a informação de todos os *plugins* e envia-a periodicamente em conjunto, sendo a mesma imediatamente armazenada na base de dados RRDTools. Posteriormente o Nagios fará uma análise dos valores constantes na base de dados, em intervalos regulares, utilizando uma filosofia de monitorização passiva. Esta aproximação traz grandes vantagens em termos de poupança de recursos pois apenas é lançado um processo de actualização para cada equipamento independentemente do número de funções a monitorizar. A nova implementação Multisite_mk adiciona ao Check_mk a capacidade de monitorização distribuída.

O OP5Monitor é a única solução comercial a ser avaliada no curso desta dissertação. Tal sucede pois existe uma versão gratuita, limitada à monitorização de até vinte endereços IP e sem admitir a utilização de quaisquer extensões, mas que poderá servir de demonstração das capacidades de uma plataforma baseada em motor de processamento Merlin com interface gráfico Ninja.

A OP5 disponibiliza a quem queira construir manualmente uma solução semelhante ao OP5Monitor os pacotes de *software* nos quais o mesmo é baseado. O processamento é realizado por um motor denominado Merlin, acrónimo de *Module for Effortless Redundancy and Loadbalancing In Nagios* que, tal como o nome indica, fornece ao Nagios capacidades de monitorização distribuída, estando a interface gráfica a cargo do Ninja, acrónimo de *Nagios Is Now Just Awesome*. O desenvolvimento da OP5 assenta primordialmente em PHP, conseguindo assim melhor desempenho que os CGIs base do Nagios.

Por último, o Icinga é um desenvolvimento de Nagios que resulta do descontentamento de vários membros da comunidade de desenvolvimento de extensões, com vários dos quais ligados ao meio académico. Devido ao lento desenvolvimento da solução base e do que foi percepcionado como um certo desinteresse da Nagios Enterprises em avançar rapidamente com o desenvolvimento do produto Nagios Core, surgiu o Icinga. Trata-se de uma versão de Nagios compatível com todos os *plugins* existentes, mas que adiciona a capacidade de monitorização distribuida, bem como uma nova interface baseada em tecnologias *Ajax* e *javascript* procurando assim introduzir maior rapidez e flexibilidade de utilização.

Todas as soluções anteriores são capazes de ajudar o gestor de rede a efectuar uma análise de tendências de utilização. Esse tipo de vista é actualmente de particular importância, pois a utilização das redes começa, cada vez mais, a ser feita com base em clientes sem fios. Estudos actuais comprovam que o padrão de utilização em termos de acesso a conteúdos se altera quando o utilizador acede à rede não a partir de um terminal da instituição, mas sim de um computador portátil pessoal ou outra solução móvel, como um *smartphone* [39]. Detecta-se que existe uma muito maior utilização de serviços sensíveis à variação do atraso (*jitter)*, sendo a utilização de largura de banda dominada pelo *YouTube* e outras emissões em *streaming*.

Nos dados do primeiro semestre de 2010 para a utilização de internet com origem em dispositivos móveis é possivel verificar-se essa tipificação do uso. Conquanto os utilizadores intensivos, responsáveis por cerca de 50% do tráfego móvel, sejam apenas 5% dos utilizadores e privilegiem a utilização de protocolos de partilha de ficheiros *peer2peer*, quando os restantes utilizadores são contabilizados verifica-se que os serviços de visualização de vídeos, nomeadamente o *YouTube,* passam a representar um fluxo superior ao da partilha de ficheiros, sendo um tipo de serviço particularmente sensível ao atraso. Espera-se que, ao longo dos próximos 3 anos, a visualização de serviços de vídeo venha a representar 66% do tráfego consumido pelos terminais móveis. [42]

Na tabela seguinte podem ser observados os dados relativos à utilização de internet móvel na Europa em 2009, quer contabilizando apenas os 5% de utilizadores intensivos, quer a utilização normalizada com a contabilização de todos os utilizadores. Trata-se da utilização média e não em hora de pico de utilização, caso em que um utilizador intensivo pode usar até 72% do seu tráfego dedicado à partilha de ficheiros, registando o tráfego normalizado um uso de 9% para redes sociais.

| Europe | | | |
|---|---|---|---|
| Normalized | | Top User | |
| Category | Percent of Traffic | Category | Percent of Traffic |
| Web Browsing | 32.6% | P2P Filesharing | 38.0% |
| Real-Time Entertainment | 31.1% | Web Browsing | 27.5% |
| P2P Filesharing | 13.5% | Real-Time Entertainment | 20.3% |
| Software Updates | 5.2% | Social Networking | 3.5% |
| Social Networking | 3.9% | Secure Tunneling | 2.4% |
| Outside Top 5 | 13.8% | Outside Top 5 | 8.3% |

Tabela 3.1 — Utilização móvel na Europa por tipo de tráfego (média). [39]

Com o aumento da utilização de *Smartphones* e computadores portáteis na rede da FEUP é de esperar que as tendências de utilização se comecem a aproximar das registadas numa numa rede generalista, com componente significativa de utilizadores móveis. Apesar de tratar-se de uma rede académica, mantendo assim as suas idiossincrasias próprias, há hoje, devido à aposta feita nos últimos anos de desmaterializar o acesso à rede, procurando privilegiar o acesso móvel com recurso a uma rede única como a *eduroam* e programas de acesso bonificado a portáteis, uma cada vez maior utilização da rede da FEUP por via exclusivamente móvel, e assim sujeita a padrões de utilização diferentes dos anteriormente observados.

Esta mudança do paradigma de utilização causada por uma maior sensação de privacidade concedida quer pela utilização em locais mais recatados quando comparados aos laboratórios de informática / salas de PCs, quer pelo facto dos ficheiros ficarem guardados no seu computador pessoal, faz com que a utilização seja muito mais aproximada à que o sujeito faz em casa, especialmente em termos de partilha de ficheiros e visualização de vídeos. [40]

Em termos de gestão da rede informática significa que essas novas tendências de utilização necessitam de ser identificadas, de forma a que possam executar-se as necessárias alterações à rede para favorecer a entrega com qualidade dos serviços mais requisitados, ou para proceder à implementação de políticas de gestão de rede que

procurem manter a qualidade de serviço do tráfego normalizado em detrimento dos utilizadores intensivos que possam estar a fazer uma utilização menos adequada da rede académica. Desta forma a capacidade de identificação de fluxos bem como da análise de tendências assumem especial relevância. Tal como referido, todas as ferramentas de monitorização consideradas encontram-se aptas a realizar essas funções.

# Capítulo 4

# Implementação

Neste capítulo serão descritas as especificidades da rede de testes e desenvolvimento criada para a análise das várias soluções estudadas. A cada uma das soluções será dada uma pequena introdução que a contextualize e justifique a sua inclusão neste lote de possibilidades avaliadas.

São posteriormente apresentadas apenas as imagens consideradas estritamente necessárias para a compreensão do ambiente da ferramenta e da sua distinção perante as demais. Imagens adicionais bem como os ficheiros de configuração principais de cada uma das soluções, quando existam, são apresentados na secção de anexos. Desta forma procura-se que esta secção não se torne excessivamente longa.

Todas as ferramentas de monitorização foram compiladas localmente a partir da última versão disponível do código fonte para uma garantia de melhor desempenho e maior quantidade de funcionalidades.

## 4.1 – Ambiente de Desenvolvimento e Testes

Para a avaliação das várias soluções foi implementada uma rede de testes que procurou incluir todos os serviços constantes do ambiente de produção para o qual a solução é

pretendida, de forma a poder aferir das capacidades de cada uma das propostas no respeitante à monitorização de um conjunto razoavelmente completo de parâmetros.

Para que as comparações das várias soluções fossem executadas em condições semelhantes foram instaladas e configuradas em ambientes de *hardware* e *software* semelhantes, sendo esse propósito conseguido através da utilização de máquinas virtuais com definições idênticas. Deu-se preferência à instalação da distribuição OpenSUSE Linux como sistema operativo devido à utilização da versão Enterprise da SUSE nos servidores em operação no CICA, sendo assim uma eventual migração da solução para o ambiente de produção bastante facilitada, bastando para tanto a replicação dos passos de instalação e configuração efectuados na máquina de testes e avaliação. As soluções foram igualmente testadas em distribuição Ubuntu Linux devido à actual popularidade do mesmo, sendo a instalação existente em várias das estações de trabalho pessoais no CICA. Uma das soluções levou também à instalação da distribuição CentOS devido a especificidades próprias que serão abordadas mais à frente neste capítulo na secção dedicada à solução baseada em Merlin e Ninja.

De forma a garantir um ambiente heterogéneo a ser monitorizado e assim aferir também da universalidade das soluções em estudo, a rede de testes incluiu também uma estação de trabalho com sistema operativo Windows 7, sendo avaliadas as capacidades de monitorização da mesma por parte das várias soluções candidatas. De notar, no entanto, que este ponto de avaliação não se encontra nos requisitos apresentados e como tal o seu resultado é de bastante menor peso que os resultados dos pontos constantes da tabela 1.1.

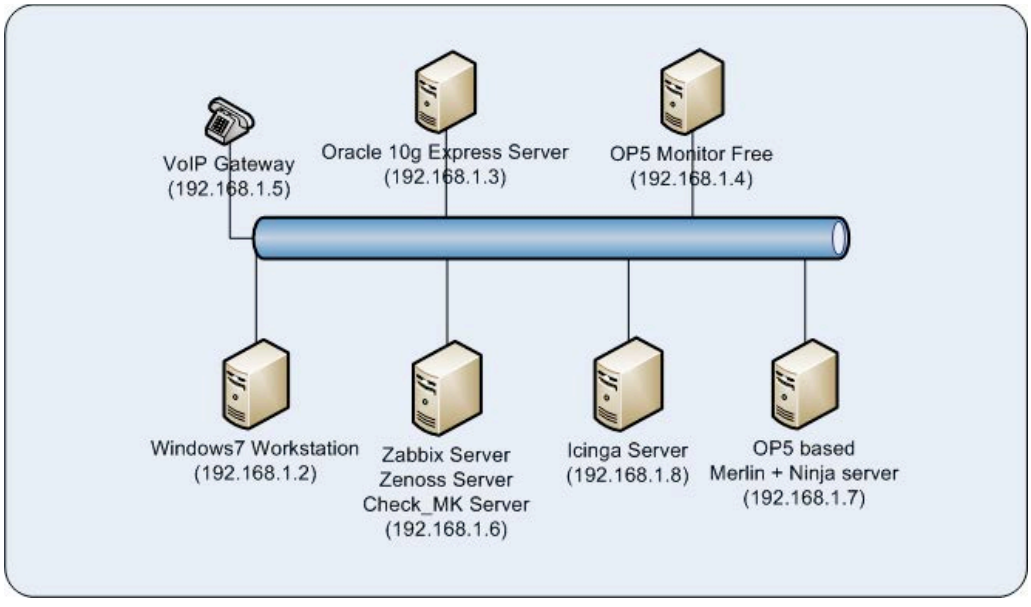



Figura 4.1 – Rede de Desenvolvimento e Testes

Os elementos da rede de testes são apresentados na figura 4.1. Cada um dos servidores de monitorização foi simultaneamente alvo das soluções a correr nos restantes. A estação de trabalho Windows, bem como a *gateway* telefónica VoIP e o servidor Oracle Express foram apenas alvos de monitorização.

As máquinas virtuais foram criadas ou executadas em Parallels Desktop 5 para Mac OS X Snow Leopard, com excepção da máquina virtual do OP5 Monitor Free, sendo aconselhada pela empresa a utilização do VMPlayer devido ao formato em que a mesma é disponibilizada.

A tabela 4.1 sintetiza as características base de todos os equipamentos ligados à rede de testes.

| Equipamento | Hardware | CPU | Memória | Sistema Operativo |
|---|---|---|---|---|
| 192.168.1.2 | S/ Marca | Single Core 3.0GHz | 1GB | Windows 7 Professional 32 bit |
| 192.168.1.3 (MV) | Máquina Virtual Parallels Desktop 5.0.9376 | Core2 Duo 2.66GHz | 2GB | Oracle 10g Express Server 1.1 (Oracle Linux) |
| 192.168.1.4 (MV) | Máquina Virtual VMWare Player 3.1.4 | Core2 Duo 2.66GHz | 2GB | CentOS 5.6 |
| 192.168.1.5 | Worldcom SIP Gateway WG4K | - | - | OjoLabs |
| 192.168.1.6 (MV) | Máquina Virtual Parallels Desktop 5.0.9376 | Core2 Duo 2.66GHz | 2GB | Open SUSE 11.4 |
| 192.168.1.7 (MV) | Máquina Virtual Parallels Desktop 5.0.9376 | Core2 Duo 2.66GHz | 2GB | CentOS 5.6 |
| 192.168.1.8 (MV) | Máquina Virtual Parallels Desktop 5.0.9376 | Core2 Duo 2.66GHz | 2GB | Open SUSE 11.4 |

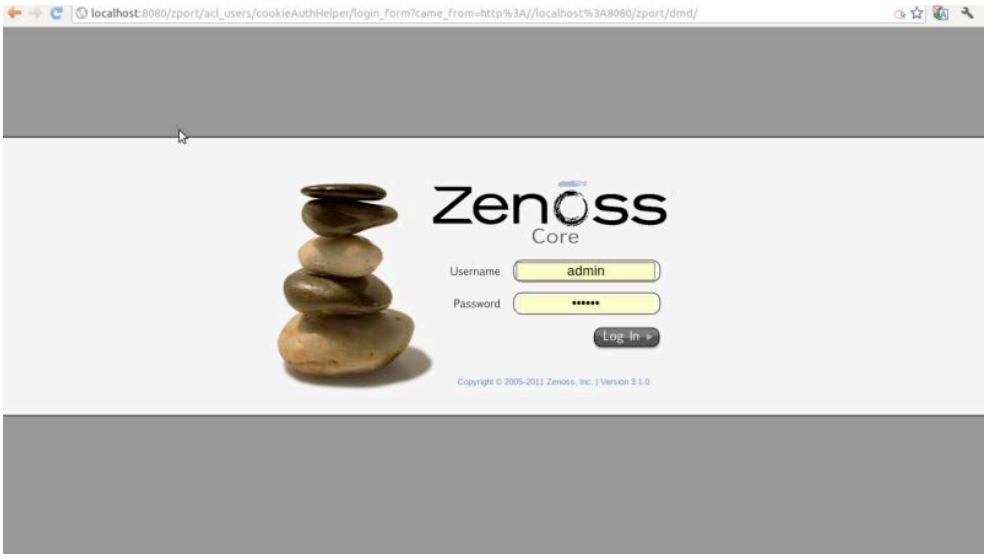
Tabela 4.1 — Equipamentos da rede de teste

Todas as soluções colocadas em máquinas virtuais com o sistema operativo Open SUSE foram igualmente instaladas em máquinas virtuais a correr Ubuntu 11.04 para recolha de configurações devido à utilização da distribuição em causa em várias estações de trabalho no CICA-FEUP e consequente possibilidade de poderem vir a ser utilizadas nesse ambiente operativo.

## 4.2 – Análise de Soluções

### 4.2.1 – Zenoss

A plataforma Zenoss, na sua vertente Enterprise tem vindo a ganhar quota de mercado nos últimos anos, tendo conquistado para o seu portefólio de clientes de monitorização as redes de empresas reconhecidas, quer pela sua dimensão, quer pela criticidade de alguns serviços que operam, os quais não se coadunam com tempos de inoperacionalidade dos mesmos. Nomes como a Motorola, Deutsche Bank, Linked In, Sugar CRM ou VMWare




A principal desvantagem da plataforma Zenoss em relação a outras soluções concorrentes assentava no facto de ambas as versões, mas em particular a sua versão Core, dependerem quase exclusivamente do protocolo SNMP para as suas acções de monitorização. Durante o decorrer desta dissertação foi lançada a versão 3 do Zenoss, que reforçou o movimento em direcção a uma maior utilização de outras formas de monitorização passando também para a versão core a utilização mais ampla dos pacotes de configuração do Zenoss denominados zpacks e a utilização de ligações aos alvos de

monitorização via telnet ou SSH, efectuando login nos mesmos e correndo comandos, utilizando a resposta desses comandos para a recolha de dados de monitorização. A estrutura de monitorização actual do Zenoss, com recurso a SNMP, zpacks e zcommands na camada de recolha de dados é a que se pode verificar na figura 4.3.

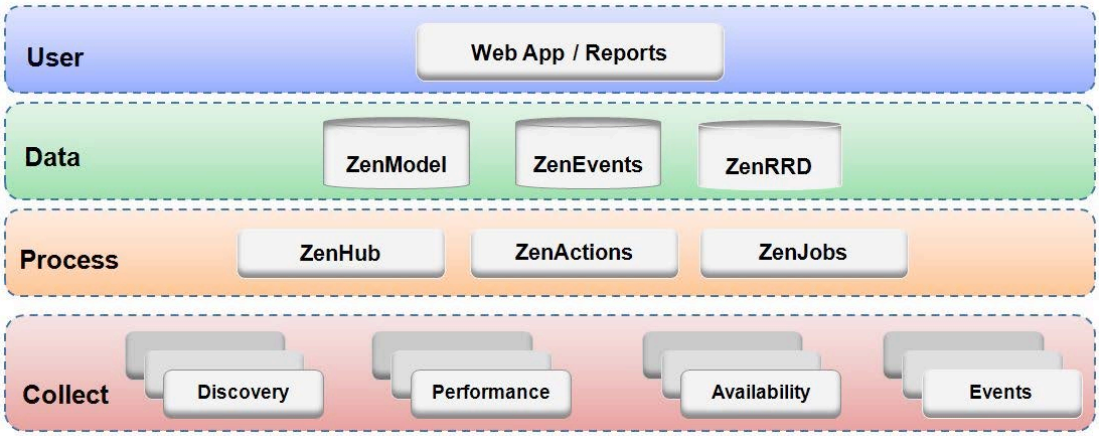



Figura 4.3 – Arquitectura da Solução Zenoss [36]

A arquitectura do Zenoss segue a típica solução por camadas, estando a camada inferior responsável pela recolha dos dados através de SNMP ou zcommands. A camada de processamento efectua a comunicação entre a camada de recolha e a de tratamento de dados, estando esta última responsável pelo armazenamento dos dados em base de dados compatível com RRD e processamento dos mesmos, passando os resultados à camada superior de visualização e interacção com o utilizador. É também responsável por manter o modelo da rede e o histórico de eventos. A camada de utilizador é responsável por toda a interacção gráfica e pela criação e processamento de relatórios.

O Zenoss destaca-se por ser um software de fácil implementação e configuração, com um interface gráfico agradável, permitindo ao administrador realizar todas as funções de adição de equipamentos a monitorizar a inclusão de zpacks ou zcommands sem necessitar nunca de editar quaisquer ficheiros de configuração. O esquema de utilização e localização de menus requer alguma habituação, mas não se poderá considerar como ponto determinante na avaliação da solução.

Ainda do ponto de vista da configuração, incluem-se entre os aspectos positivos a capacidade de descoberta de equipamentos ligados à rede local, bem como a sua fácil afectação a *templates* pré-existentes para um conjunto alargado de fabricantes e modelos. Tudo isto é feito através do interface gráfico da ferramenta, tornando a inclusão de novos equipamentos a monitorizar um processo simples e rápido, tal como está ilustrado nas figuras 4.4 e 4.5.

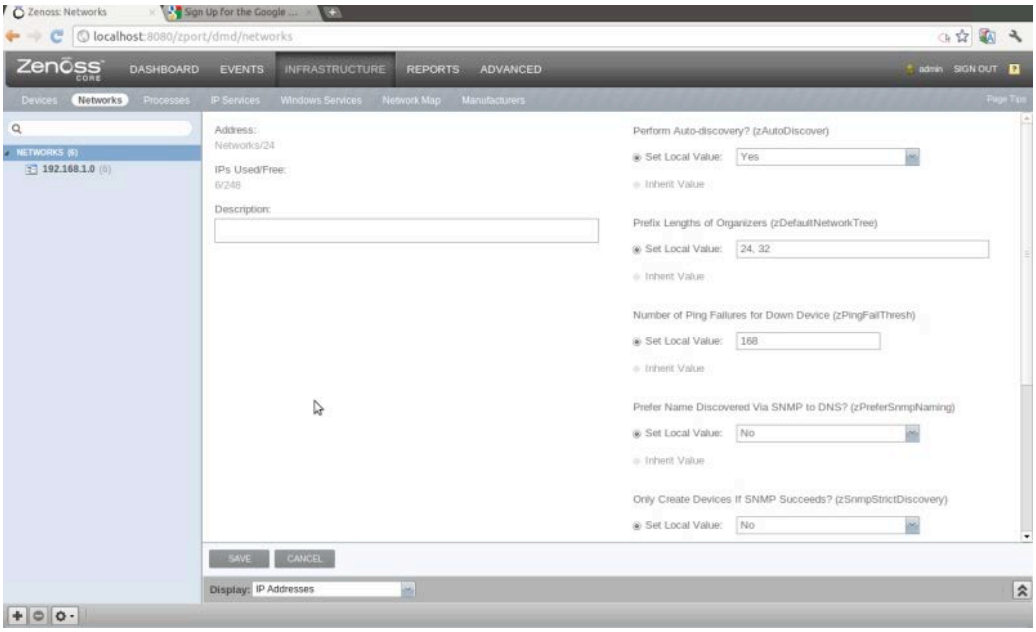



Figura 4.4 – Zenoss – Descoberta da rede local

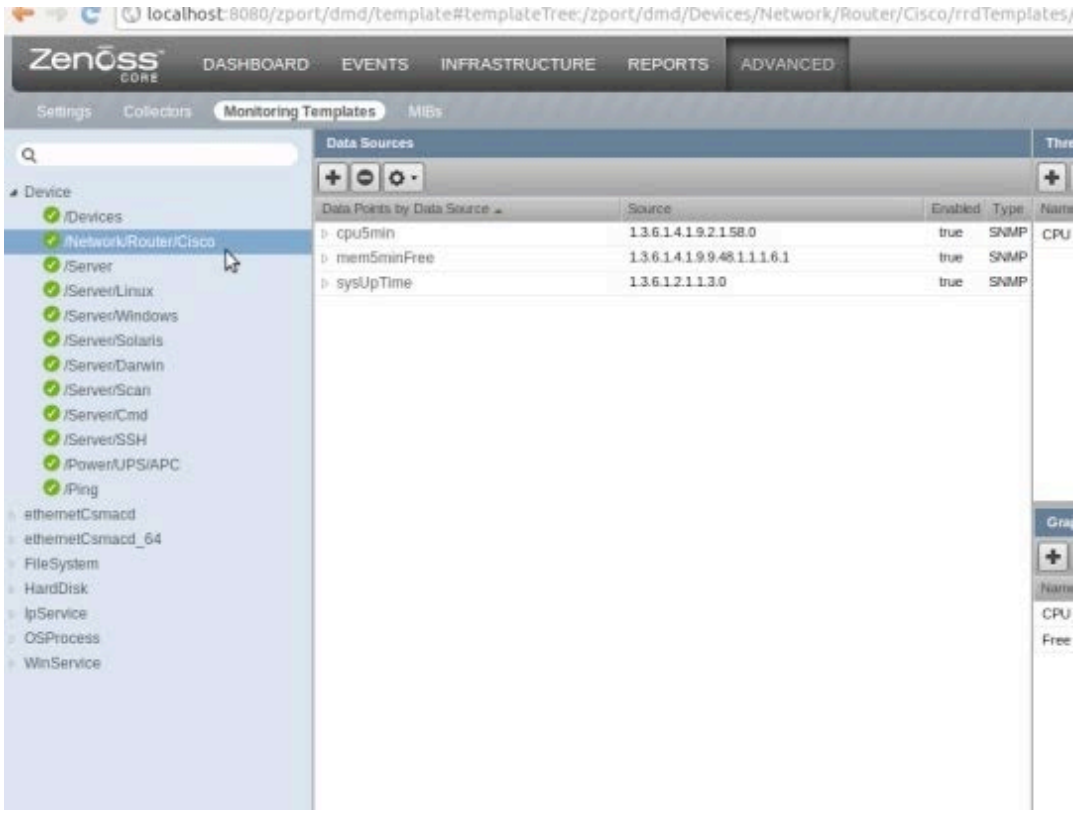
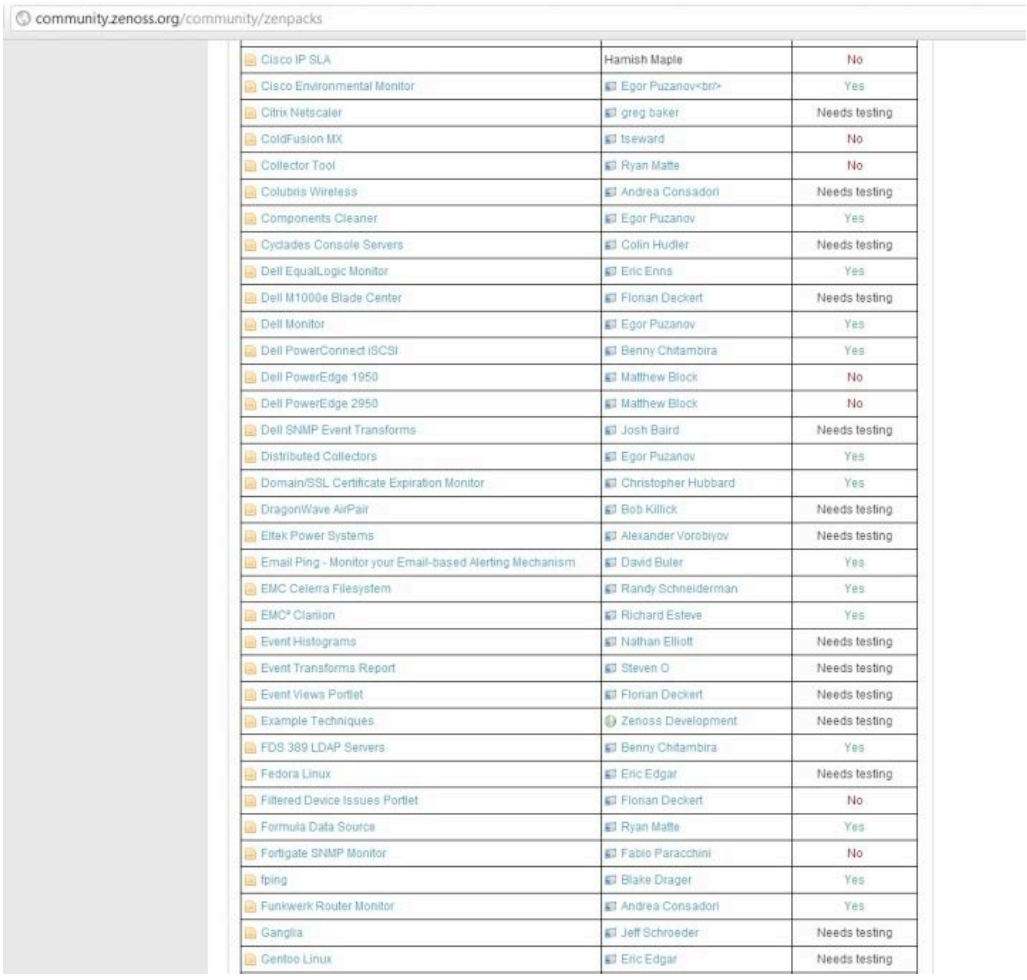



Figura 4.5 – Zenoss – Templates para equipamento diverso

No entanto, e apesar da versão 3 do Zenoss Enterprise ter surgido dotada de várias capacidades e funcionalidades acrescidas em relação à versão 2.5 anterior, esta evolução

marcou também um maior distanciamento entre as capacidades da versão Enterprise comercial em relação à versão Core gratuita. Assim, ao passo que a versão enterprise conta com um conjunto alargado de zpacks e zcommands que lhe permitem a monitorização de um conjunto bastante completo de serviços incluindo, por exemplo, o novo Cisco Unified Computing System (UCS), a versão Core vê-se com apenas vinte dois zpacks oficiais. Aliado a esta diferença, os zpacks existentes feitos pela comunidade de utilizadores para a versão 2.5 têm vindo a sofrer de um grande atraso na sua transposição para versão 3, o que deixa a versão Core de momento sem muitas das funcionalidade oferecidas não só pela versão Enterprise, como por outros concorrentes em código aberto sem custos.




| | | |
|---|---|---|
| Cisco IP SLA | Hamish Maple | No |
| Cisco Environmental Monitor | Egor Puzanov<br/> | Yes |
| Citrix Netscaler | greg baker | Needs testing |
| ColdFusion MX | tseward | No |
| Collector Tool | Ryan Matte | No |
| Colubris Wireless | Andrea Consadori | Needs testing |
| Components Cleaner | Egor Puzanov | Yes |
| Cyclades Console Servers | Colin Hudler | Needs testing |
| Dell EqualLogic Monitor | Eric Enns | Yes |
| Dell M1000e Blade Center | Florian Deckert | Needs testing |
| Dell Monitor | Egor Puzanov | Yes |
| Dell PowerConnect iSCSI | Benny Chitambira | Yes |
| Dell PowerEdge 1950 | Matthew Block | No |
| Dell PowerEdge 2950 | Matthew Block | No |
| Dell SNMP Event Transforms | Josh Baird | Needs testing |
| Distributed Collectors | Egor Puzanov | Yes |
| Domain/SSL Certificate Expiration Monitor | Christopher Hubbard | Yes |
| DragonWave AirPair | Bob Killick | Needs testing |
| Eltek Power Systems | Alexander Vorobiyov | Needs testing |
| Email Ping - Monitor your Email-based Alerting Mechanism | David Buler | Yes |
| EMC Celerra Filesystem | Randy Schneiderman | Yes |
| EMC² Clariion | Richard Esteve | Yes |
| Event Histograms | Nathan Elliott | Needs testing |
| Event Transforms Report | Steven O | Needs testing |
| Event Views Portlet | Florian Deckert | Needs testing |
| Example Techniques | Zenoss Development | Needs testing |
| FDS 389 LDAP Servers | Benny Chitambira | Yes |
| Fedora Linux | Eric Edgar | Needs testing |
| Filtered Device Issues Portlet | Florian Deckert | No |
| Formula Data Source | Ryan Matte | Yes |
| Fortigate SNMP Monitor | Fabio Paracchini | No |
| fping | Blake Drager | Yes |
| Funkwerk Router Monitor | Andrea Consadori | Yes |
| Ganglia | Jeff Schroeder | Needs testing |
| Gentoo Linux | Eric Edgar | Needs testing |

Figura 4.6 – Zenoss – Disponibilidade de Zpacks comunitários para V.3 [36]

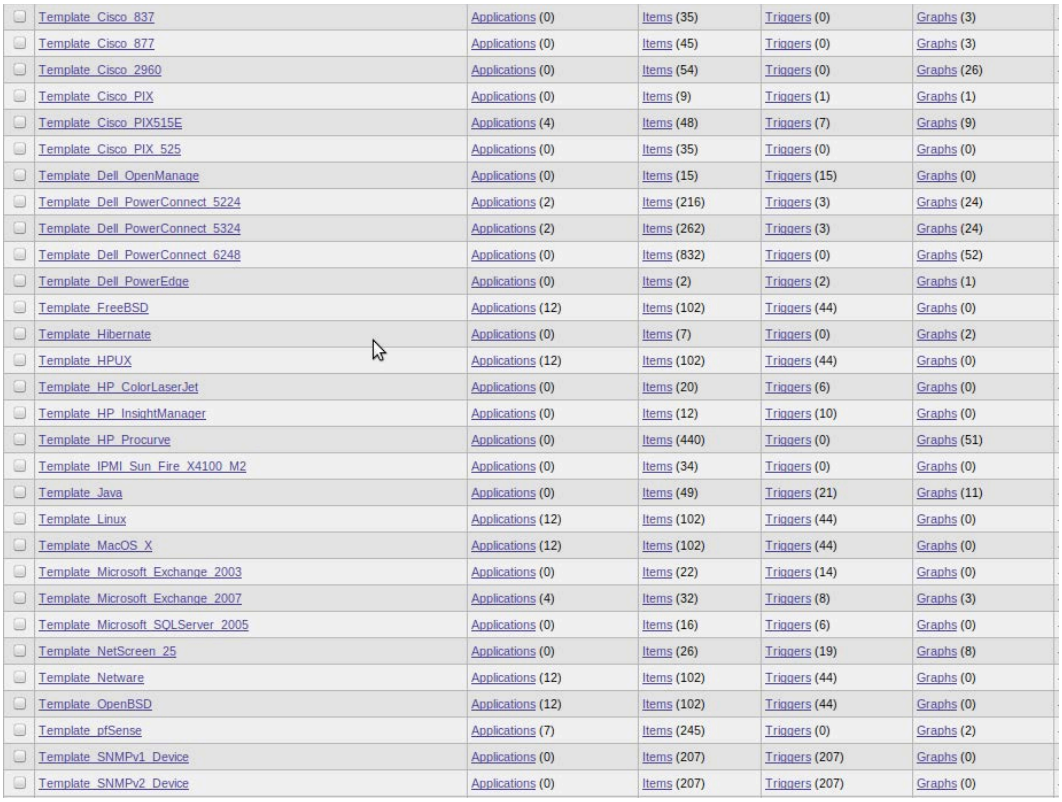
Dentro das possibilidades oferecidas pela plataforma Zenoss não se encontrou nenhuma que oferecesse uma solução aceitável para a monitorização de bases de dados Oracle. A criação de zcommands para o efeito tampouco se revelou uma boa aposta pois a própria

tecnologia inerente à comunicação de zcommands utiliza mais recursos que tecnologias de soluções concorrentes, onde foram encontradas soluções apropriadas. Desta forma o estudo desta plataforma não foi mais aprofundado. Não foram notadas quaisquer diferenças de desempenho nas instalações de Zenoss Core 3 em Ubuntu e OpenSuse.

### 4.2.2 – Zabbix

A ferramenta Zabbix apresenta-se, na sua forma base, como uma solução bastante completa. Após a criação da base de dados MySQL necessária ao seu funcionamento, à configuração do interpretador PHP com valores compatíveis com o Zabbix e posterior instalação, tudo o resto é feito em ambiente gráfico no interior da ferramenta.

O Zabbix confia nos seus agentes, que têm de ser instalados em todos os alvos de monitorização e no próprio servidor para a recolha e envio de todas as informações necessárias. O Zabbix possui um número bastante elevado de *templates* para várias marcas e modelos de equipamento bem como para uma variedade considerável de sistemas operativos e distribuições. Seleccionando o template correcto o Zabbix irá pedir ao agente instalado remotamente as informações mais relavantes para o equipamento escolhido. Os equipamentos podem ser adicionados tanto manualmente como através de busca automática na rede local por parte do Zabbix. No caso da busca automática cabe ao administrador a correcção ou afectação de novo *template* a cada equipamento.



| | | | | | |
|---|---|---|---|---|---|
| Template_Cisco_837 | Applications (0) | Items (35) | Triggers (0) | Graphs (3) | |
| Template_Cisco_877 | Applications (0) | Items (45) | Triggers (0) | Graphs (3) | |
| Template_Cisco_2960 | Applications (0) | Items (54) | Triggers (0) | Graphs (26) | |
| Template_Cisco_PIX | Applications (0) | Items (9) | Triggers (1) | Graphs (1) | |
| Template_Cisco_PIX515E | Applications (4) | Items (48) | Triggers (7) | Graphs (9) | |
| Template_Cisco_PIX_525 | Applications (0) | Items (35) | Triggers (0) | Graphs (0) | |
| Template_Dell_OpenManage | Applications (0) | Items (15) | Triggers (15) | Graphs (0) | |
| Template_Dell_PowerConnect_5224 | Applications (2) | Items (216) | Triggers (3) | Graphs (24) | |
| Template_Dell_PowerConnect_5324 | Applications (2) | Items (262) | Triggers (3) | Graphs (24) | |
| Template_Dell_PowerConnect_6248 | Applications (0) | Items (832) | Triggers (0) | Graphs (52) | |
| Template_Dell_PowerEdge | Applications (0) | Items (2) | Triggers (2) | Graphs (1) | |
| Template_FreeBSD | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_Hibernate | Applications (0) | Items (7) | Triggers (0) | Graphs (2) | |
| Template_HPUX | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_HP_ColorLaserJet | Applications (0) | Items (20) | Triggers (6) | Graphs (0) | |
| Template_HP_InsightManager | Applications (0) | Items (12) | Triggers (10) | Graphs (0) | |
| Template_HP_Procurve | Applications (0) | Items (440) | Triggers (0) | Graphs (51) | |
| Template_IPMI_Sun_Fire_X4100_M2 | Applications (0) | Items (34) | Triggers (0) | Graphs (0) | |
| Template_Java | Applications (0) | Items (49) | Triggers (21) | Graphs (11) | |
| Template_Linux | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_MacOS_X | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_Microsoft_Exchange_2003 | Applications (0) | Items (22) | Triggers (14) | Graphs (0) | |
| Template_Microsoft_Exchange_2007 | Applications (4) | Items (32) | Triggers (8) | Graphs (3) | |
| Template_Microsoft_SQLServer_2005 | Applications (0) | Items (16) | Triggers (6) | Graphs (0) | |
| Template_NetScreen_25 | Applications (0) | Items (26) | Triggers (19) | Graphs (8) | |
| Template_Netware | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_OpenBSD | Applications (12) | Items (102) | Triggers (44) | Graphs (0) | |
| Template_pfSense | Applications (7) | Items (245) | Triggers (0) | Graphs (2) | |
| Template_SNMPv1_Device | Applications (0) | Items (207) | Triggers (207) | Graphs (0) | |
| Template_SNMPv2_Device | Applications (0) | Items (207) | Triggers (207) | Graphs (0) | |

Figura 4.7 – Templates Zabbix

Após a adição das entidades de rede a monitorizar grande parte das funcionalidades do Zabbix fica activa. O Zabbix tem vindo historicamente a evoluir no sentido de uma maior facilidade de utilização não prescindindo, no entanto, de continuar a ser uma ferramenta bastante poderosa. Possui capacidade própria de geração de gráficos complexos e cálculo de SLA e outros relatórios sem necessidade de recurso a soluções de terceiros.

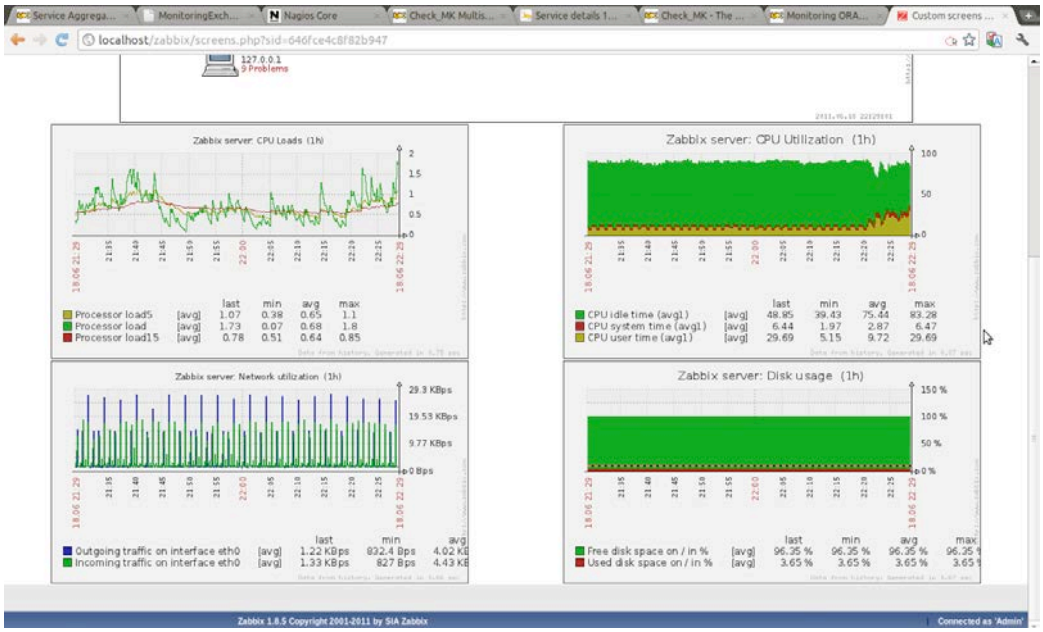



Figura 4.8 – Gráficos em Zabbix

No entanto, essa mesma independência faz com que o Zabbix possua não muito mais que as capacidades que nele são incluídas pela SIA Zabbix, empresa responsável pelo seu desenvolvimento. O envolvimento da comunidade é primordialmente limitado ao teste de versões beta e proposta de correcções para defeitos existentes. Existem alguns plugins não oficiais que são directamente introduzidos no código da ferramenta, mas não são suportados pela SIA Zabbix e tendem a sofrer de problemas com o lançamento de novas versões do software. Um dos plugins de facto reconhecidos é o Orabbix que permite a monitorização de bases de dados Oracle. Uma nova versão do Orabbix totalmente compatível com a actual versão do zabbix será lançada muito brevemente. Até lá não será considerado o zabbix como solução, apesar de ser um forte candidato nas restantes áreas. Não foram detectadas quaisquer diferenças de desempenho entre as distribuições Ubuntu e OpenSuse.

### 4.2.3 – Nagios e derivados

O Nagios é a ferramenta de monitorização mais utilizada a nível mundial tendo, de acordo com a Nagios Enterprises, efectuadas mais de um milhão de instalações e possuindo duzentos e cinquenta mil utilizadores regulares. A história da ferramenta iniciou-se no

final dos anos noventa, ainda sob o nome NetSaint, mudando posteriormente para Nagios Ain't Gonna Insist On Sainthood (NAGIOS). Tendo sido a primeira ferramenta de relevo a surgir na comunidade de software de código aberto e desde cedo utilizando uma arquitectura de funcionamento que permitia o desenvolvimento de extensões simples por parte de terceiros, o Nagios começou a gozar de grande popularidade e atrair uma grande comunidade de programadores que viram nessa flexibilidade uma grande facilidade em introduzir de forma simples e rápida no programa as funcionalidades que desejavam.

O desenvolvimento do Nagios apostou de forma contínua na expansão desta flexibilidade e facilidade de integração de soluções e funcionalidade de terceiros passando também a permitir *addons, plugins* mais complexos que permitem complementar ou mesmo substituir aspectos vários da funcionalidade no Nagios, como o seu interface gráfico, modo de armazenamento de dados ou o próprio motor de processamento. Desta forma, é possível adaptar o Nagios, construindo sobre ele soluções bastante mais completas e complexas, dando assim origem a vários produtos, quer gratuitos, quer comerciais onde a semelhança com a base de Nagios é por vezes pouco mais do que a compatibilidade com os plugins existentes para a ferramenta base.

O crescimento destes *plugins* foi de tal ordem que, hoje em dia, o Nagios conta com cerca de dois mil, contando apenas os disponíveis na página oficial, cobrindo todas as áreas da monitorização de rede e equipamentos, do hardware aos sistemas operativos, passando pelas bases de dados, clusters ou sensores vários. O número de *addons* é também já superior a quatrocentos conferindo uma alargada escolha aos utilizadores em termos de alternativas de funcionamento do Nagios.

Com argumentos em tamanha quantidade, não é de estranhar que seja a solução instalada para a monitorização, em exclusivo ou aliada a outras soluções, das redes de alguns dos nomes mais conhecidos da indústria como 3Com, Amazon, AT&T, digg, eBay, Google, HP, IBM, MySpace, StumbleUpon, Symantec, Twitter, Verisign ou Yahoo, entre muitos outros. [43]

A flexibilidade da ferramenta não se esgota, no entanto, no número de funções adicionais com que conta. A facilidade de comunicação e utilização dos recursos da ferramenta pode ser usada por terceiros para daí retirar informação recolhida pelo Nagios e utilizá-lo nos seus próprios programas ou, inversamente, exportar informação dos seus programas para a ferramenta Nagios permitindo que a mesma seja incluida nos relatórios ou visualizações do Nagios, ou mesmo que seja processada pela ferramenta base ou qualquer um dos seus plugins. A popularidade e disseminação do Nagios fazem com que fabricantes de hardware de rede, sensores, bastidores ou outro equipamento possuam plugins para Nagios. Da mesma forma produtores de software de registos, anti-vírus, segurança e outras áreas fornecem também compatibilidade com Nagios, permitindo exportar informação e alertas para a plataforma de monitorização. [44]

A documentação disponível online na página da ferramenta é abundante, existindo sempre como alternativa o acesso à comunidade de utilizadores que mantém listas de distribuição, sites em formato wiki com truques e dicas, fóruns e vários outros recursos. O próprio manual da ferramenta é de leitura extensa mas é excepcionalmente completo. O ficheiro de configuração da ferramenta permite a quem não deseje utilizar os parâmetros base afinar o comportamento da ferramenta em mais de cento e cinquenta parâmetros que permitem definir o como e quando a plataforma efectua todas as suas acções de monitorização e registo.

Para alargar a sua capacidade de monitorização, o Nagios recorre, para além de SNMP, a agentes, denominados *Plugins* remotos, instalados nos equipamentos a monitorizar e com os quais contacta através da funcionalidade Nagios Remote Plugin Executor (NRPE). Um *plugin* remoto não é mais do que um plugin normal escrito em perl ou mesmo um *shell script* que, quando executado, devolve um conjunto de valores sobre aquilo que monitoriza. Como estes plugins são exactamente os mesmos que os que se utilizam numa instalação local podem portanto devolver resultados sobre tudo o que se queira, desde ocupação da memória ou espaço em disco, a carga do processador ou qualquer outra verificação desde que a mesma seja possível de ser feita localmente. Um plugin remoto pode ser utilizado também como intermediário em situações nas quais a estaçaão de monitorização Nagios possa não ter acesso directo ao alvo a monitorizar. A título de exemplo refira-se uma situação na qual se deseja monitorizar os serviços SSH e HTTP de um servidor, mas que o mesmo apenas aceite ligações de um número restrito de clientes. Um plugin nagios instalado num desses clientes autorizados poderá ser depois questionado pela estação de monitorização e efectuará a verificação do serviço através de tentativa de conexão, devolvendo o resultado à ferramenta de monitorização. O NRPE pode ser utilizado através de conexão simples ou segura via SSL. O Nagios permite a execução dos plugins remotos igualmente através de uma ligação SSH clássica, no entanto essa opção é normalmente preterida em função do NRPE pois este último implica um gasto bastante inferior de recursos, o que se torna tão mais importante quanto maior o número de equipamentos a monitorizar.

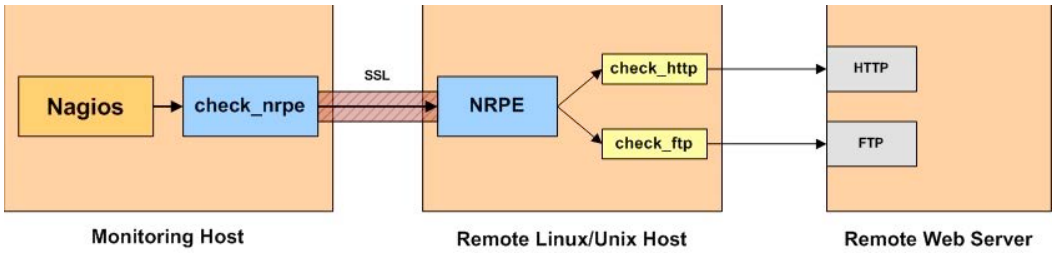



Figura 4.9 – Nagios NRPE – Verificação indirecta [35]

Junto com o Nagios é disponibilizado um pacote com um pequeno grupo de plugins que devem ser instalados junto com a ferramenta e que proporcionam um conjunto básico de cerca de cinquenta parâmetros a monitorizar, permitindo ao utilizador fazer uma ideia do panorama geral do equipamento monitorizado ao nível da conectividade e funcionamento dos principais serviços, utilização do processador, memória e espaço em disco. Sem necessidade de procurar plugins individualmente ou instalar novos, o que tem de ser feito manualmente editando ficheiros de configuração, o novo utilizador do Nagios fica assim com um sistema de monitorização que satisfaz as necessidades básicas de informação sobre um servidor.

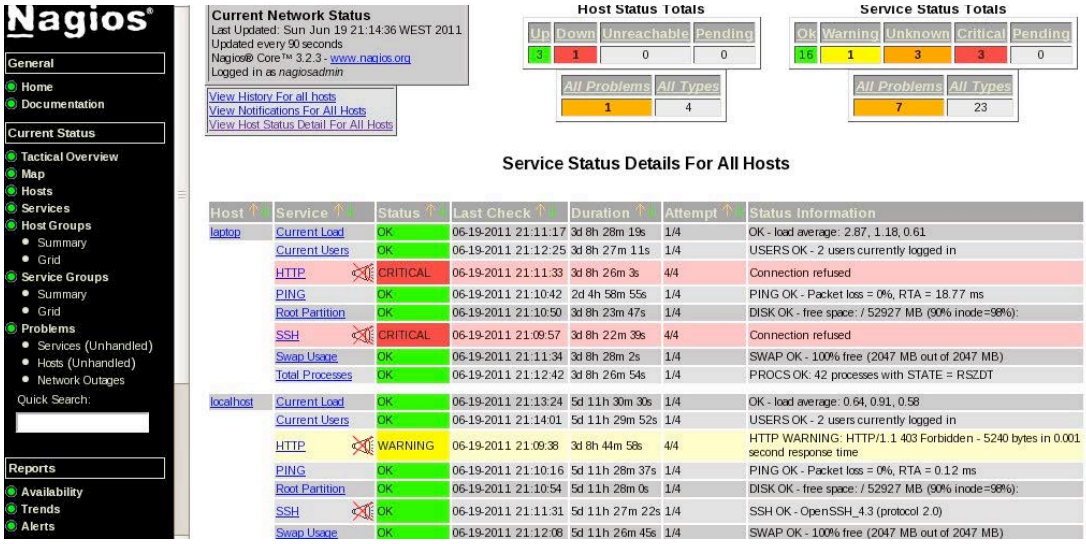



Figura 4.10 – Nagios – Vista de serviços básicos

Juntamente com o download do Nagios Core e do pacote de plugins básico é também proposta a descarga e instalação das NDOutils, um *addon* que permite que o Nagios armazene a informação recolhida numa base de dados MySQL ao invés do armazenamento de ficheiros no seu directório. O objectivo primordial das NDOutils é permitir que várias instâncias diferentes de Nagios guardem as informações recolhidas numa mesma base de dados. Assim, num futuro desenvolvimento da interface gráfica do Nagios, que se aponta venha a ser realizada em PHP, essa mesma interface consiga efectuar consultas a informação de tantas instâncias diferentes quantas estejam instaladas. A utilização das NDOutils não é tão generalizada como a da ferramenta em si, muito pelo atraso de vários

anos na reformulação do interface gráfico, o que deu origem à criação de vários grafismos e *layouts* alternativos disponibilizados por terceiros sob a forma de *addons.*

Apesar de uma aposta forte, ao longo dos últimos três anos [44], por parte da Nagios Enterprises na tentativa de envolver mais a comunidade com o Nagios, nomeadamente com o lançamento de um bugtracker público onde a comunidade pode centralizar a comunicação das descobertas de todas as falhas ou erros detectados no Nagios [45], um portal de ideias da comunidade para sugestão e votação de novas capacidades [46], acesso a testes dos plugins em desenvolvimento e locais para utilização da comunidade com novas funcionalidades, a sensibilidade de parte da comunidade é de que o desenvolvimento de partes fulcrais do produto estará estagnado em favor do desenvolvimento da versão comercial da ferramenta, denominado Nagios XI.

O efeito desta percepção levou a que vários autores e empresas tomassem em mãos o suprir das falhas que encontram actualmente no Nagios. Desse conjunto de soluções alternativas foram analisadas o check_mk, que proporciona uma nova interface gráfica, um novo método de comunicação remota com plugins e com o seu novo desenvolvimento Multisite_Mk permite também a utilização de várias instâncias de Nagios centralizadas num só interface.

O Nagios permite a utilização de *Event Brokers*, módulos que podem tomar conta de acções no Nagios. Um *Event Broker* pode fazer algo tão simples como correr um comando na *shell* sempre que um determinado evento seja detectado pelo Nagios até algo mais complexo como substituir o próprio Nagios no processamento de informação. O módulo Merlin da empresa sueca OP5 é um event broker no qual é baseado o seu produto comercial OP5 Monitor e que efectua um processamento paralelo ao Nagios, guardando os valores em base de dados e permitindo redundância e balanceamento para instalações de monitorização em redes de grande dimensão ou onde a monitorização seja crítica.

O projecto icinga assume-se como uma ruptura com o desenvolvimento do Nagios, seguindo uma nova linha de desenvolvimento com prioridades diferentes da equipa original e com o objectivo de desenvolver todo o produto e não apenas de criar módulos ou *plugins.*

No entanto, o factor comum a todas estas diferentes aproximações de "correcção" do desenvolvimento da ferramenta é o de manterem a retro-compatibilidade com todos os plugins do Nagios. Tal facto revelou-se particularmente útil no decurso desta dissertação pois foi assim possível contar-se com a vastíssima gama de plugins do Nagios para satisfazer todos os requisitos da solução pedida.

Dos requisitos pedidos verificou-se que o pacote de *plugins* básicos do Nagios supria a maior parte dos mesmos. Foram então testadas várias alternativas para a monitorização de *Web Injection* e base de dados Oracle. Para ambos os casos a solução mais completa e flexível foi a proposta pela ConSol Labs alemã [47]. Achou-se ainda conveniente

proporcionar uma solução de monitorização para computadores com sistema operativo Microsoft Windows, tendo a escolha reacaído no agente de monitorização NSClient++ com WMI.

O *plugin* de *Web Injection* permite verificar o funcionamento de qualquer página de um sítio web, quer verificando se o servidor HTTP responde, como se responde com o conteúdo desejado. Assim podem verificar-se tanto páginas com conteúdo estático como formulários ou outro tipo de serviços web conquanto se saibam os argumentos a enviar e os resultados a receber. Definindo num ficheiro XML o(s) teste(s) desejado(s) pode pedir-se à solução baseada em Nagios que execute o *plugin* de forma semelhante ao exemplo abaixo.

```
%>./check_webinject –s baseurl=http://www.fe.up.pt/si/web_page.inicial
\testcase.xml
WebInject OK - All tests passed successfully in 0.027
seconds|time=0.027;0;0;0;0 testpage=0.024;5;15;0;0
```

Verifica-se que o servidor HTTP respondeu positivamente aos testes deste exemplo hipotético no espaço de 0.027 segundos, o que causou o envio de um registo positivo ao Nagios. Analisando a restante resposta do *plugin* vemos que o mesmo está configurado para no caso da resposta demorar um período superior a cinco segundos enviar um aviso (*Warning*) ao Nagios, indicando uma possível sobrecarga no servidor, e no caso da demora ser superior a quinze segundos enviar uma comunicação de falha (*critical*).

Para a monitorização da base de dados Oracle foi novamente escolhida uma solução da ConSol labs, o *plugin* check_oracle_health. A escolha resultou do facto de ser o mais completo dos *plugins* por não só incluir as funcionalidades individuais dos restantes como também verificações adicionais não encontradas nos demais. Aliado à sua completude o plugin é bastante funcional sendo de utilização muito simples, bastando incluir a palavra chave do valor a monitorizar em frente ao argumento –mode.

Foi necessário criar na base de dados de testes, a correr em Oracle 10g Express Server, um utilizador "Nagios" com as devidas permissões de consulta de modo a que o *plugin* conseguisse ler valores. Após esse passo preparatório bastou invocar o plugin via NRPE ou outro meio de comunicação para que o mesmo devolvesse valores. Um exemplo sucinto de algumas utilizações possíveis é o que se pode ver de seguida.

```
nagios$ check_oracle_health --mode invalid-objects
OK - no invalid objects found  |
  invalid_ind_partitions=0 invalid_indexes=0
  invalid_objects=0 unrecoverable_datafiles=0
```

```
nagios$ check_oracle_health --mode tablespace-free
    --name TEST
OK - tbs TEST has 97.91% free space left |
    'tbs_test_free_pct'=97.91%;5:;2:
    'tbs_test_free'=32083MB;1638.40:;655.36:;0.00;32767.98
```

O *plugin* permite igualmente que seja enviada qualquer consulta sql desde que o seu retorno seja numérico, conferindo assim uma flexibilidade ainda maior às verificações possíveis.

As palavras chave passíveis de serem usadas com o argumento –mode são as constantes da tabela disponível na secção Check_Oracle_health dos anexos. Reproduzem-se abaixo apenas alguns dos valores possíveis a título ilustrativo.

| **Keyword** | **Description** | **Range** |
|---|---|---|
| connection-time | Determines how long connection establishment and login take | 0..n Seconds (1, 5) |
| connected-users | The sum of logged in users at the database | 0..n (50, 100) |
| session-usage | Percentage of max possible sessions | 0%..100% (80, 90) |
| process-usage | Percentage of max possible processes | 0%..100% (80, 90) |
| rman-backup-problems | Number of RMAN-errors during the last three days | 0..n (1, 2) |
| sga-shared-pool-free | Free Memory in the Shared Pool | 0%..100% (10:, 5:) |
| invalid-objects | Sum of faulty Objects, Indices, Partitions | |
| tablespace-usage | Used diskspace in the tablespace | 0%..100% (90, 98) |

| | | |
|---|---|---|
| tablespace-free | Free diskspace in the tablespace | 0%..100% (5:, 2:) |
| tablespace-fragmentation | Free Space Fragmentation Index | 100..1 (30:, 20:) |
| tablespace-remaining-time | Sum of remaining days until a tablespace is used by 100%. The rate of increase will be calculated with the values from the last 30 days. (With the parameter –lookback different periods can be specified) | Days (90:, 30:) |
| sysstat | Changes/sec for any value from v$sysstat | n/sec (10,10) |
| sql | Result of any SQL-Statement that returns a number. The statement itself is passed over with the parameter –name. A Label for the performance data output can be passed over with the parameter –name2. | n (1,5) |
| list-tablespaces | Prints a list of tablespaces | |
| list-datafiles | Prints a list of datafiles | |
| list-sysstats | Prints a list with system-wide statistics | |

Tabela 4.2 — check_oracle_health – Lista de alguns argumentos monitorizáveis [52]

Para a monitorização de computadores com sistema operativo Microsoft Windows foi utilizado o agente NSClient++, quer pela sua baixa utilização de recursos, quer pelas elevadas capacidades de monitorização visto que faz utilização não só das suas próprias capacidades como do protocolo Windows Management Instrumentation (WMI), o qual pode ser descrito como uma reinvenção por parte da Microsoft de algo com uma filosofia de funcionamento semelhante ao SNMP, mas com bastantes mais funcionalidades [48]. O WMI permite ao NSClient++ recolher informação sobre sistema operativo e hardware podendo o mesmo devolver ao Nagios informação sobre vários parâmetros, estando alguns resumidos na tabela abaixo. De notar que apesar do WMI permitir operações de escrita que alteram a configuração dos equipamentos, o NSClient++ apenas realiza operações de leitura pois são o que necessita para as funções de monitorização.

| Object | Description |
|---|---|
| Win32_Fan | Represents the properties of a fan device in the computer system. |
| Win32_TemperatureProbe | Represents the properties of a temperature sensor (electronic thermometer). |
| Win32_DiskDrive | Represents a physical disk drive as seen by a computer running the Windows operating system. |
| Win32_PhysicalMedia | Represents any type of documentation or storage medium. |
| Win32_TapeDrive | Represents a tape drive on a computer system running Windows. |
| Win32_BaseBoard | Represents a baseboard (also known as a motherboard or system board). |
| Win32_BIOS | Represents the attributes of the computer system's basic input or output services (BIOS). |
| Win32_IDEController | Represents the capabilities of an Integrated Drive Electronics (IDE) controller device. |
| Win32_MemoryArray | Represents the properties of the computer system memory array and mapped addresses. |
| Win32_OnBoardDevice | Represents common adapter devices built into the motherboard (system board). |
| Win32_Processor | Represents a device capable of interpreting a sequence of machine instructions on the computer. |
| Win32_SCSIController | Represents a small computer system interface (SCSI) controller on a computer system running Windows. |
| Win32_USBControllerDevice | Relates a USB controller and the CIM_LogicalDevice instances connected to it. |
| Win32_NetworkAdapter | Represents a network adapter on a computer system running Windows. |
| Win32_Battery | Represents a battery connected to the computer system. |
| Win32_PortableBattery | Represents the properties of a portable battery, such as one used for a notebook computer. |
| Win32_PowerManagementEvent | Represents power management events resulting from power state changes. |
| **Object** | **Description** |
| Win32_SystemDriver | Represents the system driver for a base service. |
| Win32_Directory | Represents a directory entry on a computer system running Windows. |
| Win32_DiskQuota | Tracks disk space usage for NTFS file system volumes. |
| Win32_LogicalDisk | Represents a data source that resolves to an actual local storage device. |
| Win32_Volume | Represents an area of storage on a hard disk. |
| Win32_PageFileUsage | Represents the file used for handling virtual memory file swapping on a computer system running Windows. |
| Win32_NetworkConnection | Represents an active network connection in a Windows environment. |
| Win32_NTDomain | Represents a Windows NT domain. |
| Win32_PingStatus | Represents the values returned by the standard ping command. |
| Win32_ComputerSystem | Represents a computer system operating in a Windows environment. |
| Win32_OperatingSystem | Represents an operating system installed on a computer system running Windows. |
| Win32_Process | Represents a sequence of events on a computer system running Windows. |
| Win32_ProcessStartup | Represents the startup configuration of a computer system running Windows. |
| Win32_ScheduledJob | Represents a job scheduled using the Windows NT schedule service. |
| Win32_BaseService | Represents executable objects that are installed in a registry database maintained by the SCM. |
| Win32_Service | Represents a service on a computer system running Windows. |
| Win32_LogonSession | Describes the logon session or sessions associated with a user logged on to Windows 2000 or Windows NT. |
| Win32_UserAccount | Represents information about a user account on a computer system running Windows. |
| Win32_UserInDomain | Association class |
| Win32_WindowsProductActivation | Contains properties and methods related to WPA. |
| Win32_NTEvent... | **Yes you can even check the eventlog!** |

Tabela 4.3 — NSClient++ – Lista das principais funções [48]

Tendo os *plugins* necessários à realização da monitorização de todos os requisitos escolhidos, foi feita a avaliação das soluções baseadas em Nagios previamente seleccionadas pelas capacidades anunciadas e pela sua popularidade, sendo esta última característica utilizada como putativo garante do seu futuro desenvolvimento. Os três plugins anteriores foram utilizados como parte integrante de todas as soluções abaixo.

#### 4.2.3.1 – Check_MK

O Check_mk é um *plugin* para Nagios que implementa uma interface gráfica alternativa e um conjunto de agentes próprio. A forma clássica do Nagios receber informação de um alvo remoto é inquirindo, normalmente via NRPE, cada um dos *plugins* instalados na máquina da qual se pretende obter informação. Se forem considerados cem processos monitorizados em cada computador, em cada ronda de monitorização o Nagios tem de iniciar cem processos, abrir cem ligações TCP e executar cem *plugins* na máquina remota. Com o check_mk o agente no computador remoto vai mantendo um registo do estado actual dos serviços, assim, quando o Nagios faz uma chamada ao *plugin* check_mk, inicia apenas esse processo, que abre apenas uma ligação TCP e recebe todos os dados agregados respeitantes aquele computador remoto, transmitidos apenas por um agente remoto.

A informação recebida é guardada numa base de dados *Round Robin* (RRD) e é periodicamente processada pelo plugin, que faz uma monitorização passiva dos equipamentos a monitorizar visto que apenas processa valores já recolhidos e registados previamente, enviando os resultados desse processamento ao Nagios, que os recebe como se de resultados de uma monitorização activa se tratassem, pois convém lembrar que para o Nagios, o check_mk é apenas mais um plugin.

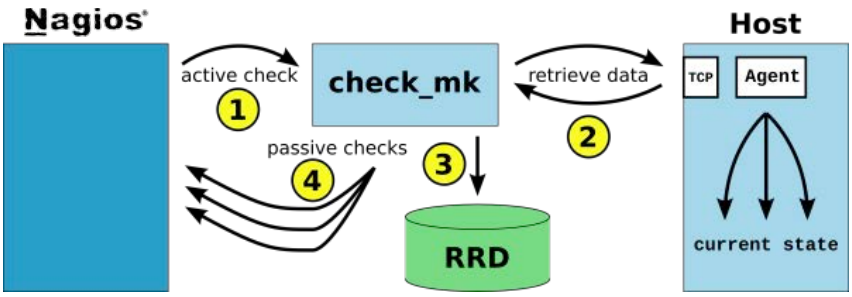



Figura 4.11 – Check_MK – Diagrama de funcionamento [52]

O check_mk é particularmente flexível pois é capaz de detectar quais os serviços que pode monitorizar em cada alvo de monitorização através da função de inventário. Correndo o check_mk com a opção –I no servidor de monitorização Nagios, é feita uma ronda por todos os computadores conhecidos no ficheiro de configuração inquirindo o agente check_mk de cada um sobre quais os serviços que consegui detectar desde o último inventário, adicionando-os à lista de valores a devolver. No caso de ser a primeira vez que o agente recebe o pedido de inventário, listará todos os serviços que conseguiu detectar até ao momento. Correndo novamente o check_mk com a opção –O, é alterado automaticamente o ficheiro de configuração do Nagios para passar a incluir a

monitorização de todos os novos serviços descobertos nos vários computadores remotos. Relembrando que o check_mk é visto como apenas mais um plugin pelo Nagios, todas as capacidades que o check_mk não possua poderão ser atribuidos a outros plugins do Nagios com os quais a plataforma comunicará via NRPE ou SSH, sendo esta configuração mista a mais comum.

Visto que todos os valores são guardados em base de dados, o check_mk avançou para um dos objectivos futuros do Nagios Core e implementou a capacidade de várias instâncias de Nagios guardarem valores na mesma base de dados através de uma aplicação de coordenação distribuida chamada MK_Livestatus, sendo os resultados dessa instalação distribuída consultáveis através de uma interface gráfica chamada Multisite_MK, solução essa que foi a instalada para avaliação nesta dissertação.


Figura 4.12 – Multisite_MK – Vista Geral

O Multisite_MK destaca-se por ter um grafismo configurável, que permite várias visualizações alternativas de quase todos os ecrãs. Em todo o interface estão sempre presentes no topo várias opções de layout, bem como links directos para a alteração de vários tempos de refrescamento da informação apresentada.

O agente Check_MK fornece um conjunto alargado de parâmetros de monitorização tanto em sistemas operativos Linux/Unix como Microsoft Windows directamente acessíveis através do link para o computador ou do menu lateral da aplicação, também ele configurável relativamente às funcionalidades presentes. Uma das novidades introduzidas pela interface Multisite_MK foram os snapins, pequenos módulos que podem ser desenvolvidos por terceiros e que se alojam no menu lateral fornecendo acesso a directo a funcionalidades ou exibindo informação. Através do botão snapins pode-se aceder à lista de snapins disponíveis e adicionar ou remover snapins do menu lateral.


Figura 4.13 – Check_MK – Serviços em computador MS Windows – Vista parcial

Outro dos melhoramentos existentes em relação à interface clássica do Nagios é a forma mais simples de criar verificações, notificações que podem ser enviadas por email ou SMS ou efectuar a marcação de periodos de inactividade ou desligamento programados (*Downtime*).

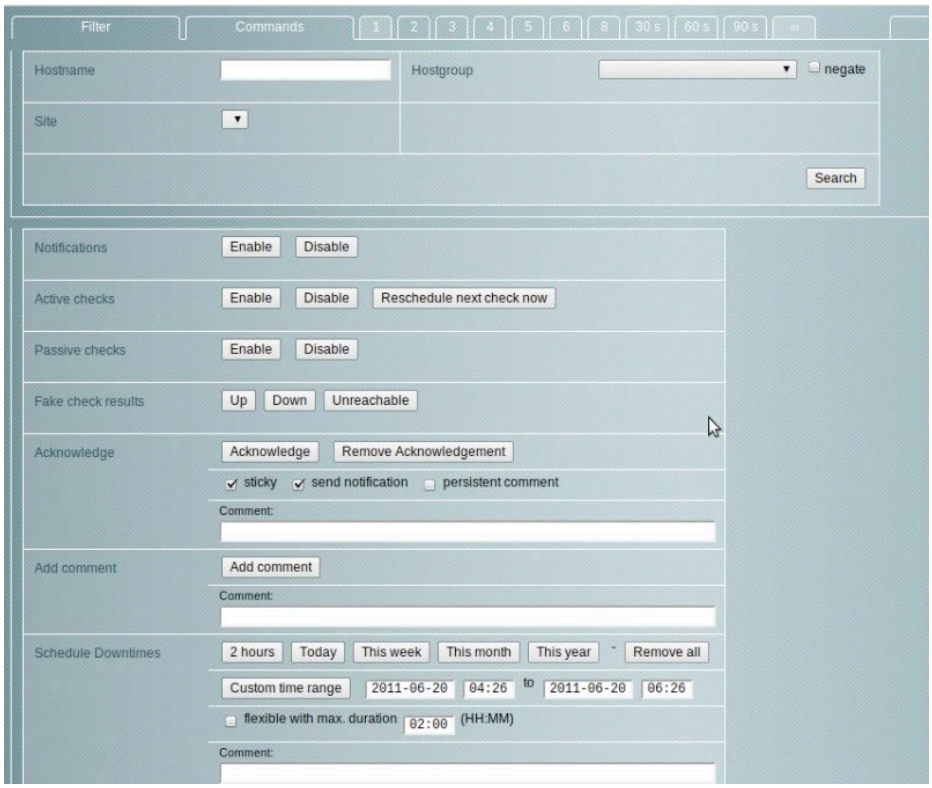



Figura 4.14 – Check_MK – Configuração de verificações e periodos de manutenção

O autor do check_mk aconselha vivamente a que se aproveite o facto do plugin guardar toda a informação numa base de dados RRD e que se instale uma das das várias ferramentas de produção de gráficos compatíveis com esse formato de base de dados, estendendo assim a capacidade de produção de informação do *plugin*. Uma das ferramentas aconselhadas é o bem conhecido PNP4Nagios, para o qual o check_mk vem, aliás já preparado. A última versão do PNP4Nagios é, no entanto, incompatível com as anteriores, o que faz com que tenha de se editar o código fonte do check_mk bem como os seus ficheiros de configuração para que o *plugin* seja capaz de produzir pedidos com a sintaxe correcta. Após a edição de alguns blocos de código fonte os gráficos produzidos pelo PNP4Nagios surgem integrados na interface do Multisite_MK, podendo ser gerados sobre qualquer uma das variáveis monitorizadas nos intervalos de tempo pretendidos. O PNP4Nagios oferece ainda a possibilidade de exportar os gráficos gerados em XML ou PDF, sendo possível fazer a ligação entre o gráfico no ecrã e os relatórios de disponibilidade internos do Nagios, bem como a listagem de alertas gerados no mesmo periodo, bastando para tal clicar num dos símbolos presentes junto ao gráfico.


Figura 4.15 – Check_MK – Gráficos PNP4Nagios

Aliada à capacidade de gerar gráficos existirá no futuro uma capacidade de melhor organizar a informação permitindo efectuar uma melhor análise de tendências. O *plugin* Check_MK_BI, de *Business Intelligence* encontra-se já em versão beta.

Resta registar que não foram detectadas quaisquer diferenças de desempenho entre a instalação Multisite_MK realizada em OpenSUSE e em Ubuntu quando monitorizando os mesmos serviços.

### 4.2.3.2 – Merlin + Ninja

Tal como previamente referido, a plataforma Nagios permite a existência de *Event Brokers*, módulos que alterando ou não o comportamento do Nagios podem aceder a toda a sua informação e complementar ou mesmo substituir o Nagios nas suas funções. A

empresa de consultadoria de redes sueca OP5 tendo identificado a necessidade de um produto configurável que respondesse às necessidades dos seus clientes escolheu o Nagios como plataforma de base para o desenvolvimento do seu produto precisamente devido à possibilidade de usar uma estrutura existente e comprovada, podendo no entanto alterar as partes que entendesse convenientes. A forma mais rápida de atingir esse objectivo e o consequente lucro financeiro foi pegando no Nagios e implementando as sugestões mais populares constantes em ideas.nagios.org colocadas pelos utilizadores. [49]

Desta forma foi criado um *Event Broker* que providencia *clustering*, balanceamento e redundância entre servidores Nagios, adiciona uma nova camada de transporte de eventos e passa a utilizar a biblioteca libdbi para comunicação com a base de dados, substituindo assim as NDOTolls do Nagios, tendo o sistema sido baptizado como Module for Efortless, Redundant and Loadbalanced Infrastructure in Nagios (MERLIN).

O módulo Merlin intercepta todas as comunicações efectuadas pela plataforma Nagios e passa-as ao *daemon* Merlin, que por sua vez as processa e armazena em base de dados. Esta estrutura é necessária, pois uma tentativa de colocar o módulo a efectuar o processamento iria causar atrasos no tempo de resposta entre o módulo e a plataforma Nagios, causando com que a mesma encravasse, um problema conhecido do Nagios quando aguarda tempo demais pela resposta de um *Event Broker*. Desta forma o módulo apenas envia as mensagens que intercepta para o *daemon*, libertando assim de imediato a plataforma Nagios enquando o *daemon* Merlin processa a informação em paralelo com o Nagios.

O *daemon* tem também a capacidade de comunicar com outros *daemons* Merlin através de sockets, sendo assim conseguida a redundância e balanceamento do sistema. A configuração dos *daemons* é extremamente simples, bastando adicionar no ficheiro de configuração os endereços IP de todos os servidores Nagios com os quais se pretende o balanceamento e redundância.

Ao contrário das restantes soluções testadas, o Merlin exige a futura versão 3.2.4 do Nagios para funcionar, sendo assim necessário recorrer ao repositório de desenvolvimento do Nagios para obter o código fonte necessário a uma instalação compatível com Merlin.

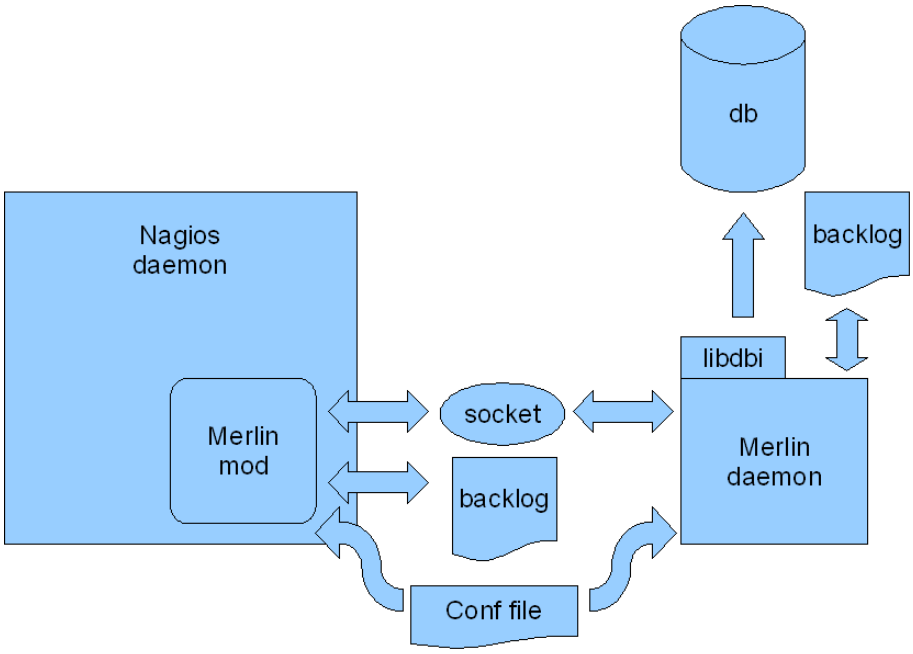



Figura 4.16 – Arquitectura sistema Merlin [51]

A não-utilização das NDOutils permitiu à OP5 criar um esquema de base de dados totalmente diferente e mais bem estruturado, registando-se apenas o necessário e criando tabelas que facilitassem as consultas mais comuns. Procurou-se que o conteúdo das tabelas fosse estático, como a lista de alvos de monitorização, ou contivesse apenas a diferença de um estado para o anterior já registado ao invés de duas informações de estado distintas e completas. Com este esquema de armazenamento foi conseguida uma enorme simplificação das consultas à base de dados, conseguindo quase linearizar-se o tempo de resposta da mesma.

- **NDOutils sample query**
  - SELECT nagios_instances.instance_id, nagios_instances.instance_name, nagios_services.host_object_id, obj1.name1 AS host_name, nagios_services.service_object_id, obj1.name2 AS service_description, nagios_servicestatus.* FROM `nagios_servicestatus` LEFT JOIN nagios_objects as obj1 ON nagios_servicestatus.service_object_id = obj1.object_id LEFT JOIN nagios_services ON nagios_servicestatus.service_object_id=nagios_services.service_object_id LEFT JOIN nagios_instances ON nagios_services.instance_id = nagios_instances.instance_id WHERE nagios_services.config_type = '1' ORDER BY instance_name ASC, host_name ASC, service_description ASC
- **Merlin sample query**
  - SELECT * FROM service ORDER BY host_name, service_description ASC

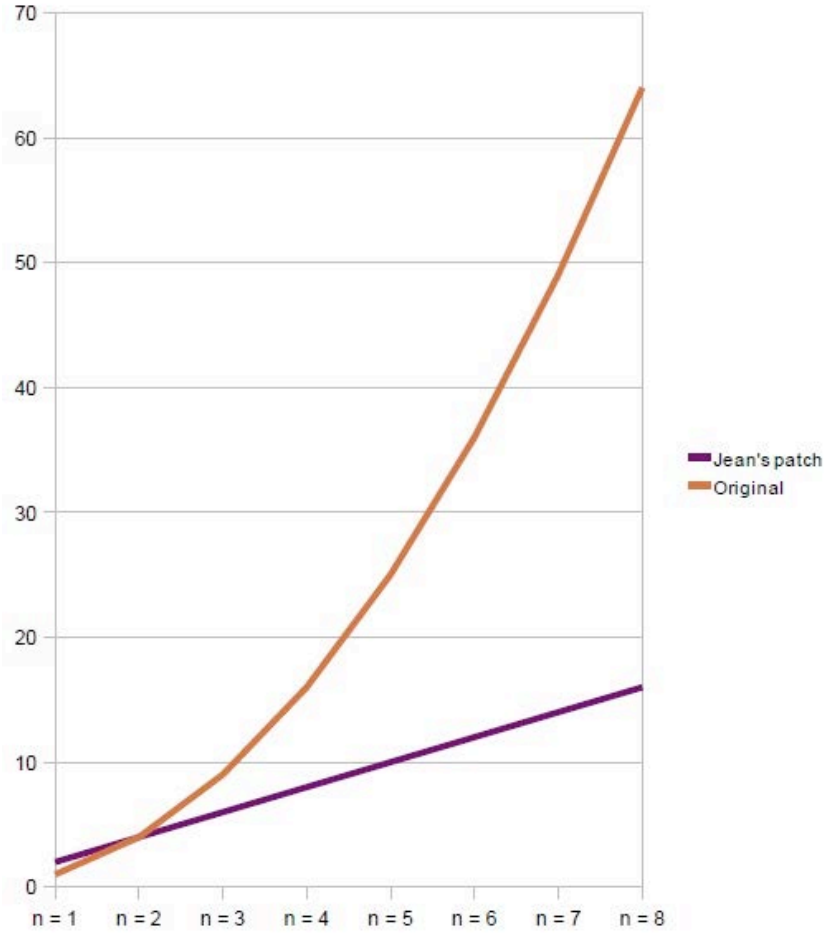



Figura 4.17 – OP5 – Comparação NDOUtils Vs. Esquema Merlin [49]

Por forma a que os utilizadores possam visualizar a informação produzida e interagir com o sistema Merlin foi criado um interface gráfico chamado Nagios Is Now Just Awesome (NINJA). A junção destes dois componentes forma o produto comercial da OP5, o OP5 Monitor. Solução que é a eleita para a monitorização de rede por parte de empresas como a Volvo, SAAB, Time Warner Cable, Comcast ou Electrolux.

Esta solução comercial está disponível em três versões, sendo a mais simples delas gratuita, mas com limitação de monitorização de vinte endereços IP e sem as funcionalidades de balanceamento ou rede distribuida, sem a possibilidade de utilização de extensões e apenas com acesso à documentação online, mas sem suporte técnico. A versão base ganha a assistência por email e a instação e actualizações, e finalmente a versão enterprise não possui quaisquer limitações. A OP5 afirma que mais de 70% das instalações de Merlin e Ninja são versões pagas do OP5Monitor apesar de a empresa dar acesso gratuito a ambos os *softwares*, sendo que qualquer pessoa pode então, na posse no

Merlin e Ninja, reproduzir a versão enterprise bastando para isso não colocar nenhuma limitação.

Esse facto justificar-se-á em parte devido às versões pagas serem as únicas que incluem suporte técnico para a instalação e actualização das versões do OP5Monitor. Na verdade, se o Merlin se apresenta como um software estável, mesmo estando em contínuas versões beta, num programa que a OP5 intitula de *stable beta*, não lançando assim nunca uma versão estável, estando essa denominação disponível apenas para a linha comercial, já o Ninja é um software em pleno desenvolvimento saindo uma nova versão em média a cada duas semanas durante o ano de 2011. O repositório de desenvolvimento sofre actualizações diárias com a resolução de vários problemas que afectam este interface e que se encontram camuflados na versão comercial.

A instalação e configuração do Ninja não se afigura trivial, tendo sidas perdidas algumas semanas até ter sido conseguida uma instalação funcional, mesmo com o apoio da lista de distribuição de utilizadores e programadores, onde os restantes utilizadores se encontravam em dificuldades semelhantes e as sugestões dos programadores não produziam os efeitos desejados.

Com efeito o Ninja é extremamente exigente não só nas versões de todo o software com que interage, incluindo versões de bibliotecas, PHP e módulos de linguagens, não podendo tratar-se de versões inferiores ou superiores à exigida, como contém também no código várias instrucções já deprecadas e que causam resultados muitas vezes imprevisíveis e à primeira vista inexplicáveis.

Após a edição e alteração de dezenas de ficheiros do código fonte do Ninja numa tentativa de resolução dos erros produzidos, foi assumida a estratégia de fazer a comparação automática das centenas de ficheiros entre a instalação de Ninja da rede de testes com os ficheiros da versão gratuita do OP5 Monitor, que é distribuido apenas em formato de imagem de máquina virtual, vindo assim já pré-instalado e funcional. Após a conversão da máquina virtual para o mesmo formato com a ajuda do VMWare converter, visto que a máquina virtual distribuida vem criada com erros sendo assim necessário efectuar uma conversão apenas para corrigir a máquina virtual, verificou-se que as diferenças entre os conteúdos dos ficheiros de código eram negligenciáveis apesar de o OP5Monitor estar funcional. A conclusão lógica foi a de que, algures no meio do código, estariam a ser usados *hard links*, com a localização absoluta de ficheiros no sistema operativo ao invés de serem passados como variáveis. Fez-se a verificação tentando a instalação do sistema Merlin e Ninja numa nova máquina a correr CentOS, a mesma distribuição da máquina virtual do OP5 Monitor. Ao fim de poucas horas, resolvendo apenas problemas triviais, não surgiram os problemas inexplicáveis vivenciados em Ubuntu e OpenSuse, ficando a instalação funcional.

O interface Ninja utiliza a framework PHP Kohana, que lhe proporciona rapidez e economia de recursos.

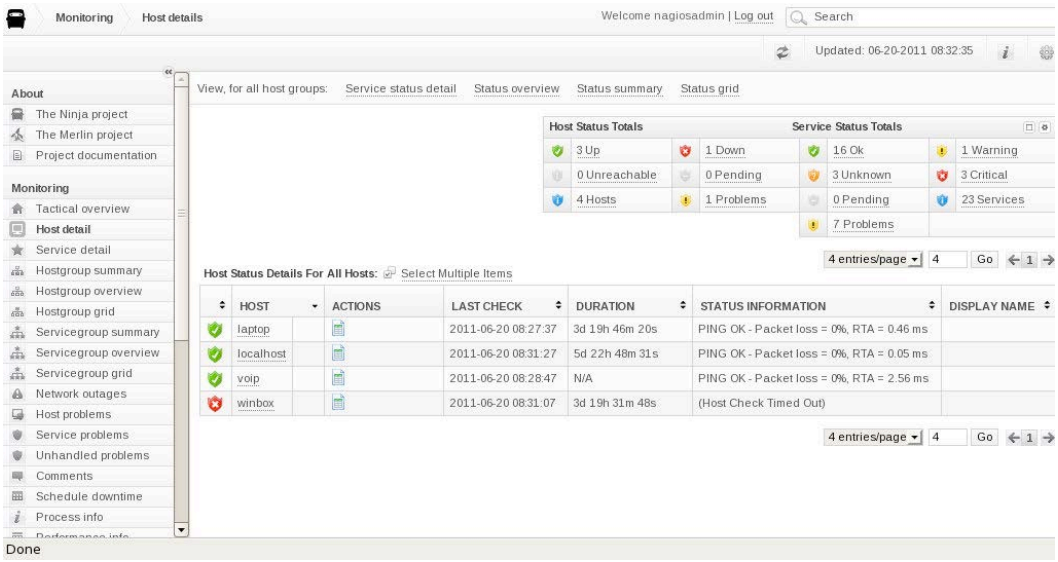



Figura 4.18 – OP5 – Interface gráfico Ninja

Visualmente, o Ninja é uma versão refrescada do ecrã típico do Nagios, com tudo em local semelhante, mudando apenas o aspecto gráfico do fundo e dos ícones, tal como sucede em qualquer um dos vários *templates* existentes para Nagios.

As funcionalidades típicas do Nagios estão presentes, podendo ser verificadas em maior detalhe nas imagens constantes na secção de anexos. Mais específica do Ninja é a apresentação de informação sobre o desempenho do sistema de monitorização.

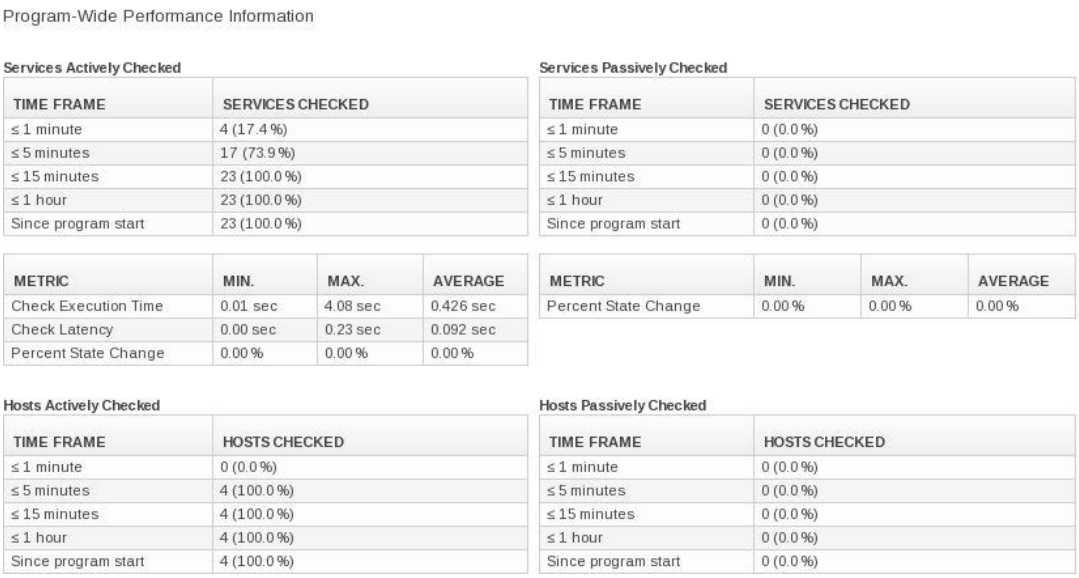



Figura 4.19 – OP5 – Informação de desempenho - Ninja

Outra especificidade digna de nota do Ninja é a forma rápida e intuitiva para cálculo de cumprimento de Service Level Agreement (SLA). Basta preencher a percentagem que se quer ver verificada em cada mês e qual o serviço ou equipamento a verificar. O Ninja gerará os gráficos indicando se o SLA foi cumprido ou não.

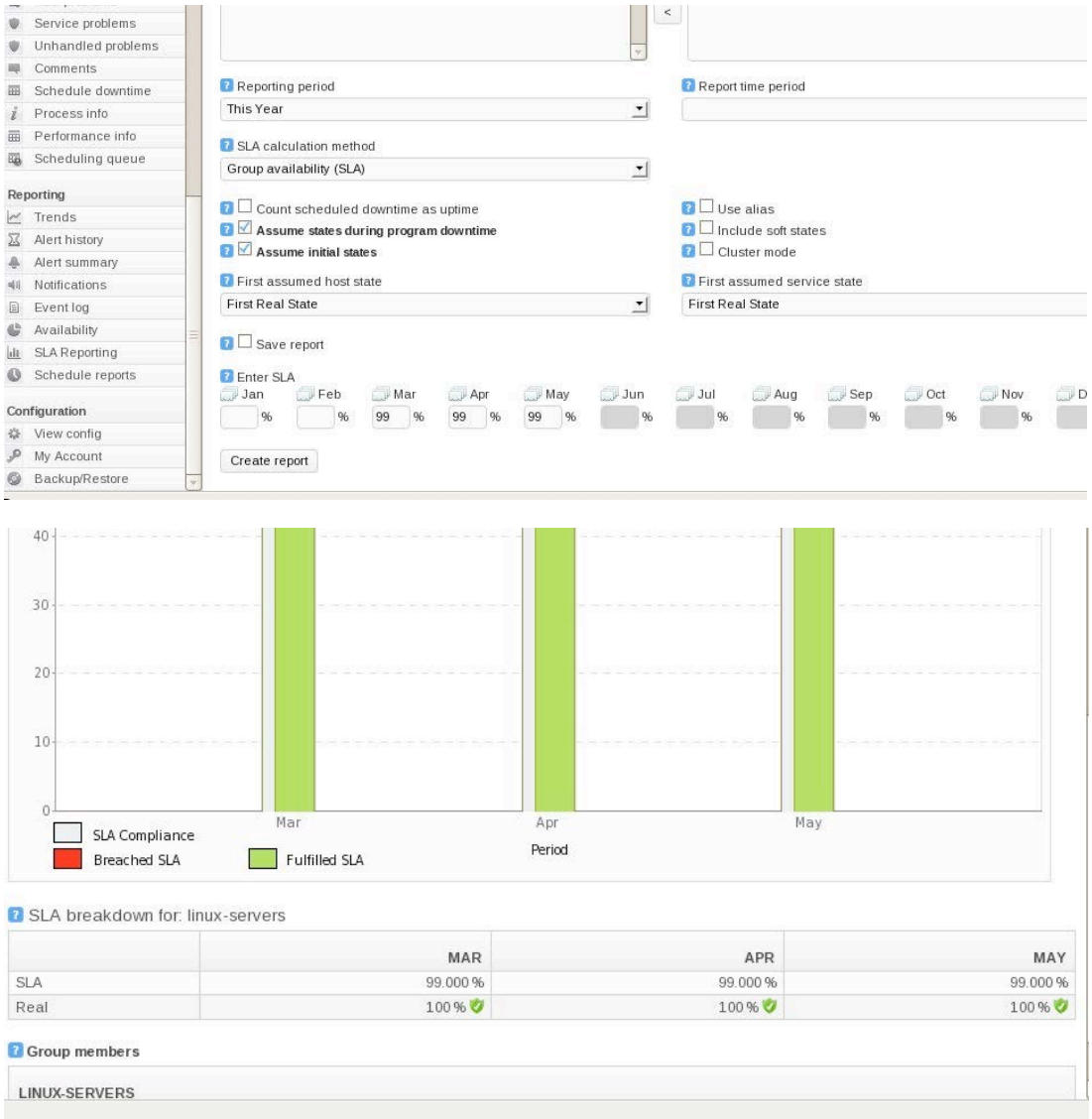



Figura 4.20 – OP5 – Informação de Cumprimento SLA - Ninja

### 4.2.3.3 – Icinga

O projecto Icinga assume-se como uma ruptura em relação ao desenvolvimento oficial do Nagios. Ao contrário das soluções avaliadas anteriormente, o Icinga não necessita de uma instalação prévia do Nagios visto que o Icinga se trata do próprio Nagios mas desenvolvido por uma equipa diferente que se propõe desenvolver uma solução exclusivamente gratuita, não a menosprezando em favor de uma solução comercial. Como compromisso de trabalho, a página web do Icinga possui um relógio em contagem decrescente marcando a cada altura o tempo para o lançamento da próxima versão.

Embora mantendo a promessa de continuar sempre compatível com os *plugins* do Nagios, razão da popularidade da plataforma, o Icinga possui já uma arquitectura de programa bastante diferente do Nagios, procurando melhorar o desempenho e ser mais modular.

Parte do sítio web do Icinga é dedicado unicamente a ilustrar as diferenças entre Icinga e Nagios em todos os aspectos, desde a arquitectura à API, passando pela documentação. A tabela seguinte fornecida pela Icinga com um resumo das principais difenças entre o Icinga e as versões Core e XI do Nagios está disponível na secção de anexos sob o título Icinga Vs. Nagios, com um aspecto semelhante ao excerto apresentado abaixo.

| | ICINGA | NAGIOS 3.2.3 | NAGIOS XI |
|---|---|---|---|
| Monitor unlimited hosts | Free | Free | $ 2,495 USD |
| CORE | | | |
| Distributed system | ✓ | ✗ | ✗ |
| Distributed monitoring | ✓ | ✓ | ✓ |
| SLA reports | ✓ | ✗ | ✓ |
| Virtual appliance | ✓ | ✗ | ✓ |
| Databases supported | MySQL<br>PostgreSQL<br>Oracle | MySQL | MySQL |

Tabela 4.4 — Tabela comparativa Icinga Vs. Nagios – Excerto [50]

Ao contrário do Nagios, cujo interface CGI comunica directamente com a ferramenta, no caso do Icinga o interface web é um software independente que consulta a base de dados e, quando necessita de comunicar com o *core*, fá-lo via API como qualquer outro plugin. Desta forma, os vários componentes da arquitectura podem estar espalhados por máquinas diferentes, se necessário, fornecendo maior segurança e redundância. No caso de um interface web falhar, outro pode de imediato assumir o seu lugar.

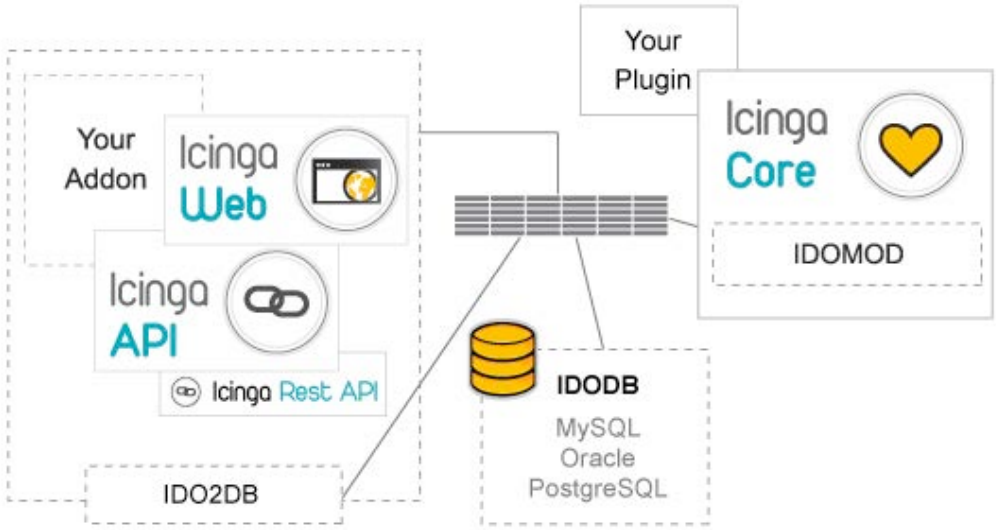



Figura 4.21 – Arquitectura Icinga [50]

A escrita de plugins para Icinga é também bastante mais facilitada pois o programador tem apenas que se preocupar com a API da ferramenta, ao contrário do Nagios onde, em addons mais complexos, o programador poderia ter de lidar com a ferramenta, o CGI do interface e as NDOutils caso utilizasse funções de base de dados, sendo o programador do addon ou plugin o responsável por fazer a tradução dos resultados da base de dados para a ferramenta ou vice versa.

A interface gráfica pode ser utilizada em modo clássico, com o painel javascript detentor de um aspecto bastante semelhante ao painel em CGI do Nagios, ou então efectuar o acesso através de Icinga Web, uma interface em Ajax que permite a adição de vários componentes por parte do utilizador denominados *Cronks* que juntam funcionalidades aos menus.

A título de exemplo, pode adicionar-se um *Cronk* que permita a visualização directa dos gráficos desenhados pelo PNP4Nagios.

Existe igualmente uma versão para dispositivos móveis, o Icinga mobile, com versões para iOS, Android e brevemente Blackberry que permitem aceder ao Icinga web num formato

conveniente sem necessidade de utilizar um PC para consultar a informação de forma clara. Sempre que novas funcionalidades forem adicionadas ao Icinga Mobile o software actualiza-se automaticamente.

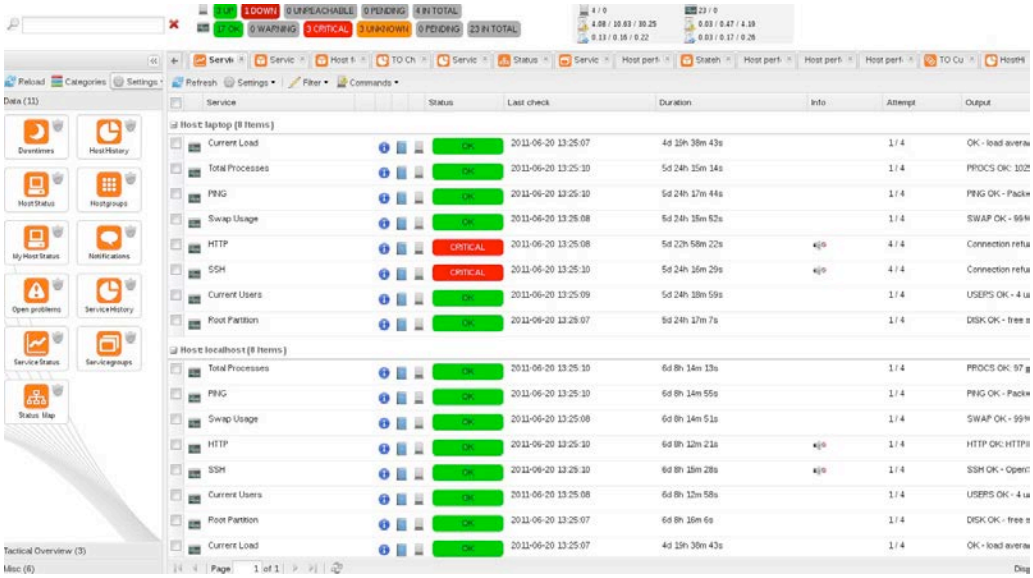


Figura 4.22 – Icinga Web – Vista geral de serviços

De referir ainda que o Icinga possui vários templates para relatórios em formato Jasper Reports Server, bastando instalar o Jasper Reports para que o mesmo consiga aceder à base de dados do Icinga e através dos templates existentes, ou outros criados pelo utilizador, crie relatórios completos sobre qualquer um dos parâmetros monitorizados. O facto do formato da base de dados do Icinga ser compatível com um dos softwares de relatórios mais conhecidos torna mais fácil a organização e interpretação da informação, particularmente útil na análise de tendências, papel para o qual o Jasper Reports é utilizado noutras áreas como, por exemplo, no SugarCRM.

Novamente não foram detectadas quaisquer diferenças de desempenho entre a instalação em Ubuntu e OpenSuse.

# Capítulo 5

# Conclusões e trabalho futuro

## 5.1 – Conclusões

Pela análise das soluções de monitorização de rede que podem encontrar-se actualmente no mercado, denota-se não só uma multiplicação do número de propostas disponíveis, mas também uma profissionalização de muitas das já existentes há bastante tempo. Este aumento de qualidade e sobretudo o início de operações comerciais da maioria das ferramentas que até há poucos anos eram exclusivamente livres é um claro reflexo da acrescida apetência do mercado da monitorização de redes informáticas. Essa apetência surge associada ao crescimento da procura por soluções de monitorização não só por grandes empresas, mas também por empresas e instituições de média dimensão, à medida que se vão apercebendo dos prejuízos de ficarem com o serviço de rede de computadores inoperacional. Actualmente cada vez mais partes das empresas se sustentam em comunicações de rede para efectuarem o seu trabalho e cada vez mais toda a comunicação da empresa se encontra unificada a nível de voz e dados, isolando por completo uma empresa que sofra um problema grave de rede.

Assim, hoje mais do que nunca, para um grande número de empresas, o tempo que se perde com uma falha de rede informática corresponde a avultados prejuízos. Se se associar a este cenário a máxima que diz que se pode usar a regra dos 80/20, 80% do tempo para descobrir onde está o problema e 20% do tempo para o resolver, então é facil de se perceber o porquê de cada vez mais empresas procurarem soluções de monitorização que permitam descobrir rapidamente qual a localização do problema. A poupança que a solução de monitorização cria nesses momentos espera-se que compense, em muito, o preço da ferramenta.

O aumento da concorrência tem também propiciado o aumento da qualidade das várias ferramentas no mercado, sejam elas livres ou comerciais. Começam cada vez mais a surgir soluções híbridas que combinam em maior ou menor parte a monitorização passiva, através da análise de registos *a posteriori*, com monitorização activa que procura espelhar o que se passa a cada momento na rede, com ou sem recurso a agentes. Ao longo dos últimos dois anos, as soluções que ainda não contemplavam instalações distribuidas ou balanceadas começaram a caminhar nesse sentido, ficando assim mais seguras e robustas.

Nos requisitos do trabalho proposto para esta dissertação era pedida uma solução de monitorização, de entre as existentes no mercado, e que respondesse às necessidades de vigilância de um sistema composto por web servers, application servers e servidores de base de dados Oracle, sendo dada grande preferência a soluções de software livre. A solução deveria avisar em tempo útil o gestor de rede da iminência ou probabilidade de uma falha e qual a sua localização. Assim, a decisão óbvia a tomar foi a de considerar as opções de monitorização activa, implementadas como software livre e que reunissem o melhor portefólio de capacidades de monitorização.

Dessa forma foram instaladas e configuradas cinco soluções completamente funcionais, de entre as de maior expressão no mercado e mais capazes e algumas delas derivadas, que apesar de não terem ainda a mesma expressão apresentam já avanços significativos em relação à sua ferramenta base. Consideram-se apenas cinco, pois tudo o que a ferramenta Nagios possa oferecer, as restantes em si baseadas oferecerão em excesso. De igual modo o OP5 Monitor free é uma versão limitada do sistema Merlin, sendo que existe uma solução não limitada em avaliação.

A primeira percepção adquirida é a da existência de um avanço cada vez mais marcado das versões comerciais em relação às versões livres "Core" da mesma aplicação. Tal afigura-se como normal, visto que as versões comerciais têm de ser apresentadas como soluções globais, não se podendo esperar que o cliente faça a configuração por si próprio, nem sequer que tenha de aguardar dias ou semanas até que a empresa instaladora da ferramenta de monitorização efectue a integração de todos os plugins. Desta forma é normal que as versões comerciais se apresentem cada vez mais como soluções completas imediatamente após a instalação e que as versões gratuitas tenham o seu progresso prejudicado pelo redirigir de esforços da equipa de desenvolvimento, pelo menos até que as versões comerciais estejam estabilizadas ao nível de integração de plugins e configurações.

Um desses casos é o Zenoss, cuja versão Enterprise possui bastantes mais funcionalidades que a versão Core. O Zenoss Core possui uma variedade de comandos remotos bastante mais pequena do que outras soluções de monitorização, baseando-se ainda principalmente em SNMP. No entanto o facto de não possuir nenhuma solução convincente para

monitorização de bases de dados oracle foi o facto que fez com que fosse de imediato preterido.

O Zabbix é uma solução de monitorização bastante completa de origem, tem capacidade SNMP e IPMI, mas a recolha de dados assenta sobretudo no seu agente remoto instalado nos computadores a monitorizar, sendo o mesmo capaz de uma elevada recolha de dados. O Zabbix poderá vir a ser uma solução que cubra por completo os requisitos previstos para o trabalho desta dissertação assim que seja lançada a nova versão do Orabbix, o plugin de monitorização de base de dados Oracle para Zabbix. No entanto, as soluções baseadas em Nagios apresentam plugins de monitorização bastante mais completos neste aspecto, o que faz com que a escolha tenda para estas últimas em detrimento do Zabbix.

Das três restantes soluções, duas delas assentam em cima de uma instalação Nagios e outra substitui-se ao próprio Nagios, sendo um novo ramo de desenvolvimento do mesmo, executado por uma equipa diferente da original.

Assim, todas elas partilharam a integração dos mesmos três plugins seleccionados para efectuar a monitorização de computadores Microsoft Windows, base de dados Oracle e testes de Web Injection. Por outro lado, todas partilham igualmente o facto de implementarem formas próprias de comunicação com plugins seus, embora mantendo a retrocompatibilidade com NRPE.

O Check_Mk consegue, através do seu agente, efectuar uma comunicação bastante mais eficaz do que o Nagios base, abrindo uma única conexão e recebendo toda a informação do computador a monitorizar de uma vez só, guardando-a numa base de dados e processando-a posteriormente passando os valores ao Nagios. A utilização desta solução híbrida consegue uma economia de processamento tão maior quanto o número de plugins que se chamariam individualmente sem o check_mk.

O Merlin é um *Event Broker* que intercepta toda a comunicação efectuada pelo Nagios e efectua o seu próprio processamento paralelo, bastante mais eficiente, estando desenhado para ser um módulo que trabalhe em instalações distribuidas e redundantes. O Merlin possui velocidades de armazenamento e consulta de dados muito interessantes, conseguindo quase obter uma relação de tempo linear. A interface gráfica compatível com o Merlin intitula-se Ninja e foi de grande dificuldade a sua instalação em Ubuntu ou OpenSuse, acabando por apenas se conseguir fazer com sucesso em CentOS. O Ninja beneficia de um desenvolvimento muito activo, com "bugs" a serem corrigidos todos os dias no repositório de desenvolvimento, sendo que os programadores resolvem rapidamente os problemas encontrados pelos utilizadores e aceitam de bom grado sugestões de solução. As sugestões de resolução de problemas no código sugeridas ao longo desta dissertação foram em geral aplicadas e submetidas no repositório de código do ninja num espaço de 2 a 3 dias. No entanto, para um ambiente de produção existem

reservas na sua recomendação, principalmente devido a problemas inesperados que costumam surgir aquando das actualizações para novas versões.

O Icinga é um desenvolvimento alternativo do Nagios, realizado por um grupo de programadores descontentes com o rumo da plataforma. Tomado em mãos o processo de desenvolvimento foi dada prioridade a pedidos vários da comunidade, foi alterada a arquitectura para um sistema modular e criada uma API que unifica a comunicação entre os plugins e toda a plataforma, sendo também alterado o método de armazenamento em base de dados. O Icinga pode ser acedido através da interface clássica ou de uma nova interface em ajax denominada Icinga-Web, que possui igualmente uma versão para dispositivos móveis. O desenvolvimento é feito por uma equipa de mais de 25 pessoas que mantém um calendário de etapas de desenvolvimento público.

Tendo em conta os avanços presentes nesta plataforma e o seu compromisso em manter a retrocompatibilidade com os plugins do Nagios, esta apresenta-se como a melhor opção para ser utilizada nas funções de monitorização requeridas ao trabalho proposto para esta dissertação. O Check_mk possui igualmente todas as condições para poder ser utilizado em segurança num ambiente de produção e seria uma opção totalmente aceitável pois teve um desempenho absolutamente sem reparos. Os únicos argumentos que justificam, de certa forma, o pendor para o Icinga são a quantidade de programadores da equipa quando comparada com o programador solitário do check_mk, prevendo-se que no futuro a velocidade de desenvolvimento do Icinga continue bastante superior e utilização de uma nova API que facilita o desenvolvimento de plugins em Icinga. Reitera-se, no entanto, a absoluta validade de ambas as opções em relação aos objectivos propostos.

A solução final apresentada e a ser implementada já fora da rede de testes criada para esta dissertação é a baseada em Icinga devido às funcionalidades apresentadas, compatibilidade com a maior base de plugins disponível em termos de monitorização devido à sua origem Nagios, facilidade de criação de novos plugins através de uma nova API, aspecto gráfico intuitivo, armazenamento de dados em formato compatível com soluções de análise de terceiros e desenvolvimento acelerado com planeamento público, permitindo aos utilizadores saber, a cada momento, quais as novas funcionalidades e quando estarão disponíveis.

## 5.2 – Satisfação de Objectivos e Trabalho Futuro

O atingir de objectivos foi satisfatório. No final da dissertação estavam configurados 3 protótipos funcionais que poderiam ser transpostos para fora da rede de testes e que cobririam todos os requisitos pedidos ao trabalho.

Ao longo do semestre transacto, e por razões alheias ao CICA, não foi possível que o desenvolvimento pudesse continuar durante algum tempo na FEUP, o que fez com que tivesse de ser criada uma nova rede de testes e reiniciado o processo de instalações, configurações e testes. Foi um atraso, no entanto, recuperável, não afectando as conclusões do trabalho.

A nível de trabalho futuro seria interessante, obviamente, a implementação de um dos sistemas testados em ambiente de produção. Será igualmente de ter em atenção a saída de plugins novos para Zabbix, que poderão dar nova força a essa solução, bem como acompanhar o desenvolvimento da interface Ninja, esperando que fique mais estável, ou pelo menos mais amigável no tocante a instalações e actualizações para que o Merlin possa ser melhor aproveitado.

Estando esta solução implementada de futuro, seria igualmente de pensar em expandir a mesma, passando a uma implementação distribuida que possa abarcar mais serviços e equipamentos dentro da rede da FEUP.

# Referências


[1] K. G. Anagnostakis, S. Ioannidis, S. Miltchev, J. Ioannidis, J. M. Smith. Efficient Packet Monitoring for Network Management. In Proceedings of IFIP/IEEE Network Operations And Management Symposium (NOMS) (2002).

[2] "Regras para a Apresentação de Dissertações de Cursos de Mestrado da FEUP", Faculdade de Engenharia da universidade do Porto (Junho de 1995).

[3] Ahsan Habib, Sonia Fahmy, Srinivas R. Avasarala, Venkatesh Prabhakar, Bharat Bhargava. On detecting service violations and bandwidth theft in QoS network domains, Computer Communications 26, pp.861–871 (2003).

[4] Alan Bivens et al. Agent-Based Network Monitoring, In Proceedings of Autonomous Agents 99 Conference, Seattle, pp. 41-53 (1999).

[5] Andrew Moore, James Hall, Christian Kreibich, Euan Harris, and Ian Pratt. Architecture of a Network Monitor. Passive & Active Measurement Workshop (2003).

[6] Mike Fisk, Steven A. Smith, Paul M. Weber, Satyam Kothapally, Thomas P. Caudell. Immersive Network Monitoring, In Proceedings of the 2003 Passive and Active Measurement Workshop (2003).

[7] Ballora, M. and D. L. Hall. "Do you see what I hear: Experiments in multi-channel sound and 3-D visualization for network monitoring?", Proceedings of SPIE--the international society for optical engineering p. 7709 (2010).

[8] Jing Li, Jing Li Chengchen, Bin Liu. Monitoring Large Flows in Network, Department of Computer Science and Technology, Tsinghua University, Beijing, P. R. China (2005).

[9] Fetahi Wuhib, Rolf Stadler, Alexander Clemm. Decentralized Service Level Monitoring using Network Threshold Crossing Alerts, IEEE Communications Magazine Vol.44, No. 10 (2006).

[10] Augusto Ciuffoletti, Yari Marchetti, Antonis Papadogiannakis, Michalis Polychronakis. End-to-end Network Monitoring Infrastructure, CoreGRID Technical Report Number TR-0126, (February 2008).

[11] Hongjie Sun. An Integrated Network Performance Monitor System, Third International Symposium on Intelligent Information Technology and Security Informatics (2010)

[12] Gion Reto Cantieni, Gianluca Iannaccone, Chadi Barakat, Christophe Diot, Patrick Thiran. Reformulating the Monitor Placement Problem: Optimal Network-Wide Sampling, Proceedings of ACM CoNEXT (2006)

[13] Wan Ming-Han, Mong-Fong Horng. An Intelligent Monitoring System for Local-Area Network Traffic, Eighth International Conference on Intelligent Systems Design and Applications (2008).

[14] J.Postel, Internet control message protocol, RFC 792, IETF (1981).

[15] J.Postel, User datagram protocol, RFC 768, IETF (1980).

[16] J.Postel, Transmission control protocol, RFC 793, IETF (1981).

[17] S.Shalunov, B.Teitelbaum, and M.Zekauskas, "A one-way delay measurement protocol. IPPM work in progress", IETF (2001).

[18] http://allendowney.com/research/clink/, consulta em Junho 2010.

[19] G.Jin, B.Tierney. "Netest: A Tool to Measure axiom Burst Size, Available Bandwidth and Achievable Throughput", Proceedings of the 2003 International Conference on Information Technology Research and Education (2003).

[20] http://www.caida.org/tools/utilities/others/pathchar, consulta em Junho 2010.

[21] http://www.kitchenlab.org/www/bmah/Software/pchar/, consulta em Junho 2010.

[22] http://www-didc.lbl.gov/NCS/, consulta em Junho 2010.

[23] T. Michael Silver. Monitoring Network and Service Availability with Open-Source Software, Information Technology and Libraries (March 2010).

[24] Rolf Stadler, Mads Dam, Alberto Gonzalez, Fetahi Wuhib. Decentralized Real-time Monitoring of Network-wide Aggregates, Proceedings of the 2nd Workshop on Large-Scale Distributed Systems and Middleware (2008).

[25] Demetris Antoniades et al. LOBSTER: A European Platform for Passive Network Traffic Monitoring, Tridentcom March 18–20, 2008, Innsbruck, Austria (2008).

[26] C. Toland, C. Meenan, M. Warnock, P. Nagy. Proactively Monitoring Departmental Clinical IT Systems with an Open Source Availability System, Journal of Digital Imaging, Vol 20, Suppl 1, pp. 119-124 (2007).

[27] Solange Lima, Paulo Carvalho, Vasco Freitas. Tuning Active Monitoring in Multi-service IP Networks, In Perf. Modelling and Evaluation of Heterogeneous Networks (HET-Nets'04) (2004)

[28] Christopher L. Hayes, Yan Luo. DPICO: A High Speed Deep Packet Inspection Engine Using Compact Finite Automata, ANCS'07 (2007).

[29] Dario Rossi, Silvio Valenti. Fine-grained traffic classification with Netflow data, IWCMC '10, June 28– July 2, 2010, Caen, France (2010).

[30] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Carey Williamson. Identifying and Discriminating Between Web and Peer to Peer Traffic in the Network Core, WWW2007 May 8–12, Canada (2007).

[31] Pere Barlet-Ros, Gianluca Iannaccone, Josep Sanjuàs-Cuxart, Josep Solé-Pareta. Robust network monitoring in the presence of non-cooperative traffic queries, Computer Networks 53, pp. 310–321 (2009).

[32] Kwang Sik Shin, Jin Ha Jung, Jin Young Cheon, Sang Bang Choi. Real-time network monitoring scheme based on SNMP for dynamic information, Journal of Network and Computer Applications 30, pp.331-353 (2007).

[33] http://oss.oetiker.ch/rrdtool/, consultado em Julho 2010.

[34] http://www.zabbix.com, consultado em Junho 2010.

[35] http://www.nagios.org, consultado em Abril 2011.
[36] http://www.zenoss.com, consultado em Abril 2011.
[37] http://www.cacti.net, consultado em Abril 2011.
[38] http://www.shinken-monitoring.org, consultado em Abril 2011.
[39] Sandvine 2010 Mobile Internet Phenomena Report. http://www.sandvine.com/news/global_broadband_trends.asp, (2010).
[40] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2009-2014. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf. (Feb. 2010).
[41] Morgan Stanley Mobile Internet Report. Rep. 15 Dec. 2009. http://www.morganstanley.com/institutional/techresearch/pdfs/2SETUP_12142009_RI.pdf. (2009)
[42] Boleslaw K. Szymanski, Alan Bivens, Yu Liu, Kiran Madnani, Anand Sastry. The Genesis Project: Network Decomposition in Monitoring and Simulation for Network Management and Intrusion Detection, Virtual World Simulation Proceedings (2002)
[43] http://www.nagios.org, consultado em Junho 2011
[44] Chris Burgess. Monitoring Anything and Everything with Nagios, The Open Web Application Security Project Summit, (200)
[45] http://tracker.nagios.org, consultado em Junho 2011
[46] http://ideas.nagios.org, consultado em Junho 2011
[47] http://labs.consol.de, consultado em Junho 2011
[48] Michael Medin. Probing the Depths of Windows, Nordic Meet on Nagios, Stockholm (2009)
[49] Andreas Ericsson. Merlin, Nordic Meet on Nagios, Stockholm (2009)
[50] http://www.icinga.org, consultado em Junho 2011
[51] http://www.op5.com e http://www.op5.org, consultado em Junho 2011
[52] http://mathias-kettner.de, consultado em Junho 2011

# Anexos

# Zenoss – Capturas de Ecrã

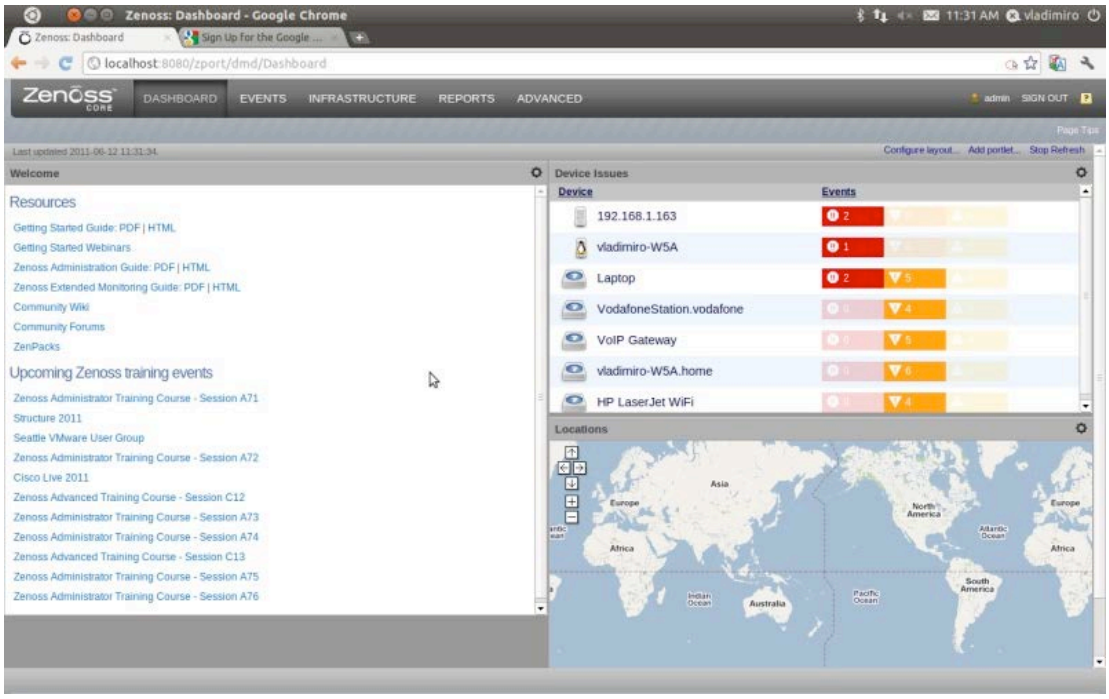



Fig. A1 – Ecrã de entrada Zenoss

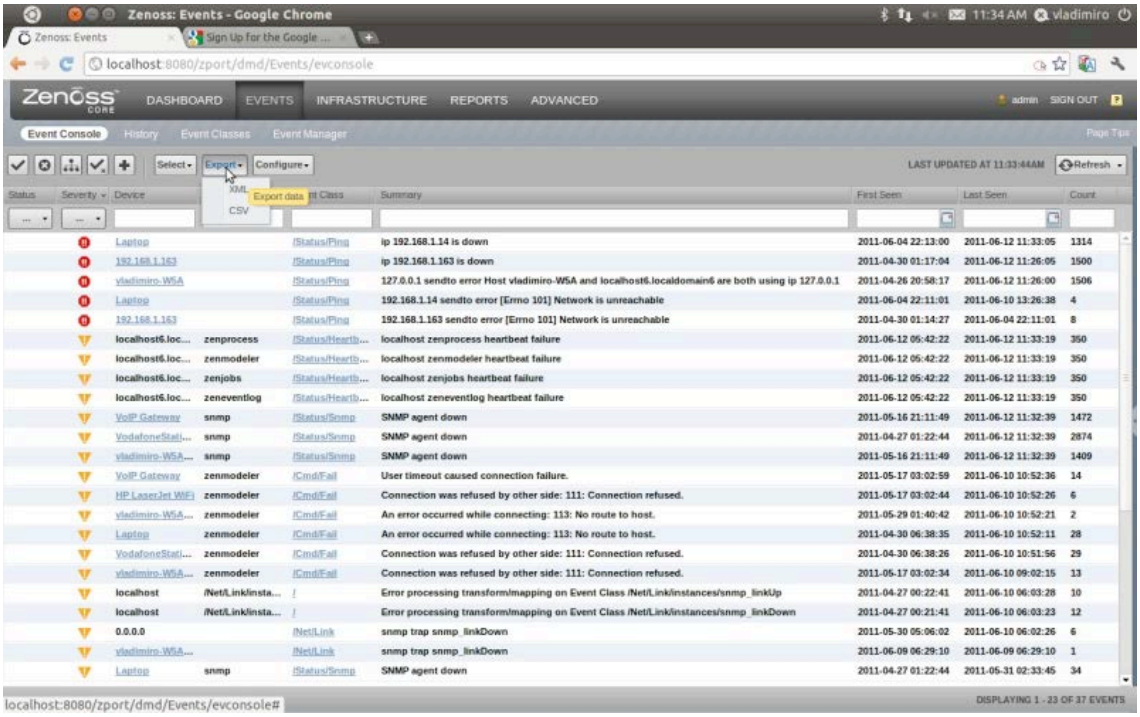



Fig. A2 – Listagem de eventos recentes Zenoss

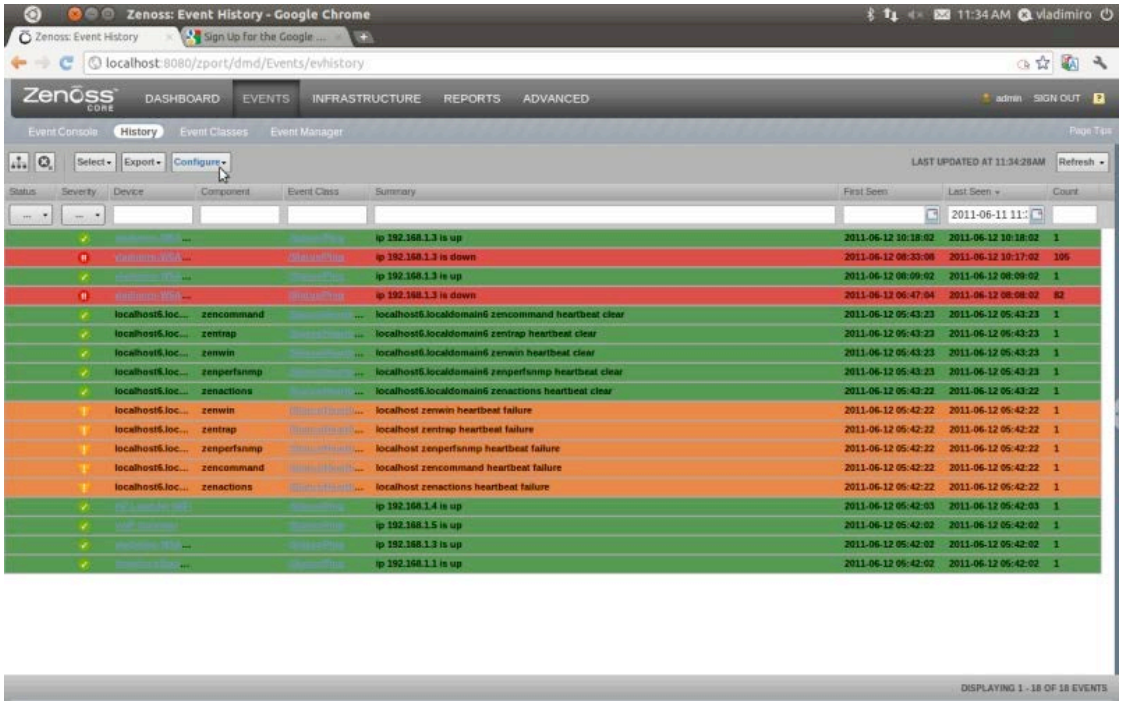



Fig. A3 – Histórico de eventos Zenoss

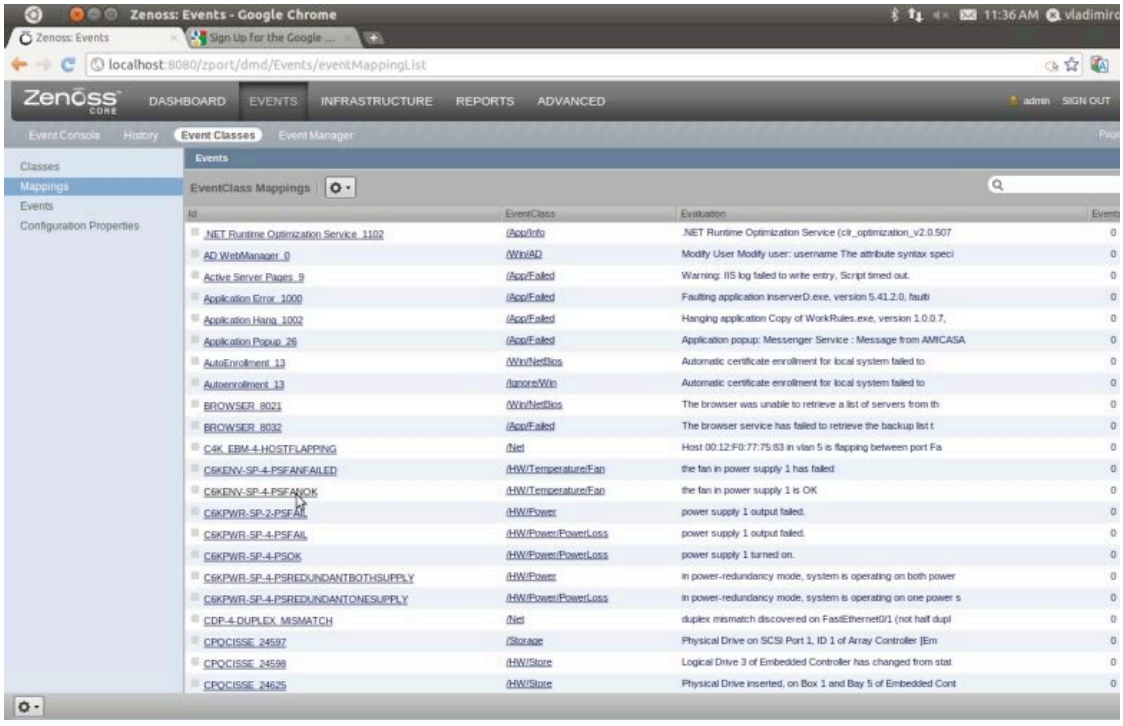



Fig.A4 – Criação e alteração de eventos Zenoss

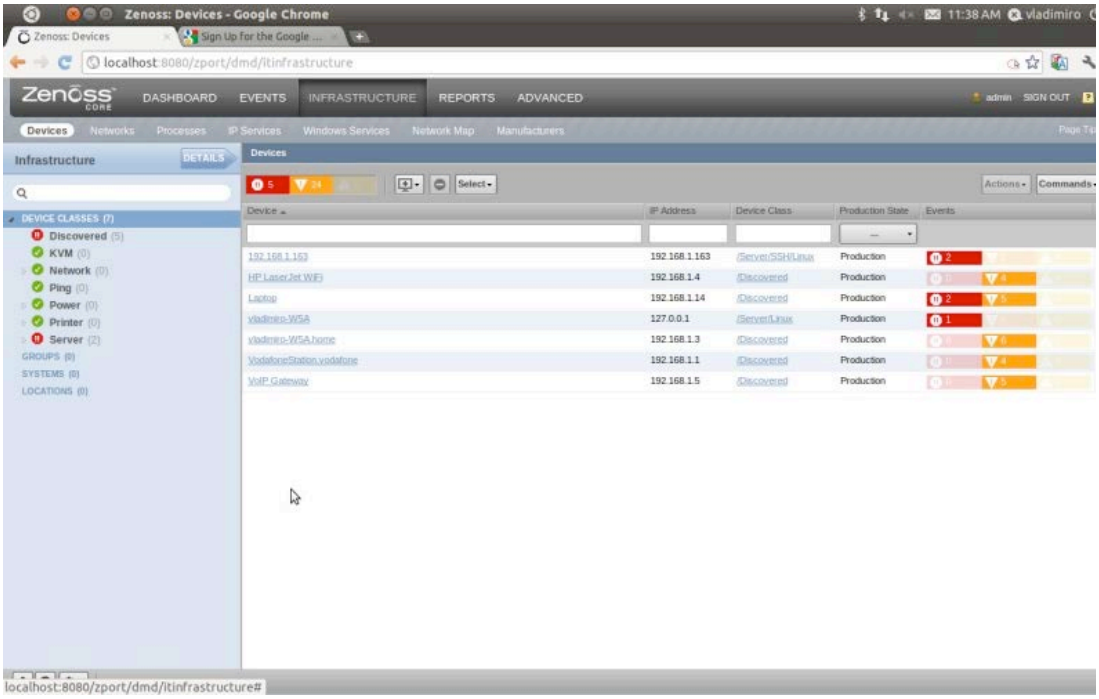



Fig. A5 – Lista de dispositivos monitorizados Zenoss

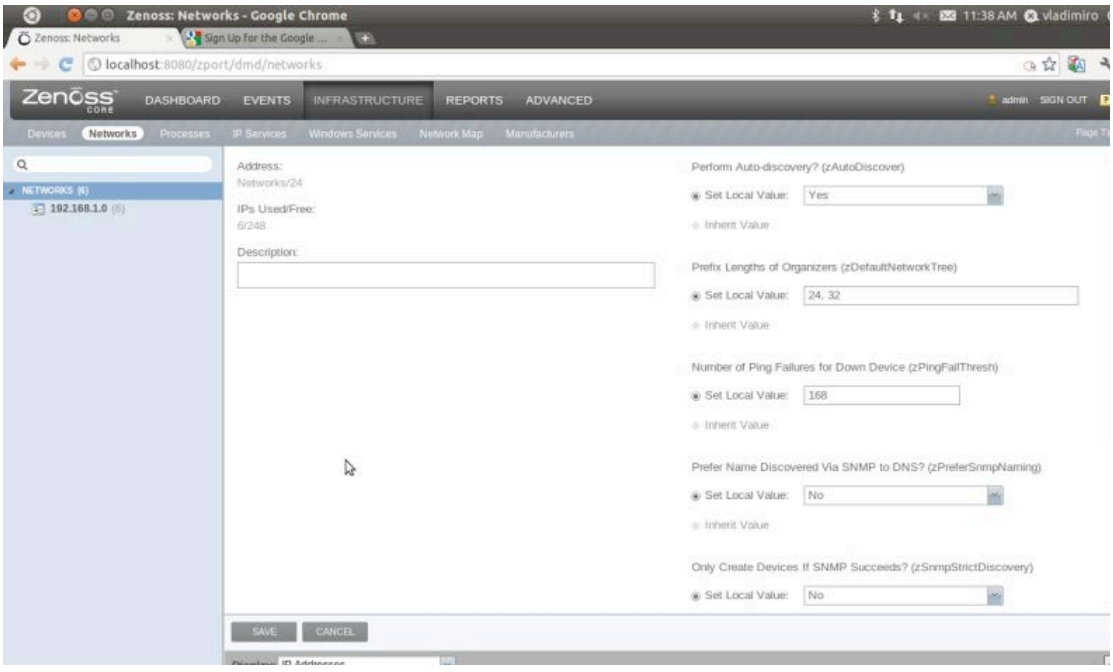



Fig. A6 – Descoberta automática da rede local

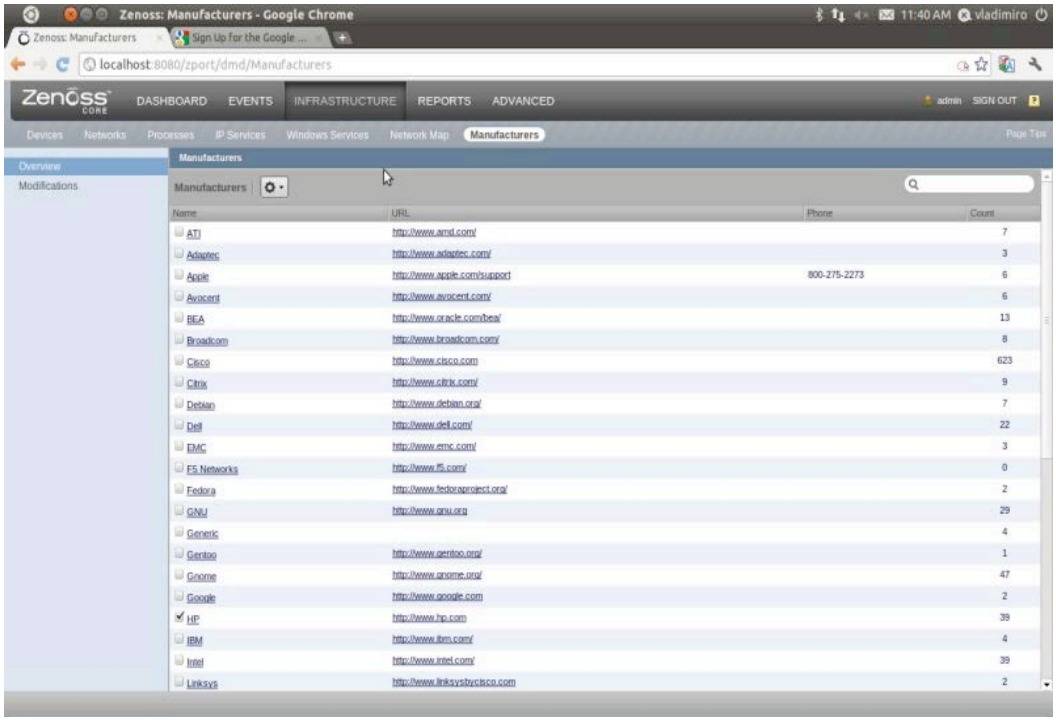



Fig A7 – Templates por fabricante

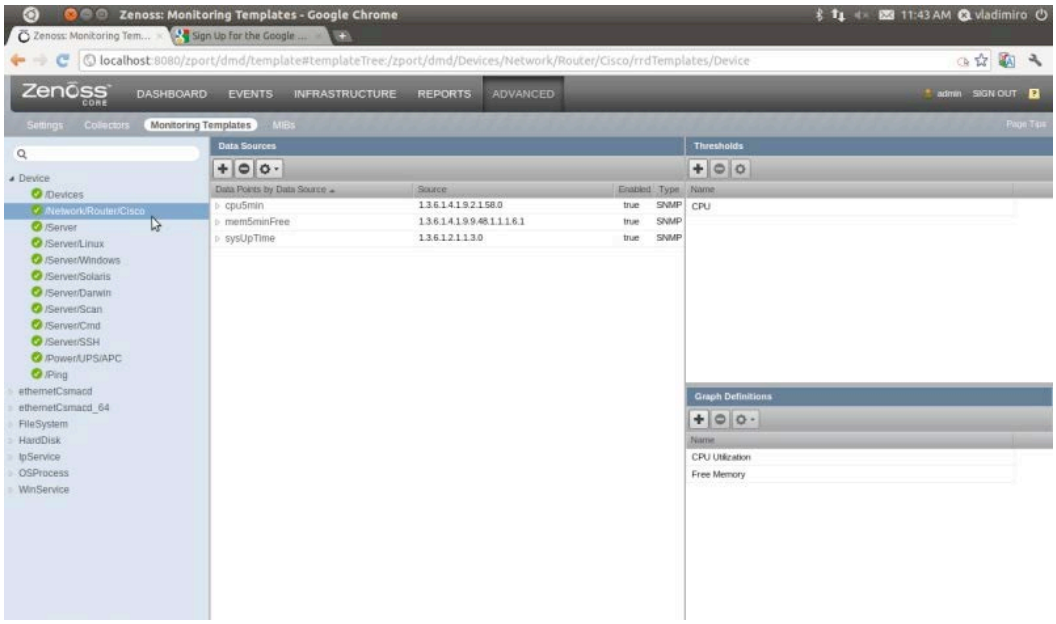



Fig. A8 – Template para router Cisco em Zenoss

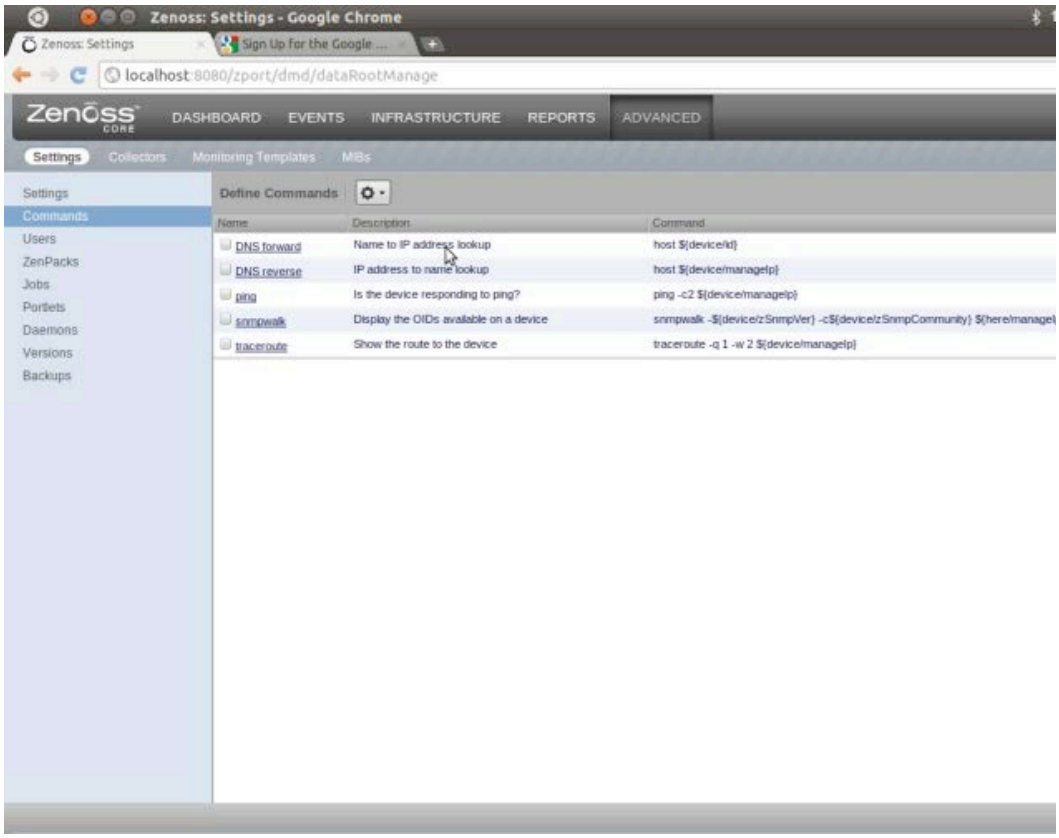



Fig. A9 – Criação de Zcommands

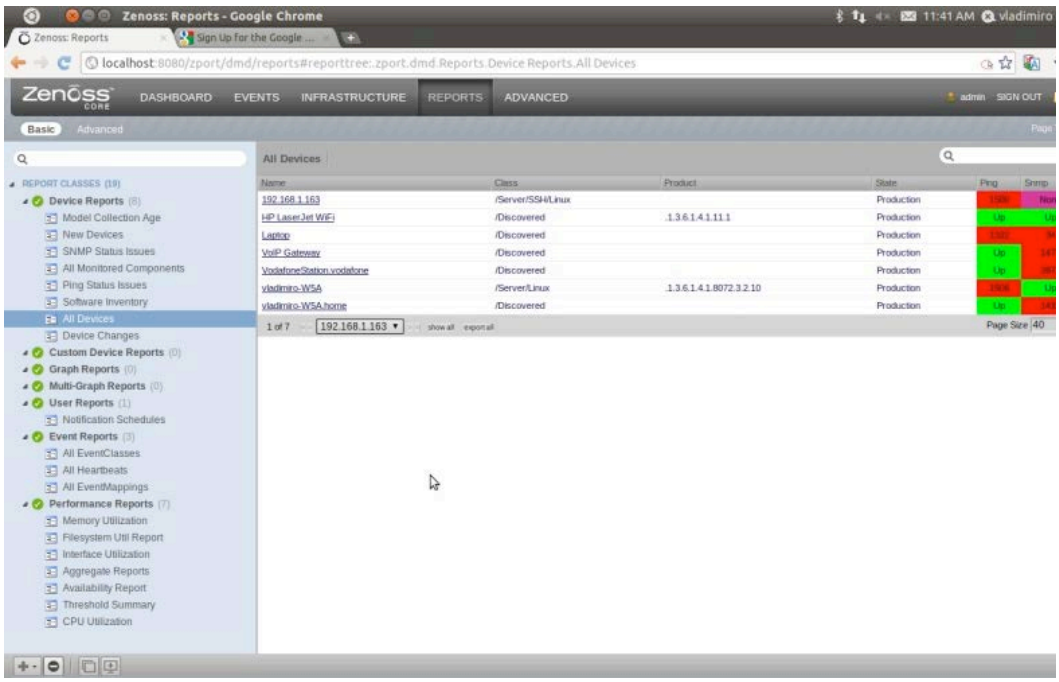



Fig. A10 – Criação de relatórios

# Zenoss – Ficheiros de Configuração

Não existem quaisquer ficheiros de configuração que necessitem de ser editados para o bom funcionamento do Zenoss. Após a criação do utilizador Zenoss no sistema e a compilação do código fonte, bastará arrancar o ficheiro binário do Zenoss que possui um servidor web próprio. Todas as configurações são feitas já dentro do interface gráfico do *software*.

# Zabbix – Capturas de Ecrã

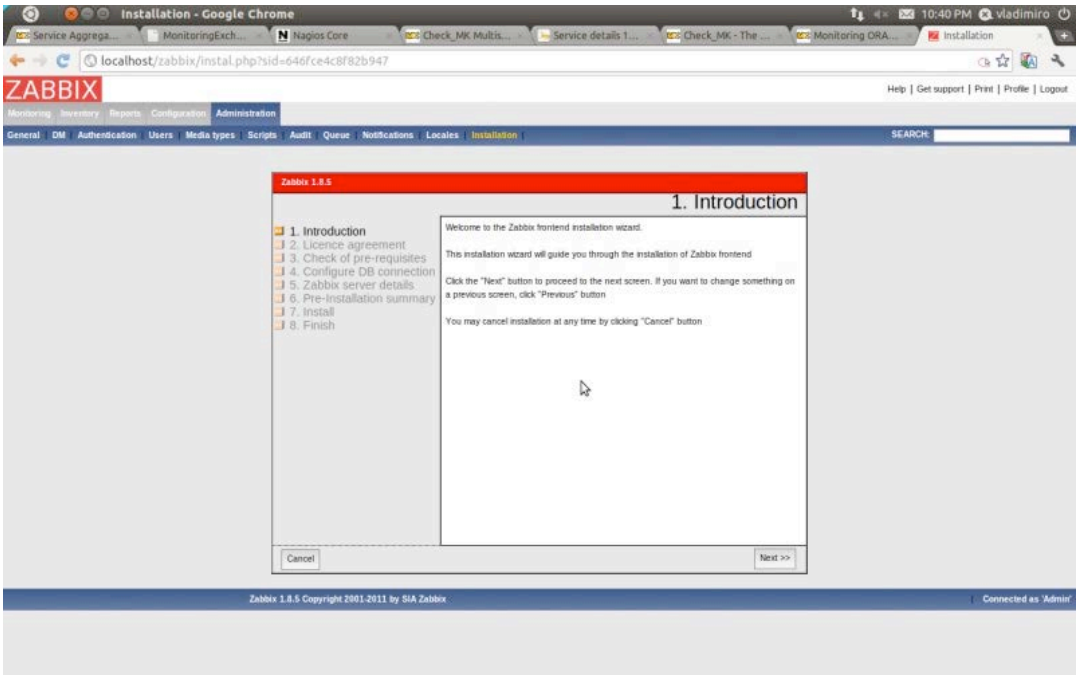



Fig. A11 – Instalação Zabbix

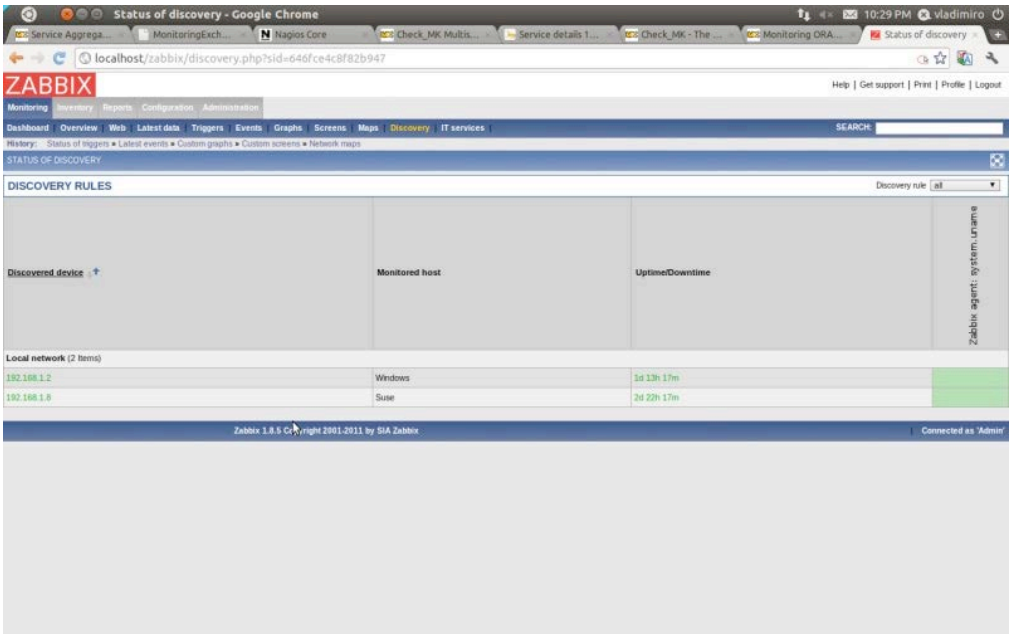



Fig. A12 – Definições de descoberta de equipamentos na rede

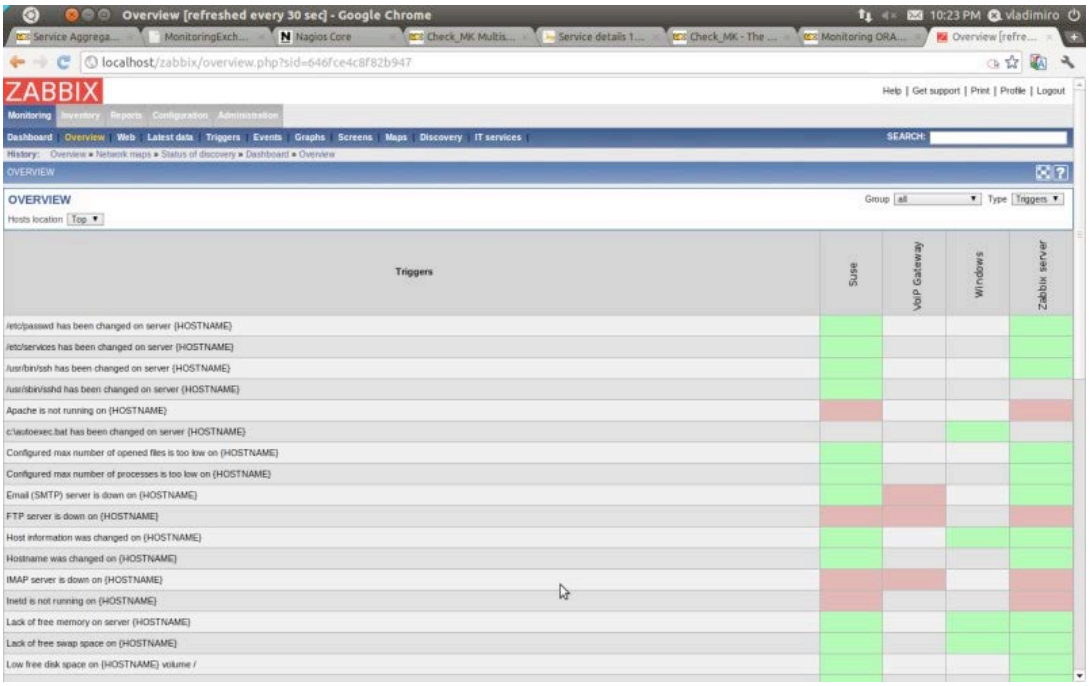



Fig. A13 – Lista de equipamentos descobertos e estado dos serviços

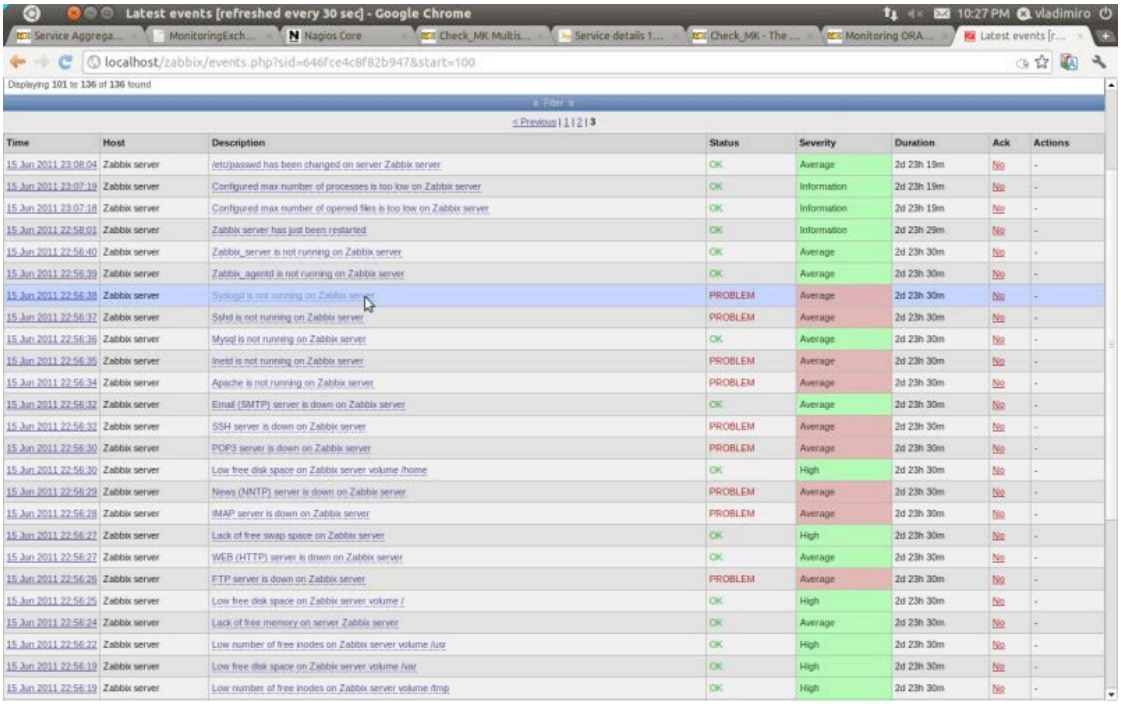



Fig. A14 – Listagem de eventos


Fig. A15 – Gráficos de serviços

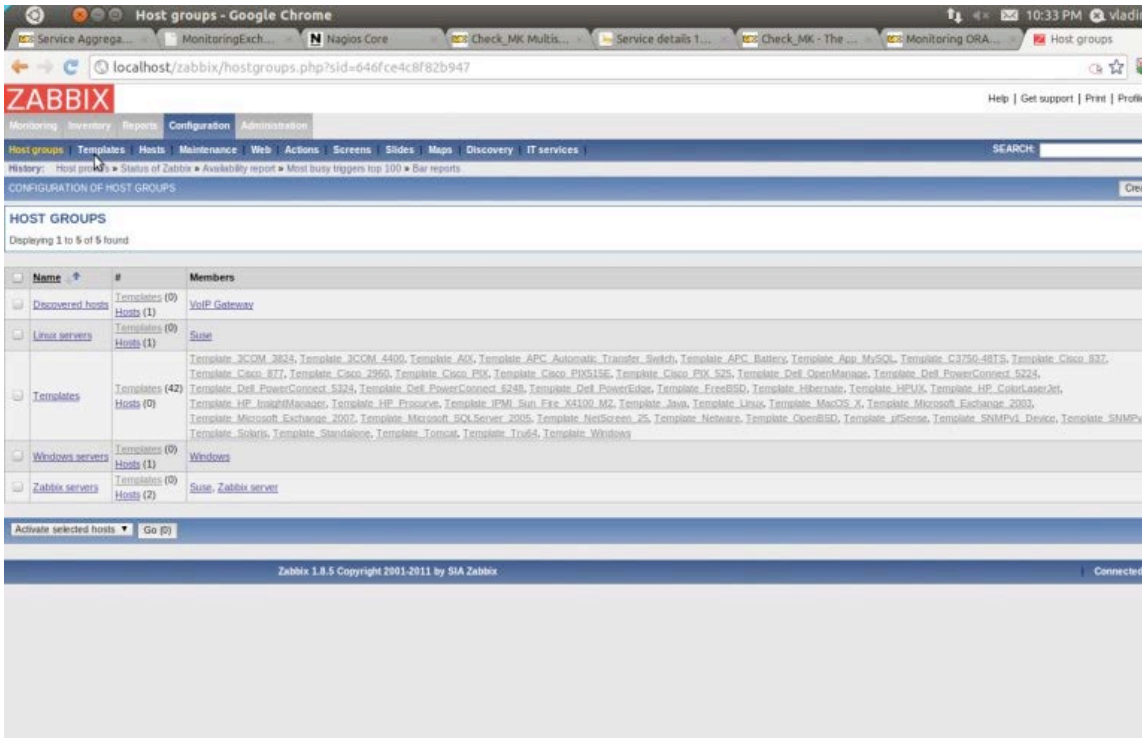



Fig. A16 – Templates para dezenas de fabricantes/modelos

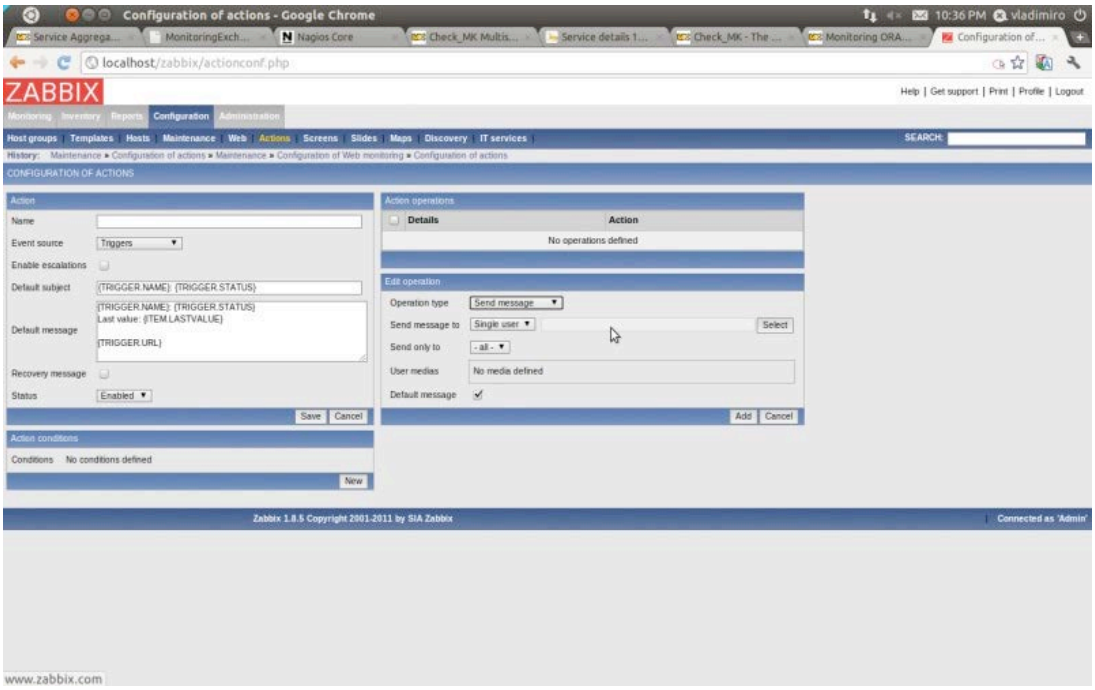



Fig. A17 – Configuração das acções a realizar em caso de alerta e modo de envio de aviso ao administrador de sistema

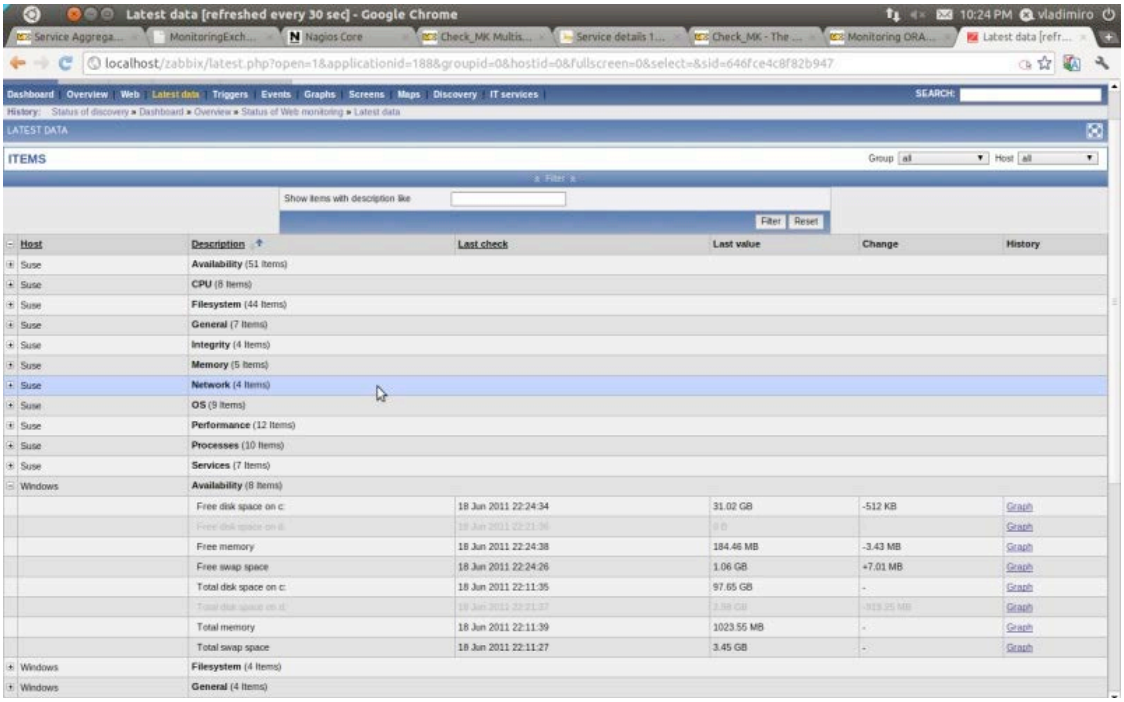



Fig. A18 – Detalhe de serviços de um equipamento com ligações para os seus gráficos

# Zabbix – Ficheiros de Configuração

A configuração do Zabbix não recorre a ficheiros de configuração. Após a criação do utilizador Zabbix no sistema operativo, é necessária a criação e preenchimendo de uma base de dados com as tabelas fornecidas junto com o código fonte do Zabbix. A ferramente exige também que se alterem os valores padrão do interpretador PHP do sistema para que possa funcionar correctamente. Realizando os passos anteriores e após a compilação do código fonte a instalação e posterior configuração do Zabbix são feitas via interface gráfica da própria ferramenta. Os valores de configuração ficam assim guardados em base de dados.

# Check_mk – Capturas de Ecrã

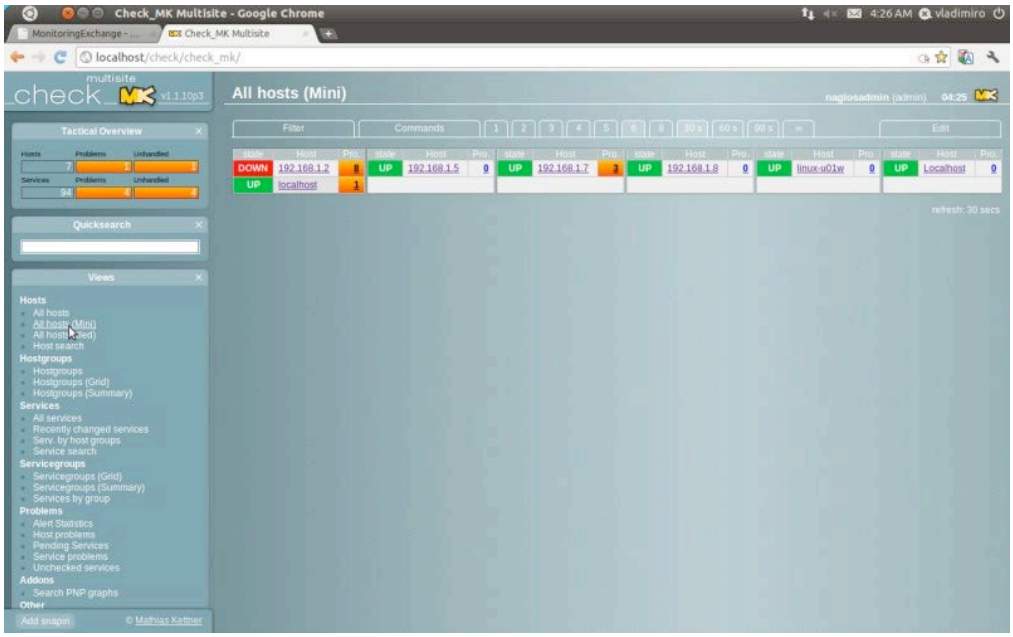



Fig. A19 – check_MK vista resumida dos equipamentos monitorizados

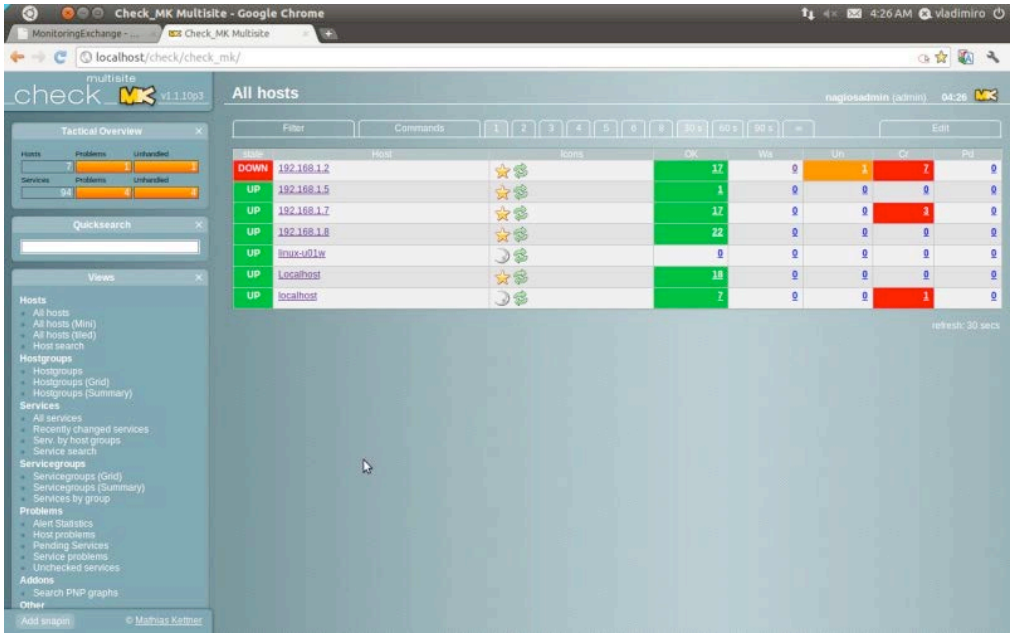



Fig. A20 – Check_MK vista geral dos equipamentos monitorizados

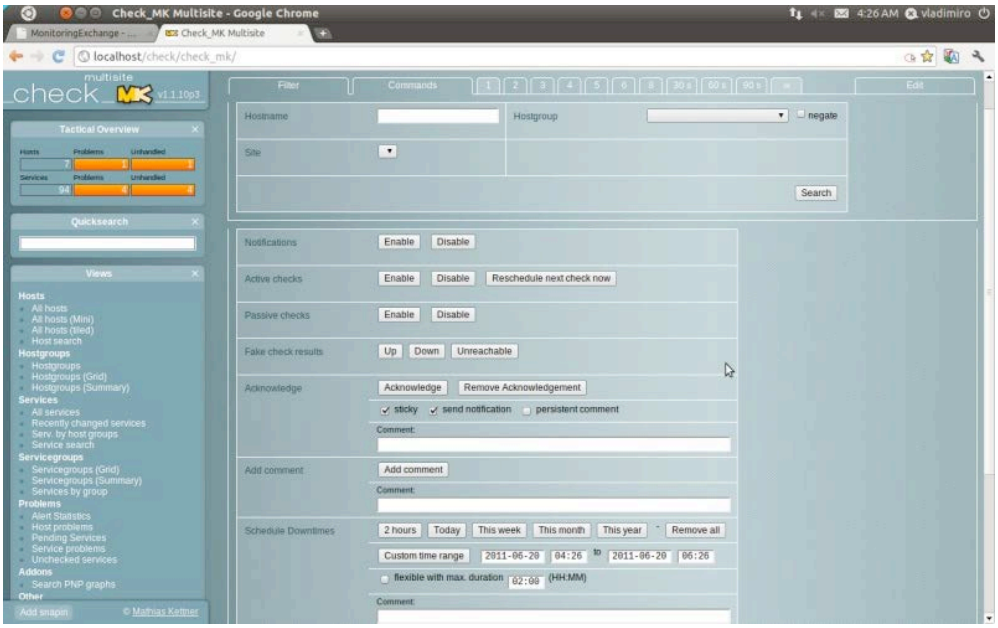



Fig. A21 – Configuração dos tipos e periodos de monitorização

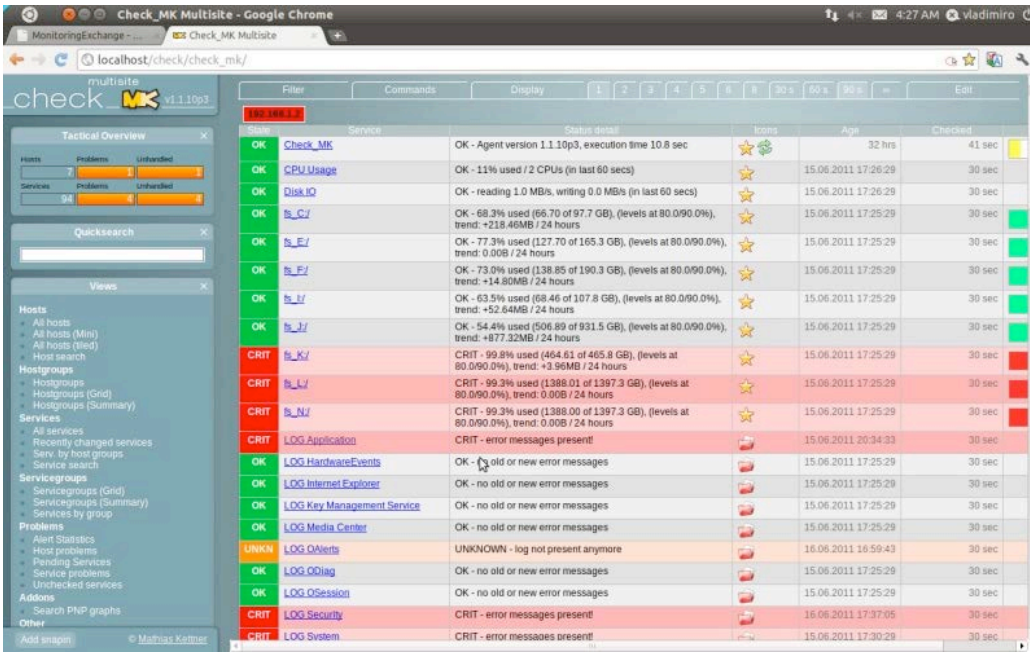



Fig. A22 – Vista do estado de todos os serviços por equipamento - Windows

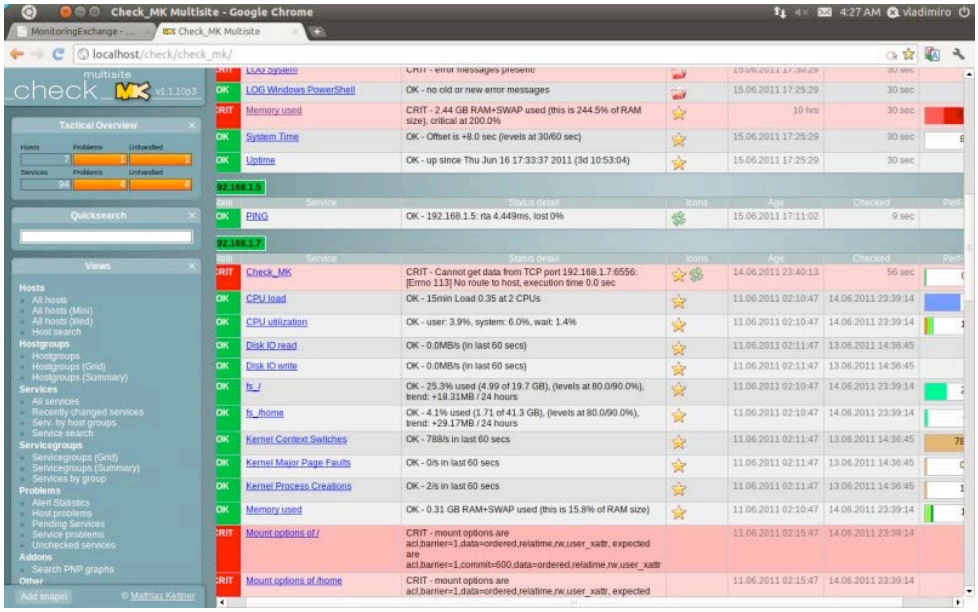



Fig. A23 – Vista do estado de todos os serviços por equipamento – Linux

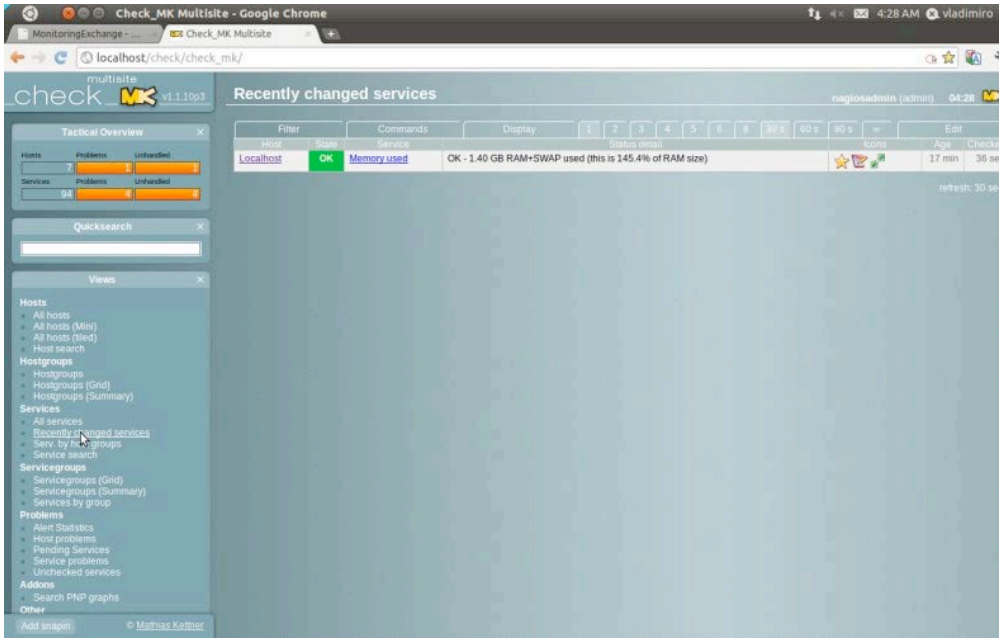



Fig. A24 – Serviços com alterações recentes

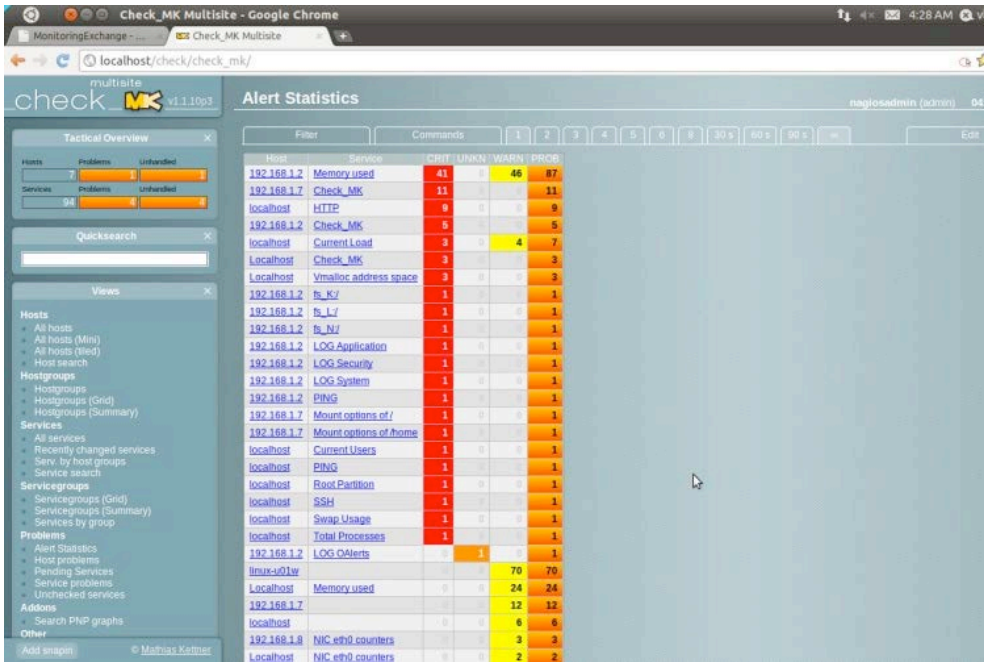



Fig. A25 – Lista de alertas


Fig. A26 – Gráficos em PNP4Nagios com acesso directo às várias funcionalidades do serviço monitorizado

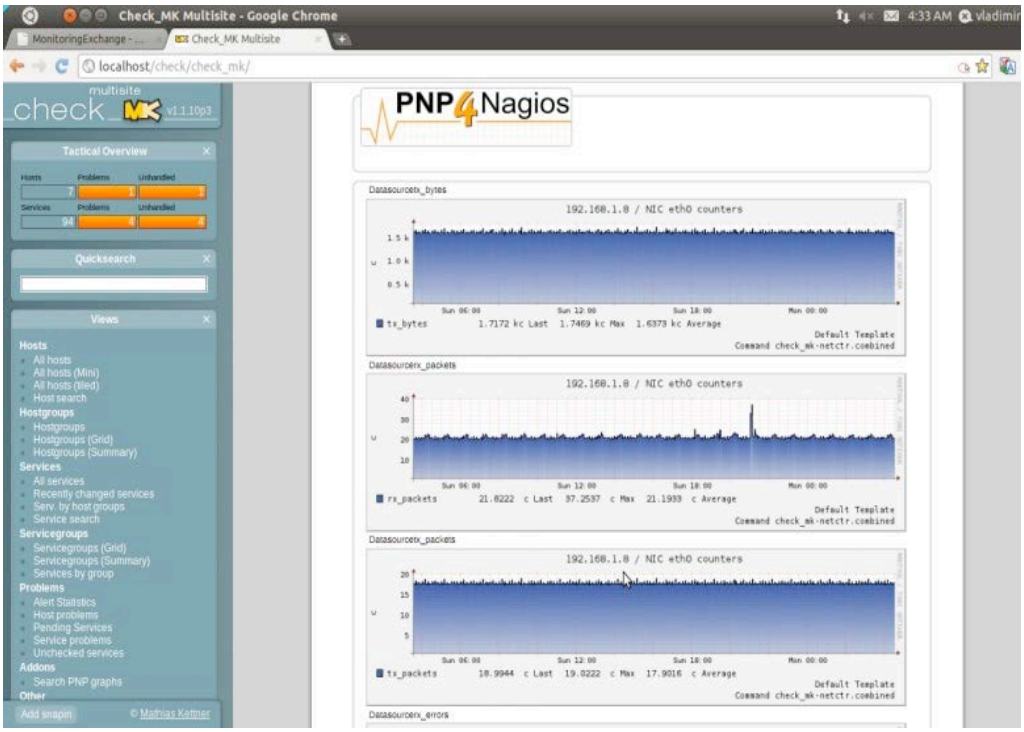



Fig. A27 – Gráficos em PNP4Nagios exportados para PDF

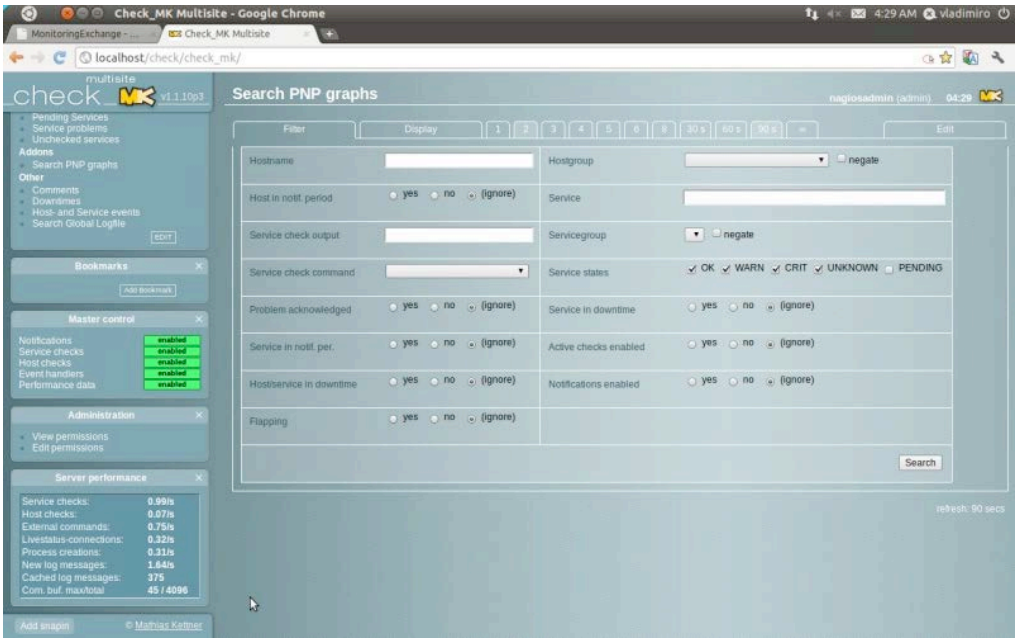



Fig. A28 – Funcionalidade de busca em gráficos de PNP4Nagios

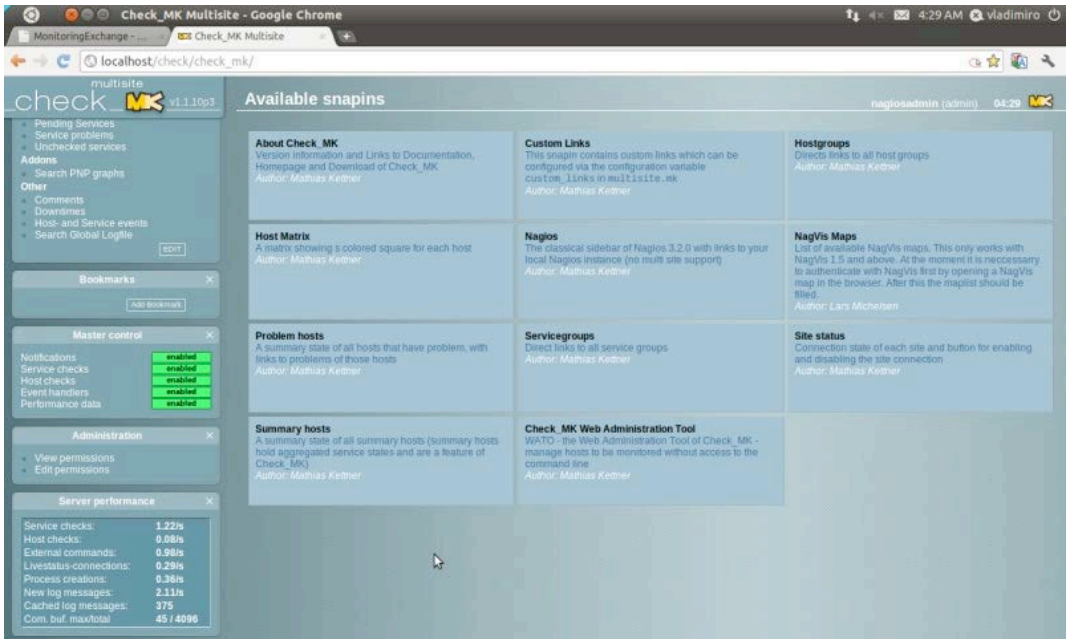



Fig. A29 – *Snapins* adicionais disponíveis para o menu lateral

# Check_MK – Ficheiros de Configuração

Check.mk - Configuração da lista de equipamentos a monitorizar

```
# Put your host names here
# all_hosts = [ "Localhost", '127.0.0.1', "192.168.1.5", "192.168.1.2",
#"192.168.1.3", "192.168.1.4", "192.168.1.7", "192.168.1.8”]
all_hosts = [ "Localhost", "192.168.1.7", "192.168.1.2", "192.168.1.8",
"192.168.1.5", "192.168.1.3", "192.168.1.4” ]
```

Check_mk_objects.cgf - Configuração dos serviços a monitorizar em cada equipamento pelo check_mk. Não surgem neste ficheiro os serviços monitorizados por *plugins* do Nagios.

```
#
# Created by Check_MK. Do not edit.
#


# ----------------------------------------------------
# Localhost
# ----------------------------------------------------

define host {
  host_name             Localhost
  use                   check_mk_host
  address               127.0.0.1
  _TAGS
  host_groups                 +check_mk
  alias                       Localhost
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
```

```
  service_description         CPU load
  check_command               check_mk-cpu.loads
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Number of threads
  check_command               check_mk-cpu.threads
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         fs_/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Disk IO read
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Disk IO write
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Kernel Context Switches
  check_command               check_mk-kernel
}
```

```
define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Kernel Major Page Faults
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Kernel Process Creations
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         CPU utilization
  check_command               check_mk-kernel.util
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Memory used
  check_command               check_mk-mem.used
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description         Vmalloc address space
  check_command               check_mk-mem.vmalloc
}

define service {
  use                   check_mk_passive
```

```
  host_name             Localhost
  service_description       Mount options of /
  check_command             check_mk-mounts
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description       NIC eth0 counters
  check_command             check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description       NIC eth1 counters
  check_command             check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description       Postfix Queue
  check_command             check_mk-postfix_mailq
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description       TCP Connections
  check_command             check_mk-tcp_conn_stats
}

define service {
  use                   check_mk_passive_perf
  host_name             Localhost
  service_description       Uptime
  check_command             check_mk-uptime
```

```
}


# Active checks

define service {
  use                   check_mk_active
  host_name             Localhost
  service_description         Check_MK
}

# ------------------------------------------------------
# 192.168.1.7
# ------------------------------------------------------

define host {
  host_name             192.168.1.7
  use                   check_mk_host
  address               192.168.1.7
  _TAGS
  host_groups                 +check_mk
  alias                       192.168.1.7
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         CPU load
  check_command               check_mk-cpu.loads
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Number of threads
  check_command               check_mk-cpu.threads
}
```

```
define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         fs_/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         fs_/home
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Disk IO read
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Disk IO write
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Kernel Context Switches
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
```

```
  service_description         Kernel Major Page Faults
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Kernel Process Creations
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         CPU utilization
  check_command               check_mk-kernel.util
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Memory used
  check_command               check_mk-mem.used
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Vmalloc address space
  check_command               check_mk-mem.vmalloc
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.7
  service_description         Mount options of /
  check_command               check_mk-mounts
}
```

```
define service {
  use                   check_mk_passive
  host_name             192.168.1.7
  service_description         Mount options of /home
  check_command               check_mk-mounts
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         NIC br0 counters
  check_command               check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         NIC eth0 counters
  check_command               check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         Postfix Queue
  check_command               check_mk-postfix_mailq
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.7
  service_description         TCP Connections
  check_command               check_mk-tcp_conn_stats
}

define service {
  use                   check_mk_passive_perf
```

```
  host_name             192.168.1.7
  service_description         Uptime
  check_command               check_mk-uptime
}


# Active checks

define service {
  use                   check_mk_active
  host_name             192.168.1.7
  service_description         Check_MK
}

# ------------------------------------------------------
# 192.168.1.2
# ------------------------------------------------------

define host {
  host_name             192.168.1.2
  use                   check_mk_host
  address               192.168.1.2
  _TAGS
  host_groups                 +check_mk
  alias                       192.168.1.2
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_C:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_E:/
```

```
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_F:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_I:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_J:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_K:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_L:/
  check_command               check_mk-df
}
```

```
define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         fs_N:/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG Application
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Application
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG HardwareEvents
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=HardwareEvents
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG Internet Explorer
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Internet%20Explor
er
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
```

```
  service_description         LOG Key Management Service
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Key%20Management%
20Service
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG Media Center
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Media%20Center
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG OAlerts
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=OAlerts
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG ODiag
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=ODiag
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG OSession
```

```
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=OSession
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG Security
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Security
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG System
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=System
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.2
  service_description         LOG Windows PowerShell
  notes_url
      /check/check_mk/logwatch.py?host=192.168.1.2&file=Windows%20PowerSh
ell
  check_command               check_mk-logwatch
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         Memory used
  check_command               check_mk-mem.used
```

```
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         System Time
  check_command               check_mk-systemtime
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         Uptime
  check_command               check_mk-uptime
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         CPU Usage
  check_command               check_mk-winperf.cpuusage
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.2
  service_description         Disk IO
  check_command               check_mk-winperf.diskstat
}


# Active checks

define service {
  use                   check_mk_active
  host_name             192.168.1.2
  service_description         Check_MK
}
```

```
# ----------------------------------------------------
# 192.168.1.8
# ----------------------------------------------------

define host {
  host_name             192.168.1.8
  use                   check_mk_host
  address               192.168.1.8
  _TAGS
  host_groups                 +check_mk
  alias                       192.168.1.8
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         CPU load
  check_command               check_mk-cpu.loads
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Number of threads
  check_command               check_mk-cpu.threads
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         CUPS Queue HP_LaserJet_Professional_P1102w
  check_command               check_mk-cups_queues
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
```

```
  service_description         CUPS Queue _192_168_1_3
  check_command               check_mk-cups_queues
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         fs_/
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         fs_/home
  check_command               check_mk-df
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Disk IO read
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Disk IO write
  check_command               check_mk-diskstat
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Kernel Context Switches
  check_command               check_mk-kernel
}
```

```
define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Kernel Major Page Faults
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Kernel Process Creations
  check_command               check_mk-kernel
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         CPU utilization
  check_command               check_mk-kernel.util
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Memory used
  check_command               check_mk-mem.used
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Vmalloc address space
  check_command               check_mk-mem.vmalloc
}

define service {
  use                   check_mk_passive
```

```
  host_name             192.168.1.8
  service_description       Mount options of /
  check_command             check_mk-mounts
}

define service {
  use                   check_mk_passive
  host_name             192.168.1.8
  service_description       Mount options of /home
  check_command             check_mk-mounts
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description       NIC br0 counters
  check_command             check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description       NIC eth0 counters
  check_command             check_mk-netctr.combined
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description       Postfix Queue
  check_command             check_mk-postfix_mailq
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description       TCP Connections
  check_command             check_mk-tcp_conn_stats
```

```
}

define service {
  use                   check_mk_passive_perf
  host_name             192.168.1.8
  service_description         Uptime
  check_command               check_mk-uptime
}


# Active checks

define service {
  use                   check_mk_active
  host_name             192.168.1.8
  service_description         Check_MK
}

# ------------------------------------------------------
# 192.168.1.5
# ------------------------------------------------------

define host {
  host_name             192.168.1.5
  use                   check_mk_host
  address               192.168.1.5
  _TAGS
  host_groups                 +check_mk
  alias                       192.168.1.5
}


define service {
  use                   check_mk_pingonly
  host_name             192.168.1.5
}
```

```
define hostgroup {
  hostgroup_name        check_mk
  alias                       Check_MK default hostgroup
}

# ------------------------------------------------------------
# Dummy check commands (controlled by generate_dummy_commands)
# ------------------------------------------------------------

define command {
  command_name               check_mk-kernel
  command_line               echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name               check_mk-uptime
  command_line               echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name               check_mk-kernel.util
  command_line               echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name               check_mk-df
  command_line               echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name               check_mk-postfix_mailq
  command_line               echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}
```

```
define command {
  command_name                  check_mk-cpu.loads
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                  check_mk-logwatch
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                  check_mk-cpu.threads
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                  check_mk-diskstat
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                  check_mk-winperf.cpuusage
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                  check_mk-mem.used
  command_line                  echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
```

```
  command_name                check_mk-mounts
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-systemtime
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-cups_queues
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-mem.vmalloc
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-winperf.diskstat
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-tcp_conn_stats
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}

define command {
  command_name                check_mk-netctr.combined
```

```
  command_line                echo "ERROR - you did an active check on
this service - please disable active checks" && exit 1
}
```

Multisite.mk – Configuração de acesso ao interface gráfico e conteúdo do menu lateral

```
# Confguration for Check_MK Multisite

# Users with unrestricted permissions
admin_users = [ "nagiosadmin" ]

# Users seeing all data but cannot do any action
# guest_users = [ "guest" ]

# A lists of all normal operational users allowed to use
# Multisite. If this variable is not set, then everybody with a correct
# HTTP login may use Multisite and gets the role "user"
# users       = [ "meier", "huber", "mueller" ]

# Users not explicitely being listed in admin_users or guest_users
# get the role "user" if they have a valid login. You can change this
# to "guest", "admin" or None by setting the following variable:
# default_user_role = "guest"

# Sites to connect to. If this variable is unset, a single
# connection to the local host is done.
#sites = {
```

```
#   # connect to local Nagios
#   "local" : {
#        "alias" : "Munich"
#   },
#
#   # connect to remote site (e.g. local OMD site 'paris')
#   "paris": {
#        "alias":          "Paris",
#        "socket":         "tcp:127.0.0.1:6557",
#        "url_prefix":     "/paris/",
#    },
#
#   # connect to remote site (site on remote host)
#   "rome": {
#        "alias":          "Rome",
#        "socket":         "tcp:10.0.0.2:6557",
#        "url_prefix":     "http://10.0.0.2/rome/",
#    },
#}

#
# NagVis
#
# The NagVis-Snapin needs to know the URL to nagvis.
# This is not always /nagvis/ - especially not for OMD
nagvis_base_url = '/nagvis'


# Restrict number of datasets queries via Livestatus.
# This prevents you from consuming too much ressources
# in case of insensible queries.
# soft_query_limit = 1000
# hard_query_limit = 5000

# Views allow to play alarm sounds according to the
# "worst" state of the show items. Configure here
# which sounds to play. Possible events: critical,
# warning, unknown, ok, up, down, unreachable,
```

```
# pending. Sounds are expected in the sounds subdirectory
# of htdocs (Default is /usr/share/check_mk/web/htdocs/sounds)
# sounds = [
#  ( "down", "down.wav" ),
#  ( "critical", "critical.wav" ),
#  ( "unknown", "unknown.wav" ),
#  ( "warning", "warning.wav" ),
#  ( None,      "ok", ),
# ]
# Note: this example has not sound for unreachable hosts.
# set sound_url to another url, if you place your sound
# files elsewhere:
# sound_url = "http://otherhost/path/to/sound/"
# or
# sound_url = "/nagios/alarms/"

# Tabs for choosing number of columns refresh
# view_option_refreshes = [ 30, 60, 90, 0 ]
# view_option_columns   = [ 1, 2, 3, 4, 5, 6, 8 ]

# Custom links for "Custom Links" Snapin. Feel free to add your
# own links here. The boolean values True and False determine
# wether the sections are open or closed by default.

# Links for everyone
custom_links['guest'] = [
  ( "Classical Nagios GUI", "../nagios/", "link_home.gif" ),
  ( "Addons", True, [
        ( "PNP4Nagios", "../../pnp4nagios/",       "link_reporting.gif"
),
        ( "NagVis", False, [
            ( "Automap",    "../nagvis/index.php?map=__automap",
"link_map.gif"),
            ( "Demo map",   "../nagvis/index.php?map=demo-map",
"link_map.gif"),
            ( "Demo Map 2", "../nagvis/index.php?map=demo2",
"link_map.gif"),
        ]),
```

```
  ]),
]

# The members of the role 'user' get the same links as the guests
# but some in addition
custom_links['user'] = custom_links['guest'] + [
  ( "Open Source Components", False, [
        ( "Multisite",     "http://mathias-
kettner.de/checkmk_multisite.html" ),
        ( "MK Livestatus", "http://mathias-
kettner.de/checkmk_livestatus.html" ),
        ( "Check_MK",      "http://mathias-kettner.de/check_mk.html" ),
        ( "Nagios",        "http://www.nagios.org/" ),
        ( "PNP4Nagios",    "http://pnp4nagios.org/" ),
        ( "NagVis",        "http://nagvis.org/" ),
        ( "RRDTool",       "http://oss.oetiker.ch/rrdtool/" ),
   ])
]

# The admins yet get further links
custom_links['admin'] = custom_links['user'] + [
  ( "Support", False, [
      ( "Mathias Kettner",              "http://mathias-kettner.de/" ),
      ( "Check_MK Mailinglists",        "http://mathias-
kettner.de/check_mk_lists.html" ),
      ( "Check_MK Portal (inofficial)", "http://check-mk-portal.org/"),
      ( "Nagios Portal (German)",       "http://nagios-portal.org"),
  ])
]

# Show error messages from unreachable sites in views. Set this
# to False in order to hide those messages.
show_livestatus_errors = True

# Hide certain views from the sidebar
# hidden_views = [ "hosttiles", "allhosts_mini" ]
# Vice versa: hide all views except these (be carefull, this
```

```
# will also exclude custom views)
# visible_views = [ "allhosts", "searchsvc" ]

# Load custom style sheet which can override styles defined in
check_mk.css
# Put your style sheet into web/htdocs/
# custom_style_sheet = "my_styles.css"

# URL to show as welcome page (in the 'main' frame).
# You can use relative URL or absolute URLs like 'http://server/url'
# Default is 'main.py'
# start_url = 'view.py?view_name=hostgroups'

# Quicksearch: Limit the number of hits to shop in dropdown.
# Default is to show at most 80 items.
# quicksearch_dropdown_limit = 80

#   __        ___  _____ ___
#   \ \      / / \|_   _/ _ \
#    \ \ /\ / / _ \ | || | | |
#     \ V  V / ___ \| || |_| |
#      \_/\_/_/   \_\_| \___/
#
# Check_MK's Web Administration Tool

# Declare files in conf.d/ to be editable with WATO. Please make
# sure, that those files exist and are writable by Apache, e.g.:
# touch /etc/check_mk/conf.d/network.mk
# chgrp www /etc/check_mk/conf.d/network.mk
# chmod 664 /etc/check_mk/conf.d/network.mk
#
# config_files = [
#   ("network.mk",    "Network, Infrastructure", [ "admin", "user" ] ),
#   ("datacenter.mk", "Servers in Datacenter",   [ "admin" ]),
# ]

# Host tags to be used in WATO
# host_tags = [
```

```
#  ( "Operating System", [
#       ( "lnx", "Linux", [ 'tcp' ]),
#       ( "win", "Windows", [ 'tcp', 'snmp' ]),
#       ( "net", "Network device", [ 'snmp' ]),
#       ( "ping", "Other PING-only device", ),
#    ]),
#  ( "Productivity", [
#       ( "prod", "Production System" ),
#       ( "test", "Test System" ),
#    ]),
#  ( "Bulkwalk (SNMP v2c)", [
#       ( None,   "simple walk (SNMP v1)"),
#       ( "bulk", "Bulkwalk (SNMP v2c)" ),
#    ]),
#
# ]
```

# Ninja – Capturas de Ecrã

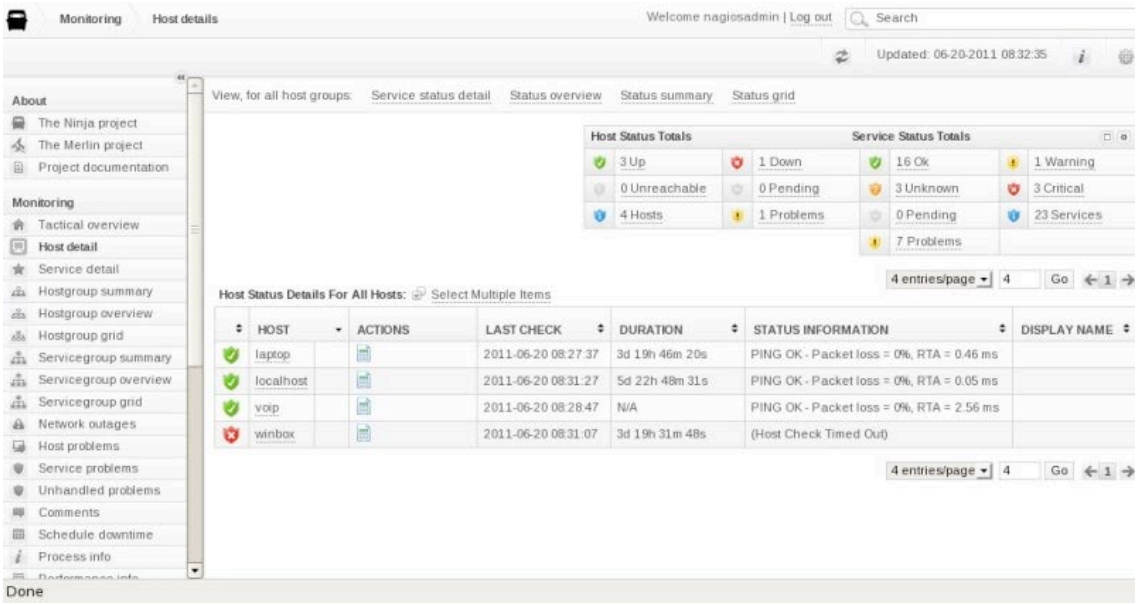



Fig. A30 – Visão geral dos equipamentos monitorizados

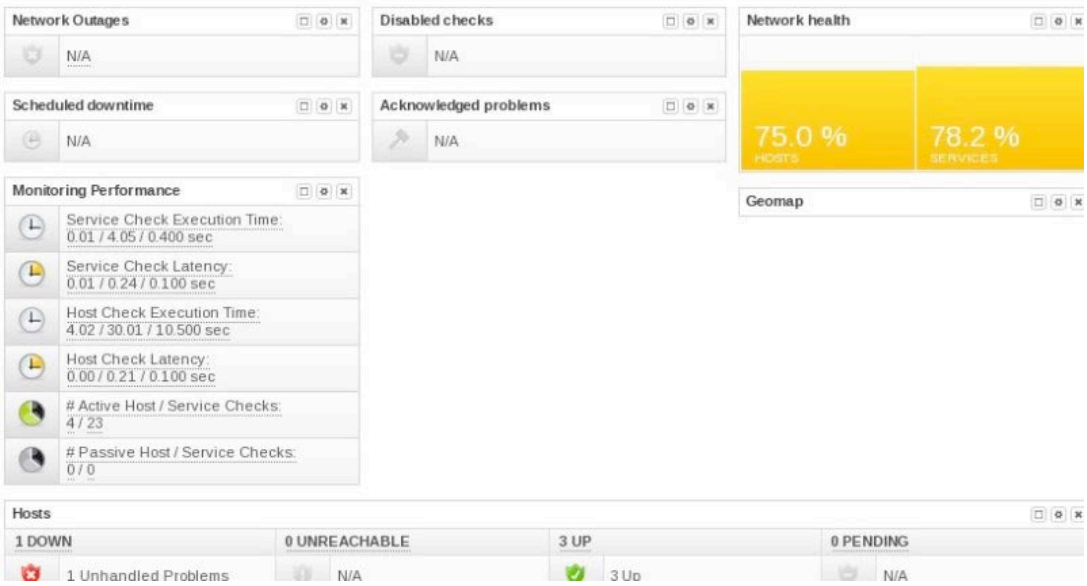
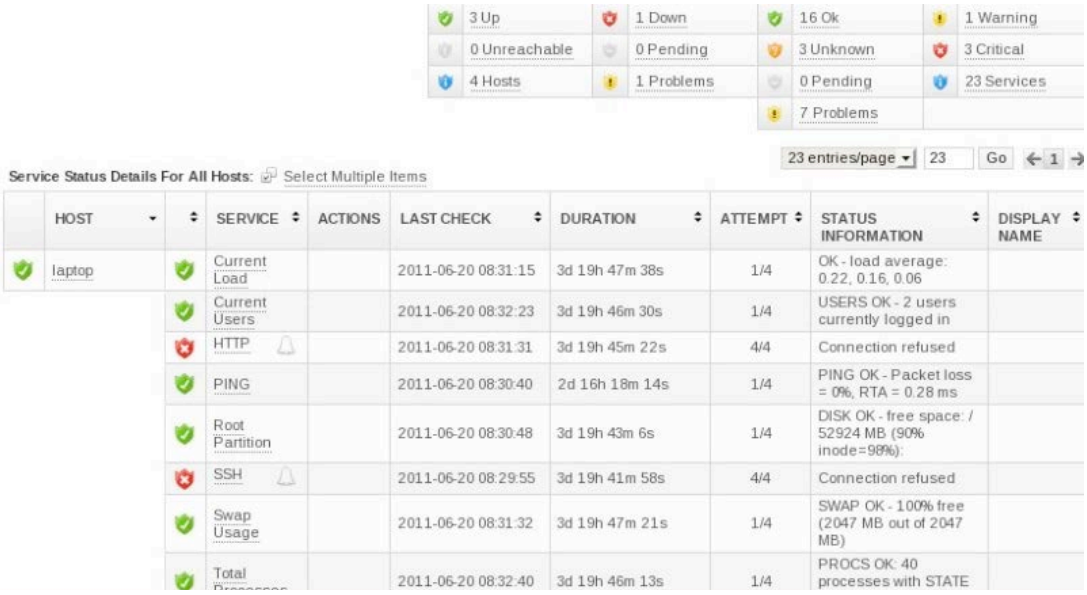



Fig. A31 – Tempo de disponibilidade de equipamentos e serviços



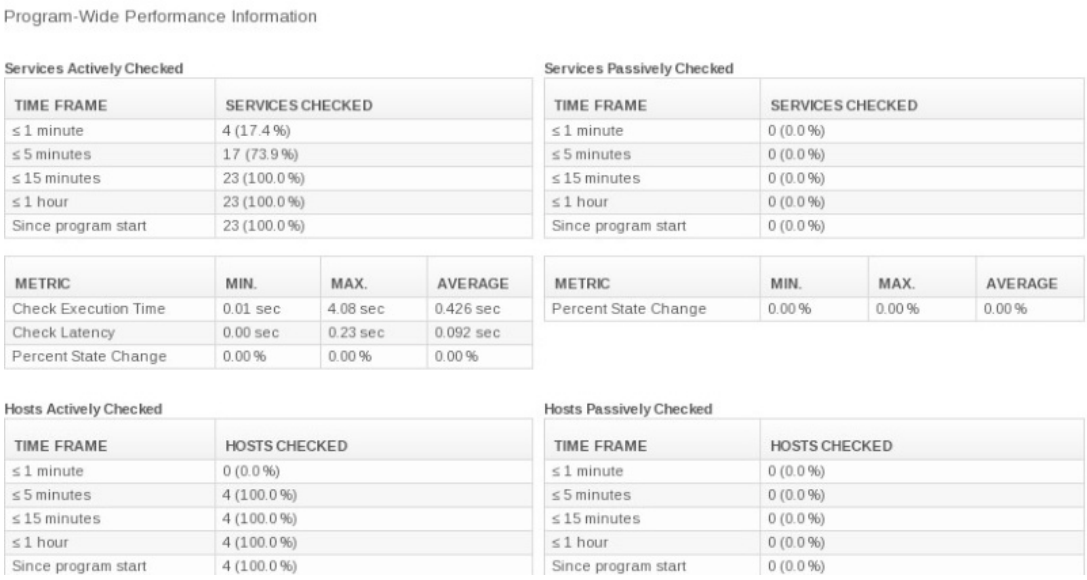
| | | | |
|---|---|---|---|
| 3 Up | 1 Down | 16 Ok | 1 Warning |
| 0 Unreachable | 0 Pending | 3 Unknown | 3 Critical |
| 4 Hosts | 1 Problems | 0 Pending | 23 Services |
| | | 7 Problems | |

23 entries/page | 23 | Go | 1

Service Status Details For All Hosts: Select Multiple Items

| HOST | SERVICE | ACTIONS | LAST CHECK | DURATION | ATTEMPT | STATUS INFORMATION | DISPLAY NAME |
|---|---|---|---|---|---|---|---|
| laptop | Current Load | | 2011-06-20 08:31:15 | 3d 19h 47m 38s | 1/4 | OK - load average: 0.22, 0.16, 0.06 | |
| | Current Users | | 2011-06-20 08:32:23 | 3d 19h 46m 30s | 1/4 | USERS OK - 2 users currently logged in | |
| | HTTP | | 2011-06-20 08:31:31 | 3d 19h 45m 22s | 4/4 | Connection refused | |
| | PING | | 2011-06-20 08:30:40 | 2d 16h 18m 14s | 1/4 | PING OK - Packet loss = 0%, RTA = 0.28 ms | |
| | Root Partition | | 2011-06-20 08:30:48 | 3d 19h 43m 6s | 1/4 | DISK OK - free space: / 52924 MB (90% inode=98%): | |
| | SSH | | 2011-06-20 08:29:55 | 3d 19h 41m 58s | 4/4 | Connection refused | |
| | Swap Usage | | 2011-06-20 08:31:32 | 3d 19h 47m 21s | 1/4 | SWAP OK - 100% free (2047 MB out of 2047 MB) | |
| | Total Processes | | 2011-06-20 08:32:40 | 3d 19h 46m 13s | 1/4 | PROCS OK: 40 processes with STATE | |

Fig. A32 – Vista de serviços por equipamento



Program-Wide Performance Information

Services Actively Checked

| TIME FRAME | SERVICES CHECKED |
|---|---|
| ≤ 1 minute | 4 (17.4 %) |
| ≤ 5 minutes | 17 (73.9 %) |
| ≤ 15 minutes | 23 (100.0 %) |
| ≤ 1 hour | 23 (100.0 %) |
| Since program start | 23 (100.0 %) |

Services Passively Checked

| TIME FRAME | SERVICES CHECKED |
|---|---|
| ≤ 1 minute | 0 (0.0 %) |
| ≤ 5 minutes | 0 (0.0 %) |
| ≤ 15 minutes | 0 (0.0 %) |
| ≤ 1 hour | 0 (0.0 %) |
| Since program start | 0 (0.0 %) |

| METRIC | MIN. | MAX. | AVERAGE |
|---|---|---|---|
| Check Execution Time | 0.01 sec | 4.08 sec | 0.426 sec |
| Check Latency | 0.00 sec | 0.23 sec | 0.092 sec |
| Percent State Change | 0.00 % | 0.00 % | 0.00 % |

| METRIC | MIN. | MAX. | AVERAGE |
|---|---|---|---|
| Percent State Change | 0.00 % | 0.00 % | 0.00 % |

Hosts Actively Checked

| TIME FRAME | HOSTS CHECKED |
|---|---|
| ≤ 1 minute | 0 (0.0 %) |
| ≤ 5 minutes | 4 (100.0 %) |
| ≤ 15 minutes | 4 (100.0 %) |
| ≤ 1 hour | 4 (100.0 %) |
| Since program start | 4 (100.0 %) |

Hosts Passively Checked

| TIME FRAME | HOSTS CHECKED |
|---|---|
| ≤ 1 minute | 0 (0.0 %) |
| ≤ 5 minutes | 0 (0.0 %) |
| ≤ 15 minutes | 0 (0.0 %) |
| ≤ 1 hour | 0 (0.0 %) |
| Since program start | 0 (0.0 %) |

Fig. A33 – Estatísticas de desempenho

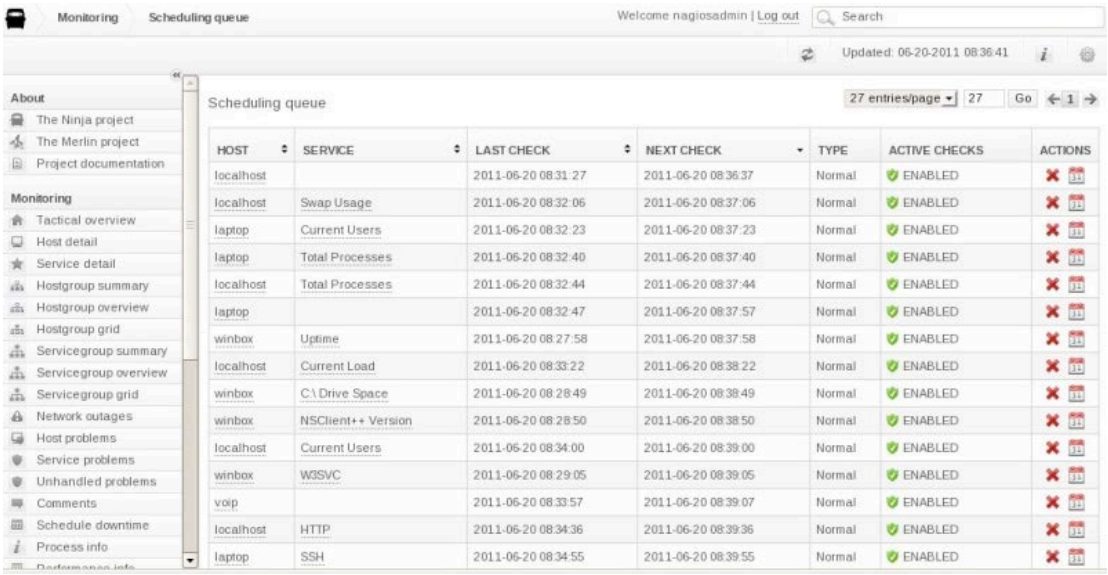



Fig. A34 – Calendarização de monitorizações

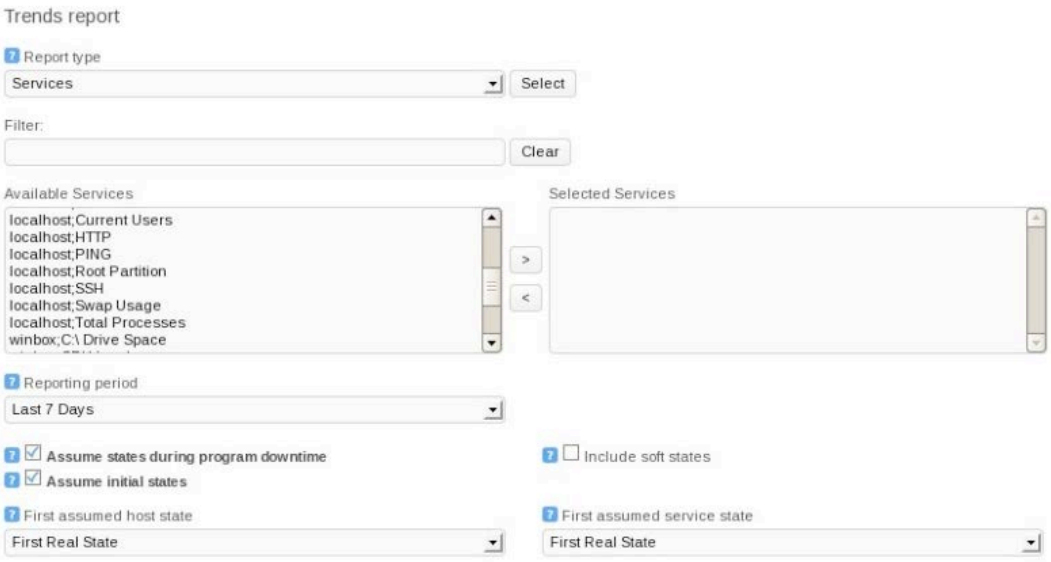



Fig. A35 – Relatórios de tendências

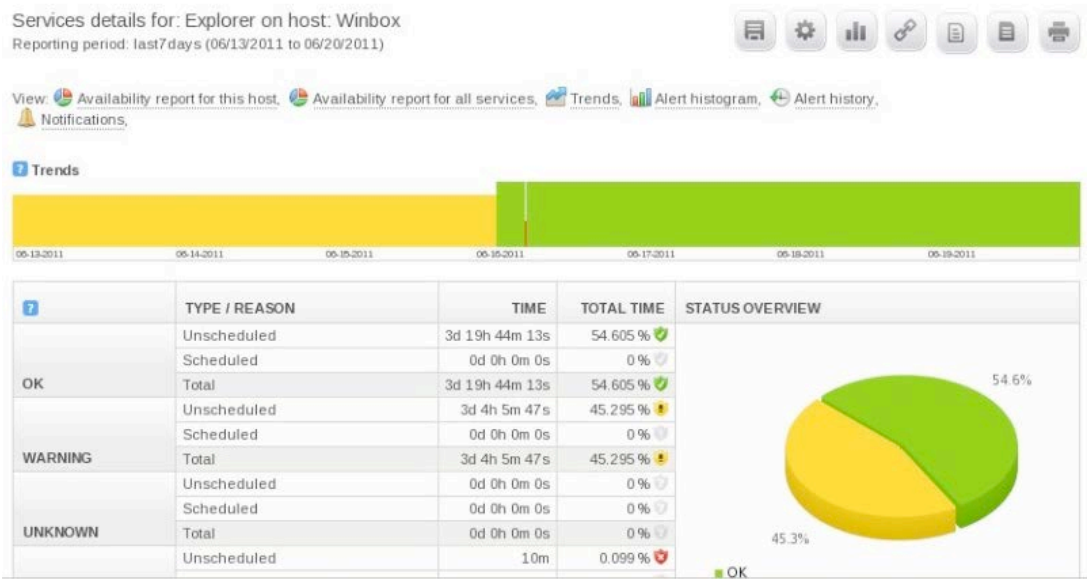



Fig. A36 – Histórico de um serviço

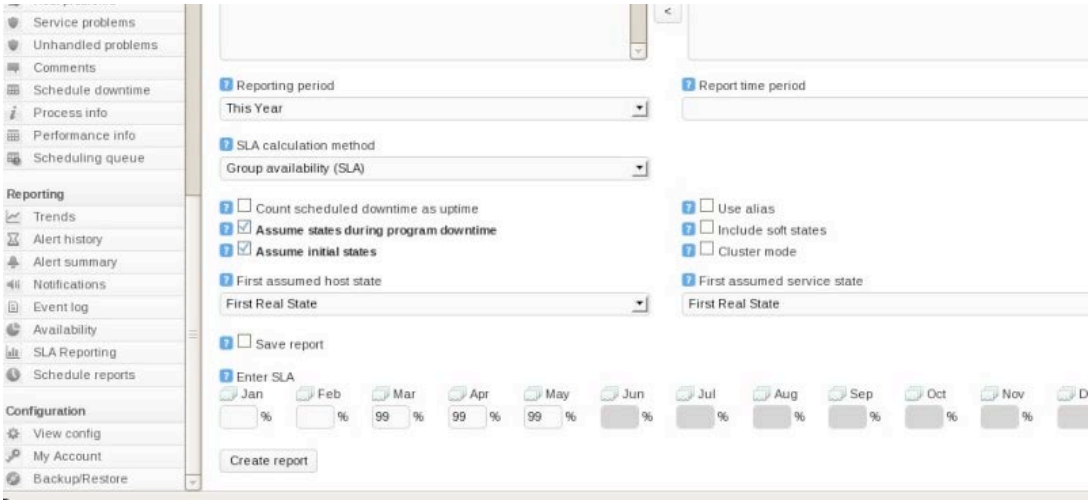
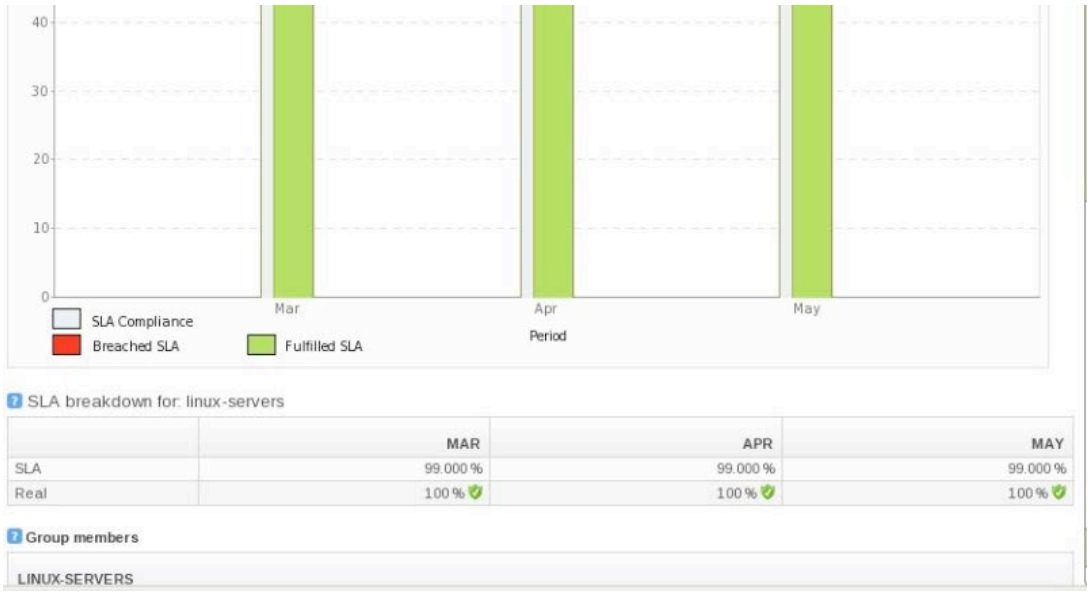



Fig. A37 – Pedido de verificação de SLA




SLA breakdown for: linux-servers

| | MAR | APR | MAY |
|---|---|---|---|
| SLA | 99.000 % | 99.000 % | 99.000 % |
| Real | 100 % | 100 % | 100 % |

Group members

LINUX-SERVERS

Fig. A38 – Gráfico de verificação de SLA

# Merlin+Ninja – Ficheiros de Configuração

merlin.conf

```
#
# Sample configuration file for merlin
#
# Default options have been commented out
#
ipc_socket = /usr/local/nagios/addons/merlin/ipc.sock;

# module-specific configuration options.
module {
      # textual log of normal hum-drum events
      log_file = /usr/local/nagios/addons/merlin/logs/neb.log;

      # determine whether we should call pthread_cancel() for the
      # reaper thread or not. Set this to "no" if you're having
      # problems with Monitor/Nagios crashing on soft reloads
      #cancel_threads = yes
}

# daemon-specific config options
daemon {
      pidfile = /var/run/merlin.pid;

      # same as the "module" section above
      log_file = /usr/local/nagios/addons/merlin/logs/daemon.log;

      # The import_program is responsible for priming the merlin database
      # with configuration information and an initial import of status
data.
      # It's invoked with the following arguments:
      # --cache=/path/to/objects.cache
      # --status-log=/path/to/status.log
      # --db-name=database_name
      # --db-user=database_user_name
```

```
# --db-pass=database_password
# --db-host=database_host
# The database parameters are taken from "database" section if such
# a section exists.
import_program = php /usr/local/nagios/addons/merlin/import.php;

# address to listen to. 0.0.0.0 is default
#address = 0.0.0.0;

# port to listen to. 15551 is default. This is a daemon
# specific config setting, as the module never listens to
# the network
port = 15551;
database {
      name = merlin;
      user = merlin;
      pass = merlin;
      host = localhost;
      type = mysql;
}

# this section describes how we handle config synchronization
# It can also be configured on a per-node basis to override the
# globals set here.
# "push" and "fetch" point to a command supposed to be run when we
# want to push configuration to any or all nodes in the network.
# They're invoked with the following arguments:
# --merlin-cfg=/path/to/merlin.conf
object_config {
      # a "push" happens for peers and pollers that we know have an
      # older configuration than we do. This is what one would
      # normally want to happen, since it originates from the node
      # that already has all the information
      push = mon oconf push

      # a "fetch" happens for peers and masters that we know
      # have a newer configuration than we do. One won't normally
      # configure this (and it currently lacks scripting support),
```

```
            # but it's available for advanced users who know what they're
            # doing
            #fetch = mon oconf fetch
      }
}
```

ninja config.php – Configuração de todos os parâmetros do interface ninja

```
<?php defined('SYSPATH') OR die('No direct access allowed.');
/**
 * Base path of the web site. If this includes a domain, eg:
localhost/kohana/
 * then a full URL will be used, eg: http://localhost/kohana/. If it only
includes
 * the path, and a site_protocol is specified, the domain will be auto-
detected.
 */
$config['site_domain'] = '/ninja/';

/**
 * Force a default protocol to be used by the site. If no site_protocol
is
 * specified, then the current protocol is used, or when possible, only
an
 * absolute path (with no protocol/domain) is used.
 */
$config['site_protocol'] = '';
```

```
/**
 * Name of the front controller for this application. Default: index.php
 *
 * This can be removed by using URL rewriting.
 */
$config['index_page'] = 'index.php';

/**
* In case anyone would like to brand their installation
* This string is shown throughout the GUI in various places
* and this is the only place you will have to change it.
*/
$config['product_name'] = 'Nagios';

/**
 * Custom version info file. Format:
 * VERSION=x.y.z
 * This info will be visible in the 'product info' link
 */
$config['version_info'] = '/etc/ninja-release';

/**
 * Fake file extension that will be added to all generated URLs. Example:
.html
 */
$config['url_suffix'] = '';

/**
 * Length of time of the internal cache in seconds. 0 or FALSE means no
caching.
 * The internal cache stores file paths and config entries across
requests and
 * can give significant speed improvements at the expense of delayed
updating.
 */
$config['internal_cache'] = FALSE;

/**
```

```
 * Enable or disable gzip output compression. This can dramatically
decrease
 * server bandwidth usage, at the cost of slightly higher CPU usage. Set
to
 * the compression level (1-9) that you want to use, or FALSE to disable.
 *
 * Do not enable this option if you are using output compression in
php.ini!
 */
$config['output_compression'] = FALSE;

/**
 * Enable or disable global XSS filtering of GET, POST, and SERVER data.
This
 * option also accepts a string to specify a specific XSS filtering tool.
 */
$config['global_xss_filtering'] = TRUE;

/**
 * Enable or disable hooks.
 */
$config['enable_hooks'] = true;

/**
 * Log thresholds:
 *  0 - Disable logging
 *  1 - Errors and exceptions
 *  2 - Warnings
 *  3 - Notices
 *  4 - Debugging
 */
$config['log_threshold'] = 0;

/**
 * Message logging directory.
 */
$config['log_directory'] = APPPATH.'logs';
```

```
/**
 * Enable or disable displaying of Kohana error pages. This will not
affect
 * logging. Turning this off will disable ALL error pages.
 */
$config['display_errors'] = TRUE;

/**
 * Enable or disable statistics in the final output. Stats are replaced
via
 * specific strings, such as {execution_time}.
 *
 * @see http://docs.kohanaphp.com/general/configuration
 */
$config['render_stats'] = TRUE;

/**
 * Filename prefixed used to determine extensions. For example, an
 * extension to the Controller class would be named MY_Controller.php.
 */
$config['extension_prefix'] = 'MY_';

$config['autoload'] = array
(
            'libraries' => 'session, database'
);

/**
 * Additional resource paths, or "modules". Each path can either be
absolute
 * or relative to the docroot. Modules can include any resource that can
exist
 * in your application directory, configuration files, controllers,
views, etc.
 */
$config['modules'] = array
(
      MODPATH.'auth',      // Authentication
```

```
        // MODPATH.'forge',     // Form generation
        // MODPATH.'kodoc',     // Self-generating documentation
        // MODPATH.'media',     // Media caching and compression
        // MODPATH.'gmaps',     // Google Maps integration
        // MODPATH.'archive',   // Archive utility
        // MODPATH.'payment',   // Online payments
        // MODPATH.'unit_test', // Unit testing
        // MODPATH.'object_db', // New OOP Database library (testing only!)
);

/**
 *    Base path to the location of Nagios.
 *    This is used if we need to read some
 *    configuration from the config files.
 *    This path sare assumed to contain the
 *    following subdirectories (unless specified below):
 *          /bin
 *          /etc
 *          /var
 *
 *    No trailing slash.
 */
$config['nagios_base_path'] = '/usr/local/nagios';

/**
 *    If the nagios etc directory is to be found outside
 *    the nagios base path, please specify here.
 *
 *    No trailing slash.
 */
$config['nagios_etc_path'] = false;

/**
 *    Path to where host logos as stored.
 *    Should be relative to webroot
 */
$config['logos_path'] = '/monitor/images/logos/';
```

```
/**
 * Theme config
 *
 * theme_path points to the views subdirectory where ALL
 * available themes are stored
 */
$config['theme_path'] = 'themes/';

/**
 * current_theme is the subdirectory to 'theme_path' above
 * that holds the currently active theme.
 */
$config['current_theme'] = 'default/';

/**
 * current_skin is the subdirectory to 'css' within the
 * theme. a skin a simple way of altering colours etc
 * in the gui.
 */
$config['current_skin'] = 'default/';

/**
 * Do we use NACOMA (Nagios Configuration Manager)?
 * If path differs from the one below but still installed
 * you could simply change it.
 */
$nacoma_real_path = '/opt/monitor/op5/nacoma/';
if (is_dir($nacoma_real_path)) {
      $config['nacoma_path'] = '/monitor/op5/nacoma/';
} else {
      $config['nacoma_path'] = false;
}

$hypermap_real_path = '/opt/monitor/share/cgi-bin/hypergraph.cgi';
if (is_file($hypermap_real_path)) {
      $config['hypermap_path'] = '/monitor.old/cgi-bin/hypergraph.cgi';
} else {
      $config['hypermap_path'] = false;
```

```
}

/**
 * Web path to Pnp4nagios
 * If installed, change path below or set to false if not
 */
$config['pnp4nagios_path'] = '/monitor/op5/pnp/';

/**
*     Path to the pnp config file 'config.php'
*     Only used if 'pnp4nagios_path' !== false
*/
$config['pnp4nagios_config_path'] = '/opt/monitor/etc/pnp/config.php';

/**
 * Do we use NagVis?
 * If path differs from the one below but still installed
 * you could simply change it.
 */
$config['nagvis_real_path'] = '/opt/monitor/op5/nagvis/';
if (is_dir($config['nagvis_real_path'])) {
      $config['nagvis_path'] = '/monitor/op5/nagvis/';
} else {
      $config['nagvis_path'] = false;
}

/**
* Add some suport for cacti/statistics
*/
$condition['cacti_real_path'] = '/opt/statistics';
if (is_dir($condition['cacti_real_path'])) {
      $config['cacti_path'] = true;
} else {
      $config['cacti_path'] = false;
}

/**
 * Default refresh rate for all pages
```

```
 */
$config['page_refresh_rate'] = 90;

/**
 * Control command line access to Ninja
 * Possible values:
 *    false             :     No commandline access
 *    true        :     Second command line argument (i.e after path)
 *                            will be used as username (default)
 *    'username'  :     The entered username will be used for
authentication
 */
$config['cli_access'] = true;

/**
* Nr of items returned for searches
*/
$config['search_limit'] = 10;

/**
* Nr of items returned for autocomplete search
*/
$config['autocomplete_limit'] = 10;

/**
*     Nr of seconds while we still are considering
*     merlin to be alive.
*/
$config['stale_data_limit'] = 60;

/**
* Control the use oof pop-ups for PNP graphs and comments
*/
$config['use_popups'] = 0;

/**
* Pop-up delay
* Milliseconds before the pop-up is shown
```

```
*/
$config['popup_delay'] = 1500;

/**
* Control whether to show display_name or not
*/
$config['show_display_name'] = 1;

/**
* Control whether to show {host,service} notes or not
* Default: 0
*/
$config['show_notes'] = 0;

/**
*     Control how many characters of the note to be displayed
*     in the GUI. The entire note will be displayed on mouseover or
click.
*     Use 0 to display everything.
*     Default: 80
*/
$config['show_notes_chars'] = 80;

# check for custom config files that
# won't be overwritten on upgrade
if
(file_exists(realpath(dirname(__FILE__)).'/custom/'.basename(__FILE__)))
{
      include(realpath(dirname(__FILE__)).'/custom/'.basename(__FILE__));
}
```

contacts.cfg – Contactos dos vários grupos de utilizadores

```
##############################################################################
######
# CONTACTS.CFG - SAMPLE CONTACT/CONTACTGROUP DEFINITIONS
#
# Last Modified: 05-31-2007
#
# NOTES: This config file provides you with some example contact and
contact
#        group definitions that you can reference in host and service
#        definitions.
#
#        You don't need to keep these definitions in a separate file from
your
#        other object definitions.  This has been done just to make
things
#        easier to understand.
#
##############################################################################
######



##############################################################################
######
##############################################################################
######
#
# CONTACTS
#
##############################################################################
######
##############################################################################
######

# Just one contact defined by default - the Nagios admin (that's you)
```

```
# This contact definition inherits a lot of default values from the
'generic-contact'
# template which is defined elsewhere.

define contact{
        contact_name                    nagiosadmin        ; Short name
of user
      use                     generic-contact         ; Inherit default
values from generic-contact template (defined above)
        alias                           Nagios Admin       ; Full name of
user

        email                           vladimiro@fe.up.pt ;
        }



################################################################################
######
################################################################################
######
#
# CONTACT GROUPS
#
################################################################################
######
################################################################################
######

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup{
        contactgroup_name       admins
        alias                   Nagios Administrators
        members                 nagiosadmin
        }
```

ninja index.php – Configuração da framework Kohana, localização de ficheiros

```
<?php
/**
 * This file acts as the "front controller" to your application. You can
 * configure your application, modules, and system directories here.
 * PHP error_reporting level may also be changed.
 *
 * @see http://kohanaphp.com
 */

$ninja_base = '/usr/local/nagios/addons/ninja';

/**
 * Define the website environment status. When this flag is set to TRUE,
some
 * module demonstration controllers will result in 404 errors. For more
information
 * about this option, read the documentation about deploying Kohana.
 *
 * @see http://docs.kohanaphp.com/installation/deployment
 */
define('IN_PRODUCTION', FALSE);

/**
 * Website application directory. This directory should contain your
application
 * configuration, controllers, models, views, and other resources.
 *
 * This path can be absolute or relative to this file.
 */
$kohana_application = "$ninja_base/application";

/**
 * Kohana modules directory. This directory should contain all the
modules used
 * by your application. Modules are enabled and disabled by the
application
```

```
 * configuration file.
 *
 * This path can be absolute or relative to this file.
 */
$kohana_modules = "$ninja_base/modules";

/**
 * Kohana system directory. This directory should contain the core/
directory,
 * and the resources you included in your download of Kohana.
 *
 * This path can be absolute or relative to this file.
 */
$kohana_system = "$ninja_base/system";

/**
 * Test to make sure that Kohana is running on PHP 5.2 or newer. Once you
are
 * sure that your environment is compatible with Kohana, you can comment
this
 * line out. When running an application on a new server, uncomment this
line
 * to check the PHP version quickly.
 */
#version_compare(PHP_VERSION, '5.2', '<') and exit('Kohana requires PHP
5.2 or newer.');

/**
 * Set the error reporting level. Unless you have a special need, E_ALL
is a
 * good level for error reporting.
 */
error_reporting(E_ALL & ~E_STRICT);

/**
 * Turning off display_errors will effectively disable Kohana error
display
```

```
 * and logging. You can turn off Kohana errors in
application/config/config.php
 */
ini_set('display_errors', TRUE);

/**
 * If you rename all of your .php files to a different extension, set the
new
 * extension here. This option can left to .php, even if this file has a
 * different extension.
 */
define('EXT', '.php');

define('MEDIAPATH', "$ninja_base/application/media");

#unset($ninja_base);
//
// DO NOT EDIT BELOW THIS LINE, UNLESS YOU FULLY UNDERSTAND THE
IMPLICATIONS.
// ----------------------------------------------------------------------
------
// $Id: index.php 3917 2009-01-21 03:06:22Z zombor $
//

$kohana_pathinfo = pathinfo(__FILE__);
// Define the front controller name and docroot
define('DOCROOT', $kohana_pathinfo['dirname'].DIRECTORY_SEPARATOR);
define('KOHANA',  $kohana_pathinfo['basename']);

// If the front controller is a symlink, change to the real docroot
is_link(KOHANA) and chdir(dirname(realpath(__FILE__)));

// If kohana folders are relative paths, make them absolute.
$kohana_application = file_exists($kohana_application) ?
$kohana_application : DOCROOT.$kohana_application;
$kohana_modules = file_exists($kohana_modules) ? $kohana_modules :
DOCROOT.$kohana_modules;
```

```
$kohana_system = file_exists($kohana_system) ? $kohana_system :
DOCROOT.$kohana_system;

// Define application and system paths
define('APPPATH', str_replace('\\', '/',
realpath($kohana_application)).'/');
define('MODPATH', str_replace('\\', '/', realpath($kohana_modules)).'/');
define('SYSPATH', str_replace('\\', '/', realpath($kohana_system)).'/');

// Clean up
unset($kohana_application, $kohana_modules, $kohana_system);

if (file_exists(DOCROOT.'install'.EXT))
{
      // Load the installation tests
      include DOCROOT.'install'.EXT;
}
else
{
      // Initialize Kohana
      require SYSPATH.'core/Bootstrap'.EXT;
}
```

localhost.cfg – exemplo de ficheiro de configuração para um equipamento a monitorizar, com a lista de todos os serviços a serem monitorizados. Existe um destes ficheiros para cada equipamento nas soluções baseadas em Nagios como Check_MK ou Icinga

```
##############################################################################
######
# LOCALHOST.CFG - SAMPLE OBJECT CONFIG FILE FOR MONITORING THIS MACHINE
#
# Last Modified: 05-31-2007
#
# NOTE: This config file is intended to serve as an *extremely* simple
#       example of how you can create configuration entries to monitor
#       the local (Linux) machine.
#
##############################################################################
######




##############################################################################
######
##############################################################################
######
#
# HOST DEFINITION
#
##############################################################################
######
##############################################################################
######

# Define a host for the local machine

define host{
```

```
        use                     linux-server            ; Name of host
template to use
                                        ; This host definition will
inherit all variables that are defined
                                        ; in (or inherited by) the
linux-server host template definition.
        host_name               localhost
        alias                   localhost
        address                 127.0.0.1
        }



################################################################################
######
################################################################################
######
#
# HOST GROUP DEFINITION
#
################################################################################
######
################################################################################
######

# Define an optional hostgroup for Linux machines

define hostgroup{
        hostgroup_name  linux-servers ; The name of the hostgroup
        alias           Linux Servers ; Long name of the group
        members         localhost     ; Comma separated list of hosts
that belong to this group
        }



################################################################################
######
```

```
##############################################################################
######
#
# SERVICE DEFINITIONS
#
##############################################################################
######
##############################################################################
######


# Define a service to "ping" the local machine

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             PING
      check_command                 check_ping!100.0,20%!500.0,60%
        }


# Define a service to check the disk space of the root partition
# on the local machine.  Warning if < 20% free, critical if
# < 10% free space on partition.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             Root Partition
      check_command                 check_local_disk!20%!10%!/
        }



# Define a service to check the number of currently logged in
# users on the local machine.  Warning if > 20 users, critical
```

```
# if > 50 users.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             Current Users
      check_command                 check_local_users!20!50
        }


# Define a service to check the number of currently running procs
# on the local machine.  Warning if > 250 processes, critical if
# > 400 users.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             Total Processes
      check_command                 check_local_procs!250!400!RSZDT
        }



# Define a service to check the load on the local machine.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             Current Load
      check_command
      check_local_load!5.0,4.0,3.0!10.0,6.0,4.0
        }
```

```
# Define a service to check the swap usage the local machine.
# Critical if less than 10% of swap is free, warning if less than 20% is
free

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             Swap Usage
      check_command                 check_local_swap!20!10
        }



# Define a service to check SSH on the local machine.
# Disable notifications for this service by default, as not all users may
have SSH enabled.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             SSH
      check_command                 check_ssh
      notifications_enabled         0
        }



# Define a service to check HTTP on the local machine.
# Disable notifications for this service by default, as not all users may
have HTTP enabled.

define service{
        use                             local-service         ; Name of
service template to use
        host_name                       localhost
        service_description             HTTP
```

```
      check_command                  check_http
      notifications_enabled          0
        }
```

windows.cfg – Semelhante ao exemplo anterior, ilustrando desta vez a utilização de um agente remoto, no caso o NSClient++

```
##############################################################################
######
# WINDOWS.CFG - SAMPLE CONFIG FILE FOR MONITORING A WINDOWS MACHINE
#
# NOTES: This config file assumes that you are using the sample
configuration
#      files that get installed with the Icinga quickstart guide.
#
##############################################################################
######
```

```
##############################################################################
######
##############################################################################
######
#
# HOST DEFINITIONS
#
```

```
################################################################################
######
################################################################################
######

# Define a host for the Windows machine we'll be monitoring
# Change the host_name, alias, and address to fit your situation

define host{
      use         windows-server    ; Inherit default values from a
template
      host_name   winbox      ; The name we're giving to this host
      alias       My Windows Server ; A longer name associated with the
host
      address           192.168.1.2 ; IP address of the host
      }




################################################################################
######
################################################################################
######
#
# HOST GROUP DEFINITIONS
#
################################################################################
######
################################################################################
######


# Define a hostgroup for Windows machines
# All hosts that use the windows-server template will automatically be a
member of this group

define hostgroup{
```

```
      hostgroup_name    windows-servers   ; The name of the hostgroup
      alias       Windows Servers   ; Long name of the group
      }



################################################################################
######
################################################################################
######
#
# SERVICE DEFINITIONS
#
################################################################################
######
################################################################################
######


# Create a service for monitoring the version of NSCLient++ that is
installed
# Change the host_name to match the name of the host you defined above

define service{
      use               generic-service
      host_name         winbox
      service_description     NSClient++ Version
      check_command           check_nt!CLIENTVERSION
      }



# Create a service for monitoring the uptime of the server
# Change the host_name to match the name of the host you defined above

define service{
      use               generic-service
```

```
	host_name		winbox
	service_description	Uptime
	check_command		check_nt!UPTIME
	}



# Create a service for monitoring CPU load
# Change the host_name to match the name of the host you defined above

define service{
	use			generic-service
	host_name		winbox
	service_description	CPU Load
	check_command		check_nt!CPULOAD!-l 5,80,90
	}



# Create a service for monitoring memory usage
# Change the host_name to match the name of the host you defined above

define service{
	use			generic-service
	host_name		winbox
	service_description	Memory Usage
	check_command		check_nt!MEMUSE!-w 80 -c 90
	}



# Create a service for monitoring C:\ disk usage
# Change the host_name to match the name of the host you defined above

define service{
	use			generic-service
	host_name		winbox
	service_description	C:\ Drive Space
```

```
      check_command           check_nt!USEDDISKSPACE!-l c -w 80 -c 90
      }



# Create a service for monitoring the W3SVC service
# Change the host_name to match the name of the host you defined above

define service{
      use               generic-service
      host_name         winbox
      service_description     W3SVC
      check_command           check_nt!SERVICESTATE!-d SHOWALL -l W3SVC
      }



# Create a service for monitoring the Explorer.exe process
# Change the host_name to match the name of the host you defined above

define service{
      use               generic-service
      host_name         winbox
      service_description     Explorer
      check_command           check_nt!PROCSTATE!-d SHOWALL -l
Explorer.exe
      }
```

nagios.cfg (excerto) – Ficheiro de configuração principal do nagios. Transcreve-se apenas o início com o propósito de mostrar a inclusão de um *event broker* que efectuará processamento em paralelo com o Nagios. O ficheiro completo constitui-se de 1330 linhas que permitem configurar todo o tipo de comportamento do nagios e dos seus agentes remotos.

```
##############################################################################
#####
#
# NAGIOS.CFG - Sample Main Config File for Nagios 3.2.3
#
# Read the documentation for more information on this configuration
# file.  I've provided some comments here, but things may not be so
# clear without further explanation.
#
# Last Modified: 12-14-2008
#
##############################################################################
#####


broker_module=/usr/local/nagios/addons/merlin/merlin.so
/usr/local/nagios/addons/merlin/merlin.conf


# LOG FILE
# This is the main log file where service and host events are logged
# for historical purposes.  This should be the first option specified
# in the config file!!!



log_file=/usr/local/nagios/var/nagios.log



# OBJECT CONFIGURATION FILE(S)
# These are the object configuration files in which you define hosts,
```

```
# host groups, contacts, contact groups, services, etc.
# You can split your object definitions across several config files
# if you wish (as shown below), or keep them all in a single config file.

# You can specify individual object config files as shown below:
cfg_file=/usr/local/nagios/etc/objects/commands.cfg
cfg_file=/usr/local/nagios/etc/objects/contacts.cfg
cfg_file=/usr/local/nagios/etc/objects/timeperiods.cfg
cfg_file=/usr/local/nagios/etc/objects/templates.cfg

# Definitions for monitoring the local (Linux) host
cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

# Definitions for monitoring a Windows machine
cfg_file=/usr/local/nagios/etc/objects/windows.cfg
cfg_file=/usr/local/nagios/etc/objects/laptop.cfg
cfg_file=/usr/local/nagios/etc/objects/voip.cfg

# Definitions for monitoring a router/switch
#cfg_file=/usr/local/nagios/etc/objects/switch.cfg

# Definitions for monitoring a network printer
#cfg_file=/usr/local/nagios/etc/objects/printer.cfg



# You can also tell Nagios to process all config files (with a .cfg
# extension) in a particular directory by using the cfg_dir
# directive as shown below:

#cfg_dir=/usr/local/nagios/etc/servers
#cfg_dir=/usr/local/nagios/etc/printers
#cfg_dir=/usr/local/nagios/etc/switches
#cfg_dir=/usr/local/nagios/etc/routers




# OBJECT CACHE FILE
```

```
# This option determines where object definitions are cached when
# Nagios starts/restarts.  The CGIs read object definitions from
# this cache file (rather than looking at the object config files
# directly) in order to prevent inconsistencies that can occur
# when the config files are modified after Nagios starts.

object_cache_file=/usr/local/nagios/var/objects.cache

[...]
```

# Icinga – Capturas de Ecrã

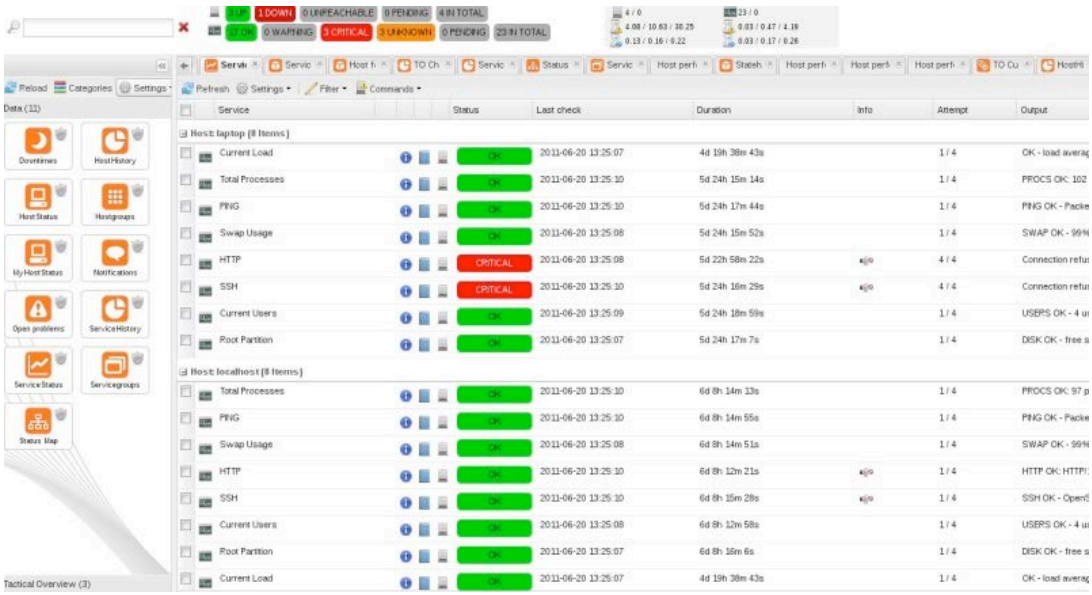


Fig. A39 – Vista geral de serviços por equipamento

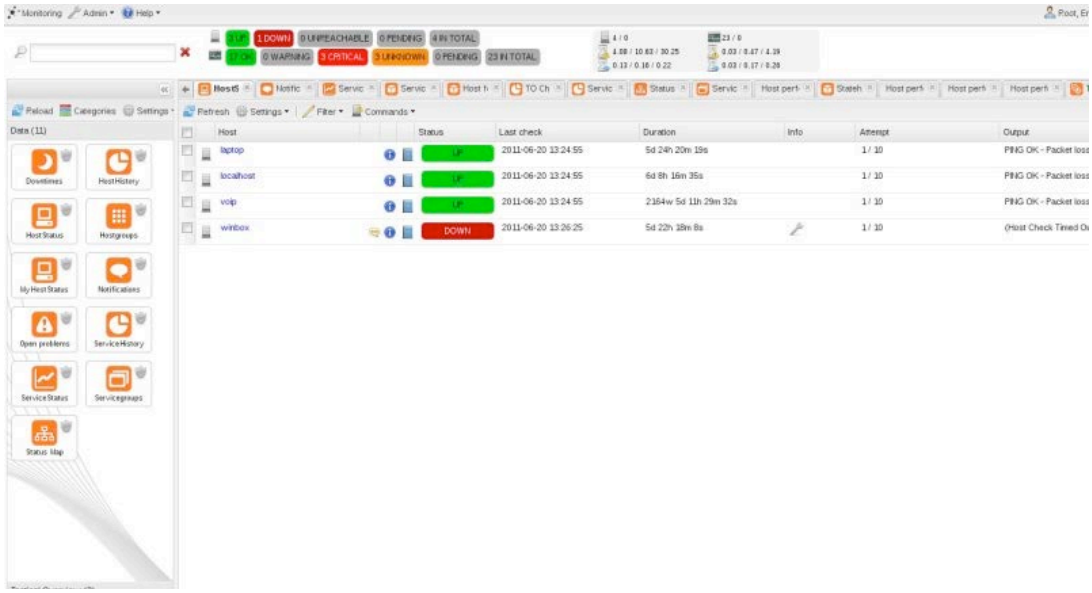


Fig. A40 – Vista geral de equipamentos


Fig. A41 – Gráficos circulares de equipamentos e serviços


Fig. A42 – Jasper Reports – Relatórios vários

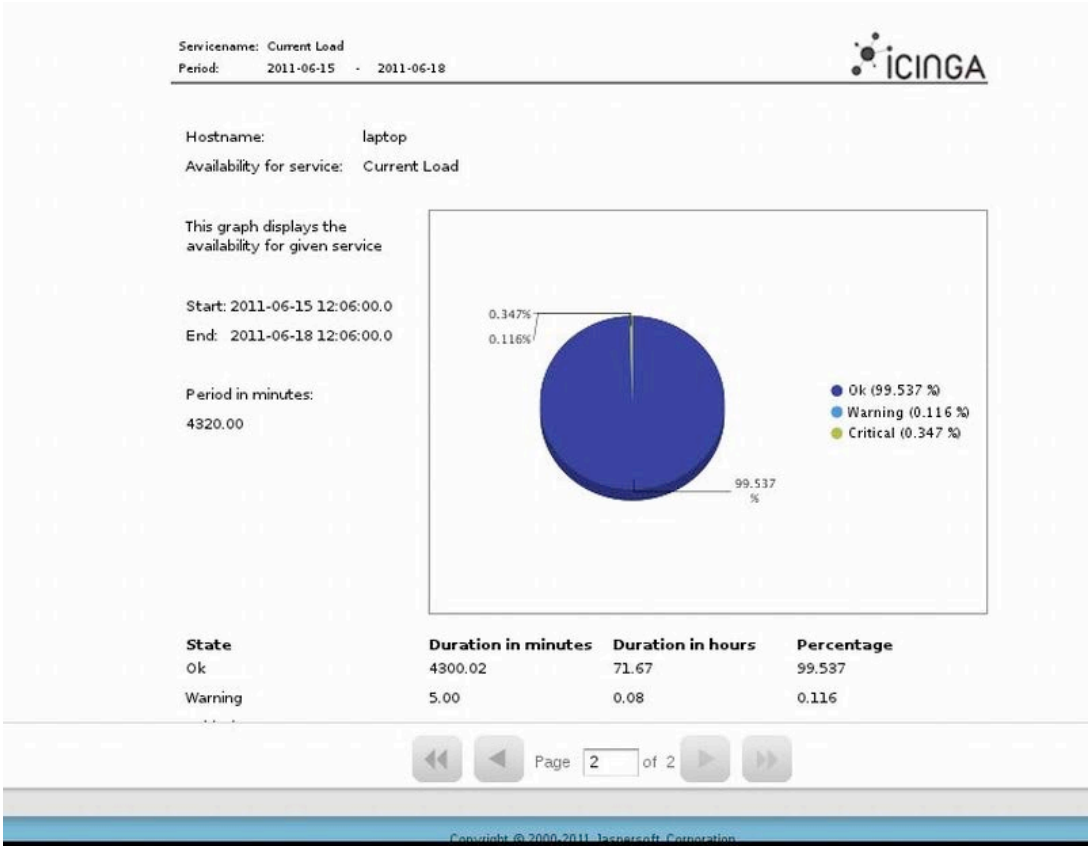



Fig A43 – Jasper Reports – Relatórios

A produção de gráficos detalhados foi implementada no Icinga através do *plugin* PNP4Nagios, produzindo assim resultados com as mesmas capacidades e similares aos apresentados nas capturas de ecrã do *plugin* Check_MK.

Cada visualização do icinga abre num novo separador do lado direito do ecrã, sendo os mesmos visíveis nas Figs. A39 e A40. Cada separador pode conter visualação de serviços ou equipamentos vários, bem como detalhes de cada um deles ou ecrãs de coefiguração, permitindo assim ao administrador de sistema manter abertos os separadores com as funcionalidades mais usadas providenciando um rápido acesso. Existe igualmente a opção de cada separador ser exibido durante algum tempo sendo de seguida mostrado o próximo, sendo assim mostrada a informação dos vários separadores sequencialmente e em ciclo.

# Icinga – Ficheiros de Configuração

Sendo o Icinga uma versão alternativa do nagios possui os mesmos ficheiros de configuração demonstrados na solução anterior baseada em Merlin e Ninja, com a excepção da referência ao *event broker* no ficheiro Nagios.conf, nesta ferramenta renomeado icinga.conf.

O mesmo não é aqui transcrito devido à sua grande extensão e à não adição de qualquer mais valia, pois as alterações efectuadas foram a nível dos ficheiros individuais de cada equipamento e de comandos, previamente ilustrados na ferramenta anterior, e não ao nível do icinga.conf.

# Check_Oracle_health

Lista dos parâmetros monitorizáveis numa base de dados Oracle, através do argumento mode no plugin check_oracle_health

| Keyword | Description | Range |
|---|---|---|
| tnsping | Listener | |
| connection-time | Determines how long connection establishment and login take | 0..n Seconds (1, 5) |
| connected-users | The sum of logged in users at the database | 0..n (50, 100) |
| session-usage | Percentage of max possible sessions | 0%..100% (80, 90) |
| process-usage | Percentage of max possible processes | 0%..100% (80, 90) |
| rman-backup-problems | Number of RMAN-errors during the last three days | 0..n (1, 2) |
| sga-data-buffer-hit-ratio | Hitrate in the Data Buffer Cache | 0%..100% (98:, 95:) |
| sga-library-cache-hit-ratio | Hitrate in the Library Cache | 0%..100% (98:, 95:) |
| sga-dictionary-cache-hit-ratio | Hitrate in the Dictionary Cache | 0%..100% (95:, 90:) |
| sga-latches-hit-ratio | Hitrate of the Latches | 0%..100% (98:, 95:) |
| sga-shared-pool-reloads | Reload-Rate in the Shared Pool | 0%..100% (1, 10) |
| sga-shared-pool-free | Free Memory in the Shared Pool | 0%..100% (10:, 5:) |
| pga-in-memory-sort-ratio | Percentage of sorts in the memory. | 0%..100% (99:, 90:) |

| | | |
|---|---|---|
| invalid-objects | Sum of faulty Objects, Indices, Partitions | |
| stale-statistics | Sum of objects with obsolete optimizer statistics | n (10, 100) |
| tablespace-usage | Used diskspace in the tablespace | 0%..100% (90, 98) |
| tablespace-free | Free diskspace in the tablespace | 0%..100% (5:, 2:) |
| tablespace-fragmentation | Free Space Fragmentation Index | 100..1 (30:, 20:) |
| tablespace-io-balanc | IO-Distribution under the datafiles of a tablespace | n (1.0, 2.0) |
| tablespace-remaining-time | Sum of remaining days until a tablespace is used by 100%. The rate of increase will be calculated with the values from the last 30 days. (With the parameter –lookback different periods can be specified) | Days (90:, 30:) |
| tablespace-can-allocate-next | Checks if there is enough free tablespace for the next Extent. | |
| flash-recovery-area-usage | Used diskspace in the flash recovery area | 0%..100% (90, 98) |
| flash-recovery-area-free | Free diskspace in the flash recovery area | 0%..100% (5:, 2:) |
| datafile-io-traffic | Sum of IO-Operationes from Datafiles per second | n/sec (1000, 5000) |
| datafiles-existing | Percentage of max possible datafiles | 0%..100% (80, 90) |
| soft-parse-ratio | Percentage of soft-parse-ratio | 0%..100% |
| switch-interval | Interval between RedoLog File Switches | 0..n Seconds (600:, 60:) |
| retry-ratio | Retry-Rate in the RedoLog Buffer | 0%..100% (1, 10) |
| redo-io-traffic | Redolog IO in MB/sec | n/sec (199,200) |

| | | |
|---|---|---|
| roll-header-contention | Rollback Segment Header Contention | 0%..100% (1, 2) |
| roll-block-contention | Rollback Segment Block Contention | 0%..100% (1, 2) |
| roll-hit-ratio | Rollback Segment gets/waits Ratio | 0%..100% (99:, 98:) |
| roll-extends | Rollback Segment Extends | n, n/sec (1, 100) |
| roll-wraps | Rollback Segment Wraps | n, n/sec (1, 100) |
| seg-top10-logical-reads | Sum of the userprocesses under the top 10 logical reads | n (1, 9) |
| seg-top10-physical-reads | Sum of the userprocesses under the top 10 physical reads | n (1, 9) |
| seg-top10-buffer-busy-waits | Sum of the userprocesses under the top 10 buffer busy waits | n (1, 9) |
| seg-top10-row-lock-waits | Sum of the userprocesses under the top 10 row lock waits | n (1, 9) |
| event-waits | Waits/sec from system events | n/sec (10,100) |
| event-waiting | How many percent of the elapsed time has an event spend with waiting | 0%..100% (0.1,0.5) |
| enqueue-contention | Enqueue wait/request-Ratio | 0%..100% (1, 10) |
| enqueue-waiting | How many percent of the elapsed time since the last run has an Enqueue spend with waiting | 0%..100% (0.00033,0.0033) |
| latch-contention | Latch misses/gets-ratio. With –name a Latchname or Latchnumber can be passed over. (See list-latches) | 0%..100% (1,2) |
| latch-waiting | How many percent of the elapsed time since the last run has a Latch spend with waiting | 0%..100% (0.1,1) |
| sysstat | Changes/sec for any value from v$sysstat | n/sec (10,10) |
| sql | Result of any SQL-Statement that returns a number. The statement itself is passed over with the parameter –name. A Label for the performance data output can be passed over with | n (1,5) |

| | | |
|---|---|---|
| | the parameter –name2. | |
| list-tablespaces | Prints a list of tablespaces | |
| list-datafiles | Prints a list of datafiles | |
| list-latches | Prints a list with latchnames and latchnumbers | |
| list-enqueues | Prints a list with the Enqueue-Names | |
| list-events | Prints a list with the events from (v$system_event). Besides event_number/event_id a shortened form of the eventname is printed out. This could be use as Nagios service descriptions. Example: lo_fi_sw_co = log file switch completion | |
| list-background-events | Prints a list with the Background-Events | |
| list-sysstats | Prints a list with system-wide statistics | |

# Icinga Vs. Nagios

Tabela comparativa disponibilizada em www.icinga.org

| | ICINGA | NAGIOS 3.2.3 | NAGIOS XI |
|---|---|---|---|
| Monitor unlimited hosts | Free | Free | $ 2,495 USD |
| CORE | | | |
| Distributed system | ✓ | ✗ | ✗ |
| Distributed monitoring | ✓ | ✓ | ✓ |
| SLA reports | ✓ | ✗ | ✓ |
| Virtual appliance | ✓ | ✗ | ✓ |
| Databases supported | MySQL<br>PostgreSQL<br>Oracle | MySQL | MySQL |
| SNMP & Syslog monitoring | Via addon | Via addon | Via addon |
| Triggers & multi-channel alerts | ✓ | ✓ | ✓ |
| WEB | | | |
| Underlying infrastructure | AJAX/ExtJs<br>PHP5/Agavi<br>HTML/CSS | CGI/C<br>HTML/CSS | AJAX/jQuery<br>PHP5/Nagios Synthesis Framework<br>HTML/CSS |
| Authentication by | LDAP<br>Database<br>HTTPBasic | HTTP Basic | Database |
| Control access to objects by | Contact group | Contact group | Contact group |

| | | | |
|---|---|---|---|
| | Host group<br>Service group<br>Custom variable | | |
| Open, modularized concept | ✓ | ✗ | ✗ |
| Dynamic dashboards | ✓ | ✗ | ✓ |
| Dynamic table views | ✓ | ✗ | ✗ |
| Live search | ✓ | ✗ | ✗ |
| Compound commands | ✓ | ✗ | ✗ |
| Dynamic drill down maps | ✓ | ✗ | ✗ |
| HTTP / REST interface | ✓ | ✗ | ✗ |
| Mobile version for iPhone & Android | ✓ | ✓ | ✗ |
| Reporting package | ✓ | ✗ | ✗ |
| Template grid and tactical overviews | ✓ | ✗ | ✗ |
| Web API for monitoring data (XML, JSON, SOAP) | ✓ | ✗ | ✗ |
| Multilingual interface | 20+ languages | ✗ | ✗ |
| INTEGRATED MODULES (OUT OF THE BOX) | | | |
| Performance charts (PNP) | ✓ | ✗ | ✓ |
| Web based configuration (eg. LConf) | ✓ | ✗ | ✗ |
| Heatmap | ✓ | ✗ | ✗ |
| Business Process View (BusinessProcessAddon) | ✓ | ✗ | ✗ |
| Hypermap | ✓ | ✗ | ✗ |

| | | | |
|---|---|---|---|
| BACKEND API | | | |
| API to Backend | ✓ | ✗ | ✗ |
| Extensions coding without parsers or queries | ✓ | ✗ | ✗ |
| DOCS / SUPPORT / DEVELOPMENT | | | |
| Community Support | ✓ | ✓ | ✗ |
| Multilingual documentation | ✓ | ✓ | ✗ |
| Flexible format (eg. Docbook) | ✓ | ✗ | ✗ |
| Public and detailed roadmap | ✓ | ✗ | ✗ |
| GIT repository | ✓ | ✗ | ✗ |
| Full OS license | ✓ | ✓ | ✗ |