INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Análise e Implementação de uma Ferramenta de Monitorização da Rede

GONÇALO DOS SANTOS CORREDOURA
Setembro de 2025

POLITÉCNICO
DO PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Análise e Implementação de uma Ferramenta de Monitorização da Rede

Gonçalo dos Santos Corredoura

Dissertação para obtenção do Grau de Mestre em

Engenharia Informática, Área de Especialização em

Cibersegurança e Administração de Sistemas

Orientador: Bertil P. Marques

Porto, setembro 2025

Declaração de Integridade

Declaro ter conduzido este trabalho académico com integridade.

Não plagiei ou apliquei qualquer forma de uso indevido de informações ou falsificação de resultados ao longo do processo que levou à sua elaboração.

Portanto, o trabalho apresentado neste documento é original e de minha autoria, não tendo sido utilizado anteriormente para nenhum outro fim.

Declaro ainda que tenho pleno conhecimento do Código de Conduta Ética do P.PORTO.

Gonçalo Corredoura

ISEP, Porto, 28 de setembro de 2025

Dedicatória

Dirigida à minha família e aos meus amigos, aqueles que mais me apoiam.

Resumo

As ferramentas de monitorização de rede desempenham um papel essencial na segurança, disponibilidade e eficiência da gestão de sistemas informáticos. Através da sua utilização é possível detetar de forma precoce incidentes, prevenir falhas e mitigar potenciais ataques cibernéticos.

A presente dissertação descreve o processo de análise, seleção e implementação de uma solução de monitorização adaptada ao contexto da empresa Arsopi, na sequência da descontinuação da ferramenta anteriormente utilizada. Para tal, foi efetuado um estudo comparativo entre três alternativas *open source* — Zabbix, CheckMK e Nagios Core —, avaliando as respetivas funcionalidades, arquiteturas, compatibilidade, processos de instalação, vantagens e limitações. A análise permitiu concluir que o Zabbix era a solução que melhor se adequava às necessidades identificadas.

Na vertente prática, procedeu-se à implementação do Zabbix num servidor dedicado, seguindo um conjunto de medidas de segurança, e à sua configuração para monitorização de servidores, dispositivos de rede e *firewall*. Foram ainda desenvolvidos itens personalizados, recorrendo a WMI, SNMP e APIs, bem como *dashboards* adaptados às necessidades da equipa de IT. Os resultados obtidos demonstraram uma melhoria significativa em termos de eficiência operacional, proatividade na deteção de problemas e redução do tempo de resposta a incidentes, sendo o *feedback* da equipa globalmente positivo.

A dissertação conclui que a adoção de uma solução de monitorização adequada, configurada de forma personalizada e segura, representa uma ferramenta estratégica para a resiliência e robustez da infraestrutura tecnológica, contribuindo para uma gestão mais eficaz e para um aumento da segurança global da rede.

Palavras-chave: Monitorização da rede, Zabbix, Segurança da Informação, Administração de Sistemas

Abstract

Network monitoring tools play a crucial role in ensuring the security, availability, and efficiency of IT systems management. Their use enables the early detection of incidents, the prevention of failures, and the mitigation of potential cyberattacks.

This dissertation describes the process of analyzing, selecting, and implementing a monitoring solution tailored to the context of Arsopi, following the discontinuation of the company's previous tool. A comparative study of three open-source alternatives — Zabbix, CheckMK, and Nagios Core — was conducted to evaluate their functionalities, architectures, compatibility, installation processes, advantages, and limitations. The analysis concluded that Zabbix was the solution best suited to the organization's requirements.

In the practical component, Zabbix was deployed on a dedicated server, accompanied by a set of security measures, and configured to monitor servers, network devices, and the firewall. Additionally, custom items were developed using WMI, SNMP, and API integrations, along with personalized dashboards adapted to the IT team's needs. The results demonstrated a significant improvement in operational efficiency, proactive incident detection, and reduced response times, with overall positive feedback from the IT staff.

The dissertation concludes that adopting a properly configured and secure monitoring solution constitutes a strategic tool for strengthening the resilience and robustness of technological infrastructures, contributing to more effective system management and enhanced overall network security.

Keywords: Network Monitoring, Zabbix, Information Security, System Administration

Agradecimentos

Em primeiro lugar, quero agradecer à minha família, especialmente à minha mãe, que sempre foi um dos meus maiores pilares, aos meus irmãos e ao meu pai.

Em segundo lugar, agradeço a todos os meus amigos que também, de uma forma ou outra, me ajudaram a atingir os meus objetivos, sempre me dando todo o apoio necessário.

De seguida, quero deixar um especial agradecimento à minha orientadora do ISEP, a professora Bertil Marques, por ter aceitado a minha proposta, mais uma vez, e me ter ajudado no desenvolvimento deste projeto.

Índice

1 Introdução 1

1.1 Contexto do problema 1

1.2 Descrição do problema 2

1.3 Objetivos 3

1.4 Análise ética do problema 4

1.5 Planeamento das atividades do projeto 5

1.5.1 Project Charter 5

1.5.2 Scope of Work 7

1.5.3 Cronograma do projeto 8

1.5.4 Procedimentos de monitorização e controlo 9

1.5.5 Identificação e gestão de riscos do projeto 10

1.5.6 Tipo de abordagem a ser utilizada 11

1.6 Estrutura do documento 12

2 Estado da arte 13

2.1 Monitorização da rede 13

2.1.1 Definição 13

2.1.2 Principais funcionalidades 14

2.1.3 Componentes e protocolos 14

2.1.4 Benefícios 17

2.2 Ferramentas de monitorização da rede 18

2.2.1 Zabbix 18

2.2.2 Checkmk 20

2.2.3 Nagios Core 23

2.3 Comparação das ferramentas apresentadas 26

2.4 Decisão final 29

3 Implementação da solução 31

3.1 Metodologia 31

3.2 Arquitetura da solução 32

3.2.1 Componentes do Zabbix 33

3.2.2 Arquitetura física 33

3.2.3 Dimensionamento e características da máquina 34

3.2.4 Integração com APIs externas 34

3.2.5 Diagrama de rede 35

3.3 Configuração inicial do servidor 36

3.4 Instalação do Zabbix 37

3.5 Preparação do ambiente para monitorização 41

3.5.1 Preparação para WMI 41
3.5.2 Preparação para SNMP 42
3.6 Configuração do Zabbix 42
3.6.1 Host Groups 43
3.6.2 Templates 45
3.6.3 Hosts e Host Groups 68
3.6.4 Notificações e alertas 86
3.6.5 Dashboards personalizados 92
3.6.6 Outros desenvolvimentos 100
4 Avaliação da solução implementada 109
4.1 Casos reais de utilização 109
4.2 Feedback da equipa de IT 110
4.3 Futuras melhorias 111
5 Conclusão 113
Referências 115

Lista de Figuras

Figura 1 - Diagrama WBS..... 8
Figura 2 - Diagrama de Gantt com cronograma do projeto..... 9
Figura 3 - Diagrama da arquitetura da solução de monitorização implementada..... 35
Figura 4 - *Dashboard* inicial..... 40
Figura 5 - Página para alteração da *password* do Zabbix *Administrator*..... 40
Figura 6 – Processo de criação de um *host group*..... 44
Figura 7 – Processo de criação de um *host*..... 45
Figura 8 –Processo de criação de um *template*..... 46
Figura 9 – Resultados obtidos para a percentagem de uso de CPU para vários servidores..... 51
Figura 10 – Demonstração dos diferentes *triggers* para o mesmo item de CPU..... 51
Figura 11 – Exemplo de obtenção do espaço livre dos discos para o servidor ASP-BD antes do tratamento dos dados..... 52
Figura 12 – Demonstração dos itens configurados a partir do item “Discos via WMI”..... 53
Figura 13 – Demonstração da tarefa em JavaScript de *preprocessing*..... 53
Figura 14 – Exemplo de resultado para o item “Discos via WMI”..... 55
Figura 15 – Demonstração dos *triggers* para o servidor ASP-BD, para o item “Discos via WMI”..... 55
Figura 16 – Exemplo de problema gerado a partir de erros do Event Viewer, mostrado no *dashboard*..... 57
Figura 17 - Demonstração de um outro problema relacionado com este item..... 57
Figura 18 – Exemplo de demonstração de texto recebido relacionado com tráfego de rede num servidor..... 58
Figura 19 – Demonstração de um dos dois itens configurados a partir do *script*..... 59
Figura 20 – Demonstração dos resultados obtidos para cada servidor dos dois itens configurados a partir do *script*..... 60
Figura 21 – Demonstração do item..... 62
Figura 22 – Demonstração dos *triggers* ativos no *template* “Arsopi Switches”..... 63
Figura 23 – *Item prototypes* do template “Arsopi Switches”..... 64
Figura 24 – Demonstração do total de bits recebidos por *switches*..... 65
Figura 25 – Configuração do *host* da *firewall* com o *template* “FortiGate by SNMP”..... 67
Figura 26 – Demonstração de exemplo de diferença de horário no mês de agosto de 2025... 76
Figura 27 – Demonstração do resultado obtido via *script* sem tarefa de pré-processamento. 77
Figura 28 – Demonstração da ativação do registo de eventos relacionados com utilizadores e grupos..... 79
Figura 29 – Demonstração do script “get_fortigate_vpn_users.sh”..... 84
Figura 30 - Demonstração da lista de utilizadores VPN SSL..... 85
Figura 31 – Demonstração, por passos, da criação de um “Media Type**”**..... 87
Figura 32 – “Message templates” adicionados ao “Media Type”..... 87
Figura 33 – *Template* da mensagem para o tipo “Problema”..... 88

Figura 34 – Demonstração do processo de criação de uma nova “media” 89
Figura 35 – Demonstração do processo de criação de uma nova “action” 90
Figura 36 – Demonstração da configuração dos detalhes da operação associada a ação 91
Figura 37 – Exemplo de email recebido após ser gerado um problema de cariz “Desastre” 91
Figura 38 – Demonstração da página “Geral” do *dashboard* “Arsopi” 94
Figura 39 – Demonstração da página “Small Info” do *dashboard* “Arsopi” 95
Figura 40 – Demonstração da página “Server Page” do *dashboard* “Arsopi” (1/2) 96
Figura 41 - Demonstração da página “Server Page” do *dashboard* “Arsopi” (2/2) 97
Figura 42 - Demonstração da página “Switch Page” quando nenhum *host* está selecionado .. 98
Figura 43 – Demonstração da página “Switch Page” quando está um *host* selecionado 98
Figura 44 – Demonstração da página “Firewall” do *dashboard* “Arsopi” 100
Figura 45 – Demonstração de um excerto de código da alteração no ficheiro “history.php” 101
Figura 46 – Demonstração de exemplo após a alteração feita no filtro 102
Figura 47 – Demonstração da página “Small Info” do *dashboard* “Arsopi” antes das alterações visuais 104

Lista de Tabelas

Tabela 1 - Comparação das ferramentas 28
Tabela 2 - Códigos relacionados com utilizadores 80
Tabela 3 - Códigos relacionados com grupos 81

Lista de Códigos

Código 1 - Obtenção de privilégios administrativos, adição do repositório oficial e atualização de pacotes 37
Código 2 - Instalação dos pacotes principais do Zabbix 37
Código 3 - Criação da base de dados e utilizador Zabbix 38
Código 4 - Importação do esquema inicial da base de dados 38
Código 5 - Reversão da configuração de segurança no MySQL 38
Código 6 - Reinício e ativação dos serviços principais 39
Código 7 – *Script* "get_pings.sh" 47
Código 8 – Script "get_cpu_usage.sh" 49
Código 9 – *Script* "get_disks_freespace.sh" 52
Código 10 – Código JavaScript da tarefa de *preprocessing* de cada item 54
Código 11 – Demonstração do *script* "get_eventviewer_errors.sh" 56
Código 12 – *Script* "get_memory_usage.sh" 57
Código 13 – *Script* "get_traffic_values.sh" 58
Código 14 – *Script* "get_trendmicro_services.sh" 60
Código 15 – Tarefa de pré-processamento em JavaScript 61
Código 16 – Demonstração da *string* retornada pelo *script* antes do pré-processamento 61
Código 17 – Chave usada no item "Total Bits Received" 64
Código 18 – Chave usada no item "Total Bits Sent" 66
Código 19 – *Script* "get_spool.sh" 69
Código 20 – Tarefa de pré-processamento JavaScript do resultado do item 69
Código 21 – *Script* "get_sql_services.sh" 70
Código 22 – Script "get_qlik_services.sh" 71
Código 23 – *Script* "get_cad_services.sh" 72
Código 24 – *Script* "get_cam_services.sh" 72
Código 25 – *Script* "get_infor_services.sh" 73
Código 26 – *Script* "get_ng_services.sh" 74
Código 27 – *Script* "get_h2st_services.sh" 74
Código 28 – *Script* "get_tasks_status.sh" 75
Código 29 – *Script* "get_localtime.sh" 76
Código 30 – Código JavaScript de pré-processamento do item de diferença de horário 77
Código 31 – *Script* "get_eventviewer_info.sh" 80
Código 32 – Script "get_password_change_info.sh" 82
Código 33 – Função "removePaddingFromContainers" 103
Código 34 – Função "removeUselessWidgets" 105
Código 35 – Função "selectHost" 107

Acrónimos e Símbolos

Lista de Acrónimos

API	*Application Programming Interface*
AP	*Access Point*
COM	*Component Object Model*
DCOM	*Distributed Component Object Model*
HTTP	*Hypertext Transfer Protocol*
ICMP	*Internet Control Message Protocol*
IP	*Internet Protocol*
ISEP	Instituto Superior de Engenharia do Porto
IT	*Information Technology*
JMX	*Java Management Extensions*
LAN	*Local Area Network*
MacOS	*Macintosh Operating System*
NRPE	*Nagios Remote Plugin Executor*
RPC	*Remote Procedure Call*
SNMP	*Simple Network Management Protocol*
SMS	*Short Message Service*
SSH	*Secure Shell*
TCP	*Transmission Control Protocol*
TelNet	*Teletype Network*
TI	Tecnologia da Informação
WMI	*Windows Management Instrumentation*
WQL	*WMI Query Language*

1 Introdução

Neste primeiro capítulo do trabalho será feita uma pequena introdução ao problema que está a ser explorado, passando por uma breve contextualização, uma descrição do problema, serão mencionados os objetivos propostos, será feita uma análise ética do problema em questão, um planeamento das atividades do projeto e, por fim, uma pequena apresentação da estrutura do documento.

1.1 Contexto do problema

A Arsopi Lda é uma empresa que trabalha principalmente no ramo metalúrgico. Apesar de não ser uma empresa diretamente de TI, faz toda a sua gestão de uma forma digital e, para isso, conta com vários dispositivos tecnológicos que facilitam esta mesma gestão, como servidores, *switches*, máquinas para os seus colaboradores dos diferentes departamentos poderem ajudar nesta gestão, APs, entre outros.

Na atualidade em que vivemos, alguns dos pontos mais importantes a manter num sistema informático são a segurança e a disponibilidade do mesmo. Cada vez mais se torna uma obrigação ter medidas de segurança implementadas em sistemas deste tipo, tais como medidas associadas à recuperação ou não perda total de dados em caso de um ataque cibernético, como *backups* de dados, medidas de prevenção deste tipo de ataques e medidas que garantam a constante disponibilidade dos serviços que são usados diariamente pelos colaboradores da empresa. Um ataque à segurança do sistema ou a interrupção de um serviço ou servidor de uma empresa podem trazer graves consequências à mesma.

Existem várias formas de prevenir e evitar este tipo de incidentes e uma delas é a utilização de ferramentas de monitorização da rede. Através da monitorização da rede de um sistema informático é possível evitar inúmeros problemas informáticos, principalmente relacionados com os dois pontos mencionados anteriormente, ou seja, problemas relacionados com a disponibilidade e boa performance de dispositivos informáticos, como servidores, e problemas relacionados com a segurança da rede.

Neste momento, a empresa conta com uma ferramenta de monitorização da rede que não está totalmente funcional e isso pode trazer graves impactos para o sistema. A ferramenta deixou de conseguir apresentar os dados que apresentou outrora e não consegue enviar alertas aquando da existência de valores fora do que é considerado "normal", impedindo assim que a equipa de administração do sistema possa ter um comportamento proativo e de prevenção, ao invés de um comportamento reativo no que toca à prevenção e resolução de incidentes.

1.2 Descrição do problema

A ferramenta de monitorização da rede que está a ser atualmente utilizada na empresa deixou de ter capacidade para suprir as nossas necessidades e, portanto, tomou-se a decisão de adotar uma nova ferramenta semelhante a esta.

Esta ferramenta era utilizada principalmente para obter métricas dos servidores tais como a percentagem de utilização de CPU, a percentagem de utilização de memória, a percentagem de utilização do(s) disco(s), os valores de tráfego de entrada e saída de rede, entre outras métricas. Assim que algum valor destes estivesse fora dos parâmetros definidos como funcionamento normal, era enviada uma notificação ou um alerta para o administrador do sistema, de forma que este pudesse agir de forma proativa e tratar do problema antes que este trouxesse consequências para o bom funcionamento do sistema.

O problema da ferramenta utilizada é que esta foi descontinuada pelo seu fabricante, deixando de receber atualizações e, consequentemente, deixando de acompanhar a evolução tecnológica que acontece constantemente nos dias de hoje. Para além disso, a ferramenta deixou de conseguir captar os valores relacionados com as métricas acima mencionadas, como os valores de CPU, memória, disco e de tráfego de rede. Tendo isto em conta, parou também de gerar os alertas que permitiam que a equipa de TI pudesse agir antes do aparecimento de um problema, reduzindo a capacidade de resposta e aumentando o risco de falhas inesperadas.

Outra questão que nos leva à troca de ferramenta é o facto de esta ser um pouco limitada no que toca à diversidade de métricas, não sendo possível fazer a monitorização de serviços, por exemplo, algo que pode ser extremamente útil e vantajoso para manter a segurança e disponibilidade do sistema.

1.3 Objetivos

O objetivo final desta dissertação consiste na implementação de uma ferramenta de monitorização da rede capaz de suprir todas as necessidades identificadas pela empresa. Será feita uma análise de algumas ferramentas existentes no mercado e será implementada aquela que cumpra com maior quantidade de requisitos identificados.

A ferramenta ou solução implementada:

1. Deve ser *open source*, não tendo qualquer custo a nível financeiro e permitindo a customização da mesma caso seja necessário;
2. Não deve acarretar qualquer custo extra a nível de recursos para as máquinas e dispositivos a serem monitorizados, por exemplo, através do uso de agentes;
3. Deve permitir monitorizar as métricas básicas de dispositivos, tais como as que foram mencionadas anteriormente neste documento (CPU, memória, discos e tráfego de rede);
4. Deve permitir, de preferência, monitorização de outras métricas, tais como a de execução de serviços, de registos de eventos do Event Viewer, entre outros. Quanto maior a diversidade de métricas disponíveis, melhor;
5. Deve permitir a monitorização não apenas de máquinas e servidores, mas também de outros dispositivos de rede, como *switches*, *access points*, entre outros;
6. Deve permitir definir os valores normais para certas métricas ou, por outro lado, definir os valores limite para os quais deixam de estar dentro dos parâmetros normais. Por exemplo, um valor normal pode ser um determinado dispositivo de rede responder a *pings* e um valor fora do normal é este deixar de responder a *pings*;
7. Deve permitir gerar alertas e enviar notificações para uma ou várias pessoas responsáveis pela administração da rede, com a informação necessária para que estas possam agir o mais rapidamente possível e em conformidade.
8. Deve permitir a apresentação de um *dashboard* simples e intuitivo que permita fazer uma análise rápida de possíveis erros e permita uma ação proativa ou uma reação mais rápida a possíveis incidentes.

Em suma, o objetivo passa por implementar uma ferramenta que possa cumprir com todos estes requisitos e possa ajudar a nossa equipa de TI a monitorizar e prevenir qualquer tipo de

incidente. Uma ferramenta que cumpra com estes objetivos e que ainda nos dê a possibilidade de monitorizar outras métricas ou de ter acesso aos valores das métricas monitorizadas de uma forma fácil e intuitiva que nos permita agir de uma forma rápida será valorizada.

1.4 Análise ética do problema

A monitorização da rede é cada vez mais uma prática essencial para assegurar o bom funcionamento e a segurança de um sistema informático. No entanto, este assunto pode levantar questões éticas que devem ser abordadas.

As ferramentas de monitorização da rede têm capacidade para obter vários tipos de informações dos equipamentos que são monitorizados e isso pode representar uma preocupação a nível ético. Caso sejam monitorizadas as máquinas dos colaboradores, é importante respeitar a privacidade dos utilizadores, monitorizando apenas as métricas estritamente necessárias para promover a segurança e a performance da rede, evitando dados que não são necessários a este propósito e que podem violar a privacidade dos utilizadores. Para além disso, é essencial garantir que os dados recolhidos não são utilizados de forma abusiva ou inadequada, isto é, para efeitos de espionagem ou vigilância excessiva. A obtenção dos dados não deve ter outro propósito que não seja a segurança e o desempenho da rede.

Da mesma forma, deve existir transparência para com os utilizadores, informando estes sobre a ação de monitorização dos seus dados, esclarecendo quais dados estão a ser obtidos, qual o tratamento desses dados e quais os objetivos. É importante garantir o consentimento dos colaboradores de forma a promover um bom ambiente profissional.

Uma medida importante é o estabelecimento de políticas internas da empresa de forma a definir quem pode aceder a estes dados obtidos através da monitorização dos equipamentos, como é que estes dados podem ser usados e o período de retenção desta informação. Deve-se definir por quanto tempo os dados serão guardados e deve-se garantir que os dados apenas sejam mantidos enquanto não forem irrelevantes. Assim que se tornarem irrelevantes, devem ser permanentemente eliminados.

A monitorização da rede e a obtenção e armazenamento dos dados obtidos devem estar de acordo com os termos legais em relação a este assunto.

Por último, os dados devem estar protegidos contra acessos não autorizados e ataques externos. Para isso, é importante o uso de criptografia para proteger tanto os dados em trânsito como os dados em repouso, e a implementação de uma política de acesso de forma que apenas pessoas autorizadas tenham acesso aos dados.

Tendo em conta o contexto e os objetivos atuais da empresa, apenas servidores e equipamentos de rede serão monitorizados e, dessa forma, não existem preocupações éticas relacionadas com os colaboradores diretamente. No entanto, de uma forma de preparar um possível futuro próximo, é importante garantir que existem um processo e normas para notificar os colaboradores e garantir que nenhuma lei ou política interna possa ser violada. De qualquer das formas, a maior parte das considerações éticas aqui mencionadas continua a aplicar-se mesmo monitorizando apenas servidores e dispositivos de rede.

1.5 Planeamento das atividades do projeto

1.5.1 Project Charter

Nesta pequena secção serão definidos alguns pontos do *Project Charter* como as partes interessadas, os benefícios, pressupostos e restrições. Os restantes pontos que fazem parte deste "documento" estão explícitos noutras partes da dissertação.

Partes Interessadas

As partes interessadas neste projeto são as seguintes:

- Equipa de IT: serão os utilizadores da ferramenta implementada. Interessa-lhes que a ferramenta facilite o seu trabalho de monitorização da rede e, de igual forma, ajude a garantir a segurança e o bom desempenho da mesma. Podem dar feedback em relação ao processo a ser desenvolvido e aos requisitos que podem ser necessários para facilitar o trabalho.
- Organização: entidade para o qual o projeto será implementado. Tem interesse em obter uma ferramenta que acrescente valor à mesma, através da melhoria do desempenho e segurança da rede. Oferece os recursos necessários à implementação da ferramenta.

Benefícios

Este tipo de ferramentas é essencial em sistemas informáticos. Ajuda na identificação precoce de problemas na rede, permitindo um comportamento proativo no que toca à resolução deste tipo de incidentes, contribuindo para um menor tempo de inatividade. Permite a monitorização contínua da rede, evitando este tipo de falhas. Melhora o desempenho da rede na medida em que permite perceber que componentes ou equipamentos possam estar em sub-rendimento. Aumenta a segurança de um sistema informático uma vez que permite ter acesso a registos que podem precaver ataques informáticos. Para além disso, acaba por ser a troca de uma ferramenta que entrou em desuso por uma ferramenta ou uma solução que está funcional e que pode vir a oferecer mais funcionalidades que a anterior.

Pressupostos

Existem alguns pontos que são assumidos uma vez que não há certeza em relação a certos aspetos, tais como:

- A ferramenta escolhida e implementada será-nos-á útil durante um grande intervalo de tempo, contribuindo para o bom funcionamento da empresa durante largos anos.
- A ferramenta apresentará uma evolução ao longo do tempo que permite acompanhar as necessidades e requisitos da empresa.
- As partes interessadas fornecerão o suporte necessário para a escolha e a implementação da ferramenta.

Restrições

Uma restrição que pode influenciar a escolha e o cumprimento dos objetivos propostos está relacionada com a exigência de se optar por soluções que possam cumprir com todos os objetivos mencionados anteriormente e que não tragam qualquer tipo de custo para a organização, como, por exemplo, custos de licenciamento.

1.5.2 Scope of Work

O método de pesquisa utilizado neste projeto foi um método de pesquisa misto. Para a parte teórica do projeto, nomeadamente o estado da arte, foi utilizada uma pesquisa bibliográfica, de forma a entender as ferramentas apresentadas. Para o resto do projeto, foi adotada uma pesquisa experimental, de forma a testar certos comportamentos e configurações da ferramenta, e uma pesquisa observacional de forma a analisar os efeitos causados no sistema informático.

Em relação ao projeto em si, este divide-se em 5 fases:

- Fase de Pesquisa: fase em que existem a análise de diversas ferramentas e a comparação entre as mesmas.
- Fase de Planeamento: fase em que a ferramenta a ser adotada é escolhida e onde são definidos os requisitos técnicos.
- Fase de Implementação: fase em que acontece a instalação e configuração da ferramenta escolhida.
- Fase de Validação: fase em que são feitos testes de forma a garantir que a ferramenta cumpra com os objetivos propostos e esteja funcional.
- Fase de Conclusão: fase após a ferramenta passar todos os testes e cumprir com todos os objetivos propostos, sendo a fase em que se cria a documentação final e se procede à entrega do projeto.

Em relação aos “entregáveis”, teremos um documento correspondente apenas à parte teórica da dissertação, um documento correspondente à parte final da dissertação e uma solução pronta a ser usada para a monitorização da rede, tal como se pode ver na figura abaixo.

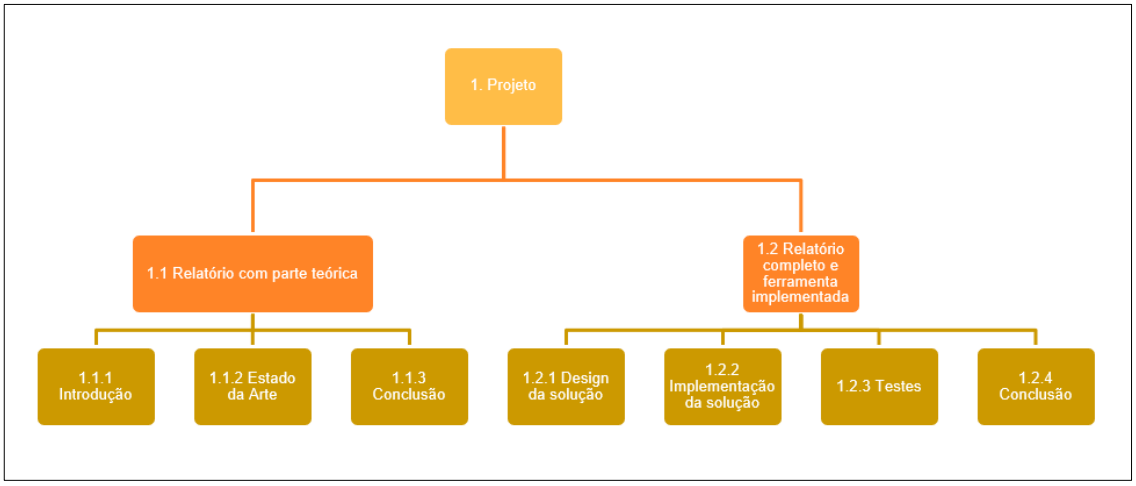


Figura 1 - Diagrama WBS

1.5.3 Cronograma do projeto

O projeto será dividido em duas fases principais, estando uma relacionada com a unidade curricular de PREPD (Preparação para Dissertação) e outra com a unidade curricular DIMEI (Dissertação). Tendo isso em conta, foi desenvolvido um diagrama de *Gantt* e foram estabelecidas as datas e durações de cada fase:

- 21-10-2024 a 04-11-2024 → 2 semanas → Levantamento de requisitos e objetivos;
- 04-11-2024 a 25-11-2024 → 3 semanas → Escolha de 3 ferramentas a serem estudadas ao pormenor;
- 25-11-2024 a 30-12-2024 → 5 semanas → Estudo profundo das 3 ferramentas e comparação das mesmas;
- 30-12-2024 a 04-01-2025 → 1 semana → Escolha de uma das ferramentas;
- 03-02-2025 a 24-02-2025 → 3 semanas → Design da solução;
- 24-02-2025 a 19-05-2025 → 12 semanas → Implementação da solução;
- 19-05-2025 a 02-06-2025 → 2 semanas → Avaliação da solução implementada.

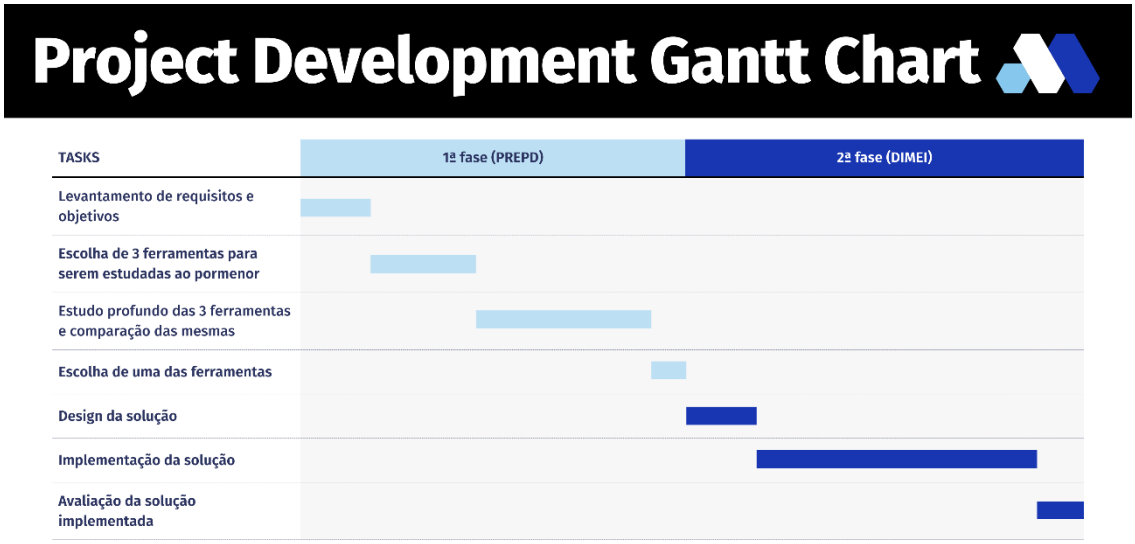


Figura 2 - Diagrama de Gantt com cronograma do projeto

1.5.4 Procedimentos de monitorização e controlo

De forma que o trabalho a ser desenvolvido possa ser controlado e monitorizado tanto pela orientadora do ISEP como pelo supervisor da organização, foram definidas algumas metodologias.

Neste caso, serão feitas reuniões com a orientadora do ISEP de 2 em 2 semanas, de forma a discutir o progresso do trabalho, o progresso da fase em que o projeto está inserido, esclarecer possíveis dificuldades e dúvidas e reavaliar prazos caso seja necessário. Estas reuniões serão *online*, de forma a facilitar o agendamento para as partes envolvidas e serão feitas através da plataforma Microsoft Teams.

Da mesma forma, serão feitas reuniões com o supervisor da organização, para garantir que os objetivos da ferramenta implementada estão a ser cumpridos, discutir os requisitos visto que pode existir a necessidade de acrescentar objetivos, dar a perceber a forma como a ferramenta está a ser implementada e obter validação por parte do supervisor acerca dos métodos de trabalho e decisões tomadas. Estas reuniões serão presenciais e serão marcadas de 2 em 2 semanas de igual forma.

É importante ressaltar que, apesar das reuniões serem marcadas em períodos de 2 semanas, ambas as partes mostram disponibilidade para a marcação de reuniões extra caso estas sejam necessárias, para o esclarecimento de dúvidas ou outras questões.

Para além disso, será utilizada uma ferramenta de acompanhamento do trabalho, ainda a ser decidida, à qual ambas as partes terão acesso de forma a poderem acompanhar as tarefas que estão a ser executadas no momento, que já foram executadas e as que ainda estão por fazer.

1.5.5 Identificação e gestão de riscos do projeto

Apesar da preparação envolvente na gestão de um projeto, existem sempre riscos inerentes à execução do mesmo que não podem ser totalmente evitados.

Alguns riscos identificados neste momento do projeto são:

- Incompatibilidade da ferramenta com a infraestrutura: a ferramenta implementada pode não funcionar como era expectável no ambiente real. Devido a diversos fatores, como, por exemplo, um gasto de recursos da máquina maior do que era esperado, fatores esses que não são totalmente planeáveis. De forma a mitigar este risco, é importante testar num ambiente de teste antes de passar para o ambiente real.
- Falta de conhecimento técnico: a equipa de desenvolvimento pode não ter os conhecimentos técnicos necessários à implementação, configuração e manutenção da ferramenta escolhida. Este risco pode causar um atraso em relação aos prazos estimados, devido ao facto de ser necessária a formação da equipa de desenvolvimento.
- Falta de opções na ferramenta escolhida: apesar de ser feito um estudo para perceber qual a melhor ferramenta tendo em conta os objetivos que tem de cumprir, pode haver pequenos detalhes que a ferramenta não consegue oferecer por predefinição, podendo ser necessária a personalização da mesma através de alteração do código. No entanto, este risco pode ser eliminado reservando tempo extra para a fase de implementação.

De uma forma geral, existem medidas que podem ser tomadas para gerir estes riscos de uma forma que permita que estes não alterem o plano de desenvolvimento do projeto:

- Revisão periódica dos riscos: fazer uma análise periódica, por exemplo, semanalmente, aos riscos associados ao desenvolvimento do projeto pode ser uma forma de ajudar a mitigar ou antecipar estes riscos.
- Monitorização contínua dos fatores internos e externos: fazer uma monitorização contínua de fatores que podem afetar o desenvolvimento do projeto é uma forma de prevenir este tipo de riscos, solucionando os problemas associados a estes de uma forma antecipada.

1.5.6 Tipo de abordagem a ser utilizada

Em relação ao tipo de abordagem a ser utilizada no desenvolvimento do projeto, a metodologia "*Waterfall*" poderá ser a que mais se adequa ao tipo de trabalho a ser desenvolvido. Isto porque, uma vez que temos todas as fases do projeto bem delineadas, assim como todos os objetivos e requisitos, podemos seguir este tipo de metodologia em que apenas seguimos para uma próxima fase depois de terminar a anterior.

No entanto, tal como a tecnologia que está em constante evolução, assim também pode estar este projeto e poderão ser encontrados novos requisitos a serem implementados na solução. Neste caso, e como ao longo do desenvolvimento do mesmo serão feitas várias reuniões tanto com o supervisor da organização como com a orientadora do ISEP para perceber a evolução do trabalho e a possível necessidade de novas funcionalidades no projeto, ter-se-á de ter em conta que podem originar novos requisitos destas reuniões e de forma a ser possível concretizar tudo, foi deixado um longo prazo entre o planeamento de entrega da solução e a real *deadline,* de forma a que estes pequenos "imprevistos" não alterem a execução atempada do projeto.

Em suma, pensa-se que a abordagem a ser considerada durante o desenvolvimento do projeto seja um pouco uma mistura entre "Agile" e "Waterfall", ou seja, uma abordagem híbrida, seguindo uma ordem e um planeamento, mas estando aberta à inserção de novas funcionalidades e a possível necessidade de ter de retroceder nas fases do projeto para as implementar.

1.6 Estrutura do documento

Esta dissertação encontra-se dividida em seis capítulos, de forma a assegurar uma leitura clara e uma melhor compreensão do trabalho desenvolvido.

No primeiro capítulo é feita uma introdução ao tema e ao problema em estudo, apresentando-se o contexto da organização, os objetivos delineados, a motivação e relevância do projeto, bem como uma breve análise ética e o planeamento da investigação.

O segundo capítulo corresponde ao estado da arte, onde são introduzidos os conceitos fundamentais de monitorização de redes e analisadas três ferramentas *open source*. É ainda realizada uma comparação entre as mesmas, com o objetivo de selecionar a solução mais adequada às necessidades identificadas.

O terceiro capítulo apresenta a metodologia seguida, descrevendo a abordagem adotada para a implementação da solução, os requisitos técnicos e funcionais definidos, bem como as práticas de segurança a ter em consideração e a preparação da infraestrutura.

O quarto capítulo descreve a implementação prática da solução, abordando em detalhe o processo de instalação e configuração do Zabbix, a preparação do ambiente de monitorização, a definição de itens e *templates* personalizados, bem como a criação de *dashboards* específicos para a Arsopi.

O quinto capítulo dedica-se à avaliação da solução implementada, recorrendo a casos reais de utilização e exemplos de incidentes detetados, ao feedback da equipa de IT relativamente à eficácia da ferramenta e à identificação de possíveis melhorias futuras.

Por fim, o sexto capítulo apresenta a conclusão do trabalho, sintetizando os resultados alcançados, destacando os contributos obtidos e identificando perspetivas de evolução e continuidade da solução desenvolvida.

2 Estado da arte

Este capítulo é referente ao estado da arte, onde é feita uma introdução teórica sobre ferramentas de monitorização da rede, uma análise detalhada de três ferramentas diferentes disponíveis no mercado que possam cumprir com os objetivos propostos, uma enumeração de vantagens e desvantagens de cada uma e uma comparação entre as mesmas.

2.1 Monitorização da rede

A monitorização da rede é um processo essencial na gestão de infraestruturas de TI. Permite identificar problemas, otimizar o desempenho do sistema e garantir a segurança dos dados transmitidos entre dispositivos conectados. Esta secção irá abordar a definição, principais funcionalidades, componentes, protocolos e benefícios da monitorização da rede (Ogogo, 2021; VMWare, s.d.).

2.1.1 Definição

Uma ferramenta de monitorização da rede consiste num software ou conjunto de aplicações concebido para observar, analisar e gerir o desempenho, a disponibilidade e a integridade de uma rede de computadores. Este tipo de ferramentas recolhe informação em tempo real e de forma contínua sobre dispositivos, conexões, tráfego e eventos de rede, permitindo identificar problemas de desempenho, falhas ou ameaças à segurança. Tem como objetivo garantir que a rede funcione de uma forma eficiente e segura (Cisco, s.d.; VMWare, s.d.).

Para ser possível monitorizar uma rede de computadores, é necessário configurar previamente a aplicação de forma a especificar os dispositivos da rede que devem ser monitorizados e, para cada um destes dispositivos, as métricas que se pretendem monitorizar. Exemplos de métricas são a disponibilidade de um dispositivo ou serviço, através do envio de pacotes ICMP (*ping*), a percentagem de uso de CPU, memória ou disco, o tráfego de entrada e saída de rede, entre outros (Ogogo, 2021).

2.1.2 Principais funcionalidades

Existem diversas ferramentas de monitorização da rede no mercado, cada uma com as suas particularidades. No entanto, há funcionalidades consideradas essenciais, que estão presentes na maioria dessas soluções.

Estas ferramentas devem ser capazes de monitorizar uma ampla variedade de dispositivos de rede, utilizando diferentes métricas para avaliar o seu desempenho. Com a evolução da tecnologia, tornou-se possível acompanhar várias métricas em dispositivos diversificados. Quanto maior for a variedade de métricas disponíveis, maior será a capacidade da solução de atender a diferentes necessidades.

Para cada métrica e dispositivo, podem ser definidos limites que determinam se os valores estão dentro de um intervalo aceitável. Quando esses limites são ultrapassados, é possível configurar ações automáticas que são iniciadas imediatamente (dos Santos Souza Silva, et al., 2024).

Uma funcionalidade indispensável é a capacidade de notificar o administrador ou a equipa de TI sempre que uma métrica exceder os limites definidos. Essas notificações, enviadas por email ou exibidas na interface web, ajudam a reduzir o tempo de resposta para resolver e prevenir incidentes. Além disso, a apresentação dos dados monitorizados de forma visual e intuitiva, como por exemplo, através do uso de *dashboards*, é uma funcionalidade valiosa. Essa abordagem facilita a análise da informação e permite que o administrador aceda rapidamente aos detalhes do dispositivo afetado, possibilitando uma intervenção ágil e eficaz, minimizando os impactos do incidente (Phiri & Mfupa, 2016).

2.1.3 Componentes e protocolos

As ferramentas de monitorização da rede são compostas por diversos elementos que trabalham de forma integrada para recolher, processar e apresentar dados sobre o estado e o desempenho da rede. Para que isso seja possível, têm de recorrer a protocolos de comunicação para interagir com dispositivos e sistemas de rede.

Tipicamente, um sistema de monitorização da rede é composto por um servidor de monitorização, onde é instalada a tecnologia a ser utilizada para monitorizar as máquinas-alvo,

e as máquinas a serem monitorizadas, que podem ou não precisar de instalar um agente para recolher as informações da própria máquina.

Após a preparação do servidor e instalação do sistema no mesmo, é necessário acrescentar os clientes, tanto com a instalação do agente nas máquinas-alvo como através de protocolos de comunicação, como ICMP, SNMP, entre outros (Alia Abdi & Boubeghel, 2024; Koskinen, 2024).

Depois disto, é necessário definir as métricas e a frequência com que estes valores são extraídos da máquina-alvo.

Finalmente, depois de tudo configurado, o processo de recolha da informação é repetitivo e contínuo, ou seja, o servidor de monitorização recolhe a informação das máquinas que estão a ser monitorizadas, com a frequência configurada anteriormente, tanto através de protocolos como através do uso do respetivo agente, armazena todos os dados na base de dados configurada na instalação da ferramenta e, por fim, mostra estes dados através da sua interface *web*, seja através de gráficos, relatórios ou *dashboards*, dependendo daquilo que é configurado e desejado pelo cliente e do que a ferramenta permite (Alia Abdi & Boubeghel, 2024; Koskinen, 2024).

Deve-se notar que nem sempre o processo de obtenção de dados dos dispositivos monitorizados é exatamente desta forma, tal como poderemos comprovar com a análise de uma das ferramentas mais à frente.

Para que todo este processo seja possível, existem os protocolos de comunicação que são o elo entre o servidor de monitorização e os vários dispositivos de rede a serem monitorizados.

Alguns desses protocolos são o SNMP, ICMP e WMI.

SNMP

O SNMP (*Simple Network Management Protocol*) é um protocolo que atua na camada de aplicação e que usa um sistema de pedido e resposta para verificar o estado de diversos tipos de dispositivos, tais como *switches*. Este protocolo pode ser usado para monitorizar e configurar dispositivos de rede remotamente (Case, et al., 1990; Fortra, 2025; Harrington, et al., 2002).

Este protocolo foi evoluindo ao longo do tempo, pelo que existem neste momento 3 versões diferentes (Fortinet, s.d.):

- SNMPv1: primeira versão, mais simples e de fácil implementação, mas com fragilidades a nível de segurança, uma vez que recorre apenas a *community strings* em texto simples para autenticação.
- SNMPv2: segunda versão que introduziu melhorias no que toca à eficiência da comunicação e novas operações de gestão, mas manteve as mesmas limitações no que diz respeito à segurança do protocolo.
- SNMPv3: versão mais recente que acrescenta mecanismos de autenticação, encriptação e controlo de acesso, garantindo maior confidencialidade e integridade de comunicações.

ICMP

O ICMP (*Internet Control Message Protocol*) é um protocolo que atua na camada de rede e que é usado em conjunto com o protocolo IP para fornecer informações de conectividade e acessibilidade de dispositivos de rede. Este protocolo foi projetado para fornecer mensagens de controlo e diagnóstico e é amplamente utilizado em ferramentas como *ping* e *traceroute* (Cloudflare, s.d.).

Por sua vez, o IP (Internet Protocol) é o principal protocolo da camada de rede do modelo TCP/IP e é responsável por garantir a comunicação entre dispositivos numa rede. Este é o protocolo que define as regras para o endereçamento e roteamento dos pacotes de dados entre dispositivos (Cloudflare, s.d.).

WMI

O WMI (*Windows Management Instrumentation*) é uma tecnologia desenvolvida pela Microsoft usada para monitorizar e controlar dispositivos e aplicações em sistemas baseados no Windows através de consultas baseadas em WQL (*WMI Query Language*). Esta tecnologia, que pode ser usada por ferramentas de monitorização da rede, faz uso do protocolo DCOM (*Distributed Component Object Model*) de forma a permitir a comunicação entre o servidor e um cliente (Microsoft, 2021).

O protocolo DCOM é um protocolo da Microsoft que permite a comunicação entre componentes de software de diferentes computadores de uma rede. Este é uma extensão do COM (*Component Object Model*) que funciona apenas localmente, com a diferença de permitir a expansão da sua funcionalidade para sistemas distribuídos em redes locais (LANs) ou até em redes maiores. A comunicação do DCOM é feita via RPC (*Remote Procedure Call*), isto é, o cliente faz uma solicitação para aceder a um objeto num servidor remoto e o RPC é utilizado para enviar essa solicitação pela rede. O servidor executa a ação solicitada e retorna o resultado para o cliente. O RPC é um protocolo de comunicação que permite a programas executarem funções ou procedimentos num computador remoto, tal como se fossem executados localmente (BasuMallick, 2023; IBM, 2025).

O WMI pode ser muito útil para ferramentas de monitorização da rede uma vez que permite obter os valores de diversas métricas, incluindo os indicadores de desempenho de uma máquina, permite consultar informações acerca dos processos ativos e dos serviços em execução, permite obter informações acerca dispositivos físicos como discos rígidos, permite a consulta de registos e eventos do "Event Viewer", entre outros.

Apesar de ser uma tecnologia a ter em conta para ser utilizada juntamente com ferramentas de monitorização da rede, para que a mesma funcione, o utilizador com o qual se pretende executar as consultas WMI deve ter certas permissões na rede para o conseguir fazer. Se o utilizador for administrador da rede, conseguirá fazer as pesquisas, mas por questões de segurança, pode ser mais indicado criar um utilizador próprio apenas para a execução destas consultas. Para que isto funcione, devem ser concedidas algumas permissões a este utilizador, como por exemplo, permissões relacionadas com o DCOM (serverfault, 2024).

2.1.4 Benefícios

O uso de ferramentas de monitorização da rede é cada vez mais fundamental para uma boa gestão de uma rede de computadores. Vantagens como o facto de estas permitirem uma deteção proativa de incidentes, tais como falhas ou ameaças à segurança, reduzindo o tempo de resposta a este tipo de ocorrências, permitirem um maior controlo e otimização dos recursos de dispositivos como servidores e contribuírem para a melhoria da segurança do sistema informático são uma mais-valia no que toca à gestão de uma rede de computadores (Ogogo, 2021).

2.2 Ferramentas de monitorização da rede

Atualmente, existem várias ferramentas de monitorização da rede disponíveis no mercado que podem ser importantes na gestão de um sistema informático. Nesta secção será feita uma análise detalhada a três ferramentas. Estas três ferramentas correspondem às opções que foram dadas pelo supervisor da organização.

2.2.1 Zabbix

O Zabbix é uma ferramenta de monitorização da rede *open source* criada por Alexei Vladishev no início dos anos 2000. A sua primeira versão estável foi lançada em 2001 e em 2005 foi fundada a Zabbix *SIA Company*, empresa responsável pelo suporte técnico e comercial do produto. A empresa está sediada na Letónia e, atualmente, conta com mais quatro escritórios, um no Japão, um nos EUA, um no Brasil e outro no México (Zabbix, s.d.).

Funcionalidades

O Zabbix apresenta uma grande variedade de funcionalidades que podem ser úteis para uma gestão fácil e intuitiva de uma rede. Permite a monitorização de vários tipos de dispositivos de rede, tais como computadores, servidores, *routers*, *switches* e *access points*. Para além disso, permite ainda a monitorização de serviços, aplicações, *websites*, bases de dados, máquinas virtuais, entre outros.

Conta também com diferentes formas de coleta de dados, tais como SNMP, ICMP, WMI, SSH, Telnet, JMX, HTTP, entre outras. Permite ainda a coleta de dados através da execução de *scripts*.

Depois de configurar máquinas para serem monitorizadas com as devidas métricas e de serem definidos os limites de valores, é possível classificar os problemas associados a essas métricas em cinco níveis diferentes e, posteriormente, é possível configurar o sistema de alerta para enviar uma notificação via email para o endereço desejado e de acordo com a severidade do problema. Também é possível configurar os alertas para serem enviados por SMS ou através de um *webhook.*

O Zabbix tem disponível na sua interface web um *dashboard* pré-configurado para a visualização de alguns dados, *dashboard* esse que pode ser alterado. Caso seja mais adequado, é possível criar *dashboards* do zero, assim como mapas de rede e gráficos. Para a criação dos

dashboards, o Zabbix oferece uma grande variedade de *widgets* que podem ser editados e adicionados de acordo com os dados coletados a partir das máquinas que foram previamente preparadas para serem monitorizadas.

Esta plataforma conta ainda com alguns processos de automatização, tais como a descoberta automática de dispositivos e serviços na rede. A sua interface é facilmente configurável e, pelo facto de ser uma ferramenta de código aberto, permite a sua alteração total, desde que haja conhecimento técnico para o fazer (Zabbix, s.d.).

Arquitetura

O Zabbix está dividido em vários componentes que trabalham juntos em prol de fornecer as informações de que o utilizador final necessita para poder monitorizar a sua rede: servidor Zabbix, clientes Zabbix, interface *web* e base de dados (Zabbix, s.d.).

O servidor Zabbix consiste no serviço que processa todos os dados coletados por outros componentes e faz a gestão dos mesmos. Os clientes Zabbix são os dispositivos que estão a ser monitorizados, que podem ou não conter o agente Zabbix, que consiste num agente que é instalado nesses mesmos dispositivos de forma a facilitar a coleta de dados. Deve-se notar que o agente Zabbix consiste num serviço que fica a correr nos dispositivos clientes e, dessa forma, consome recursos extras da máquina, podendo ser um obstáculo à sua utilização. A base de dados é usada para armazenar todos os dados históricos e de configuração do Zabbix. Por último, a interface *web* consiste no *website* que permite a configuração e a visualização dos dados resultantes da coleta de dados dos dispositivos monitorizados. Aqui é possível fazer toda a configuração do Zabbix, desde a parte da sua instalação até toda a configuração após a instalação. É também a partir deste *website* que se pode ter acesso aos *dashboards* e gráficos com os dados inerentes à monitorização dos vários dispositivos (dos Santos Souza Silva, et al., 2024).

Como já foi referido anteriormente, a coleta de dados dos dispositivos pode ser feita recorrendo ao agente Zabbix, ou pode ser feita recorrendo a outros métodos de coleta de dados já aqui mencionados, sendo assim considerada uma monitorização *agentless* (sem agente).

Instalação

O Zabbix tem um processo de instalação relativamente fácil e bem documentado. Da mesma forma, os requisitos de hardware e software para a instalação e a operação estão bem explicados na documentação oficial do site. Considera-se todo o processo de instalação fácil (Zabbix, s.d.).

Compatibilidade

Em relação à compatibilidade do Zabbix, o seu servidor é suportado apenas em sistemas operativos Linux, já os agentes oferecem mais compatibilidade (Windows, MacOS, Unix) (Zabbix, s.d.).

Vantagens

Para além de algumas vantagens já mencionadas ao longo desta análise, tais como o facto de ser uma ferramenta de código aberto, ter uma grande quantidade de funcionalidades, ser compatível com diferentes ambientes, ter uma instalação fácil, acesso rápido a documentação de ajuda e permitir o uso da aplicação no modo *agentless*, existem outras que também devem ser tidas em conta. Entre as quais estão a sua escalabilidade, o facto de contar com uma interface intuitiva e o facto de ter uma grande comunidade ativa que pode ser uma grande ajuda na hora de resolver problemas ou esclarecer algumas dúvidas (Guo, et al., 2024).

Desvantagens

Em relação a desvantagens do Zabbix, a sua curva de aprendizagem, uma vez que inicialmente pode ser difícil e trabalhoso perceber e configurar o mesmo, apesar de toda a documentação e ajuda disponível à distância de uma pesquisa. O consumo de recursos também pode ser um ponto contra quando se trata de grandes ambientes. Por fim, a possível necessidade de personalização através de alterações no código, caso se tenha a intenção de alterar algo que pode não ser diretamente alterável na interface *web* (Lavagnoli & Gonçalves, 2024).

2.2.2 Checkmk

O Checkmk é uma ferramenta de monitorização da rede que tem 4 versões diferentes: *Raw*, *Enterprise*, *Cloud* e *MSP*. Apenas uma destas 4 versões, neste caso a *Raw,* é que é totalmente

gratuita e *open source*, logo, tendo em conta os objetivos desta dissertação, será esta versão o foco da análise da ferramenta (Checkmk, s.d.).

Esta solução foi desenvolvida pela Tribe29 e atualmente é suportada por uma comunidade ativa que colabora na sua manutenção e melhoria. Tem a sua sede na Alemanha e outro escritório localizado nos EUA (Checkmk, s.d.).

Funcionalidades

O Checkmk, de forma semelhante ao Zabbix, tem capacidade para monitorizar vários tipos de dispositivos, desde computadores e servidores até dispositivos de rede como *switches* e *access points.* Permite também a monitorização de serviços, aplicações, *websites*, dispositivos de armazenamento, entre outros (Koskinen, 2024; Checkmk, 2025).

Em relação aos métodos de coleta de dados, o Checkmk permite obter informação dos seus clientes através do uso do seu agente próprio e via SNMP. Outros métodos de coleta de dados como ICMP ou WMI não estão diretamente disponíveis para esta versão do produto, apesar de ser possível contornar este obstáculo por meio de *scripts*.

De forma a informar o administrador da rede acerca de um problema, o Checkmk também tem a possibilidade de gerar alertas assim que um valor relacionado com uma métrica de um dispositivo a ser monitorizado esteja fora da sua normalidade. Estes alertas podem ser enviados via email, SMS ou através de softwares de terceiros.

O Checkmk tem na sua interface web a possibilidade de visualizar os dados oriundos das métricas definidas para os vários dispositivos a serem monitorizados em *dashboards* simples e intuitivos. Permite também a alteração desses *dashboards* e a criação de novos. Oferece ainda a possibilidade de visualizar os dados em forma de gráficos.

Permite também a coleta de registos de eventos e a visualização dos mesmos na interface *web* da aplicação, de forma direta e fácil.

Por fim, o Checkmk também tem a possibilidade de implementar alguma automação no seu processo, por exemplo, através da descoberta automática de serviços e métricas disponíveis em *hosts* adicionados ao sistema.

Arquitetura

A arquitetura do Checkmk é muito semelhante à arquitetura do Zabbix. Contém um servidor, que faz o processamento dos dados, os clientes que são os dispositivos ou serviços que estão a ser monitorizados pelo servidor, que podem ou não se conectar ao servidor através do uso do agente próprio da solução, uma base de dados para armazenar toda a informação vinda da coleta de dados dos dispositivos monitorizados e, por fim, uma interface web onde é possível fazer toda a configuração e visualizar os dados (Koskinen, 2024; Checkmk, 2023).

O agente próprio do Checkmk é um serviço que, quando ativo, acarreta gastos extra a nível de recursos. No entanto, o Checkmk permite o modo de operação *agentless*, através do uso de outros protocolos para possibilitar a comunicação entre o servidor e o cliente, tais como SNMP.

Instalação

O Checkmk, tal como o Zabbix, tem um processo de instalação relativamente fácil, uma vez que tem todo o processo documentado no seu site oficial, assim como os requisitos de hardware físico e software necessários para a instalação do mesmo (Checkmk, 2023).

Compatibilidade

O Checkmk, de forma semelhante ao Zabbix, tem compatibilidade para a instalação do seu servidor apenas em Linux. No entanto, o seu agente pode ser instalado noutros sistemas operativos, como Windows, MacOS e Unix. (Checkmk, 2025).

Vantagens

O Checkmk pode ser uma boa opção a ter em conta aquando da escolha de um software de monitorização da rede. O facto de ser uma ferramenta *open source,* ter uma interface intuitiva e de fácil uso, com funcionalidades úteis ao assunto em questão, ter uma alta compatibilidade com diferentes sistemas, ser facilmente escalável, permitir monitorização sem agente, e conter funcionalidades de autodescoberta de *host* e serviços na rede são pontos que podem pesar na escolha desta ferramenta (Koskinen, 2024).

Desvantagens

Por outro lado, o Checkmk também tem pontos menos bons que se devem ter em conta. Apresenta, de forma similar ao Zabbix, uma curva de aprendizagem inicial acentuada, apesar

de ter diversas funcionalidades, a sua versão *open source* acaba por ter algumas limitações quando comparada com as versões pagas e o facto de não ter protocolos como WMI e ICMP de forma nativa (Koskinen, 2024).

2.2.3 Nagios Core

O Nagios foi uma das primeiras ferramentas de monitorização de rede a aparecer no mercado, tendo sido lançado pela primeira vez em 1999, com o nome NetSaint, desenvolvido por Ethan Galstad (Nagios, s.d.).

Algumas ferramentas de monitorização, como é o caso do Checkmk, foram baseadas no Nagios Core. O Nagios acabou por perder alguma força no mercado para outras ferramentas capazes de oferecer mais do que o Nagios podia oferecer, como por exemplo, uma interface *web* moderna e a possibilidade de fazer a configuração de uma forma automática, ao invés do uso de configuração manual.

O Nagios, à semelhança do Checkmk, tem uma versão gratuita e *open source*, que é o Nagios Core, e outras versões pagas. O estudo será feito com foco na versão gratuita (Nagios, s.d.).

Funcionalidades

Tal como as ferramentas anteriormente apresentadas, o Nagios Core também oferece a possibilidade de monitorizar uma grande variedade de dispositivos de rede e serviços (Nagios, s.d.).

Permite a recolha de dados tanto através do seu agente, o NRPE (*Nagios Remote Plugin Executor*), como através de outros protocolos, como é o caso de SNMP, ICMP, HTTP e WMI, entre outros, sendo que para fazer a monitorização com estes protocolos tem de recorrer a plugins.

Da mesma forma que as ferramentas anteriores, permite o envio de notificações pelas mais diversas vias possíveis quando algum dispositivo falha ou apresenta valores fora dos seus parâmetros normais. Mais uma vez, este envio de notificações depende de plugins.

Apresenta uma interface *web* que permite a visualização de dados, apesar desta ser bastante básica. A sua interface não tem capacidade para mostrar as informações em forma de gráficos nem *dashboards*, nem para criar este tipo de forma de visualização de dados. Para se poder

contornar esta situação, que é possível, é necessária a instalação de plugins que permitem a amostra de gráficos, apesar de serem gráficos básicos e simples. É necessário atuar da mesma forma para a apresentação de *dashboards*.

Pelo facto de ser um sistema baseado em *plugins*, este apresenta uma grande flexibilidade de personalização. É possível adicionar plugins da comunidade ou criar *plugins* do zero. Da mesma forma, é possível alterar os arquivos de configuração do Nagios Core, logo é possível personalizar praticamente tudo.

Arquitetura

A arquitetura do Nagios Core difere um pouco da arquitetura das ferramentas apresentadas até então. O Nagios Core segue uma arquitetura modular.

Esta é composta pelo *Core Engine*, que é o seu componente central, os seus *plugins*, os seus arquivos de configuração e por uma interface *web*.

O *Core Engine*, juntamente com o *Daemon Nagios,* que é o seu processo principal, é o responsável pela gestão de todo o processo de monitorização, incluindo a coleta de dados e as notificações de alertas.

Os *plugins* são os responsáveis por fazer a recolha de informação nos dispositivos e serviços que estão a ser monitorizados. É através destes que se obtêm os valores das métricas que posteriormente são usadas pela interface *web* para mostrar ao utilizador final.

Os arquivos de configuração são responsáveis por toda a configuração do sistema, incluindo as definições globais do Nagios. É possível alterar todas as configurações do sistema através da alteração destes arquivos de configuração.

Por fim, a interface *web* acaba por ser semelhante às interfaces *web* das outras ferramentas, apesar de ser um pouco mais simples e básica.

De notar que o Nagios Core não tem uma base de dados para armazenar os dados coletados a partir da monitorização dos vários dispositivos. Em vez disso, os dados são armazenados em ficheiros de texto, ficheiros esses que podem ser consultados pela interface *web* para apresentar os dados (Johnson & Elizabeth, 2017).

Instalação

O processo de instalação do Nagios Core, apesar de estar bem documentado, acaba por ser ligeiramente mais complicado e complexo comparado com as outras ferramentas, o que seria de esperar tendo em conta que todo o processo de instalação e configuração é feito manualmente. Em relação a informações acerca dos requisitos de hardware mínimos, não existe grande informação na documentação, mas é possível encontrar algumas referências relativamente a esse assunto no fórum do seu site (Nagios, s.d.).

Compatibilidade

O Nagios Core apenas pode ser instalado em servidores Linux/Unix. Apesar de ser possível monitorizar máquinas com outros sistemas operativos, a sua instalação não pode ser feita noutros sistemas operativos, como Windows e MacOS (Nagios, s.d.).

Vantagens

O Nagios Core conta com algumas vantagens que podem ser importantes no momento da escolha de uma ferramenta de monitorização da rede. O facto de ser uma ferramenta gratuita e *open source*, apresentar uma grande flexibilidade para criar *scripts* e ter a possibilidade de integrar qualquer serviço ou dispositivo, ser uma ferramenta leve e eficiente, ideal para infraestruturas pequenas e médias e, por fim, o facto de conter inúmeros documentos que permitem personalizar a solução, como é o caso de plugins, disponíveis em sites de suporte à ferramenta, criados pela comunidade (Canzari, et al., 2019; Alia Abdi & Boubeghel, 2024).

Desvantagens

Por outro lado, existem outros pontos menos bons que devem ser levados em consideração, tais como o facto da sua interface *web* ser bastante simples e não permitir a visualização dos dados de forma nativa através de *dashboards* e gráficos e a sua configuração ser completamente manual, a partir dos arquivos de configuração. Para além disso, a ferramenta não é facilmente escalável, uma vez que pode encontrar problemas de desempenho para ambientes maiores. O facto de ser uma ferramenta cujo uso tem decrescido nos últimos anos faz com que a documentação e o suporte não sejam tão prestáveis quanto antes. Por fim, o facto de ter uma versão paga pode ser uma limitação, uma vez que existem funcionalidades que apenas estão disponíveis nas versões pagas do Nagios, logo envolvem custos adicionais (Alia Abdi & Boubeghel, 2024).

2.3 Comparação das ferramentas apresentadas

Após a análise de cada uma das três ferramentas, é possível perceber que, apesar de todas cumprirem com a maioria dos requisitos necessários numa ferramenta de monitorização da rede, estas apresentam diferenças que podem ser importantes na hora de escolher uma para aplicar a um determinado contexto (Alia Abdi & Boubeghel, 2024; Koskinen, 2024; Lavagnoli & Gonçalves, 2024).

Tendo em conta os objetivos propostos para esta ferramenta, será agora feita uma comparação para perceber quais desses objetivos são cumpridos pelas três ferramentas apresentadas e para esclarecer as principais diferenças entre elas.

Todas as ferramentas são *open source* e todas elas não têm quaisquer custos associados na versão em que foram apresentadas. Pelo facto de serem plataformas de código aberto, todas permitem a sua personalização caso seja necessário. Apesar disto, o Nagios Core poderá ser a ferramenta mais facilmente personalizável, visto que toda a sua configuração é feita manualmente, nos arquivos de configuração (Johnson & Elizabeth, 2017; Kumar, 2024).

O funcionamento de todas as ferramentas origina um ligeiro aumento no que toca ao consumo de recursos, tanto usando o agente próprio de cada uma das ferramentas, como utilizando outros métodos de comunicação. Por norma, utilizar outros métodos de coleta de dados não é tão pesado e todas as ferramentas aqui apresentadas permitem operar sem agente, através de outros protocolos, como é o caso de SNMP, WMI, ICMP. Neste aspeto, a única ferramenta que tem todos estes protocolos de forma nativa é o Zabbix, tendo o Checkmk apenas o SNMP e podendo ter outras via *scripts* e o Nagios ter todas apenas via *plugins* (dos Santos Souza Silva, et al., 2024).

As três ferramentas permitem fazer monitorização de métricas de desempenho, tais como percentagem de uso do processador, percentagem do uso de memória, percentagem de uso do(s) disco(s), tráfego de rede, entre outros. Para além disso, todas elas permitem monitorizar outras métricas através de diferentes métodos. Uma questão importante é que todas elas também permitem o uso de *scripts* ou *plugins* como forma de monitorização, aumentando assim o leque de métricas diferentes disponíveis. Todas elas permitem também a integração com APIs de softwares de terceiros. Para além disso, todas permitem fazer diferentes tipos de monitorização, isto é, monitorização de máquinas, servidores, *switches*, *access points*,

impressoras, *websites*, aplicações, serviços, entre outros (Johnson & Elizabeth, 2017; Kumar, 2025).

As três ferramentas aqui apresentadas permitem a definição de limites para as diferentes métricas e o envio de alertas para os administradores assim que um valor estiver fora desses limites. Permitem também o envio destes alertas por email, SMS e por *scripts* externos, para além de permitirem também receber a notificação na interface *web* (Canzari, et al., 2019; Gonçalves Coelho, et al., 2020).

No que toca à questão da apresentação de dados na interface web em *dashboards*, gráficos e outros tipos de formas de visualização mais fáceis e intuitivas, o Zabbix e o Checkmk permitem enquanto o Nagios Core não tem essa funcionalidade nativa, sendo necessária a implementação por meio de *plugins*. Da mesma forma, o Nagios Core tem uma interface mais simples enquanto o Zabbix e o Checkmk têm uma interface web mais completa e funcional (Guo, et al., 2024; Koskinen, 2024).

Em relação a outras questões fora dos objetivos, o Zabbix e o Checkmk têm um processo de instalação considerado mais fácil pelo facto de terem toda a informação necessária na sua documentação, incluindo a informação de requisitos de hardware, enquanto o Nagios Core não tem essa informação de forma tão acessível. Todas as ferramentas oferecem suporte através das suas comunidades ativas, mas deve-se realçar que o Zabbix tem uma comunidade mais forte, seguindo-se o Checkmk e, por fim, o Nagios Core, que apresenta a comunidade e suportes gratuitos mais fracos. O Zabbix pode conter uma curva de aprendizagem mais acentuada no início, assim como o Checkmk, em comparação com o Nagios Core, mas, por outro lado, estes apresentam uma maior variedade de personalização, incluindo formas de mostrar a informação e de configurar a interface de forma nativa (dos Santos Souza Silva, et al., 2024; Lavagnoli & Gonçalves, 2024). O Zabbix e o Checkmk têm uma arquitetura muito semelhante, ao passo que o Nagios Core tem uma arquitetura diferente e que é considerada um pouco mais complicada pelo facto de toda a sua configuração ter de ser feita manualmente. Por outro lado, o Nagios Core é a ferramenta das três mais flexível no que toca à personalização da mesma. (Alia Abdi & Boubeghel, 2024; Johnson & Elizabeth, 2017).

De seguida, é apresentada uma tabela com uma comparação resumida entre as três ferramentas, de forma a ser mais fácil perceber as principais diferenças entre as mesmas.

Tabela 1 - Comparação das ferramentas

	Zabbix	**Checkmk**	**Nagios Core**
Licença	*Open Source*	*Open Source*	*Open Source*
Custos	Gratuito	Gratuito	Gratuito
Versão Paga	Suporte	Funcionalidades e Suporte	Funcionalidades e suporte
Arquitetura	Servidor, cliente, BD e interface web	Servidor, cliente, BD e interface web	Servidor, interface web, plugins e ficheiros de configuração
Compatibilidade (S.O.)	Linux	Linux	Linux / Unix
Protocolos suportados	ICMP, SNMP, WMI, HTTP, etc	SNMP	ICMP, SNMP, WMI, HTTP, etc via
Monitorização *Agentless*	Sim	Sim	Sim
Curva de Aprendizagem	Difícil	Difícil	Média
Configuração / Personalização	Alta	Alta	Muito Alta
Configuração manual	Não	Não	Sim
Interface Gráfica	Moderna	Média	Simples
***Dashboards* e gráficos**	Sim	Sim	Apenas via plugins
Escalabilidade	Alta	Alta	Baixa
Usabilidade / Facilidade	Alta	Alta	Baixa
Base	N/A	Nagios Core	N/A
Instalação	Fácil	Fácil	Difícil
Variedade de *widgets* e personalização	Alta	Alta	Média

2.4 Decisão final

Após a análise detalhada das três ferramentas aqui apresentadas e de ter sido feita uma comparação entre as mesmas, tomou-se a decisão em conjunto com o supervisor da organização de se avançar por aquela que parece dar mais garantias tanto no imediato como futuramente por ser a que apresenta mais variedade no que toca a funcionalidades. Optou-se então pela escolha do Zabbix. Fatores como o facto de apresentar um conjunto de funcionalidades que responde aos principais requisitos, ter uma interface que permite a criação e edição de *dashboards* que facilitam a visualização dos dados, ter uma boa documentação online e uma comunidade vasta e bastante ativa pesaram na decisão, uma vez que, apesar desta ferramenta oferecer suporte por valores relativamente baixos, é do interesse da organização não ter qualquer custo com a implementação e manutenção da ferramenta de monitorização da rede.

3 Implementação da solução

Após a análise comparativa e a decisão de adotar o Zabbix como ferramenta de monitorização da rede, procede-se à implementação prática da solução no contexto da Arsopi. Este capítulo descreve de forma detalhada todas as etapas do processo, desde a metodologia de implementação, definição da arquitetura e preparação da máquina até à instalação, configuração e personalização da ferramenta.

O objetivo é apresentar de forma clara e sistemática as opções técnicas adotadas, as configurações realizadas e as razões subjacentes a cada decisão. Para além da instalação e parametrização do Zabbix, são também descritos os mecanismos de segurança implementados, a integração com outros sistemas e a criação de *dashboards* personalizados que permitem uma monitorização eficaz e adaptada às necessidades da organização.

3.1 Metodologia

Após a análise comparativa das diferentes ferramentas de monitorização da rede, apresentada no capítulo anterior, concluiu-se que o Zabbix é a solução que melhor responde aos requisitos identificados pela organização. A escolha baseou-se na diversidade de funcionalidades oferecidas, na flexibilidade de integração, na disponibilidade de documentação extensa e de uma comunidade ativa, bem como no facto de ser uma ferramenta *open source*, sem custos de licenciamento.

Este capítulo tem como objetivo apresentar a metodologia adotada para a implementação prática da solução, detalhando a forma como foi estruturado o processo de instalação, configuração e validação do Zabbix no contexto da infraestrutura da Arsopi.

A abordagem seguida pode ser dividida em várias fases:

1. **Definição da Arquitetura da Solução** – conceção da arquitetura lógica e física, escolha dos componentes necessários (servidor, base de dados, *web server*) e dimensionamento dos recursos de hardware.

2. **Configuração Inicial da Máquina** – preparação da máquina virtual que aloja a solução, aplicação de medidas de segurança (políticas de utilizadores, permissões, certificação digital, *backups*, entre outras).
3. **Instalação do Zabbix** – instalação do servidor, *frontend* e base de dados, seguida da configuração inicial e do acesso via interface web.
4. **Configuração da Ferramenta** – adaptação do Zabbix ao ambiente da organização através da criação de *templates*, *hosts* e métricas específicas, bem como integração com outros sistemas (APIs externas, dispositivos de rede).
5. **Gestão de Notificações e Alertas** – definição de *media types*, ações e regras de escalonamento, permitindo que os administradores recebam alertas em tempo real.
6. ***Dashboards* Personalizados** – criação de interfaces gráficas adaptadas às necessidades da equipa de IT, com foco em métricas críticas e visualização intuitiva.
7. **Avaliação da Solução** – análise dos resultados obtidos, apresentação de casos reais de utilização, recolha de feedback da equipa de IT e identificação de melhorias futuras.

Com esta organização, procura-se garantir que a descrição da implementação é apresentada de forma lógica e progressiva, permitindo ao leitor acompanhar todas as etapas do processo, desde a conceção até à validação final da solução.

Nos capítulos seguintes, será explorado em detalhe cada um destes pontos, começando pela definição da arquitetura da solução e pela configuração inicial da máquina que serve de base à implementação do Zabbix.

3.2 Arquitetura da solução

A definição da arquitetura da solução é um passo fundamental para garantir que a ferramenta implementada responde aos requisitos funcionais e não funcionais identificados, nomeadamente em termos de escalabilidade, segurança, fiabilidade e facilidade de manutenção. Nesta secção são apresentados os principais componentes do sistema, a arquitetura física adotada, as características técnicas da máquina virtual onde a solução foi instalada, as integrações externas realizadas e, por fim, o diagrama de rede que ilustra a forma como a solução se enquadra na infraestrutura da Arsopi.

3.2.1 Componentes do Zabbix

O Zabbix é composto por diversos módulos que trabalham em conjunto de forma integrada. No entanto, para esta implementação foram considerados apenas os três componentes indispensáveis:

- **Zabbix Server** – núcleo central da solução, responsável pelo processamento de dados provenientes dos dispositivos monitorizados e pela gestão das operações da plataforma.
- **Base de Dados** – armazena toda a informação histórica e de configuração. Para esta implementação foi escolhida a base de dados MySQL, atendendo à experiência prévia da equipa com esta tecnologia e à sua robustez.
- **Servidor Web (Apache)** – utilizado para disponibilizar a interface gráfica da solução, através da qual são feitas a configuração, gestão e visualização de métricas e *dashboards*.

Optou-se por não recorrer à instalação de agentes nas máquinas finais, uma vez que a infraestrutura monitorizada não é extensa e o Zabbix permite a monitorização *agentless*, recorrendo a protocolos como ICMP, SNMP e WMI. Esta decisão minimiza o impacto no desempenho das máquinas alvo e simplifica a gestão da solução.

3.2.2 Arquitetura física

A solução foi instalada numa máquina virtual com sistema operativo Ubuntu 24.04 (Noble). A escolha por um ambiente virtualizado justifica-se por diversos fatores:

- Facilidade de criação e replicação de máquinas virtuais pela equipa de IT.
- Flexibilidade e escalabilidade, permitindo aumentar os recursos alocados sempre que necessário.
- Alta disponibilidade e redundância, asseguradas pela infraestrutura de virtualização da empresa.

- Facilidade de acesso e configuração, dado que a gestão da máquina virtual pode ser feita remotamente.

Foi considerada suficiente a utilização de uma única máquina para alojar a solução, dado que o volume de métricas a monitorizar não é elevado a ponto de justificar a separação de componentes em diferentes servidores. Esta decisão é também suportada pela política de *backups* regulares, que garante a recuperação rápida em caso de incidente. Importa salientar que, embora a ferramenta de monitorização não seja considerada crítica para o funcionamento da empresa, é reconhecida a sua importância estratégica para a prevenção de falhas e incidentes de segurança.

3.2.3 Dimensionamento e características da máquina

De acordo com a documentação oficial do Zabbix, para uma instalação de pequena dimensão (até 1000 métricas), os requisitos mínimos incluem 2 CPU *cores* e 8 GB de memória (Zabbix, s.d.). Contudo, com base na experiência da equipa e tendo em conta margens de segurança, foram definidos os seguintes recursos para a máquina virtual:

- 4 CPUs
- 8 GB de memória RAM
- 150 GB de armazenamento em disco

O cálculo do espaço em disco considerou as estimativas apresentadas pelo Zabbix para diferentes cargas de trabalho, incluindo histórico de 30 dias, armazenamento de *trends* por 5 anos e até 1 evento por segundo. Embora este último valor seja exagerado para a realidade da empresa, foi considerada uma reserva de espaço superior ao mínimo, de forma a evitar limitações futuras.

3.2.4 Integração com APIs externas

Para complementar a monitorização nativa do Zabbix, foram implementadas integrações com APIs externas, permitindo a recolha de métricas adicionais relevantes para a segurança e desempenho da rede:

- API da Fortigate – utilizada para recolher informações relacionadas com as VPNs e sessões ativas, fornecendo maior visibilidade sobre acessos remotos e tráfego de rede.
- API da IPInfo – utilizada para mapear geograficamente os endereços IP dos utilizadores remotos, permitindo identificar de forma imediata a origem dos acessos.

Estas integrações ampliam o âmbito da monitorização e fornecem dados complementares à equipa de IT, contribuindo para uma gestão mais eficaz da infraestrutura.

3.2.5 Diagrama de rede

A figura 3 apresenta o diagrama de rede da solução implementada, ilustrando a integração do servidor Zabbix na infraestrutura Arsopi, os dispositivos monitorizados e os protocolos de comunicação utilizados.

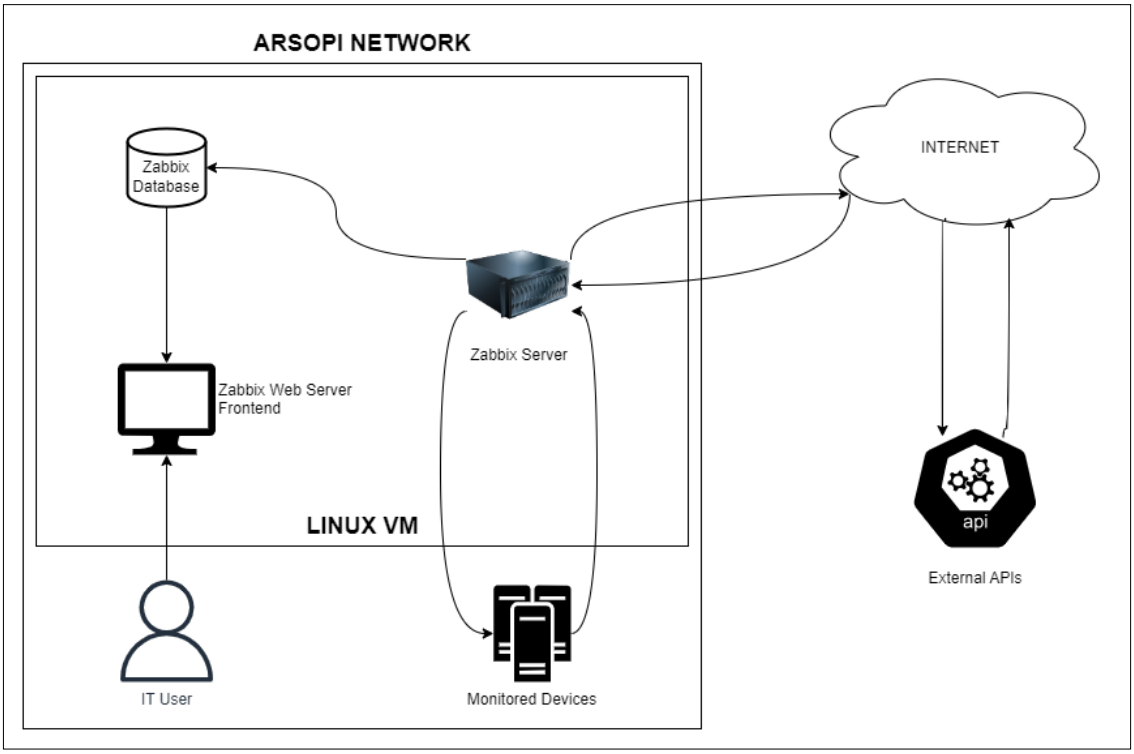


Figura 3 - Diagrama da arquitetura da solução de monitorização implementada

3.3 Configuração inicial do servidor

Após o provisionamento da máquina virtual com as características previamente descritas, foram aplicados procedimentos de segurança com o objetivo de garantir a confidencialidade, integridade e disponibilidade do sistema e, em particular, da rede corporativa.

Em primeiro lugar, foi criada uma conta de acesso dedicada, protegida por uma palavra-passe forte, assegurando a proteção contra acessos não autorizados. Adicionalmente, procedeu-se à execução das atualizações de segurança disponíveis no sistema operativo, de forma a mitigar vulnerabilidades conhecidas e reduzir a probabilidade de exploração.

Optou-se por não integrar esta máquina no domínio da empresa. Esta decisão visa impedir que eventuais vulnerabilidades do Zabbix, ou do próprio servidor, possam ser exploradas para comprometer o controlador de domínio ou outros sistemas críticos da infraestrutura, reduzindo assim a superfície de ataque e os impactos de um eventual comprometimento.

As atualizações automáticas foram desativadas, uma vez que podem originar indisponibilidades inesperadas ou incompatibilidades entre versões em sistemas críticos. Assim, a atualização do sistema passa a ser efetuada manualmente, garantindo maior controlo e previsibilidade no processo.

De seguida, procedeu-se à instalação dos componentes necessários para o funcionamento do Zabbix, nomeadamente MySQL, Apache, PHP e respetivas extensões. Foram utilizadas versões compatíveis entre si, assegurando a estabilidade e o correto funcionamento da solução.

Após a instalação do Apache, configurou-se o acesso seguro à interface *web* através da ativação do protocolo HTTPS. Para tal, foi integrado um certificado digital emitido pela autoridade certificadora interna da empresa, juntamente com a respetiva chave privada. Esta configuração garante a cifragem das comunicações entre os utilizadores e o servidor, prevenindo ataques de interceção (*man-in-the-middle*) e assegurando a confidencialidade e integridade dos dados trocados.

Finalmente, foi configurada a *firewall* de forma a bloquear acessos externos não autorizados ao servidor. Esta medida reforça a proteção contra potenciais atacantes e contribui para a manutenção de um ambiente seguro e controlado.

Para complementar as medidas de segurança aplicadas, foi ainda definida uma política de *backups* para o servidor que aloja o Zabbix. Esta é uma medida bastante importante para garantir a disponibilidade do serviço no caso de ocorrência de algum incidente, sendo, desta forma, facilmente reposto o servidor e voltando a estar ativo e a funcionar corretamente. A política de *backups* definida conta com um *backup* completo do servidor uma vez por semana, ao fim de semana e um *backup* incremental nos outros dias da semana, de 24h em 24h. As cópias de segurança são armazenadas num servidor de *backups* imutáveis. Para além disso, todas estas cópias são também guardadas num serviço *cloud* contratado, que também tem características de imutabilidade, garantindo assim uma cópia deslocalizada. Por fim, se se pretender obter informação mais recente de uma cópia, é possível repor informação ou o servidor completo dos *snapshots* feitos de hora em hora pela *storage* que armazena as cópias.

3.4 Instalação do Zabbix

De seguida, procedeu-se à instalação do Zabbix. O acesso à máquina foi sempre feito via SSH. Para isso, seguiu-se a documentação oficial do Zabbix, que tem um guia bem detalhado, passo a passo, com todo o processo de instalação (Zabbix, s.d.).

Começou-se por obter privilégios administrativos e, de seguida, instalar o repositório Zabbix:

```
# sudo -s
# wget
https://repo.zabbix.com/zabbix/7.2/release/ubuntu/pool/main/z/zabbix-
release/zabbix-release_latest_7.2+ubuntu24.04_all.deb
# dpkg -i zabbix-release_latest_7.2+ubuntu24.04_all.deb
# apt update
```

Código 1 - Obtenção de privilégios administrativos, adição do repositório oficial e atualização de pacotes

Logo a seguir procedeu-se à instalação do servidor Zabbix, do *frontend* e do agente:

```
# apt install zabbix-server-mysql zabbix-frontend-php zabbix-apache-conf
zabbix-sql-scripts zabbix-agent
```

Código 2 - Instalação dos pacotes principais do Zabbix

Após a instalação dos três componentes, passou-se para a configuração da base de dados. Para isso, executaram-se os comandos do código 3.

```
# mysql -uroot -p
password
mysql> create database zabbix character set utf8mb4 collate utf8mb4_bin;
mysql> create user zabbix@localhost identified by 'password';
mysql> grant all privileges on zabbix.* to zabbix@localhost;
mysql> set global log_bin_trust_function_creators = 1;
mysql> quit;
```

Código 3 - Criação da base de dados e utilizador Zabbix

A *password* definida não foi "password", mas sim uma palavra-passe que, como é de esperar, não será revelada por questões de segurança. Após a configuração inicial da base de dados, com a criação da base de dados para o Zabbix, a criação do utilizador "zabbix" e a concessão de permissões a este utilizador, passou-se para a importação do esquema inicial e respetivos dados:

```
# zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mysql --
default-character-set=utf8mb4 -uzabbix -p zabbix
```

Código 4 - Importação do esquema inicial da base de dados

Após a importação, voltou-se a desativar a opção "log_bin__trust_function_creators", por questões de segurança:

```
# mysql -uroot -p
password
mysql> set global log_bin_trust_function_creators = 0;
mysql> quit;
```

Código 5 - Reversão da configuração de segurança no MySQL

De seguida, procedeu-se a fornecer ao servidor do Zabbix a informação que necessita para fazer uso da base de dados, ou seja, a *password*. Esta informação foi alterada no ficheiro /etc/zabbix/zabbix_server.conf, alterando a "DBPassword" para a *password* correta.

Por fim, reiniciou-se os serviços do servidor Zabbix, do agente e do apache e ativou-se os mesmos:

```
# systemctl restart zabbix-server zabbix-agent apache2
# systemctl enable zabbix-server zabbix-agent apache2
```

Código 6 - Reinício e ativação dos serviços principais

Com os serviços ativos, o próximo passo consistiu em aceder à interface *web* do Zabbix, através do endereço IP ou do nome interno do servidor. Seguiram-se os passos do instalador web (Zabbix, s.d.):

1. Ecrã de boas-vindas.
2. Verificação de pré-requisitos.
3. Configuração da ligação à base de dados (utilizador “zabbix” e respetiva palavra-passe).
4. Definição do nome do servidor (“Arsopi”).
5. Resumo das opções escolhidas.
6. Confirmação da instalação concluída com sucesso.

Na primeira autenticação, utilizaram-se as credenciais padrão “Admin/zabbix”, que foram de imediato substituídas por credenciais personalizadas, garantindo maior segurança.

Após o login, é feito um encaminhamento para o *dashboard* principal que vem pré-definido, semelhante ao da figura 4.

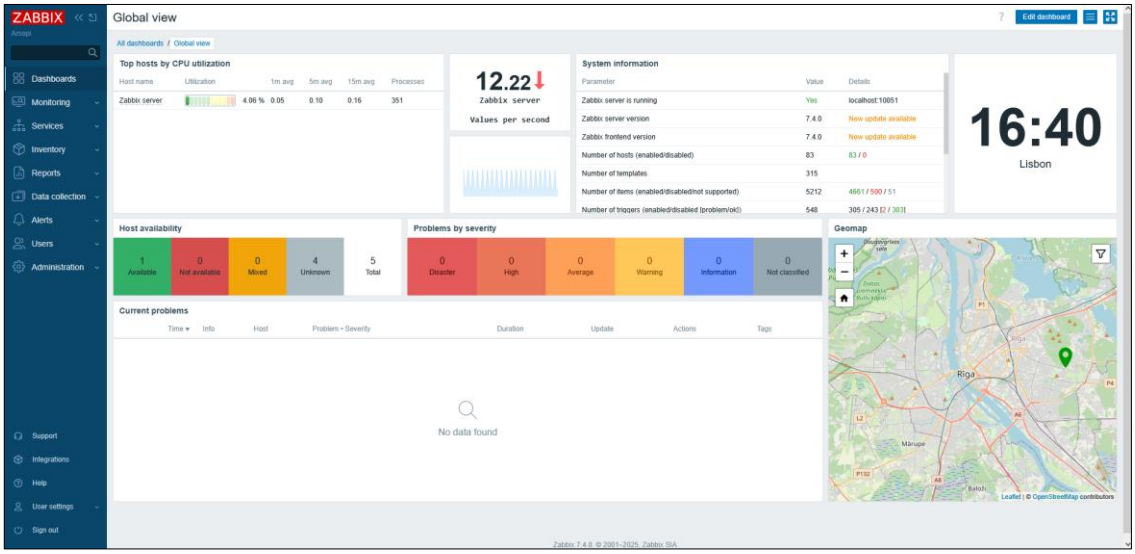


Figura 4 - *Dashboard* inicial

Procedeu-se, por fim, à alteração da *password* de administrador que vem como “zabbix” por defeito. Para isso, acedeu-se, no menu lateral, ao submenu “User settings” e acedeu-se à página “Profile”. Aqui, clicou-se na opção “Change password”, procedendo à alteração da mesma.

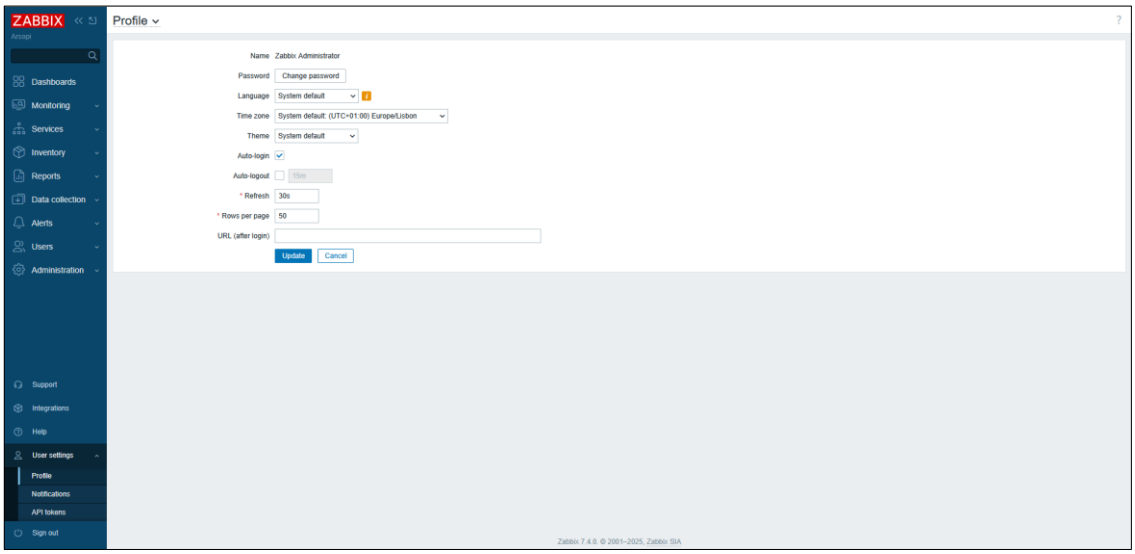


Figura 5 - Página para alteração da *password* do Zabbix *Administrator*

3.5 Preparação do ambiente para monitorização

Este ponto tem como objetivo detalhar alguns processos executados necessários para que a monitorização dos dispositivos fosse possível. De reparar que estes não estão documentados com o auxílio de imagens por questões de segurança.

3.5.1 Preparação para WMI

Para possibilitar a monitorização de servidores Windows através de Windows Management Instrumentation (WMI), foi necessário efetuar uma preparação prévia do ambiente, tanto ao nível do domínio como do servidor de monitorização.

Em primeiro lugar, por questões de segurança, foi criado um utilizador dedicado exclusivamente para operações WMI. Este utilizador não pertence ao grupo de administradores do domínio, mas foi-lhe atribuída a concessão mínima de permissões necessárias para aceder remotamente a métricas do sistema, tais como utilização de CPU, memória, discos e registos do "Event Viewer". O utilizador criado foi adicionado aos grupos "Utilizadores do monitor de desempenho" e "Utilizadores do registo de desempenho". As permissões tiveram de ser aplicadas de forma individual em cada servidor a monitorizar, garantindo o princípio do menor privilégio. Para isso, para cada servidor, foi necessário ir ao "wmimgmt.msc", ir a propriedades de WMI Control (Local), na aba "Security, abrir "Root", clicar em "CIMV2" e clicar de seguida em "Security". Adicionar o utilizador criado com as permissões "Enable Account" e "Remote Enable"; de seguida, executar "dcomcnfg", seguir pelo caminho "Component Services > Computers > My Computer" e nas propriedades, aceder a "COM Security" e fazer "Edit Limits" para ambos os casos que aparecem, dando permissões de "remote access", "remote launch" e "remote activation" para o utilizador (serverfault, 2024).

Adicionalmente, foi configurado no servidor de monitorização um ambiente Python virtualizado, onde foram instaladas bibliotecas e *scripts* específicos que permitem a execução de queries WMI a partir do Zabbix. Este ambiente, identificado no diretório /home/zabbix/myenv/, contém diversos utilitários, incluindo o *script* wmiquery.py, que possibilita a execução de consultas WMI diretamente a partir da linha de comandos.

3.5.2 Preparação para SNMP

Para permitir a monitorização de dispositivos de rede, como *switches*, *routers* e pontos de acesso, foi necessário preparar previamente o ambiente ao nível do Simple Network Management Protocol (SNMP).

Em primeiro lugar, foi ativado o protocolo nos dispositivos-alvo. Este passo é essencial, uma vez que muitos equipamentos de rede não têm o SNMP ativo por defeito. A configuração foi realizada com a versão mais recente suportada por cada equipamento, preferencialmente SNMPv3, uma vez que esta introduz mecanismos de autenticação e encriptação inexistentes nas versões anteriores (SNMPv1 e SNMPv2). Esta decisão garante maior proteção contra ataques de interseção ou exploração de tráfego SNMP.

Nos casos em que apenas era possível recorrer a SNMPv2c, definiu-se uma *community string* específica, que funciona como uma forma de autenticação simples, permitindo que apenas clientes que conheçam essa chave consigam aceder às métricas. Apesar de não oferecer encriptação, a utilização de uma *community string* forte e não predefinida contribui para mitigar riscos de acessos indevidos.

Desta forma, garantiu-se que os dispositivos de rede estavam corretamente preparados para serem integrados no sistema de monitorização, assegurando simultaneamente a compatibilidade funcional e o cumprimento das melhores práticas de segurança.

3.6 Configuração do Zabbix

Uma vez instalado, o Zabbix precisa de ser configurado de acordo com a nossa rede de forma a podermos tirar o máximo proveito da ferramenta. Para isso, foi usada a interface *web* oferecida pela solução.

Na interface *web*, depois de fazer *login*, temos acesso ao *dashboard* inicial que vem pré-configurado com a solução e aos menus laterais. A partir dos menus laterais, em “Data collection”, é possível ver e criar *hosts*, *host groups*, *templates*, *template groups*, entre outros.

De forma a ter uma boa organização da estrutura e ter acesso aos hosts de forma rápida e prática, optou-se pela criação de *templates* e de *host groups*.

3.6.1 Host Groups

Começando pelos *host groups*, já existem alguns por default que foram e estão em uso, como os "Problem Hosts", "Zabbix Servers" e "Windows Servers", sendo este último mais para o âmbito de testes. Procedeu-se então à criação de outros grupos, sendo eles os seguintes:

- Arsopi APs
- Arsopi Firewalls
- Arsopi nGs
- Arsopi Servers
- Arsopi Switches
- Arsopi VPNs

Estes grupos, tal como os seus nomes indicam, servem para diferenciar os vários *hosts* adicionados à ferramenta para serem monitorizados, podendo ser *access points*, *firewalls*, nGs (dispositivos táteis espalhados pela empresa que servem maioritariamente para os colaboradores do chão de fábrica marcarem os seus tempos em cada trabalho específico), servidores, *switches* e VPNs (túneis criados para acesso às outras empresas do grupo).

3.6.1.1 Processo de criação de um *host group*

A criação de um *host group* é bastante simples e, como já foi referenciado, é inevitável para uma boa organização dos *hosts*. O processo passa por, na aba lateral, ir a "Data collection", "Host groups" e, posteriormente, na página dos *host groups*, clicar no botão situado no canto superior direito denominado "Create host group". De seguida, basta dar um nome ao grupo e clicar em "Add" para adicionar o mesmo, ficando este pronto a ser selecionado na página dos *hosts*.

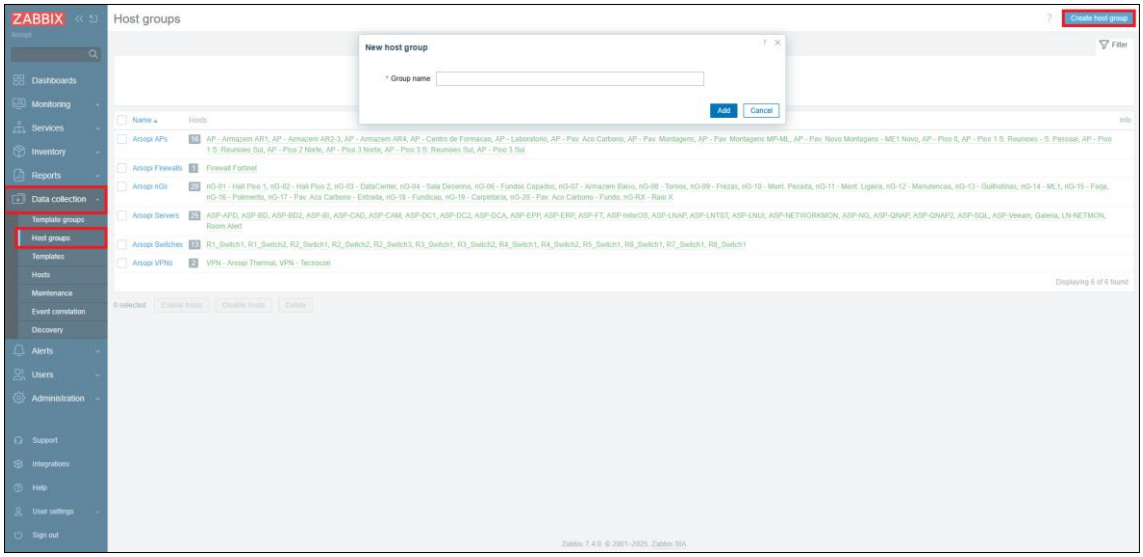


Figura 6 – Processo de criação de um *host group*

3.6.1.2 Processo de criação de um *host*

O processo de criação de um *host* segue uma lógica semelhante à descrita anteriormente. Para iniciar, é necessário aceder à página de "Hosts", disponível na aba lateral em "Data collection", e selecionar a opção "Hosts". Em seguida, procede-se à criação de um novo *host* através da opção "Create host", localizada no canto superior direito, sendo então solicitado o preenchimento da informação associada. Nesta fase, atribui-se um nome ao *host*, podendo ainda ser definido um "visible name" distinto, caso se justifique. É igualmente possível associar um ou mais *templates* e selecionar um ou mais *host groups*.

Relativamente às interfaces, devem ser especificados o tipo de interface e o endereço IP correspondente ao *host*. Adicionalmente, pode ser incluída uma descrição, configurado o modo de monitorização (*server*, *proxy* ou *proxy group*), bem como definida a opção de criação como ativo ou inativo. As restantes abas disponibilizadas apresentam configurações facultativas, cuja aplicação depende do contexto de utilização e das necessidades específicas de cada *host*. No âmbito do presente trabalho, na maioria dos casos foi adotado um processo simplificado de criação de *hosts*, em conformidade com o procedimento aqui descrito.

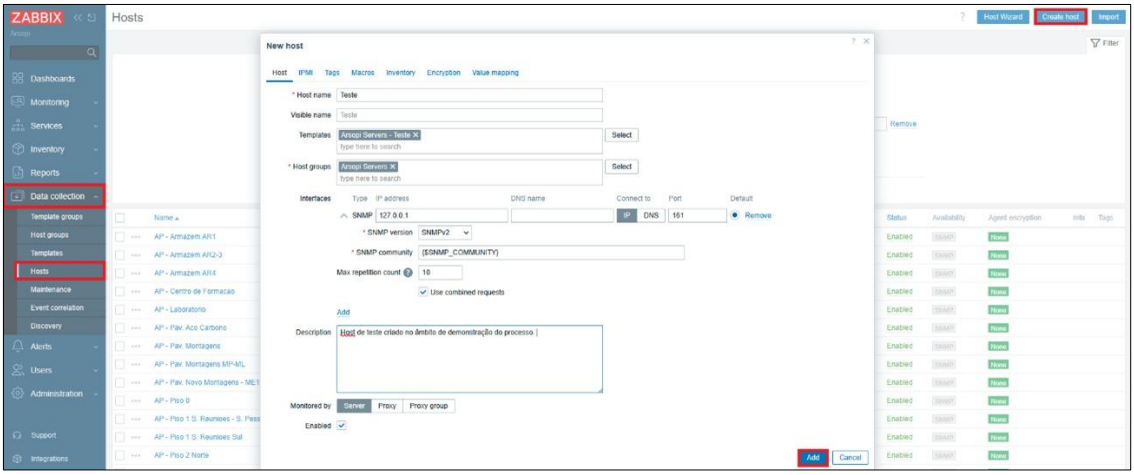


Figura 7 – Processo de criação de um *host*

3.6.2 Templates

Em relação a *templates*, a lógica foi semelhante, apesar de existirem menos *templates*. Os *templates* agregam os itens, *triggers* e gráficos que queremos monitorizar e, por esse motivo, foram criados apenas 3 *templates*, sendo eles:

- Arsopi Ping
- Arsopi Servers
- Arsopi Switches

Por existirem muitos dispositivos em que apenas interessa saber se estão ativos e em funcionamento, um simples "ping" já resolve a situação, pelo que este é o *template* com mais hosts associados. De frisar que um host pode ter mais que um *template* associado e, da mesma forma, pode ter itens, *triggers* ou gráficos associados para além daqueles configurados nos *templates*. Para além destes *templates* que foram criados, está em uso um outro *template* da FortiGate que já vem com a solução, denominado "FortiGate by SNMP". Este é utilizado para nos dar acesso a algumas métricas da nossa *firewall* FortiGate.

3.6.2.1 Processo de criação de um *template*

A criação de *templates* é, tal como os processos de criação anteriores, bastante simples, como se pode visualizar na Figura 8. Basta ir à página dos *templates* através do menu lateral do Zabbix, clicar em "Create template", dar um nome ao mesmo, associar um *template group* e clicar em "Add".

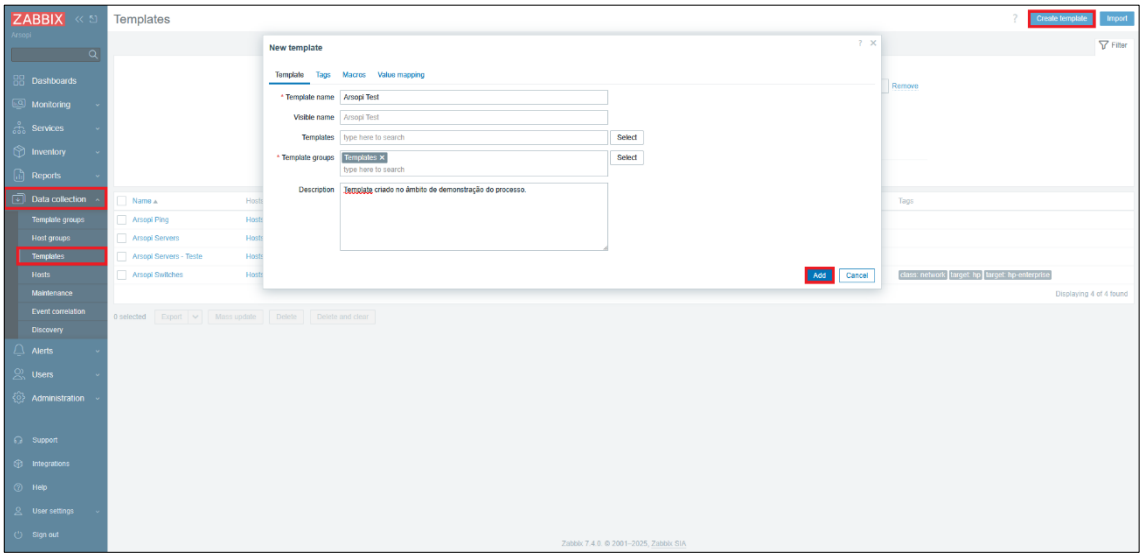


Figura 8 –Processo de criação de um *template*

3.6.2.2 Arsopi Ping

Tal como foi mencionado anteriormente, "Arsopi Ping" é o *template* mais utilizado. Contém apenas uma métrica, ou seja, um item, um *trigger* e um gráfico. Esta métrica, denominada "Ping", avalia se o dispositivo em causa responde a *pings* através de um "External Check" que no fundo é um *script* Bash.

O Zabbix oferece várias formas de obter esta métrica de forma nativa, isto é, sem ser preciso recorrer a um *script* externo, por exemplo, através de um item do tipo "Simple Check" com a key "icmpping". Este foi o cenário adotado numa fase inicial, com um intervalo de tempo reduzido a 10 segundos. O problema encontrado para esta forma de obter esta métrica foi a sobrecarga associada a uma avaliação tão frequente. Para melhorar este aspeto, adotou-se um

outro método, que consiste na chamada do *script* externo que obtém o "ping" do dispositivo de 1 em 1 minuto, caso o valor do "ping" seja 1 (ativo). Caso o valor do "ping" seja 0, o *script* espera 30s e volta a fazer o teste. Se continuar a 0, devolve 0 como valor e lança o *trigger*, caso contrário, devolve 1 e está tudo dentro do expectável. Esta medida foi adotada algum tempo depois de se optar por intervalos de tempo de 1 minuto, pela razão explicada anteriormente associada à sobrecarga e para eliminar falsos problemas que eram lançados por vezes. Isto porque, por vezes, pode haver falha de *ping* por 1 segundo e não quer dizer necessariamente que o dispositivo ou serviço esteja em baixo, e era o que acontecia muitas vezes. Desta nova forma, só devolve o valor a 0 e lança o *trigger* caso não haja resposta positiva ao *ping* num determinado momento e 30 s depois, reduzindo os falsos problemas.

Como já se percebeu, o *trigger* associado a esta métrica é lançado quando o valor do *ping* é igual a 0, o que quer dizer que o sistema está em baixo, originando um problema de severidade "Desastre".

```
#!/bin/bash

TARGET=$1
COUNT=4

if [ -z  "$TARGET" ]; then
    exit 1
fi

ping -c $COUNT $TARGET > /dev/null 2>&1

if [ $? -eq 0 ]; then
    echo 1
else
    sleep 30
    ping -c $COUNT $TARGET > /dev/null 2>&1
    if [ $? -eq 0 ]; then
        echo 1
    else
        echo 0
    fi
fi
```

Código 7 – *Script* "get_pings.sh"

A criação do *script* deve ser feita na pasta pré-definida do Zabbix para guardar *scripts* externos, sendo esta no caminho "/usr/lib/zabbix/externalscripts/". Desta forma, pode-se configurar a chave do item com o nome do script quando se trata de um item do tipo "External check". De

salientar que, como é expectável, é necessário também dar permissões de execução ao ficheiro para que este possa ser executado.

3.6.2.3 Arsopi Servers

Em relação ao *template* Arsopi Servers, este é composto por 6 itens, 4 *triggers* e 3 gráficos. Estas são as métricas comuns a todos os servidores que estão a ser monitorizados pela ferramenta, sendo que, posteriormente, cada servidor pode ter outros itens, *triggers* ou gráficos associados, dependendo do contexto.

3.6.2.3.1 Percentagem de Uso de CPU

Este é um dos itens do *template* e tem como objetivo obter a percentagem de utilização do CPU do servidor. Este item tem um intervalo de tempo de 5 minutos e é do tipo "External check", uma vez que consiste num *script* Bash que faz uso do *script* de Python de WMI para obter o valor periodicamente.

Para este item, o *script* vai buscar o valor através da função "get_cpu_usage()", fazendo a limpeza da *string* que é retornada.

- "tr -d '\n'" serve para remover as quebras de linha;
- "tr -cd '\11\12\15\40-\176'" serve para manter apenas caracteres ASCII imprimíveis e remover caracteres indesejados;
- "tr -d '|'" para remover os "|" da *string*;
- "sed 's/ */ /g'" para substituir múltiplos espaços por um espaço único;
- "tr ' ' '\n' para substituir espaços por quebras de linhas, fazendo com que cada palavra fique numa linha;
- "tail -n1" para obter a última linha, ou seja, a última palavra.

Deve-se notar que, como se pode verificar no código 8, as credenciais do utilizador com permissões para obter informação dos vários *hosts* via WMI foram substituídas por

“{UTILIZADOR}” e “{PASSWORD}”, por motivos óbvios de segurança. Da mesma forma, algumas informações em algumas figuras também foram omitidas.

```
#!/bin/bash

get_cpu_usage(){
    /home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
    {UTILIZADOR}:{PASSWORD}@$1 -query "SELECT PercentProcessorTime FROM
    Win32_PerfFormattedData_PerfOS_Processor WHERE Name = '_Total'" | tr
    -d '\n' | tr -cd '\11\12\15\40-\176' | tr -d '|' | sed 's/  */ /g' |
    tr ' ' '\n' | tail -n1
}

hora_atual=$(date +"%H%M")

if [ "$1" == '10.10.10.13' ]; then
    if [ "$hora_atual" -ge 0600 ] && [ "$hora_atual" -lt 0800 ]; then
        limite_cpu=74
    else
        limite_cpu=49
    fi
else
    if [ "$1" == '10.10.0.234' ]; then
        limite_cpu=2
    else
        limite_cpu=69
    fi
fi

cpu_usage=$(get_cpu_usage $1)

if [ "$cpu_usage" -gt "$limite_cpu" ]; then
    #echo "$cpu_usage"

    sleep 10

    cpu_usage=$(get_cpu_usage $1)

    if [ "$cpu_usage" -gt "$limite_cpu" ]; then
        echo "$1 --> $(date '+%Y-%m-%d %H:%M:%S')" >>
/usr/lib/zabbix/externalscripts/logs_cpu.txt
    fi

fi

echo "$cpu_usage"
```

Código 8 – Script “get_cpu_usage.sh”

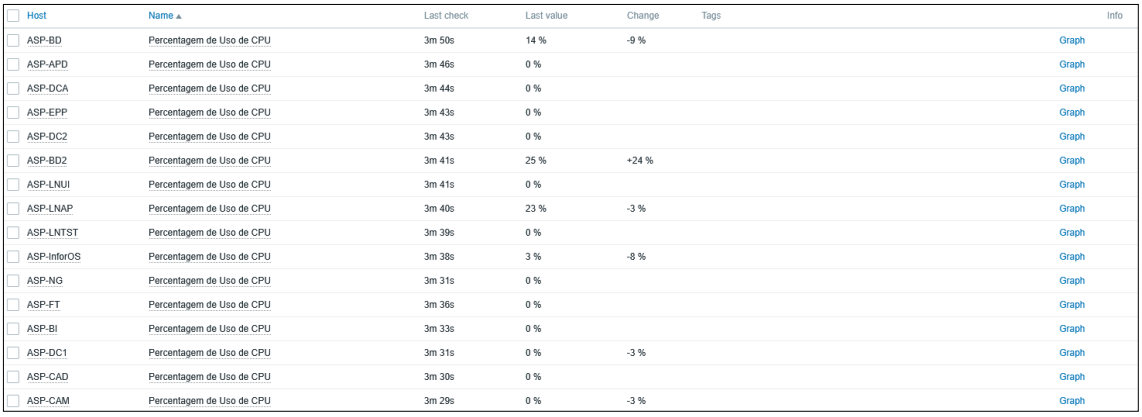
Com esta limpeza, a função retorna apenas um número entre 0 e 100 que corresponde à percentagem de utilização de CPU no momento.

O *script* começa por ir buscar a hora atual. De seguida, tem uma condição que verifica se se trata de um determinado servidor com um determinado endereço IP. Caso seja este servidor, e caso a hora esteja entre as 6h e as 08h, define o valor "74" para a variável de limite de CPU.

Caso esteja fora desse horário, define o valor "49". Caso se trate de outro servidor qualquer, assume-se o valor "69" para o limite de CPU. Isto acontece porque ao longo do tempo, antes desta condição ser feita, o servidor com aquele IP disparava o *trigger* frequentemente entre as 06h e as 08h por conta de uma carga de trabalho maior que apresentava nesta altura, carga essa que não apresenta em mais nenhuma altura do dia, daí o valor limite para o que resta do dia ser 49. Adotou-se esta medida para aumentar a "veracidade" e "realismo" dos problemas gerados pelos alertas do Zabbix, uma vez que este na realidade não era um problema, mas sim um comportamento expectável.

De seguida vai buscar o valor associado à percentagem de utilização de CPU e caso este esteja abaixo do limite, devolve o mesmo ao servidor Zabbix. Caso contrário, volta a repetir o "teste" 10 segundos depois e devolve este novo valor. O servidor Zabbix, depois de receber o valor, dispara ou não o *trigger*, dependendo do valor de CPU e do valor limite definido para o servidor em questão.

O valor definido para o *trigger* associado a este item no *template* é de 70, ou seja, qualquer valor maior ou igual a 70 vai disparar o *trigger* e lançar alerta. De frisar que este *trigger* do *template* pode ser desativado para um host específico e ser criado outro *trigger* com um valor diferente, de forma a poder haver uma adaptação de valores de acordo com o servidor/dispositivo em questão.



Host	Name ▲	Last check	Last value	Change	Tags	Info
ASP-BD	Percentagem de Uso de CPU	3m 50s	14 %	-9 %		Graph
ASP-APD	Percentagem de Uso de CPU	3m 46s	0 %			Graph
ASP-DCA	Percentagem de Uso de CPU	3m 44s	0 %			Graph
ASP-EPP	Percentagem de Uso de CPU	3m 43s	0 %			Graph
ASP-DC2	Percentagem de Uso de CPU	3m 43s	0 %			Graph
ASP-BD2	Percentagem de Uso de CPU	3m 41s	25 %	+24 %		Graph
ASP-LNUI	Percentagem de Uso de CPU	3m 41s	0 %			Graph
ASP-LNAP	Percentagem de Uso de CPU	3m 40s	23 %	-3 %		Graph
ASP-LNTST	Percentagem de Uso de CPU	3m 39s	0 %			Graph
ASP-InforOS	Percentagem de Uso de CPU	3m 38s	3 %	-8 %		Graph
ASP-NG	Percentagem de Uso de CPU	3m 31s	0 %			Graph
ASP-FT	Percentagem de Uso de CPU	3m 36s	0 %			Graph
ASP-BI	Percentagem de Uso de CPU	3m 33s	0 %			Graph
ASP-DC1	Percentagem de Uso de CPU	3m 31s	0 %	-3 %		Graph
ASP-CAD	Percentagem de Uso de CPU	3m 30s	0 %			Graph
ASP-CAM	Percentagem de Uso de CPU	3m 29s	0 %	-3 %		Graph

Figura 9 – Resultados obtidos para a percentagem de uso de CPU para vários servidores

Exemplo de *triggers* definidos para o servidor “ASP-BD”:

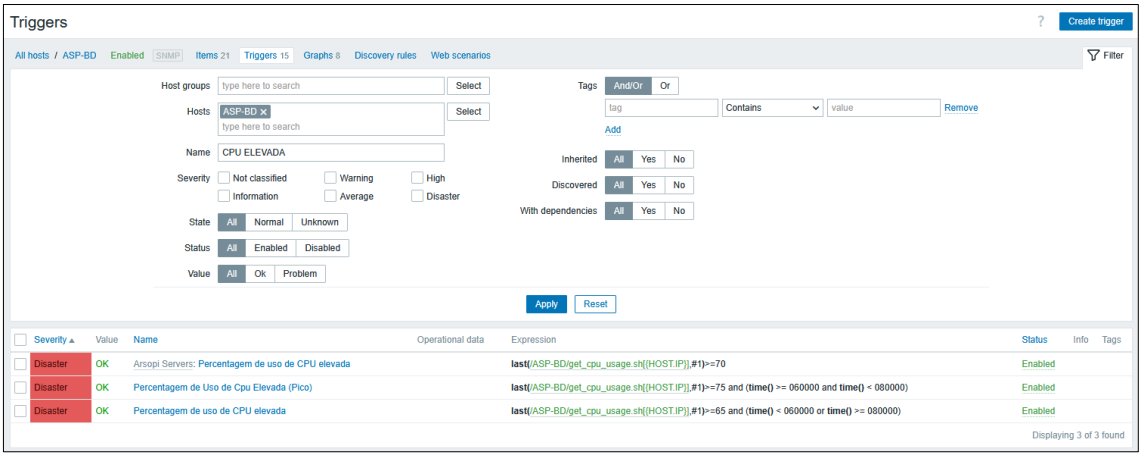


Figura 10 – Demonstração dos diferentes *triggers* para o mesmo item de CPU

3.6.2.3.2 Discos via WMI

Este é um item que tem como objetivo obter uma *string* que consiste na lista de discos do servidor e, para cada disco, obter o tamanho do mesmo e o espaço livre, entre outras características.

No *template*, o item não tem nenhum *trigger* nem gráfico associado, isto porque optou-se por associar *triggers* individualmente para cada servidor, uma vez que nem todos os servidores têm o mesmo número de discos.

Tem um intervalo de tempo de 1 dia e é do tipo "External check". Mais uma vez, este item é obtido através de um *script* em Bash que usa WMI para obter a informação. O *script* é bastante simples, tendo apenas uma linha de código:

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@1 -query "SELECT Caption, FreeSpace, Size,
FileSystem FROM Win32_LogicalDisk"
```

Código 9 – *Script* "get_disks_freespace.sh"

Na figura 11, pode-se ver um exemplo da *string* que é recebida para um determinado servidor:

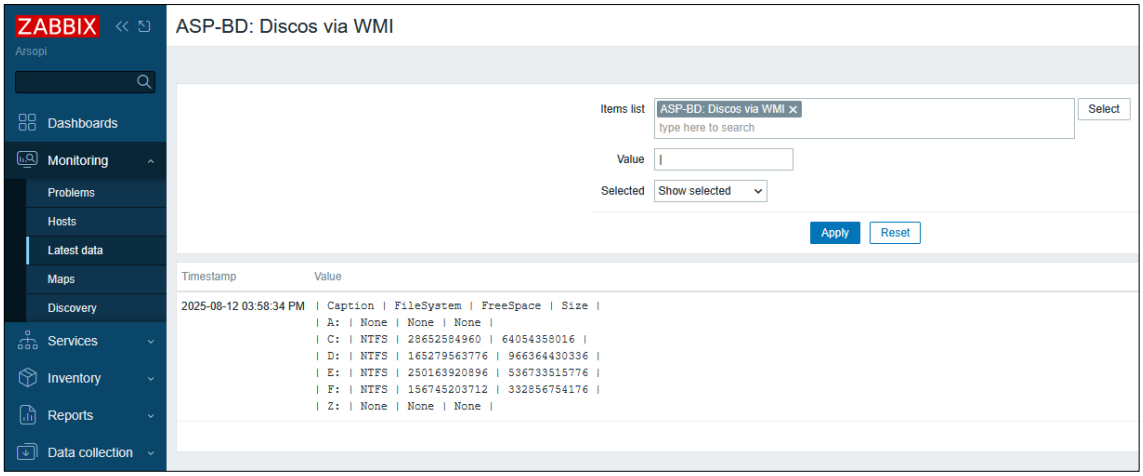


Figura 11 – Exemplo de obtenção do espaço livre dos discos para o servidor ASP-BD antes do tratamento dos dados

Depois de se obterem estes valores, cada servidor tem itens individuais configurados para dividir este texto por disco e para definir *triggers* para cada um deles, tal como mostram as seguintes figuras.

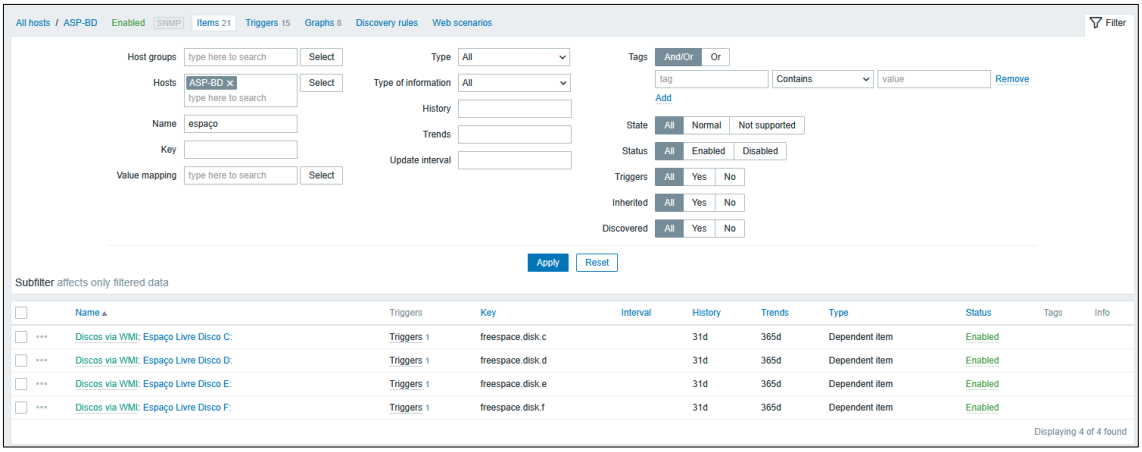


Figura 12 – Demonstração dos itens configurados a partir do item “Discos via WMI”

Estes itens criados para cada um dos servidores são do tipo “Dependent item”, ou seja, são itens que dependem do item “Discos via WMI” vindo do *template*. Estes itens não têm um intervalo de tempo definido, uma vez que são executados logo após que o item “pai” é executado. Cada um deles tem uma “tarefa” de *preprocessing*, um pequeno passo para transformar a *string* vista na figura 11 em apenas um número correspondente ao espaço livre. Este passo é uma função em JavaScript, tal como se pode ver na figura 13.

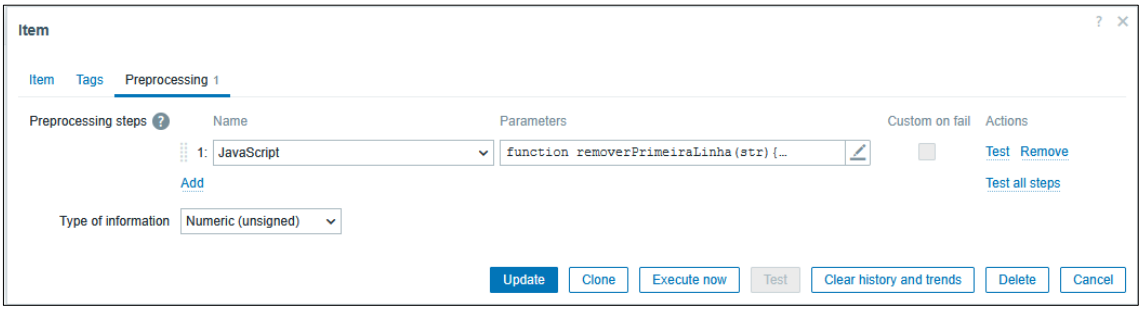


Figura 13 – Demonstração da tarefa em JavaScript de *preprocessing*

E o código JavaScript lá inserido é o seguinte:

```
function removerPrimeiraLinha(str){
    const colunas = str.split(/\s{2,}/);
    const dadosRestantes = colunas.slice(4);
    return dadosRestantes.join(' ');
}

const resultado1 = value.replace(/\|/g, ' ');
const resultado = removerPrimeiraLinha(resultado1);

const array = resultado.split(/\s+/);

return (array[7]/array[8]*100);
```

Código 10 – Código JavaScript da tarefa de *preprocessing* de cada item

O que este código faz é transformar uma *string* do género da presente na figura 11 em vários valores inteiros, um para cada disco associado. Ou seja, passo a passo:

- A função "removerPrimeiraLinha(str)" faz exatamente o que o seu nome diz, remove a primeira linha da *string* que é praticamente o título das colunas. Isto é conseguido com 3 linhas de código. A primeira divide toda a *string* por espaços (2 ou mais) e guarda este *array* na constante "coluna". A segunda linha da função cria um *array* denominado "dadosRestantes" e guarda apenas a informação a partir do *index* 4, ou seja, deixando ficar para trás as 4 partes iniciais do *array* que correspondem à primeira linha. A terceira linha devolve este *array* em formato de *string* novamente, juntando cada parte do *array* com um espaço;
- O código começa por substituir todas os caracteres "|" por espaço, dando origem à *string* "resultado1";
- De seguida, aplica a esta *string* "resultado1" a função descrita anteriormente, resultando uma *string* denominado "resultado" que já não tem a primeira linha;
- De seguida, mais uma vez, transforma a *string* "resultado" num *array* denominado "array" que é separado por um ou mais espaços, ficando assim apenas com os valores para cada disco;
- Por fim, vai buscar os 2 relativos a um determinado disco e faz a conta para calcular a percentagem de espaço livre no disco para cada um. Ou seja, este código é repetido o número de vezes igual ao número de discos existentes na máquina, diferenciando

apenas a última linha em que vai devolver valores diferentes para valores diferentes do *array*.

Esta é uma função bastante utilizada na configuração dos itens, pelo que será mencionada mais vezes. O resultado está explícito na figura 14.

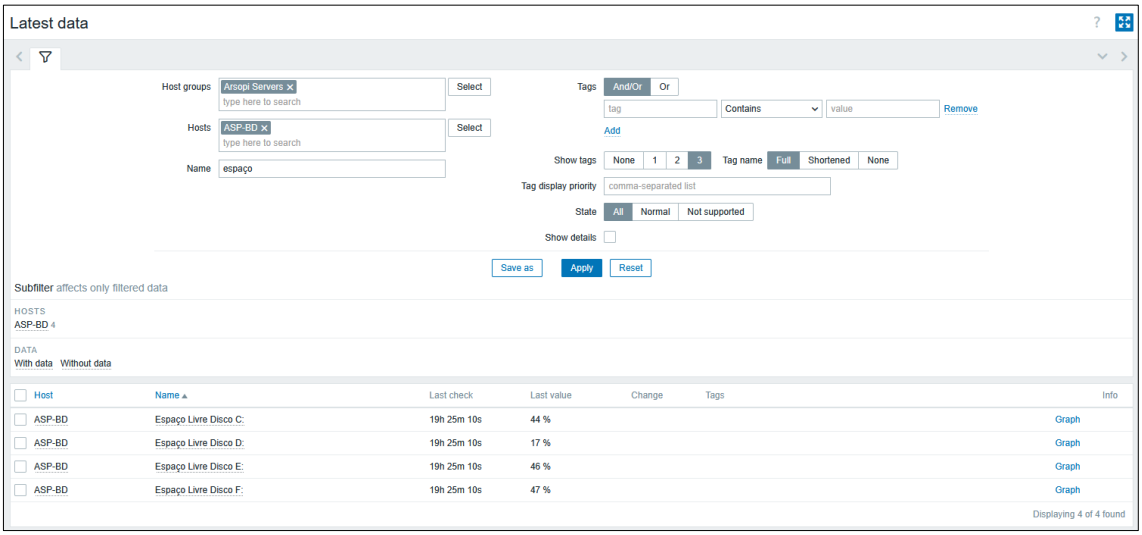


Figura 14 – Exemplo de resultado para o item “Discos via WMI”

E os *triggers* definidos para estes itens são os seguintes, todos com grau de severidade “Disaster” e com um limite de espaço livre de 10%, ou seja, se o espaço livre em disco for igual ou menor que 10%, dispara o *trigger* e envia o alerta.

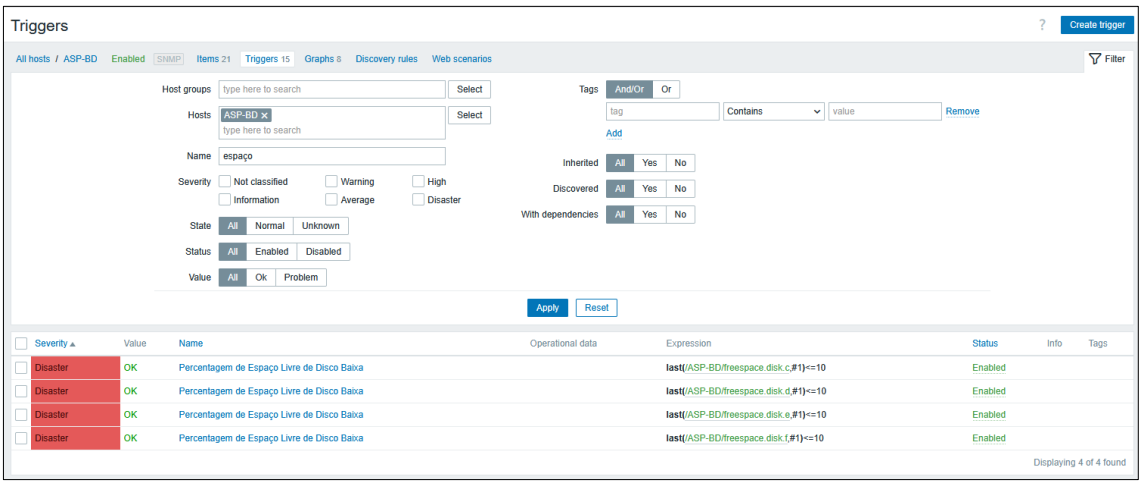


Figura 15 – Demonstração dos *triggers* para o servidor ASP-BD, para o item “Discos via WMI”

3.6.2.3.3 Erros do Event Viewer

Este item tem como objetivo obter os erros de Event Viewer dos servidores, dentro de algumas restrições. Tem um intervalo de tempo de 5 minutos e é do tipo "External Check", uma vez que os erros são obtidos via WMI, através de um *script* Bash.

O *script* é como mostra o código 11:

```
#!/bin/bash

last_minutes=$(date --utc +"%Y%m%d%H%M%S.000000-000" --date="5 minutes ago")

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT RecordNumber, TimeGenerated,
Message, SourceName FROM Win32_NTLogEvent WHERE EventType = 1 AND
TimeGenerated >= '$last_minutes' AND NOT (SourceName LIKE '%Perflib%' OR
SourceName LIKE '%Schannel%' OR SourceName LIKE '%SharePoint%' OR SourceName
LIKE '%DistributedCOM%' OR SourceName LIKE '%DCOM%' OR Message LIKE
'%KRB_AP_ERR_MODIFIED%' OR SourceName LIKE '%Sp.Feedback%' OR Message LIKE
'%The AppFabric Caching Service%' OR Message LIKE '%AddLegacyDriverFiles%'
OR Message LIKE '%The computer with the IP address 10.10.252.1 did not allow
the name to be claimed by this computer.%' OR Message LIKE '%ARSOPI<1B>%')"
```

Código 11 – Demonstração do *script* "get_eventviewer_errors.sh"

Este *script* guarda na variável "last_minutes" uma *string* no formato necessário para a *query* do *timestamp* de 5 minutos antes do momento em que corre o *script*. O objetivo passa por, de 5 em 5 minutos, a *script* ser executada e se houver algum erro dentro das restrições impostas, este erro vai ser mostrado e um alerta será lançado, uma vez que o *trigger* deste item está configurado para ser disparado caso o retorno da *string* não seja vazio, o que significa que há erros no visualizador de eventos do servidor em causa.

Em relação às restrições, este vai buscar todos os erros, visto que o tipo de evento 1 corresponde a "Erro", ocorridos nos últimos 5 minutos, ignorando alguns erros que são insignificantes e, por esse motivo, são considerados "ruído".

Os problemas gerados por este item são de severidade "Alta". Tendo em conta a nossa política de alertas e o facto de ser mais frequente este tipo de alertas/problemas, decidiu-se classificar os problemas associados a este item desta forma para evitar uma sobrecarga excessiva de emails. Desta forma, os alertas gerados a partir deste item aparecem na página principal do nosso *dashboard*, de fácil visualização, e são resolvidos na hora pela nossa equipa.

Figura 16 – Exemplo de problema gerado a partir de erros do Event Viewer, mostrado no *dashboard*

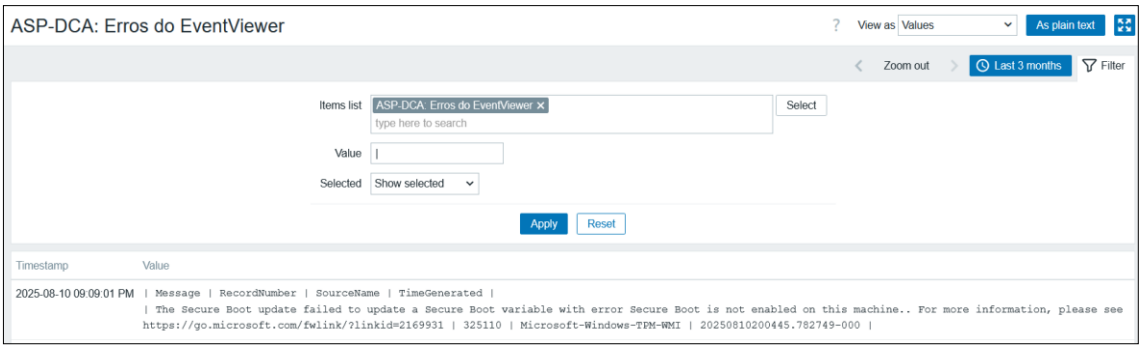


Figura 17 - Demonstração de um outro problema relacionado com este item

3.6.2.3.4 Percentagem de Uso de Memória

Permite obter o valor da percentagem de uso de memória de uma determinada máquina. Fá-lo por via de um *script* Bash com uso de WMI de 5 em 5 minutos e tem um *trigger* configurado para ser disparado se o valor da percentagem for maior ou igual que 95, originando um problema de severidade “Desastre”.

O código da *script* é o seguinte:

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT TotalVisibleMemorySize,
FreePhysicalMemory FROM Win32_OperatingSystem"
```

Código 12 – *Script* “get_memory_usage.sh”

Este código retorna uma *string* que posteriormente é tratada para se obter apenas o valor desejado. Isto é feito através de uma função em JavaScript semelhante à presente no código 10.

3.6.2.3.5 Tráfego de Rede (Enviado/Recebido)

Este item foi implementado através de um *script* em Bash que recorre ao WMI, sendo executado periodicamente a cada 10 minutos. O seu objetivo consiste em registar os valores de tráfego de rede de cada servidor, recolhendo métricas relativas aos bytes enviados e recebidos por segundo. Os dados são armazenados com um período de retenção de 30 dias. Este item não possui *triggers* associadas, servindo apenas como fonte de consulta em caso de incidentes. O *script* em Bash é o seguinte:

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name, BytesReceivedPerSec,
BytesSentPerSec FROM Win32_PerfFormattedData_Tcpip_NetworkInterface"
```

Código 13 – *Script* “get_traffic_values.sh”

A informação obtida a partir do *script* acima apresentado no Zabbix é a seguinte para cada servidor:

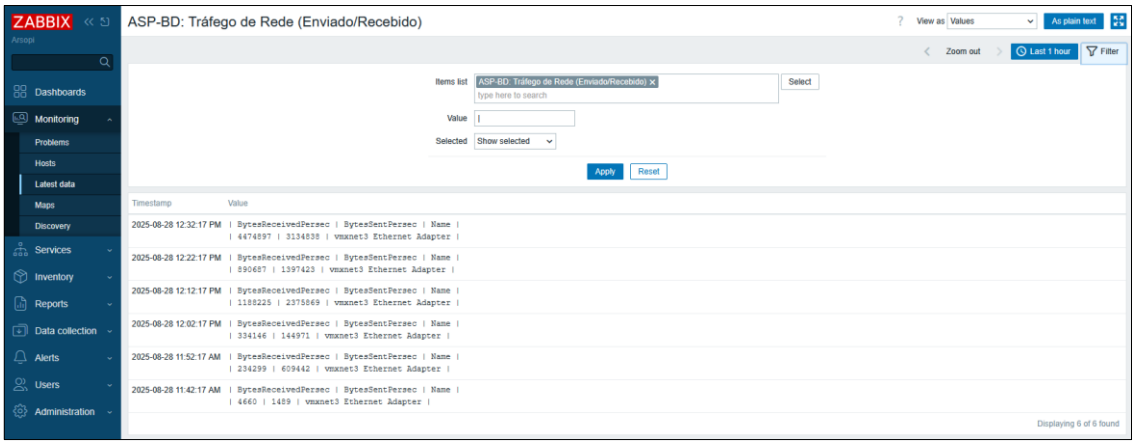


Figura 18 – Exemplo de demonstração de texto recebido relacionado com tráfego de rede num servidor

Importa referir que se trata de um item intermédio, a partir do qual são gerados itens dependentes para cada servidor, de forma semelhante ao que foi realizado na monitorização da métrica de utilização de discos. Neste caso concreto, cada *host* pertencente ao grupo "Arsopi Servers" possui dois itens dependentes configurados:

- um destinado a extrair os valores correspondentes aos bytes enviados por segundo;
- outro relativo aos bytes recebidos por segundo.

Estes itens processam a informação recolhida pelo item "pai" e transformam-na em métricas individuais, mais simples de consultar, visualizar e analisar.

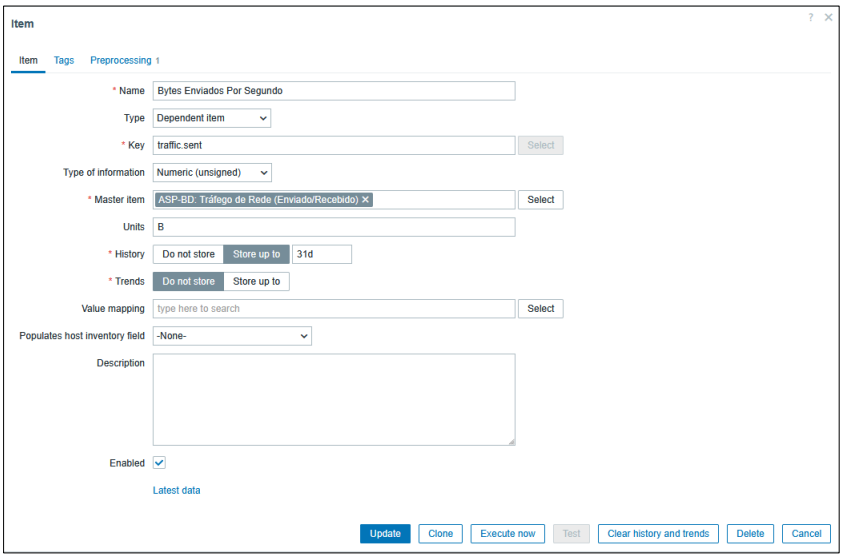


Figura 19 – Demonstração de um dos dois itens configurados a partir do *script*

Para que seja possível converter o texto devolvido pelo item principal num valor numérico único, foi aplicado um passo de pré-processamento, recorrendo a um excerto de código em JavaScript semelhante aos utilizados em outros casos análogos.

O resultado encontra-se representado na figura 20, onde é possível observar que cada servidor monitorizado apresenta duas métricas distintas: uma referente aos bytes recebidos por segundo e outra relativa aos bytes enviados por segundo.

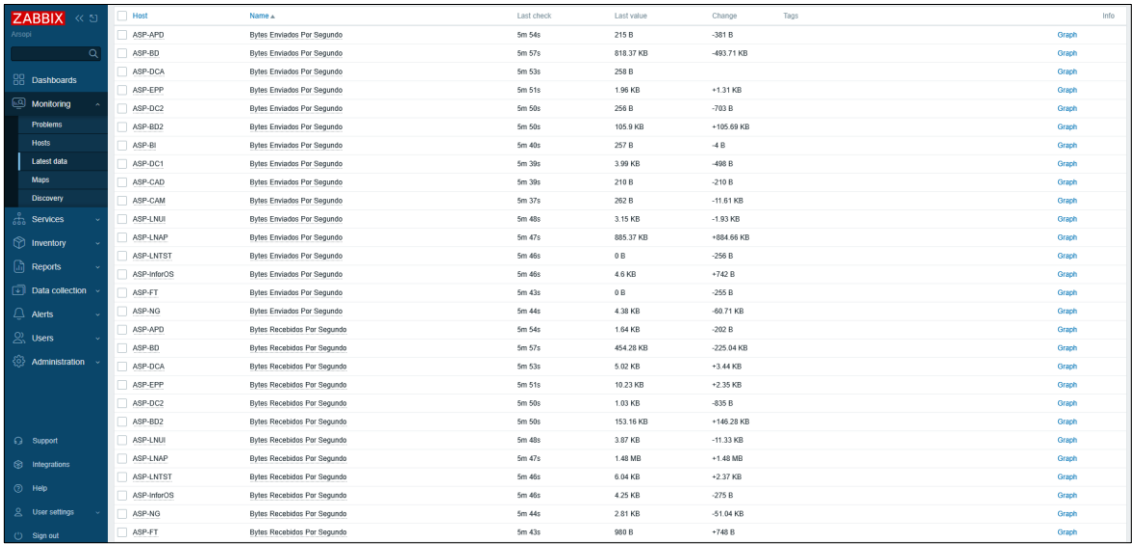


Figura 20 – Demonstração dos resultados obtidos para cada servidor dos dois itens configurados a partir do *script*

3.6.2.3.6 Serviços de Trend Micro

Este é um item do tipo “External check”, uma vez que consiste num *script* Bash que faz uso de WMI para obter informações acerca dos serviços do antivírus Trend Micro que se encontram em execução numa determinada máquina.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query “SELECT Name FROM Win32_Process WHERE Name
LIKE ‘tmlisten%’ OR Name LIKE ‘ntrtscan%’”
```

Código 14 – *Script* “get_trendmicro_services.sh”

Todos os itens cujo nome é “Serviços de…” envolvem a mesma lógica, ou seja, é feito um estudo prévio da aplicação que se pretende monitorizar, percebendo desta forma quais serviços precisam de estar em execução para que esta funcione corretamente. O objetivo passa por recolher a lista de serviços que estão a correr cujo nome é semelhante ao das condições impostas e garantir que o número de processos ativos coincide com o mínimo necessário para

garantir a operação normal da aplicação. Qualquer valor abaixo do número de serviços necessários para o bom funcionamento da aplicação origina o lançamento de um alerta do tipo "Desastre", alertando para a possibilidade de mau funcionamento da aplicação em causa.

O *script* acima devolve apenas uma lista dos serviços procurados que se encontram em execução na máquina, sendo necessário, posteriormente, aplicar uma tarefa de pré-processamento em JavaScript para transformar esta lista de serviços num valor numérico.

```
const result1 = value.replace(/\|/g, ' ');
const result2 = result1.replace(/\n/g, ' ');
const result3 = result2.trim();
const result4 = result3.replace(/\s+/g, ' ');
const colunas = result4.split(' ');
const tamanho = colunas.length - 1;
return tamanho;
```

Código 15 – Tarefa de pré-processamento em JavaScript

O que este código JavaScript faz, passo a passo, é:

- Substituir todos os "|" por espaços;
- Substituir todas as quebras de linha por espaços também;
- Eliminar espaços a mais existentes no início e no fim da *string*;
- Converter múltiplos espaços em um único espaço;
- Transformar a *string* num *array* em que o separador é um espaço;
- Guardar o valor do tamanho do *array* tirando 1 e retornar esse valor (tira-se 1 que corresponde ao título da coluna e os restantes já são serviços por isso devem ser contabilizados)

Com isto, transforma-se uma *string* que consiste numa lista de serviços em execução, do tipo:

```
| Name |
| NTRTScan.exe |
| TmListen.exe |
```

Código 16 – Demonstração da *string* retornada pelo *script* antes do pré-processamento

Num número apenas, que neste caso, é 2.

Neste caso, foi configurado um alerta que é acionado sempre que o número de serviços for inferior a dois, indicando uma potencial falha no agente Trend Micro.

Este item, tal como todos os itens semelhantes a este que envolvem a monitorização de serviços, é executado a cada 10 minutos, garantindo uma monitorização regular do antivírus nos servidores, neste caso.

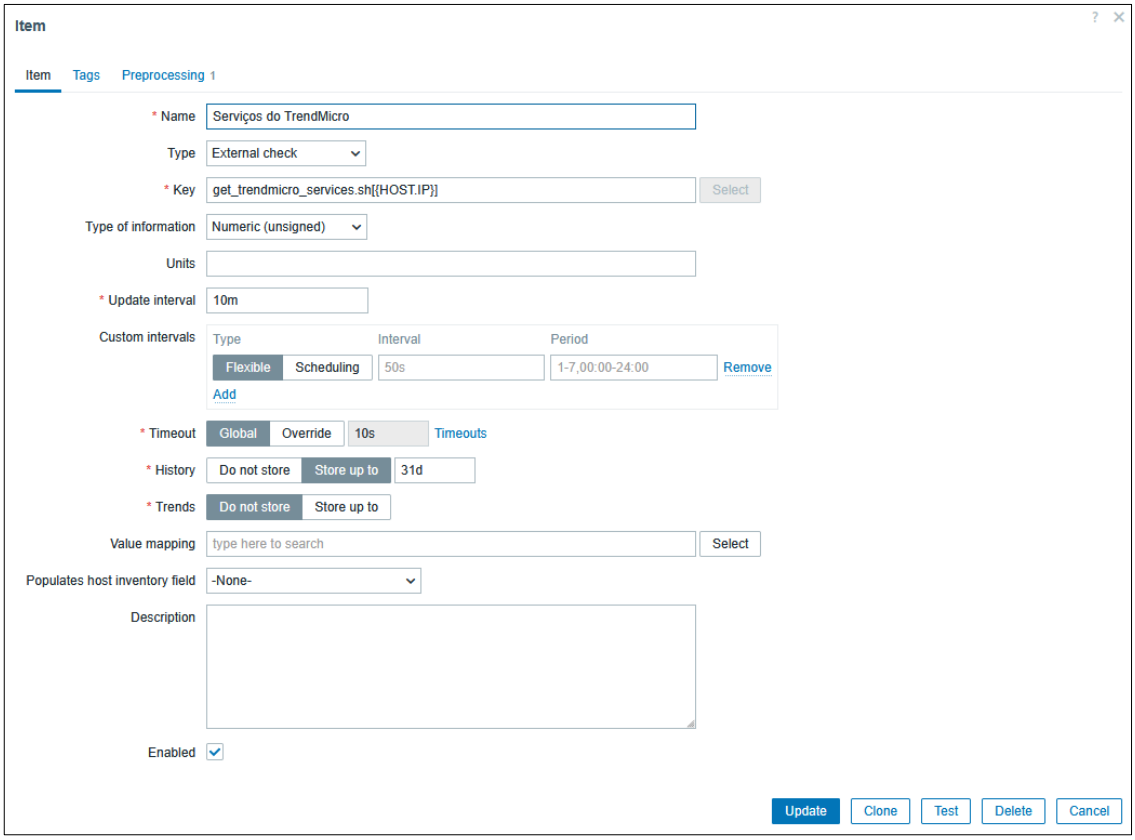


Figura 21 – Demonstração do item

3.6.2.4 Arsopi Switches

O *template* "Arsopi Switches" foi criado a partir da clonagem do *template* nativo "HP Enterprise Switch by SNMP". Após uma análise comparativa aos *templates* existentes, verificou-se que este cumpria a maioria dos requisitos definidos para a monitorização de *switches*. A opção pela clonagem resulta de uma boa prática adotada pela equipa, que consiste em evitar a utilização direta de *templates* ou itens nativos do Zabbix, de modo a permitir adaptações sem restrições e garantir maior flexibilidade na manutenção futura.

O *template* resultante inclui um conjunto de itens, alarmes (*triggers*) e regras de descoberta (*discovery rules*) considerados úteis para a infraestrutura em questão. A maioria dos elementos originais foi mantida, tendo-se ainda procedido à adição de dois itens em falta:

- Total bits received
- Total bits sent

Estes itens permitem recolher o valor total de bits por segundo recebidos e enviados, respetivamente, sendo descritos em detalhe nos pontos 3.6.2.4.1 e 3.6.2.4.2.

Relativamente a *triggers*, apenas um se encontra ativo: o alarme associado à resposta a *pings*, configurado com severidade "Desastre". Este alarme é acionado sempre que não exista resposta positiva a *pings*, funcionando como um indicador de indisponibilidade do dispositivo.

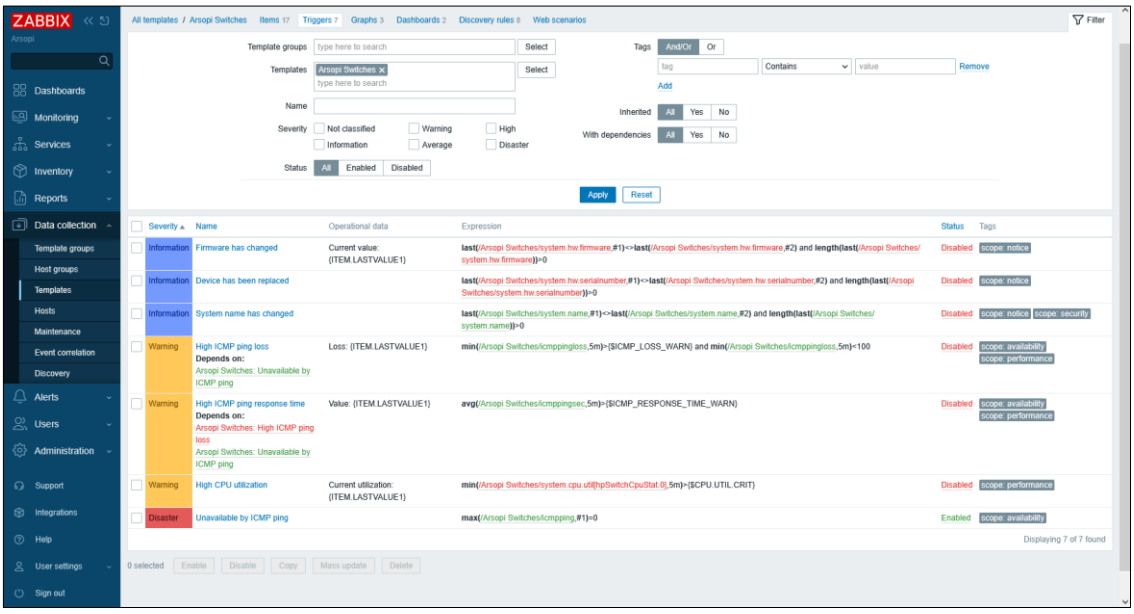


Figura 22 – Demonstração dos *triggers* ativos no *template* "Arsopi Switches"

Embora o *template* inclua *dashboards* pré-configurados, estes não estão atualmente em utilização. No que respeita a *discovery rules*, apenas a regra "Network interfaces discovery" se encontra ativa. Esta regra tem como função detetar todas as interfaces de cada *switch* e recolher as métricas associadas, definidas através de *item prototypes*.

Os *item prototypes* existentes permitem monitorizar, entre outros aspetos, a velocidade de ligação de cada posto de trabalho, a quantidade de tráfego por porta e o estado operacional das interfaces. Associados a estes itens encontram-se *trigger prototypes*, que se encontram desativados por não serem considerados necessários no contexto atual.

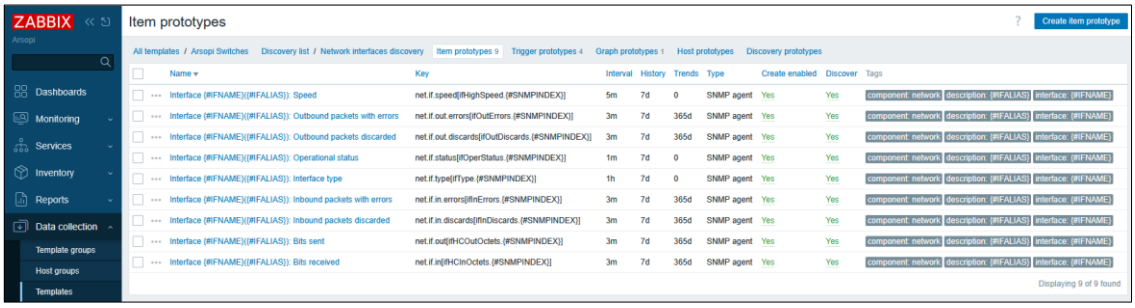


Figura 23 – *Item prototypes* do template "Arsopi Switches"

Por fim, o *template* inclui ainda um *graph prototype* associado ao tráfego de rede, composto por duas linhas: uma relativa ao tráfego de entrada e outra ao tráfego de saída. Este gráfico é gerado automaticamente para cada interface de rede descoberta, constituindo uma ferramenta útil para a monitorização detalhada da rede a nível de porta ou posto de trabalho. Os exemplos gráficos serão apresentados posteriormente na secção dedicada aos *dashboards*.

3.6.2.4.1 Total Bits Received

Este item foi adicionado ao *template* "Arsopi Switches" com o objetivo de calcular o tráfego total de rede recebido por um *switch*. Trata-se de um item do tipo "Calculated", dado que o seu valor resulta do processamento de métricas já recolhidas por outros itens. A chave definida para o item é "total.traffic.in", sendo o resultado um valor numérico calculado de acordo com a seguinte fórmula:

```
sum(last_foreach(//net.if.in[*]))
```

Código 17 – Chave usada no item "Total Bits Received"

As unidades de medição são *bps* (bits por segundo), sendo o item atualizado com uma periodicidade de 3 minutos. Para garantir a coerência dos valores obtidos, foram configurados dois passos de pré-processamento:

- Change per second;
- Custom Multiplier: 8.

O funcionamento é o seguinte: o item recolhe o último valor de tráfego recebido em cada interface do *switch* e efetua a soma de todos os valores. Em seguida, aplica as regras de pré-processamento, convertendo o total em tráfego por segundo e multiplicando por 8, de forma a transformar a unidade padrão (bytes) em bits.

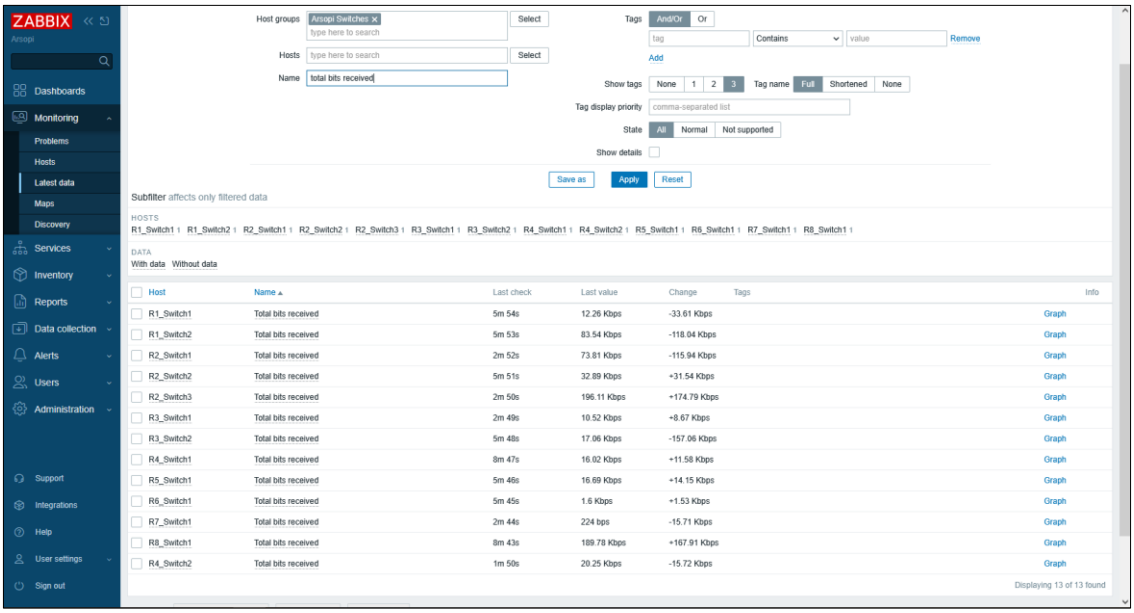


Figura 24 – Demonstração do total de bits recebidos por *switches*

Embora não possua *trigger* associado, este item inclui um gráfico próprio, utilizado nos *dashboards*. O seu principal objetivo é disponibilizar um indicador de consulta que permite acompanhar, de forma consolidada, o tráfego recebido em cada *switch*, ainda que não exista a necessidade de definir limites de alarme específicos.

3.6.2.4.2 Total Bits Sent

Este item apresenta uma configuração semelhante ao item anterior "Total Bits Received", diferenciando-se apenas pelo facto de monitorizar o tráfego total de rede enviado pelo *switch*. A chave definida é "total.traffic.out", sendo o valor calculado de acordo com a fórmula seguinte:

```
sum(last_foreach(//net.if.out[*]))
```

Código 18 – Chave usada no item "Total Bits Sent"

Tal como no item anterior, são aplicados os mesmos dois passos de pré-processamento. Estes passos garantem que os valores recolhidos são expressos em bits por segundo, convertendo corretamente os dados obtidos em bytes.

O item possui igualmente um gráfico associado, utilizado nos *dashboards* para representar a evolução do tráfego enviado. À semelhança do item "Total Bits Received", não foi configurado qualquer *trigger*, dado tratar-se de um indicador destinado à consulta e análise, sem limites de alarme pré-estabelecidos.

3.6.2.5 FortiGate By SNMP

Para além da monitorização de servidores, *switches* e outros dispositivos de menor relevância, identificou-se a necessidade de incluir também a *firewall* na plataforma de monitorização. Para esse efeito, foi adotado o *template* nativo do Zabbix "FortiGate by SNMP", que se revelou adequado por disponibilizar um conjunto alargado de métricas relevantes, dispensando a criação de um *template* de raiz.

Este *template* permite acompanhar métricas consideradas essenciais para a operação da firewall, nomeadamente:

- percentagem de utilização de CPU;
- percentagem de utilização de memória;
- tráfego de rede (LAN e WAN);

- métricas de VPN, incluindo o número de sessões IPsec ativas e o número de utilizadores ligados via VPN SSL.

Para que a monitorização via SNMP fosse viável, foi necessário proceder a uma configuração prévia na *firewall*:

- Ativação do agente SNMP na interface de gestão;
- Criação de um utilizador SNMPv3, definindo nome e palavra-passe;
- Configuração de um "Trusted Host", restringindo o acesso SNMP ao servidor onde o Zabbix se encontra instalado;
- Ativação do SNMP na interface LAN, garantindo a recolha de métricas pela rede interna.

No Zabbix, foi criado um *host* denominado "Firewall Fortinet", associado aos *templates* "Arsopi Ping" e "FortiGate by SNMP", e inserido no grupo "Arsopi Firewalls", com configuração de interface SNMP, conforme ilustrado na figura 25.

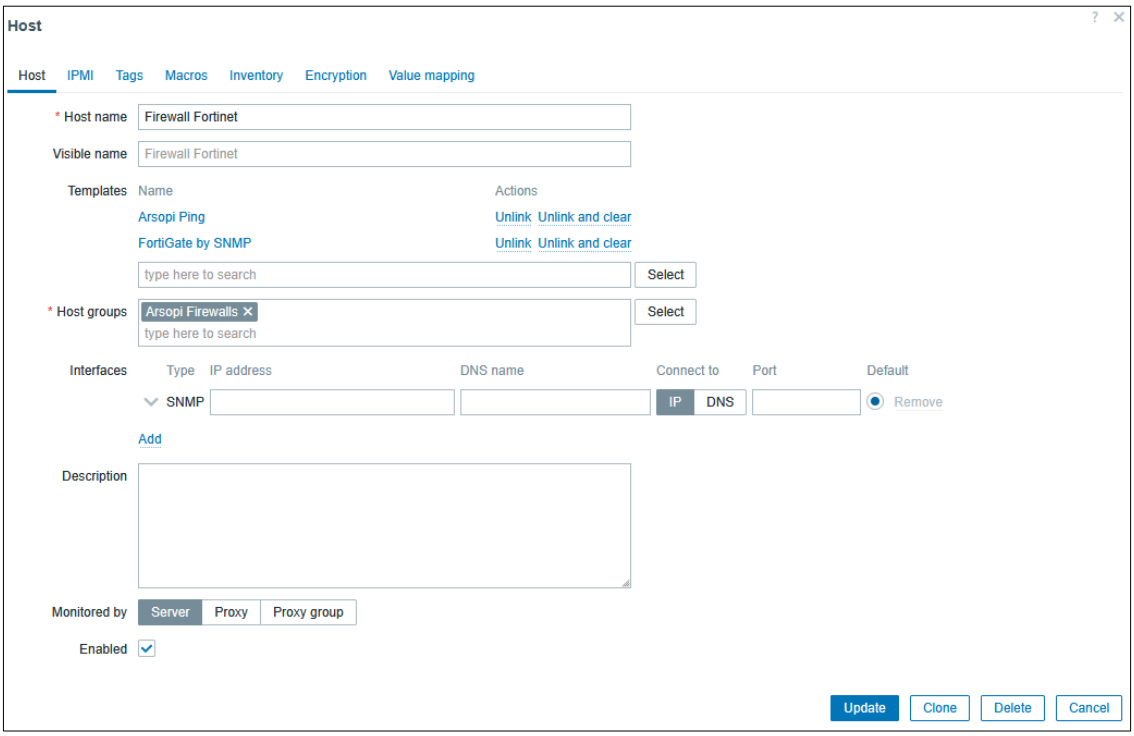


Figura 25 – Configuração do *host* da *firewall* com o *template* "FortiGate by SNMP"

O *template* "FortiGate by SNMP" é bastante completo, incorporando centenas de itens, alarmes e *discovery rules*. No entanto, a sua utilização foi ajustada às necessidades da infraestrutura, sendo que uma parte significativa dos itens e regras de descoberta encontra-se desativada, por não serem considerados relevantes, enquanto os restantes permanecem ativos, por fornecerem informação necessária aos *dashboards* ou por poderem ser úteis em situações de incidente.

3.6.3 Hosts e Host Groups

3.6.3.1 Outras métricas

Como referido anteriormente, cada *host* possui, por norma, pelo menos um *template* associado, o que garante a existência de um conjunto de métricas pré-definidas e uniformes dentro do grupo a que pertence. No entanto, é igualmente possível que determinados *hosts* necessitem de métricas adicionais, específicas ao seu contexto, que não se aplicam de forma generalizada ao grupo. Nessas situações, essas métricas não são integradas em *template*s, mas configuradas individualmente no próprio *host* — tal como exemplificado no caso das métricas relacionadas com discos e tráfego de rede.

Adicionalmente, existem métricas que, embora sejam semelhantes e partilhadas por diversos *hosts*, apresentam pequenas variações que justificam a sua criação individual. Desta forma, assegura-se que cada *host* é monitorizado de acordo com as suas particularidades, sem comprometer a consistência global da monitorização.

3.6.3.1.1 Spooler

Esta métrica está associada ao servidor responsável pela gestão de impressoras da rede. Consiste na execução periódica (a cada 10 minutos) de um *script* em Bash que devolve a lista de processos do *spooler* atualmente em execução. Foi configurado um *trigger* de severidade

"Desastre", acionado sempre que o número de serviços em execução relacionados com o *spooler* for igual a zero, por indicar indisponibilidade do serviço de impressão.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name FROM Win32_Process WHERE Name
LIKE 'Spool%'"
```

Código 19 – *Script* "get_spool.sh"

Dado que o resultado devolvido pelo script é textual e não diretamente numérico, aplica-se o pré-processamento em JavaScript, tal como mostra o código 20.

```
const result1 = value.replace(/\|/g, ' ');
const result2 = result1.replace(/\n/g, ' ');
const result3 = result2.trim();
const result4 = result3.replace(/\s{2,}/, ' ');
const colunas = result4.split(' ');
const tamanho = colunas.length - 1;
return tamanho;
```

Código 20 – Tarefa de pré-processamento JavaScript do resultado do item

De forma concisa, esse pré-processamento:

- substitui todos os caracteres «|» e quebras de linha por espaços;
- remove espaços redundantes no início e no fim da string e normaliza múltiplos espaços para um só;
- converte a string num array (separador: espaço);
- calcula o tamanho do array, subtraindo uma unidade correspondente ao cabeçalho;
- devolve, assim, o número de processos identificados.

Desta forma, a métrica apresenta um valor inteiro facilmente analisável e compatível com a lógica do alarme definida.

3.6.3.1.2 Serviços de SQL Server

Este item, assim como os próximos até ao ponto 3.6.3.1.8, inclusive, insere-se no conjunto de métricas relacionadas com serviços, pelo que será bastante semelhante ao item do ponto 3.6.2.3.6, apenas com algumas diferenças. O objetivo deste item é obter o número de processos ativos relacionados com SQL Server, logo o *script* Bash utilizado terá de ser ligeiramente diferente:

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name FROM Win32_Process WHERE Name
LIKE 'sqlservr%'"
```

Código 21 – *Script* "get_sql_services.sh"

Tal como nos restantes itens desta categoria, aplica-se o mesmo passo de pré-processamento em JavaScript previamente descrito, que transforma a lista de serviços devolvida pelo WMI num valor numérico representativo do total de processos ativos.

Como foi explicado anteriormente, a chave deste item é "get_sql_services.sh[{HOST.IP}]". Por fim, este item conta com um *trigger* que gera um alerta assim que o número de serviços em execução relacionados com SQL Server for menor que 3, originando um problema de grau "Desastre". Este item está associado a 2 servidores, sendo eles o ASP-BD e o ASP-BD2.

3.6.3.1.3 Serviços de Qlik Sense

Este item integra igualmente o grupo de métricas associadas a serviços críticos, sendo configurado de forma semelhante ao item do ponto 3.6.2.3.6 (Serviços de Trend Micro). Neste caso, o objetivo passa por verificar o correto funcionamento da aplicação Qlik Sense, através da contagem do número de processos indispensáveis que devem estar em execução.

Para tal, foi desenvolvido um *script* em Bash adaptado especificamente à identificação dos serviços do Qlik Sense.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name FROM Win32_Process WHERE Name
LIKE 'Engine%' OR Name LIKE 'Printing%' OR Name LIKE 'Proxy%' OR Name LIKE
'pg_ctl%' OR Name LIKE 'Repository%' OR Name LIKE 'Scheduler%' OR Name LIKE
'ServiceDispatcher%'"
```

Código 22 – Script "get_qlik_services.sh"

À semelhança dos restantes itens desta categoria, é aplicado o mesmo processo de pré-processamento em JavaScript, já detalhado anteriormente, que converte a lista textual de processos devolvida pelo WMI num valor numérico. A chave associada a este item é "get_qlik_services.sh[{HOST.IP}]". Foi ainda configurado um *trigger* que é acionado sempre que o número de serviços ativos seja inferior a nove, limiar considerado mínimo para garantir o funcionamento estável da aplicação. Este item encontra-se associado ao servidor ASP-BI.

3.6.3.1.4 Serviços de CAD (Computer-Aided Design)

Este item enquadra-se também no conjunto de métricas do tipo "Serviços de …", funcionando de forma semelhante ao modelo descrito no ponto 3.6.2.3.6 (Serviços de Trend Micro). O objetivo é garantir que todos os serviços essenciais ao correto funcionamento dos softwares CAD se encontram ativos, através da contagem do número de processos em execução. A recolha desta informação é efetuada por um *script* em Bash desenvolvido especificamente para identificar os serviços associados ao CAD.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name FROM Win32_Process WHERE Name
LIKE 'tomcat10%' OR Name LIKE 'eDoc%' OR Name LIKE 'eGrouServiceManager%' OR
Name LIKE 'FNPLicensingService%' OR Name LIKE 'NLMServer%' OR Name LIKE
'hasplms%' OR Name LIKE 'SimLabFloatingServer%' OR Name LIKE 'lmgrd%' OR
Name LIKE 'ArchiveServerService%' OR Name LIKE 'sqlservr%'"
```

Código 23 – *Script* "get_cad_services.sh"

À semelhança dos casos anteriores, aplica-se o mesmo mecanismo de pré-processamento em JavaScript, já explicado anteriormente, o qual converte a lista devolvida pelo WMI num valor numérico representativo do total de serviços ativos. A chave definida para este item é "get_cad_services.sh[{HOST.IP}]". Foi configurado um alarme que é acionado quando o número de serviços ativos desce abaixo de catorze, valor considerado o mínimo aceitável para garantir a operacionalidade desta aplicação. Este item está associado ao servidor ASP-CAD.

3.6.3.1.5 Serviços de CAM (Computer-Aided Manufacturing)

Tal como os restantes itens do tipo "Serviços de …", este elemento segue a mesma metodologia descrita no ponto 3.6.2.3.6 (Serviços de Trend Micro). O objetivo é monitorizar os serviços essenciais dos softwares CAM, assegurando que o número mínimo de processos em execução corresponde ao necessário para o seu funcionamento regular. Para tal, foi criado um *script* em Bash ajustado aos serviços específicos do CAM.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT Name FROM Win32_Process WHERE
Name LIKE 'FNPLicensing%' OR Name LIKE 'hasplms%' OR Name LIKE 'lservnt%' OR
NAME LIKE 'sqlservr%' OR Name LIKE 'lmgrd%' OR Name LIKE 'Sp.Refresh3D%' OR
Name LIKE 'Sp.MaintenancePlan%' OR Name LIKE 'Sp.Feedback%' OR Name LIKE
'Trumpf.Analytics%' OR Name LIKE 'ProgService_Cut%' OR Name LIKE
'erlsrv.exe' OR Name LIKE 'nssm%'"
```

Código 24 – *Script* "get_cam_services.sh"

A informação devolvida pelo WMI é posteriormente sujeita ao pré-processamento em JavaScript já apresentado, que converte a lista de serviços em execução num valor numérico. Este item tem um *trigger* associado que dispara caso o número de serviços em execução seja inferior a catorze, originando um problema "Desastre". Este item está associado ao servidor ASP-CAM.

3.6.3.1.6 Serviços de Infor

Este é mais um item do mesmo tipo que os anteriores, uma vez que segue a mesma metodologia descrita no ponto 3.6.2.3.6. O seu propósito é garantir que os serviços Infor dispõem sempre do número mínimo de serviços em execução necessário ao seu funcionamento. Tal como nos casos anteriores, a recolha é realizada através de um *script* em Bash configurado para identificar os serviços específicos de Infor.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Name FROM Win32_Process WHERE Name
LIKE 'asm_srv%' OR Name LIKE 'jobd%' OR Name LIKE 'rexecd%' OR Name LIKE
'og%' OR Name LIKE 'shmserv%' OR Name LIKE 'ReportingControllerService%' OR
Name LIKE 'Slmserver%' OR Name LIKE 'tomcat8%' OR Name LIKE
'ODMVaultServer%' OR Name LIKE 'Mis.DWCore.Log.Service%' OR Name LIKE
'Infor.Bi.Repository.RegistrationService%' OR Name LIKE
'Edge.SystemTray.SendFilesService%'"
```

Código 25 – *Script* "get_infor_services.sh"

Este item é aplicado a dois servidores diferentes, sendo eles o ASP-LNAP e o ASP-LNUI. O *trigger* definido não é o mesmo para os dois servidores, disparando o primeiro caso existam menos de dez serviços em execução, enquanto para o segundo dispara com menos de 2 serviços em execução. Tanto para um servidor como para o outro, o disparo do *trigger* origina um problema de cariz "Desastre".

3.6.3.1.7 Serviços de nG

Este item segue a mesma metodologia aplicada a todos os itens do tipo "Serviços de …", descrita inicialmente no ponto 3.6.2.3.6 (Serviços de Trend Micro). O objetivo consiste em monitorizar os serviços essenciais dos softwares relacionados com relógios de ponto e ecrãs táteis de marcação de horas e ordens de serviço, assegurando que o número de processos ativos se mantém dentro dos valores esperados para o correto funcionamento da solução. A recolha da informação é realizada através de um *script* em Bash.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT Name FROM Win32_Process WHERE
Name LIKE 'dcsSRVTRF%' OR Name LIKE 'dcsSupremaServiceDispatcher%' OR Name
LIKE 'svchost%'"
```

Código 26 – *Script* "get_ng_services.sh"

É devolvida uma lista do WMI que é sujeita a um pré-processamento JavaScript já detalhado anteriormente, o qual transforma os resultados textuais num valor numérico representativo do total de serviços em execução. A chave definida para este item é "get_ng_services.sh[{HOST.IP}]". Este item está associado ao servidor ASP-NG e, neste caso, foram configurados dois *triggers* distintos:

- Um de severidade "Elevado", acionado quando o número de serviços ativos desce abaixo de 16;
- Outro de severidade "Desastre", disparado quando o número de serviços em execução cai abaixo de 3, valor que representa o funcionamento mínimo aceitável da aplicação.

3.6.3.1.8 Serviços de H2ST

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT Name FROM Win32_Process WHERE
Name LIKE 'H2ST_Sync%'"
```

Código 27 – *Script* "get_h2st_services.sh"

Tal como os restantes itens do tipo "Serviços de …", este elemento segue a metodologia explicada no ponto 3.6.2.3.6 (Serviços de Trend Micro). O seu objetivo é assegurar que a aplicação H2ST possui pelo menos um serviço ativo, condição mínima necessária ao seu correto funcionamento. A monitorização é realizada por meio de um *script* em Bash configurado para identificar os serviços do H2ST em execução.

À semelhança dos casos anteriores, a lista devolvida pelo WMI é tratada através do processo de pré-processamento em JavaScript já descrito, que converte o resultado textual num valor numérico. A chave deste item é "get_h2st_services.sh[{HOST.IP}]". Foi configurado um alarme de severidade "Desastre", acionado sempre que o número de serviços em execução for inferior a um, uma vez que a aplicação requer apenas um serviço ativo para se manter operacional. Este item está associado ao servidor ASP-NG.

3.6.3.1.9 Erros do Task Scheduler

Esta métrica encontra-se aplicada aos servidores ASP-LNAP e ASP-NG, responsáveis pelo funcionamento do ERP e de diversos serviços *web* da empresa. O seu propósito é alertar a equipa de administração de sistemas sempre que ocorra um erro no Task Scheduler que impeça a execução correta de uma tarefa agendada ou de qualquer outro processo associado. Desta forma, garante-se uma deteção precoce deste tipo de falhas, que muitas vezes podem passar despercebidas e originar consequências significativas a médio ou longo prazo.

A recolha da informação é realizada por meio de um *script* em Bash com recurso a WMI, configurado para ser executado de hora em hora. O *script* extrai os erros registados no Event Viewer durante a última hora, filtrando-os de acordo com os títulos das tarefas que se pretende monitorizar. Esta filtragem reduz consideravelmente a ocorrência de mensagens irrelevantes, garantindo que apenas os erros de interesse são apresentados.

```
#!/bin/bash

last_hour=$(date --utc +"%Y%m%d%H%M%S.000000-000" --date="1 hour ago")

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT RecordNumber,TimeGenerated,Message
FROM Win32_NTLogEvent WHERE LogFile = 'Microsoft-Windows-
TaskScheduler/Operational' AND EventType = 1 AND TimeGenerated >=
'$last_hour' AND (Message LIKE '%Email Anomalias CTH%' OR Message LIKE
'%Email Birth%' OR Message LIKE '%Email Docs%' OR Message LIKE '%Arsopi-
Thermal%' OR Message LIKE '%Tecnocon%' OR Message LIKE '%TBFilesAPI%' OR
Message LIKE '%Start JobD%' OR Message LIKE '%Update Time - UL%' OR Message
LIKE '%Limpar LN tmps%' OR Message LIKE '%Limpar PDFs%' OR Message LIKE
'%Limpar SQL old BAKs%' OR Message LIKE '%Excel Alertas%')"
```

Código 28 – *Script* "get_tasks_status.sh"

O item possui ainda um alarme associado, que é acionado sempre que o *script* devolve uma *string* não vazia, sinalizando a presença de erros que requerem atenção imediata. Caso não sejam detetados erros, o valor devolvido pelo *script* é vazio. O grau de severidade configurado para este alarme é "Desastre", em conformidade com a criticidade deste tipo de falhas.

3.6.3.1.10 Diferença de Horário Local

Esta métrica foi implementada numa fase posterior à configuração do sistema, com o objetivo de resolver um problema identificado no servidor ASP-LNAP. Verificou-se que, em determinadas ocasiões, este servidor apresentava uma diferença entre o seu horário interno e o horário local, que poderia variar entre alguns segundos e vários minutos, tendo-se registado discrepâncias de até nove minutos.

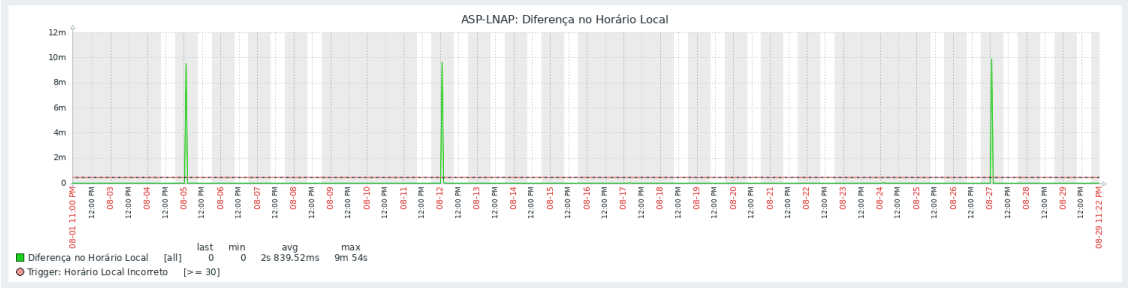


Figura 26 – Demonstração de exemplo de diferença de horário no mês de agosto de 2025

Estas diferenças temporais ocasionavam incidentes operacionais, dado que certas tarefas eram executadas fora do momento previsto, o que poderia gerar falhas ou impactos indiretos noutros serviços. Para mitigar este risco, foi criada uma métrica destinada a monitorizar continuamente a diferença entre o horário do servidor e o horário local, possibilitando a sua correção sempre que necessário. O item é obtido através de um script em Bash com recurso a WMI, executado de hora em hora.

```
#!/bin/bash

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT Year, Month, Day, Hour, Minute,
Second FROM Win32_LocalTime"
```

Código 29 – *Script* “get_localtime.sh”

Caso a diferença detetada seja superior a 30 segundos, é gerado um alarme de severidade “Desastre”, de forma a alertar a equipa para a necessidade de intervenção.

O *script* retorna apenas o horário local da máquina em questão. O resultado inicial do *script*, ilustrado na figura 27, consiste em duas linhas: uma relativa ao cabeçalho e outra contendo os valores de cada campo.

```
arsopi@lnnetmon:/usr/lib/zabbix/externalscripts$ ./get_localtime.sh
| Day | Hour | Minute | Month | Second | Year |
| 1 | 16 | 9 | 9 | 24 | 2025 |
arsopi@lnnetmon:/usr/lib/zabbix/externalscripts$
```

Figura 27 – Demonstração do resultado obtido via *script* sem tarefa de pré-processamento

O processamento lógico subsequente é realizado diretamente no Zabbix, através de uma tarefa de pré-processamento em JavaScript. Esta transforma estas linhas num valor numérico único, correspondente à diferença horária em segundos.

```
function removerPrimeiraLinha(str){
    const colunas = str.split(/\s{2,}/);
    const dadosRestantes = colunas.slice(6);
    return dadosRestantes.join(' ');
}

const resultado1 = value.replace(/\|/g, ' ');
const resultado = removerPrimeiraLinha(resultado1.trim());
const array = resultado.split(/\s+/);

const day = array[0];
const hour = array[1];
const minute = array[2];
const month = array[3];
const second = array[4];
const year = array[5];

const date = new Date(year, month - 1, day, hour, minute, second);

const timestampInSeconds = Math.floor(date.getTime() / 1000);

const now = new Date();
const currentTimestampInSeconds = Math.floor(now.getTime() / 1000);

return Math.abs(timestampInSeconds - currentTimestampInSeconds);
```

Código 30 – Código JavaScript de pré-processamento do item de diferença de horário

O código em JavaScript, apresentado no Código 30, executa as seguintes operações:

- Substitui todos os caracteres "|" por espaços;
- Remove a primeira linha da *string* (cabeçalho), recorrendo à função "removerPrimeiraLinha" já descrita em métricas anteriores;
- Divide a *string* resultante por espaços, obtendo um *array* com os valores necessários;
- Armazena os valores do *array* em variáveis auxiliares;
- Cria um objeto correspondente à data atual do servidor, aplicando a lógica de mês *zero-based* do JavaScript (janeiro = 0, fevereiro = 1, etc.);
- Converte a data em segundos desde 1 de janeiro de 1970, utilizando a função "getTime()" e dividindo o resultado por 1000, com arredondamento através de "Math.floor()";
- Por fim, obtém novamente a data atual em segundos e devolve o valor absoluto da diferença entre os dois registos temporais.

Desta forma, a métrica permite identificar rapidamente discrepâncias relevantes entre o horário local e o do servidor, assegurando a consistência temporal necessária ao correto funcionamento das tarefas agendadas.

3.6.3.1.11 Informações do Event Viewer (Utilizadores e/ou Grupos)

Esta métrica foi introduzida no Zabbix na sequência de uma auditoria interna de segurança, a qual identificou como ponto crítico a inexistência de monitorização de eventos relacionados com a gestão de contas de utilizadores e grupos do domínio. A sua implementação teve como objetivo reforçar os mecanismos de controlo, permitindo detetar tentativas de adição, alteração ou remoção de utilizadores e/ou grupos. Desta forma, é possível identificar rapidamente situações potencialmente anómalas, como a inclusão indevida de um utilizador num grupo com permissões elevadas, sabendo quem realizou a ação e quando ocorreu.

Para viabilizar esta monitorização, foi necessário proceder inicialmente à configuração do registo de eventos no Windows. A partir de um dos servidores que funciona como Domain

Controller, acedeu-se ao “Group Policy Management”, selecionaram-se as políticas aplicáveis e, no editor correspondente, ativaram-se as seguintes opções no caminho:

“Computer Configuration → Policies → Windows Settings → Security Settings → Advanced Audit Policy Configuration → Audit Policies → Account Management”

Foram então configuradas as subcategorias “Audit Security Group Management” e “Audit User Account Management”, ativando-se as opções de auditoria para “Success” e “Failure”. Com esta configuração, os eventos passam a ser registados no Visualizador de Eventos, em “Windows Logs → Security”.

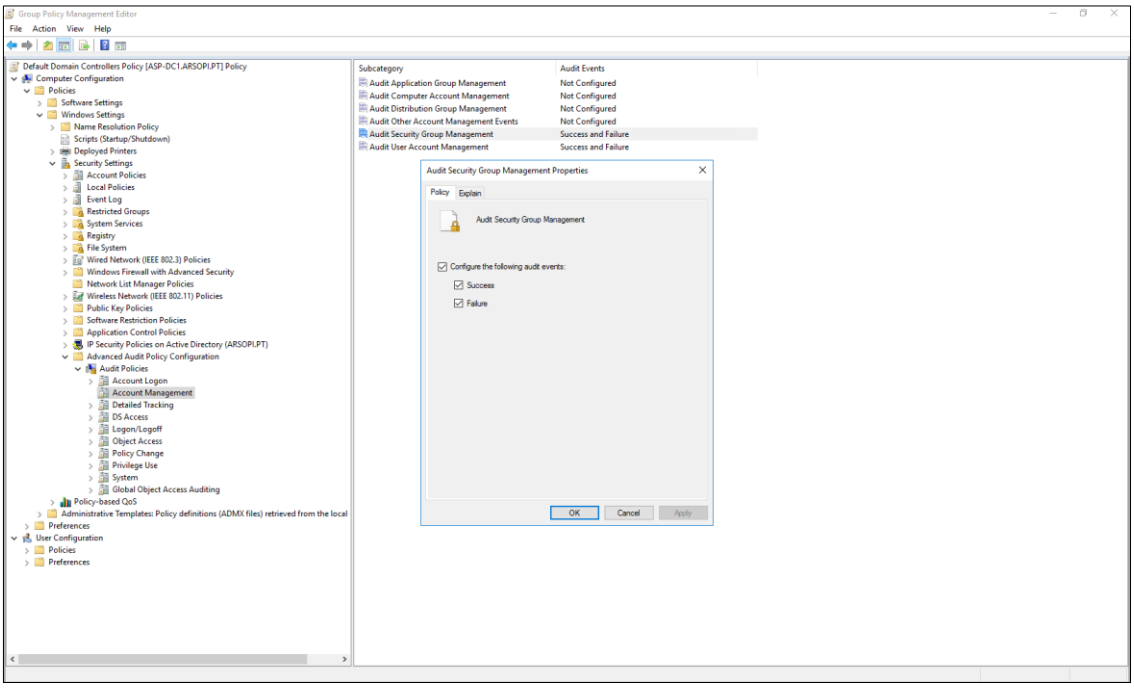


Figura 28 – Demonstração da ativação do registo de eventos relacionados com utilizadores e grupos

Concluída esta configuração, foi desenvolvida um *script* em Bash com recurso a WMI, responsável por recolher os registos e transmiti-los ao Zabbix. O item criado é do tipo “External check”, executado a cada 5 minutos, e devolve uma resposta em formato textual que corresponde ao conteúdo registado no Visualizador de Eventos.

```
#!/bin/bash

last_minutes=$(date --utc +"%Y%m%d%H%M%S.000000-000" --date="5 minutes ago")

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD} @$1 -query "SELECT
EventType,RecordNumber,TimeGenerated,Message,SourceName,EventCode FROM
Win32_NTLogEvent WHERE LogFile = 'Security' AND (EventType = 4 OR EventType
= 5) AND ((EventCode >= 4720 AND EventCode <= 4740 AND EventCode <> 4721 AND
EventCode <> 4736 AND EventCode <> 4739 AND EventCode <> 4723) OR EventCode
= 4767 OR EventCode = 4780) AND TimeGenerated >= '$last_minutes'"
```

Código 31 – *Script* "get_eventviewer_info.sh"

O excerto de código desenvolvido é semelhante ao utilizado no item relativo a erros do Visualizador de Eventos, mas com algumas adaptações:

- Inclusão do campo "tipo de evento", permitindo distinguir entre eventos de sucesso (código 4) e de falha (código 5);
- Verificação explícita de que a informação é extraída do ficheiro de registos de segurança;
- Filtragem por data e por códigos de evento relevantes, previamente identificados através de um estudo que determinou quais os eventos de interesse para monitorização.

Tabela 2 - Códigos relacionados com utilizadores

Código	Descrição do Código
4720	Uma conta de utilizador foi criada.
4722	Uma conta de utilizador foi ativada.
4724	Foi feita uma tentativa de redefinir a senha de uma conta (admin)
4725	Uma conta de utilizador foi desativada.
4726	Uma conta de utilizador foi removida.
4738	Uma conta de utilizador foi alterada.
4740	Uma conta de utilizador foi bloqueada.
4767	Uma conta de utilizador foi desbloqueada.
4780	A ACL foi definida em contas que são membros de grupos de administradores.

Tabela 3 - Códigos relacionados com grupos

Código	Descrição do Código
4727	Um grupo global habilitado para segurança foi criado.
4728	Um membro foi adicionado a um grupo global habilitado para segurança.
4729	Um membro foi removido de um grupo global habilitado para segurança.
4730	Um grupo global habilitado para segurança foi removido.
4731	Um grupo local habilitado para segurança foi criado.
4732	Um membro foi adicionado a um grupo global habilitado para segurança.
4733	Um membro foi removido de um grupo global habilitado para segurança.
4734	Um grupo local habilitado para segurança foi removido.
4735	Um grupo local habilitado para segurança foi alterado.
4737	Um grupo global habilitado para segurança foi alterado.

Estes códigos de evento são os que se encontram atualmente sob vigilância, permitindo que a equipa seja alertada sempre que ocorram alterações relevantes ao nível de utilizadores e grupos de domínio (Microsoft, 2025).

Foi ainda configurado um alarme de severidade “Desastre”, acionado sempre que o resultado devolvido pelo *script* for diferente de vazio, isto é, sempre que sejam detetados eventos correspondentes. Este alarme gera notificações por correio eletrónico, garantindo que nenhuma destas ocorrências passe despercebida. A métrica foi aplicada aos dois Domain Controllers da infraestrutura: ASP-DC1 e ASP-DC2, assegurando uma cobertura completa e redundante deste tipo de eventos críticos.

3.6.3.1.12 Tentativa de Alteração da *Password* por parte do Utilizador

Este item foi desenvolvido a partir da métrica relativa às alterações de utilizadores e grupos do domínio, uma vez que se considerou útil monitorizar especificamente os eventos de troca de *password*. Na organização, encontra-se em vigor uma política de segurança que obriga os utilizadores a alterar a sua palavra-passe a cada seis meses. Contudo, verificou-se que, por vezes,

este processo origina problemas colaterais, como falhas em atualizações de sistema ou indisponibilidade temporária de determinados serviços. Assim, a monitorização deste evento permite à equipa de IT antecipar potenciais queixas dos utilizadores e intervir de forma mais proativa.

Este item foi configurado em separado do anterior, dado que, na perspetiva da equipa, não possui a mesma criticidade. Por esse motivo, foi atribuído ao alarme um grau de severidade "Alto", em vez de "Desastre", evitando a receção de notificações por correio eletrónico que poderiam introduzir ruído excessivo, considerando o número potencialmente elevado de ocorrências deste tipo de evento.

```
#!/bin/bash

last_minutes=$(date --utc +"%Y%m%d%H%M%S.000000-000" --date="5 minutes ago")

/home/zabbix/myenv/bin/python /home/zabbix/myenv/bin/wmiquery.py
{UTILIZADOR}:{PASSWORD}@$1 -query "SELECT
EventType,RecordNumber,TimeGenerated,Message,SourceName,EventCode FROM
Win32_NTLogEvent WHERE LogFile = 'Security' AND (EventType = 4 OR EventType
= 5) AND (EventCode = 4723) AND TimeGenerated >= '$last_minutes'"
```

Código 32 – Script "get_password_change_info.sh"

O código do *script* utilizado é bastante semelhante ao do item anterior, diferindo apenas na filtragem por código de evento. Neste caso, são considerados exclusivamente os eventos com o código "4723", correspondentes a alterações de *password*. A métrica é do tipo "External check", executada com uma periodicidade de 5 minutos. O alarme é acionado sempre que o resultado da execução do *script* for diferente de vazio, indicando que ocorreu pelo menos uma alteração de *password* no intervalo considerado.

3.6.3.1.13 Lista de Utilizadores VPN SSL

Este item foi introduzido na *firewall*, como complemento ao *template* "FortiGate by SNMP". Embora o *template* já disponibilizasse a métrica relativa ao número de utilizadores ligados via VPN SSL, considerou-se relevante melhorar esta funcionalidade, passando a obter também a lista detalhada de utilizadores ativos, de modo a permitir que a equipa de administração tivesse

acesso, em tempo real, à informação mais completa sobre quem se encontra conectado à rede corporativa.

Para viabilizar esta métrica foi necessário recorrer à API da *firewall*, uma vez que a informação pretendida não se encontrava disponível por SNMP. O processo envolveu uma configuração semelhante à descrita anteriormente para o SNMP, nomeadamente:

- ativação das conexões HTTPS (já previamente configuradas);
- criação de um utilizador com permissões restritas para leitura de informação relativa à VPN;
- definição de um perfil administrativo denominado “VPN_Reader”;
- criação de uma REST API Admin associada a este perfil, com restrição de acesso (“Trusted host”) apenas ao servidor do Zabbix.

Da criação deste utilizador resultou um *token* de autenticação, utilizado posteriormente pelos *scripts* para obtenção da informação.

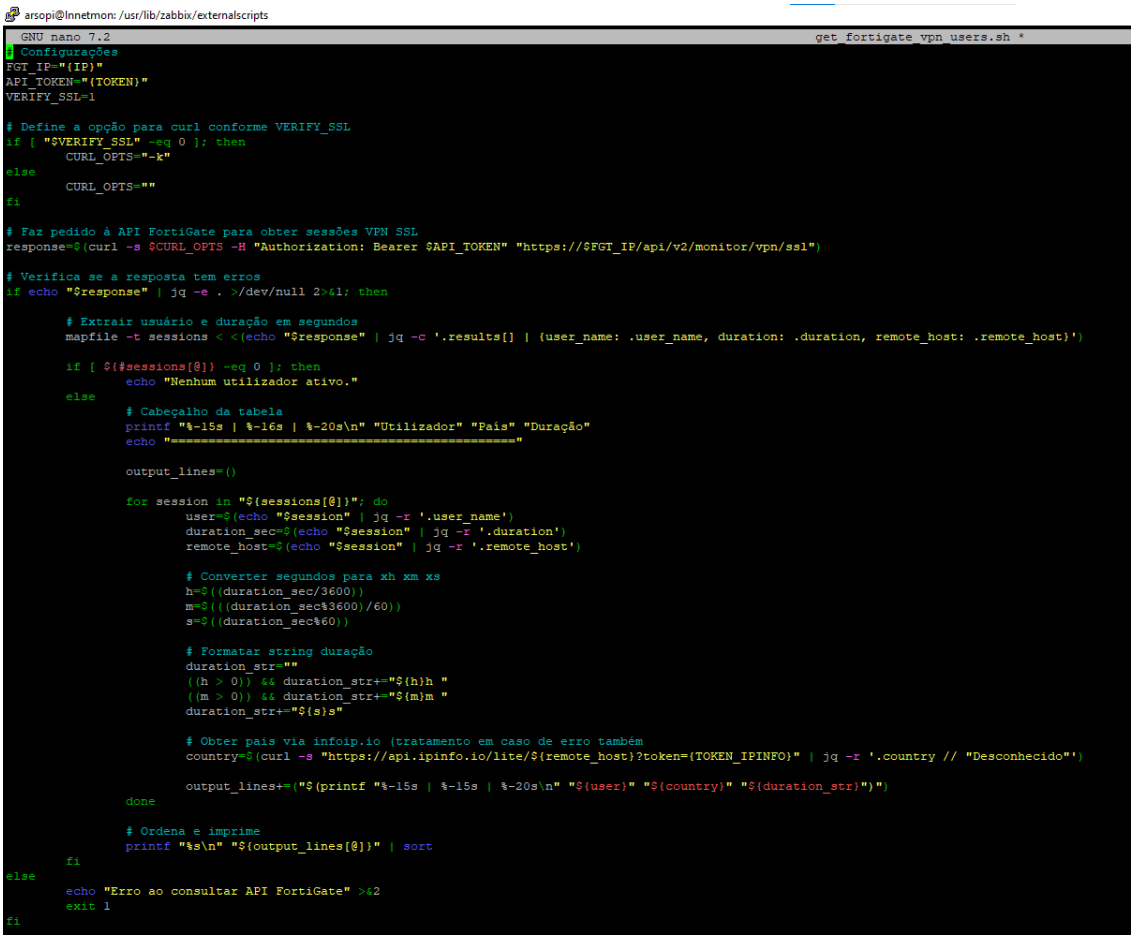


Figura 29 – Demonstração do script “get_fortigate_vpn_users.sh”

O objetivo deste script consiste em recolher, diretamente na *firewall*, a lista de utilizadores conectados por VPN SSL e disponibilizar essa informação ao Zabbix sob a forma de uma tabela. Para tal, o script define inicialmente as variáveis necessárias à autenticação na API da *firewall* e, em seguida, consulta o endereço “https://$FGT_IP/api/v2/monitor/vpn/ssl”.

A resposta devolvida pela API é armazenada num *array* de sessões (*sessions*), no qual cada entrada corresponde a um utilizador ativo. Para cada sessão são recolhidos três elementos, sendo eles o nome de utilizador, a duração da ligação e o endereço IP remoto. Após a recolha, o *script* executa uma verificação:

- Se não existirem utilizadores ativos, devolve a mensagem “Nenhum utilizador ativo”;
- Caso existam, gera uma tabela formatada onde, para cada utilizador, a duração da ligação é convertida para o formato “hh:mm:ss” e o IP remoto é traduzido para o país de origem da ligação através de uma API de geolocalização.

Posteriormente, foi criado no Zabbix o item correspondente, definido como verificação externa de periodicidade de 1 minuto. Considerando que se trata de um indicador de carácter informativo, este item não possui alarme associado, funcionando como ferramenta de consulta em tempo real para os administradores de rede.

A informação final obtida no Zabbix é a seguinte:

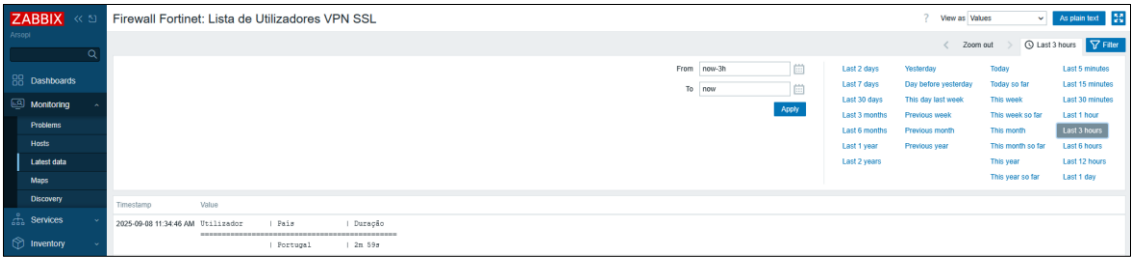


Figura 30 - Demonstração da lista de utilizadores VPN SSL

3.6.3.1.14 Lista de Utilizadores VPN IPsec

De forma análoga ao item referente à VPN SSL, esta métrica surgiu como melhoria relativamente ao *template* da *firewall*, o qual já disponibilizava a contagem do número de sessões IPsec ativas. O objetivo passou a ser mais detalhado, consistindo na obtenção da lista de conexões IPsec em tempo real, em vez de apenas um valor agregado.

Tirando partido do trabalho já realizado para o item anterior, foi necessário apenas desenvolver um novo *script*, denominado "get_fortigate_ipsec_connections.sh", atribuir-lhe as permissões adequadas e criar o item correspondente no Zabbix. A estrutura do *script* é praticamente idêntica, alterando-se apenas o *endpoint* da API utilizada, sendo neste caso "https://$FGT_IP/api/v2/monitor/vpn/ipsec", e os parâmetros processados para cada conexão.

Um aspeto particular das ligações IPsec é que, em alguns casos, não existe nome de utilizador associado à sessão. Nestes cenários, o campo é substituído pelo nome da conexão, de forma a evitar valores em branco e, sobretudo, garantir que a equipa de administração identifica corretamente a origem da ligação. Este comportamento é particularmente útil em túneis VPN entre empresas, como aqueles criados para aceder a recursos das restantes empresas do grupo.

No Zabbix, restou apenas configurar o item “Lista de Utilizadores VPN IPsec”, do tipo “External check”, com a chave “get_fortigate_ipsec_connections.sh[{HOST.IP}]” e com um intervalo de tempo de um minuto igualmente, devolvendo a informação no formato textual, com uma tabela idêntica à do item anterior. Tal como o item relativo à VPN SSL, também este não possui *trigger* associado, funcionando exclusivamente como ferramenta de consulta para a equipa de IT, que acompanha diariamente os *widgets* associados.

3.6.4 Notificações e alertas

As notificações geradas aquando da ocorrência de um incidente são das funcionalidades mais importantes e úteis que esta solução do Zabbix oferece aos seus utilizadores. Para além dos *widgets* disponíveis onde se pode ver com facilidade os problemas existentes nas máquinas/serviços monitorizados, oferece várias formas de receber alertas personalizados.

Para o nosso caso, optou-se por configurar o Zabbix de forma que os membros responsáveis pela administração da rede recebam um email alertando-os acerca do acontecimento de problemas, ajudando a equipa a ter uma reação mais rápida e minimizando possíveis consequências.

Desta forma, para se conseguir definir os alertas enviados por email, foram necessários alguns passos de configuração que envolvem a criação e configuração de um “media type”, configuração da parte dos utilizadores, respetivos emails e condições na parte das notificações e a criação de uma “action” que trata de fazer o envio da notificação para os meios configurados. O passo que falta neste “fluxo” é a criação de *triggers* que, ao serem disparados, originem problemas, sendo que esta parte já foi configurada anteriormente na parte dos itens e métricas.

3.6.4.1 Criação de Media Type

O processo responsável por fazer a criação e configuração de uma forma de envio de notificação que neste caso se trata de email. Para isso, foi necessário ir, na aba lateral do menu do Zabbix, a “Alerts” e dentro deste submenu, a “Media Types”. Clicou-se, no canto superior direito em “Create Media Type” e procedeu-se ao preenchimento dos campos, dando o nome “Email”,

metendo o servidor SMTP da empresa, assim como a porta, algo parecido com o que é possível ver na figura 31.

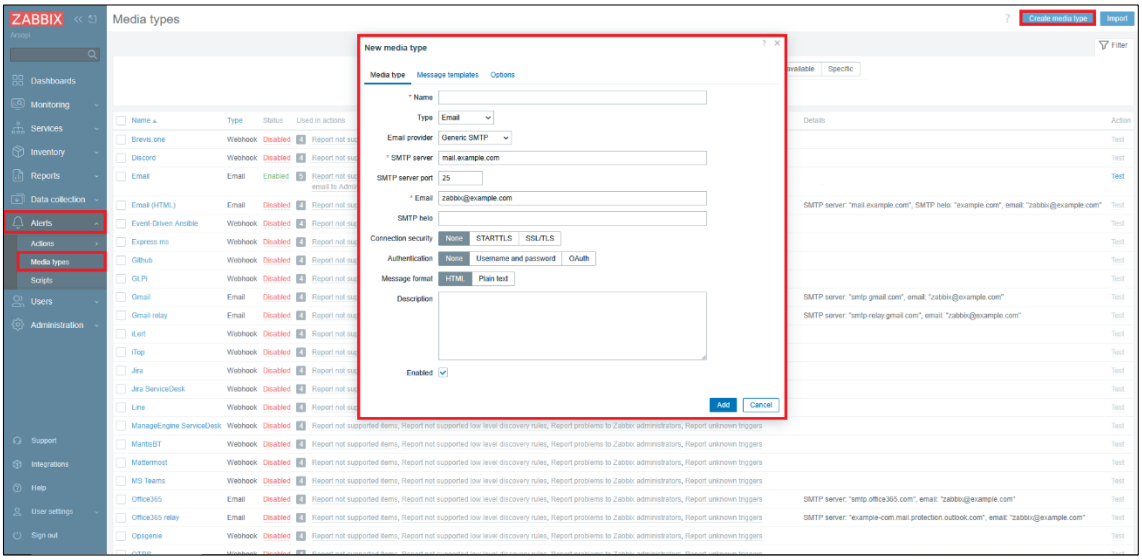


Figura 31 – Demonstração, por passos, da criação de um "Media Type"

De seguida, adicionaram-se os "Message Templates" necessários. Neste caso, adicionaram-se todos os *templates* da figura 32, apesar de apenas o *template* do problema em si estar em uso, uma vez que apenas se pretende receber emails acerca de problemas e nada mais.

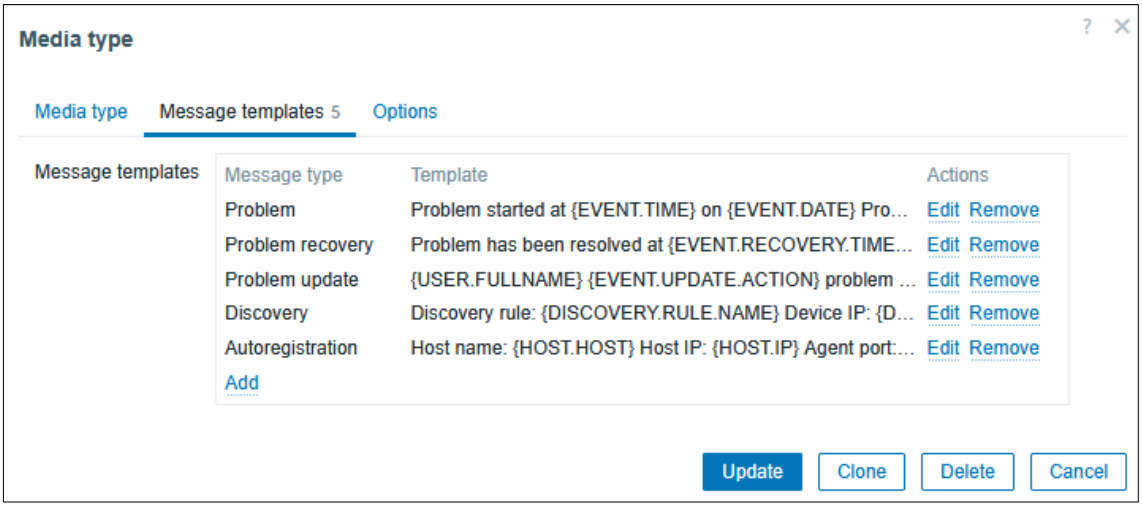


Figura 32 – "Message templates" adicionados ao "Media Type"

Este *template* tem o conteúdo da mensagem, que foi deixado como vem pré-definido, visto que cumpre as nossas necessidades de perceber o problema em questão e de partir para uma resolução o mais rápido possível:

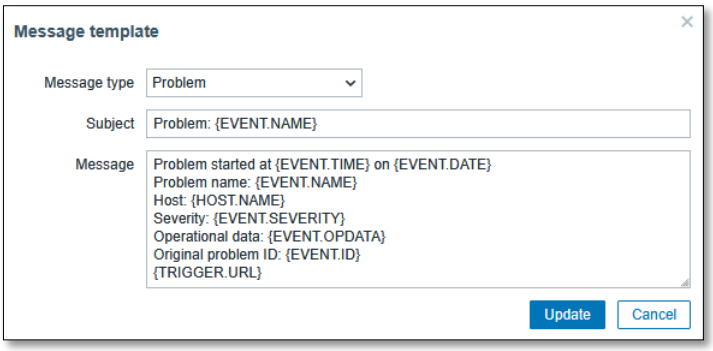


Figura 33 – *Template* da mensagem para o tipo "Problema"

3.6.4.2 Configuração dos utilizadores

Após a criação do "*media type*", visto em cima, foi necessário configurar os utilizadores que irão receber as notificações via email. Para isso, foi necessário ir ao submenu "*User settings*", à página "*Notifications*". Nesta página, na aba "*Media*", é possível configurar o "*media type*" desejado, o endereço de email, a altura do dia em que se pretende receber notificações e o tipo de notificações que se pretende receber de acordo com a severidade do problema.

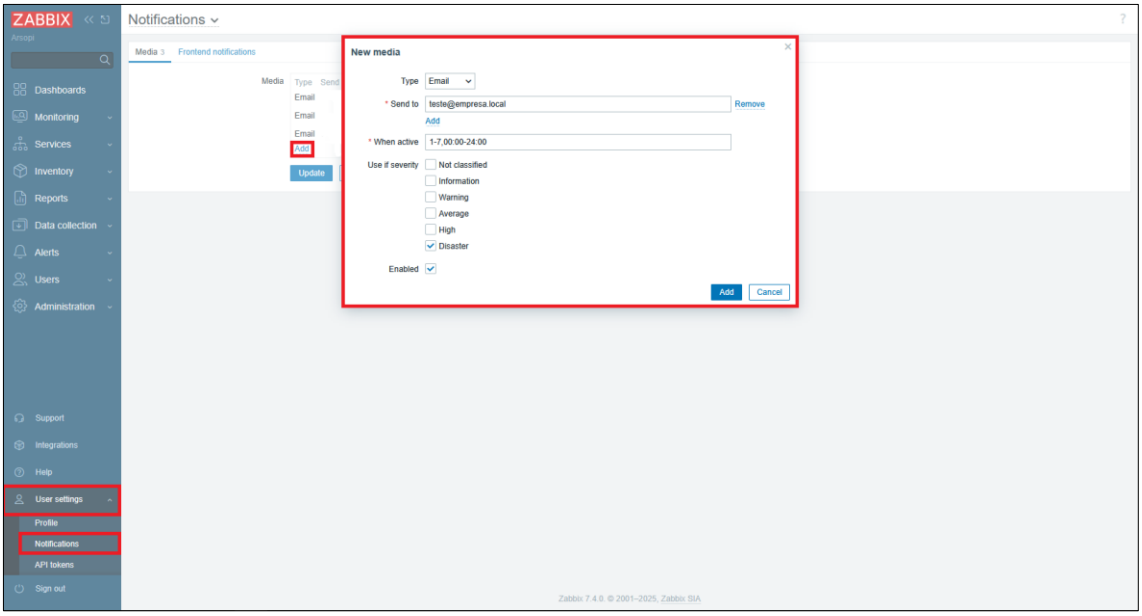


Figura 34 – Demonstração do processo de criação de uma nova “media”

Aqui, foram adicionados todos os membros pertencentes à equipa de administração da rede, garantindo que todos estejam a par dos incidentes na rede e possam atuar o mais rápido possível. Decidiu-se que todos os utilizadores apenas recebem emails para problemas de severidade “Desastre”. Optou-se por configurar desta forma para evitar o recebimento de excesso de emails que possam não ser tão importantes e para que os emails recebidos sejam apenas os mais importantes e que necessitam de atenção da nossa parte. Por esta razão, a maior parte dos itens que, da nossa perspetiva, são essenciais ao bom funcionamento da rede, têm *triggers* associados que, ao serem disparados, originam problemas desta severidade, enviando emails para os administradores da rede. Os outros itens que possam não ser tão importantes ou podem originar problemas mais frequentemente sem ser preciso uma ação imediata, são considerados com severidades menores e aparecem apenas no *dashboard* que será demonstrado e analisado posteriormente.

3.6.4.3 Configuração da ação

Por último, é necessário configurar a ação que envia os emails para os utilizadores definidos anteriormente, com as condições impostas e com os *templates* configurados.

Para isso, foi necessário ir a “Alerts”, “Actions” e aceder à página “Trigger actions”. Nesta página, adicionou-se uma nova action, clicando no botão presente no canto superior direito, deu-se o nome à mesma, não se adicionaram condições, visto que as condições já estão definidas na parte dos utilizadores, e é a forma mais fácil e direta de criar condições de envio de email de acordo com a severidade.

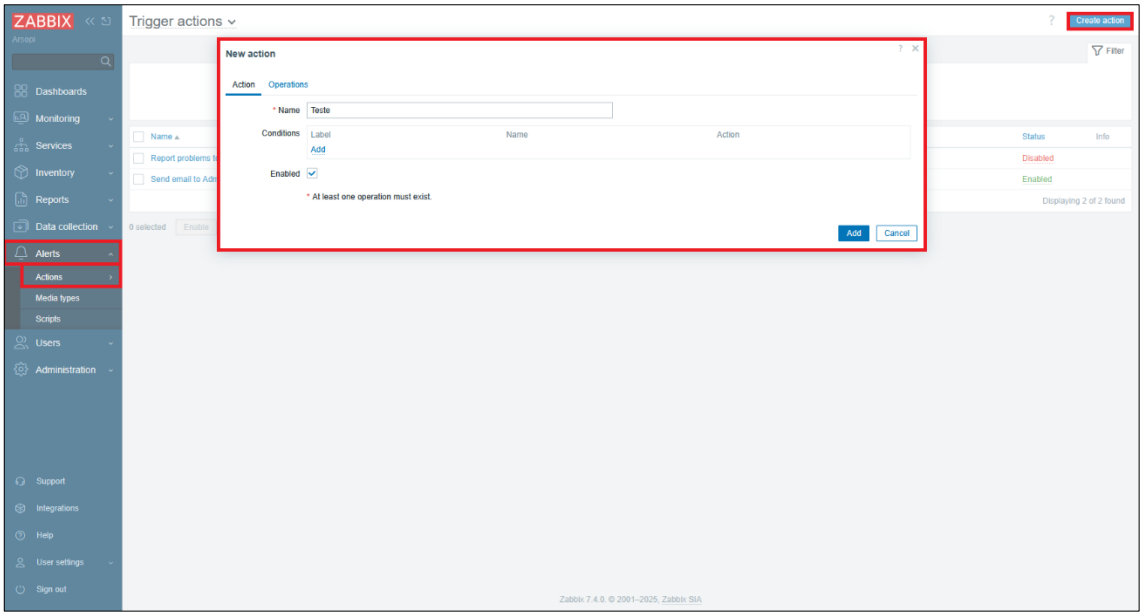


Figura 35 – Demonstração do processo de criação de uma nova “action”

Antes de adicionar a ação, foi necessário configurar as operações. Para o nosso caso, foi apenas necessário adicionar uma operação, sendo essa operação de envio de mensagem. Através dos campos de “Step”, foi definido que o email apenas é enviado para cada endereço uma vez e é imediatamente após o *trigger* ser disparado. Esta mensagem é do tipo “Email” e é enviada para o utilizador Admin (Zabbix Administrator). De forma a clarificar conceitos, todos os emails configurados no ponto anterior, são emails associados a este utilizador, o utilizador Admin, que é o utilizador usado pela nossa equipa.

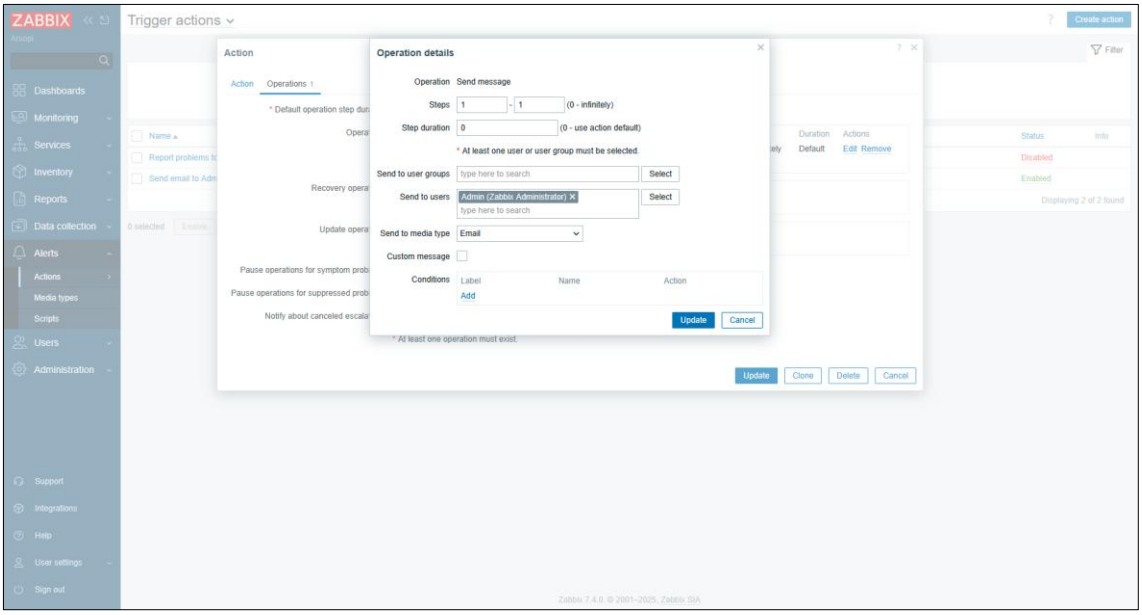


Figura 36 – Demonstração da configuração dos detalhes da operação associada a ação

Conclui-se assim o processo de configuração de envio de notificações via email para os utilizadores da equipa de administração da rede. O resultado é o recebimento de um email semelhante a este:

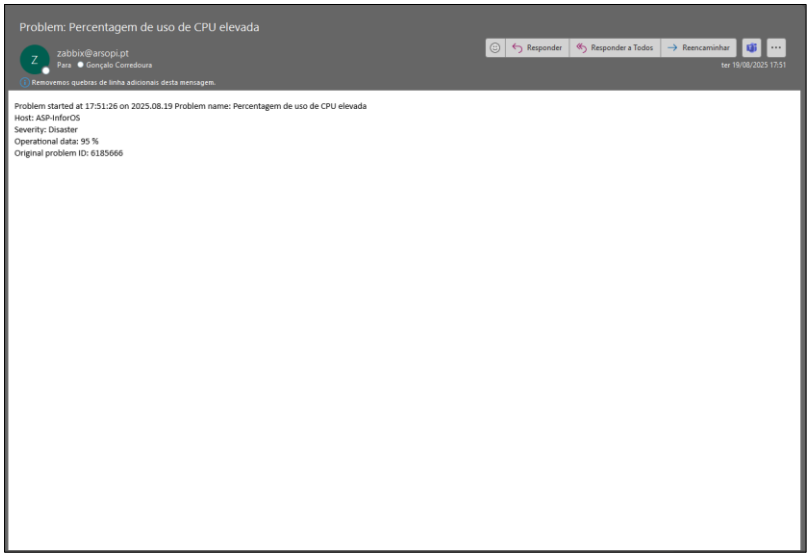


Figura 37 – Exemplo de email recebido após ser gerado um problema de cariz “Desastre”

3.6.5 Dashboards personalizados

Uma das funcionalidades mais relevantes do Zabbix é a possibilidade de criação e personalização de *dashboards*, que permitem uma visualização centralizada e intuitiva do estado da infraestrutura. A conceção de um *dashboard* claro, objetivo e que contenha apenas a informação essencial constitui um desafio, mas traz vantagens significativas em termos de rapidez na identificação e resolução de incidentes, bem como na capacidade de análise proativa.

Embora o Zabbix disponibilize um *dashboard* pré-definido após a instalação, este não satisfazia os requisitos definidos pela equipa de IT. Assim, foi desenvolvido um *dashboard* personalizado, com o objetivo de apresentar de forma simples e eficaz as métricas mais relevantes para o funcionamento da rede e dos sistemas críticos da organização.

3.6.5.1 *Dashboard* Arsopi

O *dashboard* criado, designado "Arsopi", foi estruturado em cinco páginas distintas, cada uma com um propósito específico:

- **Página "Geral"**: apresenta os principais *widgets*, oferecendo uma visão global dos problemas identificados em toda a rede.
- **Página "Small Info"**: disponibiliza métricas de CPU, memória e tráfego de rede de todos os servidores, bem como o tráfego de rede agregado dos *switches*.
- **Página "Server Page"**: fornece informação detalhada relativa a um servidor específico, incluindo todas as métricas já referidas (CPU, memória, discos, tráfego de rede, entre outras).
- **Página "Switch Page"**: semelhante à anterior, mas orientada para os *switches*. Apresenta tanto uma visão geral do tráfego de rede de cada dispositivo como uma análise detalhada por porta/interface de um *switch* selecionado.
- **Página "Firewall"**: contém *widgets* dedicados à monitorização da *firewall*, permitindo acompanhar métricas críticas associadas à sua operação.

Um aspeto relevante desta configuração, ilustrado nas figuras 42 e 43, é a possibilidade de selecionar um *host* específico através de um parâmetro disponível no canto superior direito do *dashboard*. Quando um *host* é selecionado, os *widgets* são automaticamente atualizados de acordo com o parâmetro "Override Host", apresentando apenas informação relativa ao dispositivo escolhido. Esta funcionalidade acrescenta flexibilidade à análise, permitindo transitar facilmente de uma visão global para uma visão detalhada.

3.6.5.1.1 Página "Geral"

Como se pode observar na figura 38, o *dashboard* "Geral" foi concebido para fornecer uma visão centralizada dos principais componentes da infraestrutura, permitindo uma identificação imediata de problemas críticos. A primeira coluna apresenta informação relacionada com a *firewall*, incluindo métricas de tráfego de rede, percentagem de utilização de CPU e memória, bem como o número de utilizadores ligados por VPN SSL e IPsec. Através de um atalho funcional, ao clicar no título "Firewall", o utilizador é automaticamente redirecionado para o *dashboard* dedicado à *firewall*, onde é possível consultar informação mais detalhada. As colunas seguintes apresentam, respetivamente, listas de servidores, *access points*, *switches* e nGs. Caso algum destes dispositivos apresente anomalias, surge junto ao seu nome um ícone de alerta, cuja cor corresponde ao nível de severidade do problema identificado. Na parte inferior da página encontra-se um *widget* adicional que lista todos os problemas ativos, incluindo detalhes como a hora da ocorrência, o estado, o *host* afetado e a severidade associada. Este *widget* facilita ainda o acesso direto ao gráfico da métrica ou à informação adicional do problema, permitindo a sua análise e resolução de forma célere.

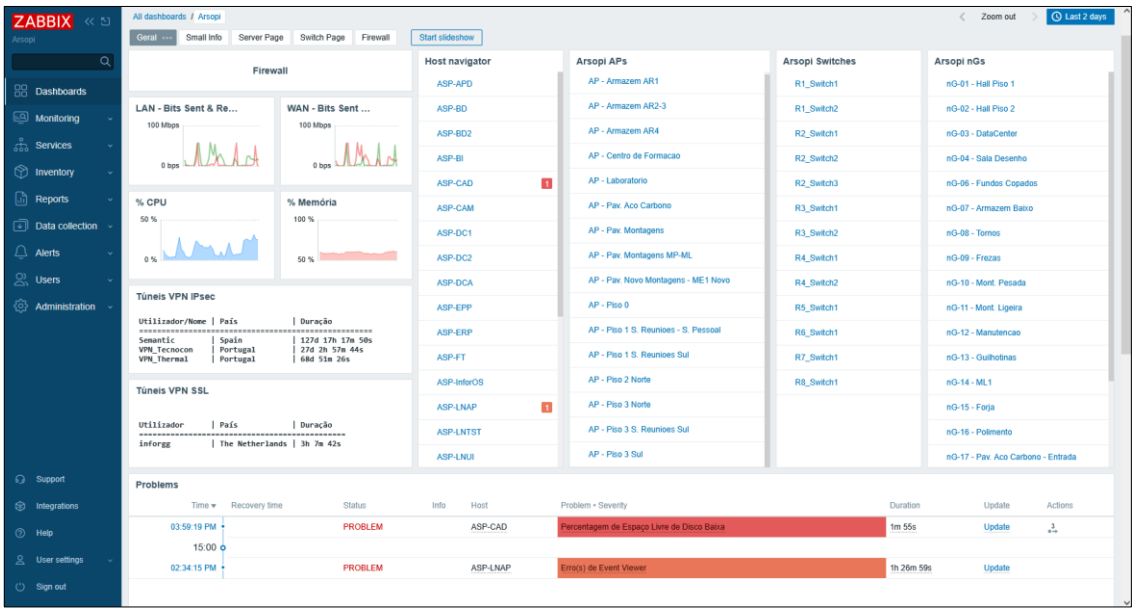


Figura 38 – Demonstração da página “Geral” do *dashboard* “Arsopi”

De modo a aumentar a eficácia da monitorização e reduzir informação redundante, foi definida a regra de apenas exibir problemas classificados como “Elevado” ou “Desastre”, ocultando ocorrências de menor impacto que poderiam introduzir ruído e desviar a atenção da equipa.

Outra medida importante adotada consistiu na não resolução automática de problemas. Por norma, o Zabbix fecha automaticamente um problema quando a métrica retorna a valores normais. No entanto, essa abordagem poderia induzir em erro, uma vez que problemas intermitentes poderiam desaparecer do *dashboard* sem que a equipa tivesse conhecimento da sua ocorrência (por exemplo, durante um fim de semana). Para evitar esta situação, foi implementada a resolução manual obrigatória: apenas a equipa de IT pode encerrar um problema após confirmar que este foi efetivamente resolvido. Esta prática assegura um histórico mais fidedigno e uma maior consciência situacional acerca do estado real da rede.

3.6.5.1.2 Página “Small Info”

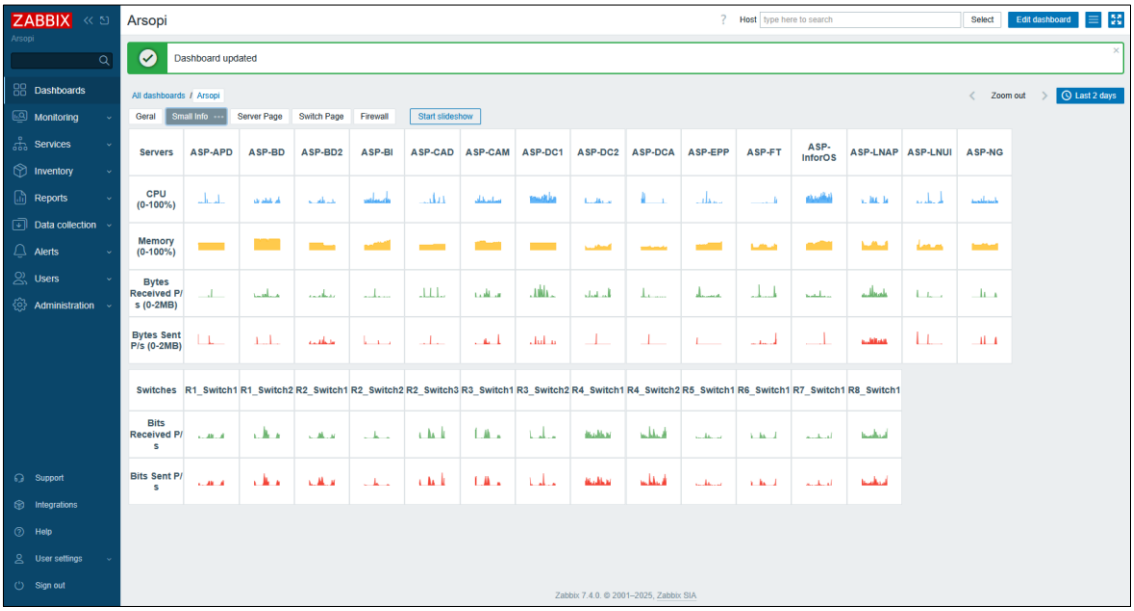


Figura 39 – Demonstração da página “Small Info” do *dashboard* “Arsopi”

A página “Small Info” foi concebida com um caráter intencionalmente simples, apresentando de forma compacta a informação considerada mais relevante relativa a servidores e *switches*. O objetivo é proporcionar uma visão sintética e imediata sobre o estado geral destes equipamentos, sem sobrecarregar a equipa com dados secundários. Os gráficos disponibilizados permitem uma análise rápida, possibilitando a identificação de tendências de utilização. Ao passar o cursor sobre cada gráfico, o utilizador tem acesso ao valor exato da métrica e ao respetivo momento temporal (dia e hora). Esta funcionalidade acrescenta detalhe sem comprometer a simplicidade da página. Adicionalmente, foi implementada uma funcionalidade de navegação direta: ao clicar sobre o nome de um servidor ou de um *switch*, o utilizador é encaminhado para a respetiva “Server Page” ou “Switch Page”, onde se encontra disponível informação mais detalhada e segmentada para o dispositivo selecionado.

3.6.5.1.3 Página “Server Page”

A “Server Page” foi desenvolvida com o objetivo de disponibilizar informação detalhada acerca de um servidor específico, permitindo uma monitorização adaptada às características individuais de cada máquina.

A página está configurada para apresentar ou ocultar *widgets* de forma dinâmica, em função das métricas disponíveis para o servidor selecionado. Por exemplo, ao selecionar o servidor ASP-APD, não são exibidos gráficos relativos a serviços de SQL, CAD ou CAM, uma vez que essas métricas não lhe são aplicáveis. Em contrapartida, surge o gráfico referente ao Spooler, métrica que se encontra ativa neste servidor.

Este comportamento dinâmico garante que a “Server Page” apresenta sempre informação relevante e contextualizada, evitando a exibição de dados desnecessários e facilitando a análise específica de cada servidor.

Figura 40 – Demonstração da página “Server Page” do *dashboard* “Arsopi” (1/2)

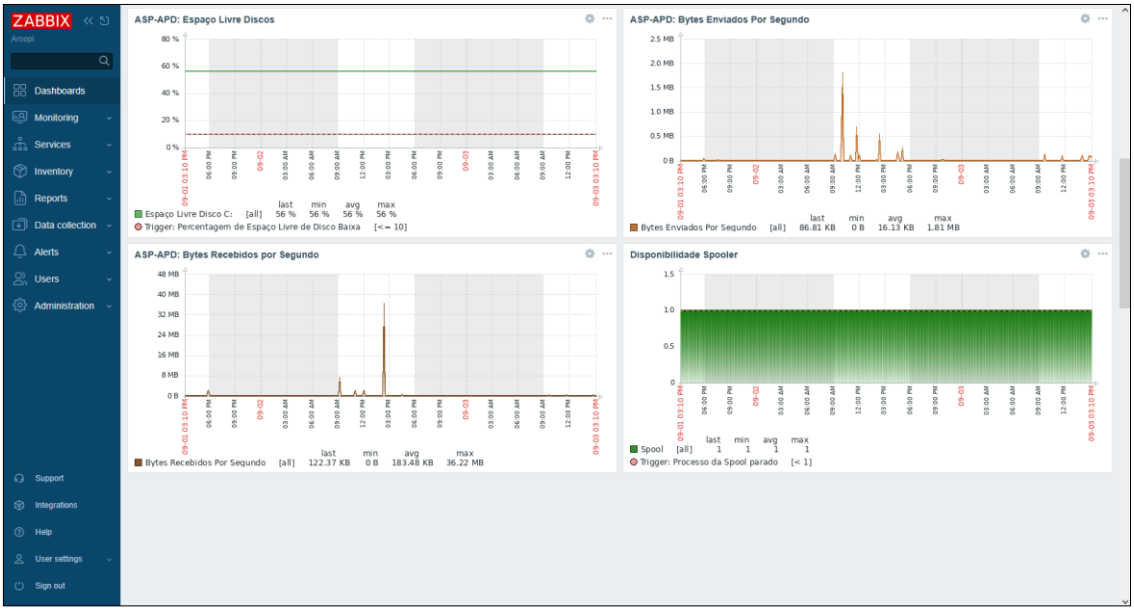


Figura 41 - Demonstração da página “Server Page” do *dashboard* “Arsopi” (2/2)

3.6.5.1.4 Página “Switch Page”

A “Switch Page” apresenta uma lógica semelhante à da “Server Page”, mas orientada para a monitorização de dispositivos de rede. Esta página foi concebida para disponibilizar informação útil tanto quando não está selecionado nenhum *host* como quando um *switch* específico é escolhido.

- Sem *host* selecionado: a página apresenta um gráfico de tráfego de rede para cada *switch* da infraestrutura, permitindo visualizar os bits enviados e recebidos por cada dispositivo. Esta configuração fornece uma visão global do desempenho de todos os *switches* em simultâneo.

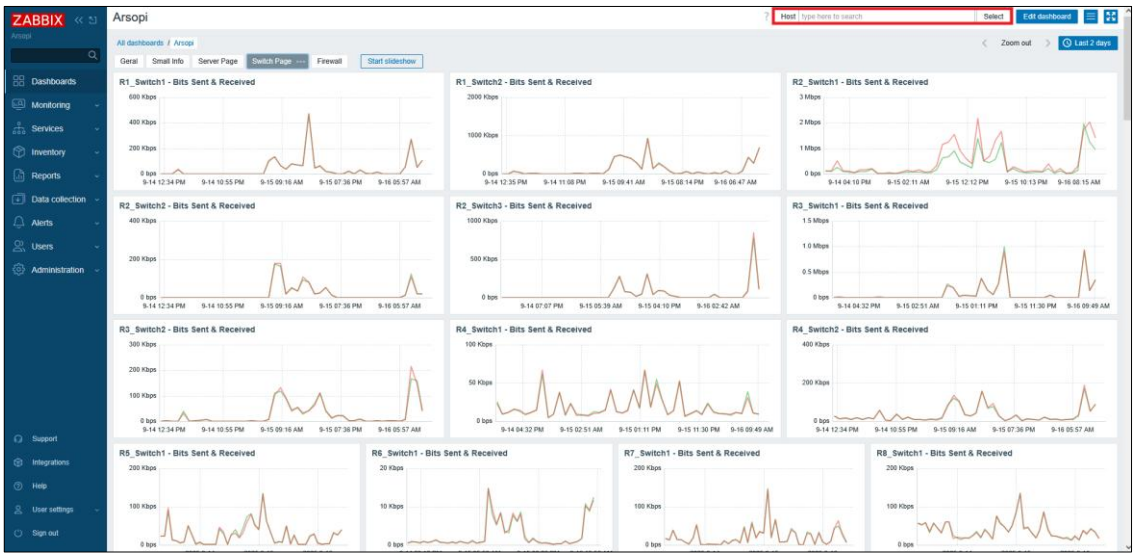


Figura 42 - Demonstração da página “Switch Page” quando nenhum *host* está selecionado

- Com *host* selecionado: neste caso, são exibidos dois gráficos gerais relativos ao *switch* escolhido — um para os bits enviados e outro para os bits recebidos. Seguidamente, são apresentados os gráficos de tráfego de rede referentes a cada interface ativa do *switch*. As interfaces sem tráfego associado permanecem ocultas, evitando informação redundante.

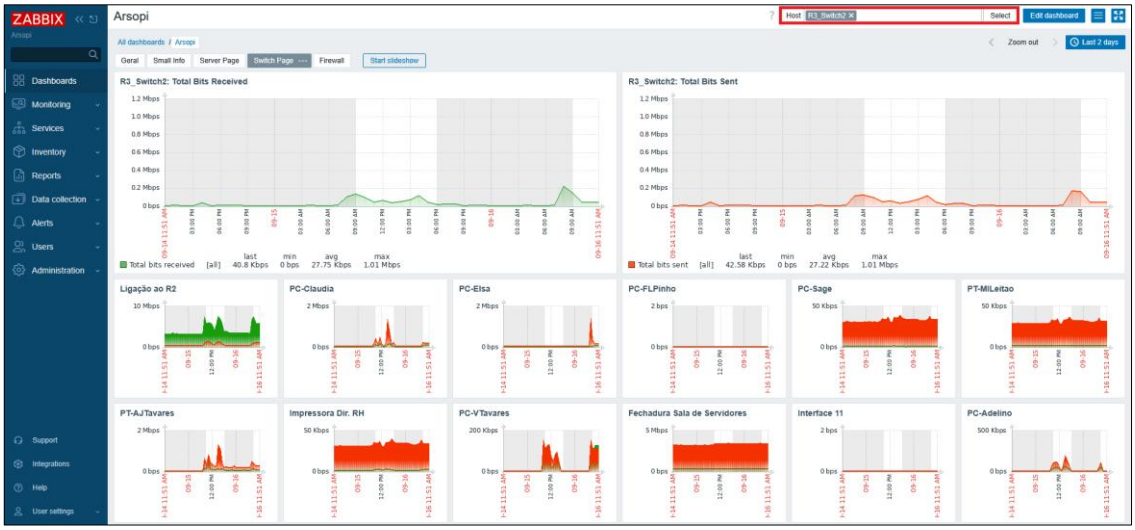


Figura 43 – Demonstração da página “Switch Page” quando está um *host* selecionado

Um aspeto relevante desta configuração, ilustrado nas figuras 42 e 43, é a alternância dinâmica entre gráficos:

- quando não existe *host* selecionado, são exibidos apenas os gráficos globais de todos os *switches* (Figura 42);
- ao selecionar um *host*, esses gráficos são ocultados e são mostrados os gráficos específicos do *switch* escolhido (Figura 43).

Além disso, para a maioria das interfaces, o título do gráfico corresponde ao nome do dispositivo ligado à porta em questão, permitindo identificar diretamente PCs, portáteis (PT), *workstations* (WS) ou outros equipamentos. Existem, no entanto, algumas exceções em que os nomes não são alterados, devido ao facto de se tratar de portas utilizadas de forma variável para diferentes dispositivos.

3.6.5.1.5 Página “Firewall”

A página “Firewall” foi concebida para apresentar informação detalhada sobre o dispositivo de segurança principal da infraestrutura. Apesar da existência de mais do que uma *firewall* no *datacenter*, por razões de redundância, apenas uma é monitorizada ativamente, pelo que nesta página não é necessária a seleção de *hosts* para a apresentação da informação.

Este *dashboard* é acedido diretamente a partir da página Geral, através do atalho associado ao título “Firewall”. A página disponibiliza um conjunto abrangente de métricas, nomeadamente:

- Disponibilidade da *firewall*, verificada através de respostas a *pings*;
- Estado dos túneis de VPN, em linha com a informação já visível na página Geral;
- Número de sessões IPv4 ativas;
- Tráfego de rede segmentado por interfaces LAN e WAN;
- Percentagem de utilização de CPU e memória.

Com esta organização, a página da Firewall centraliza as métricas críticas de operação do dispositivo, permitindo uma análise rápida da sua disponibilidade, desempenho e utilização de recursos.

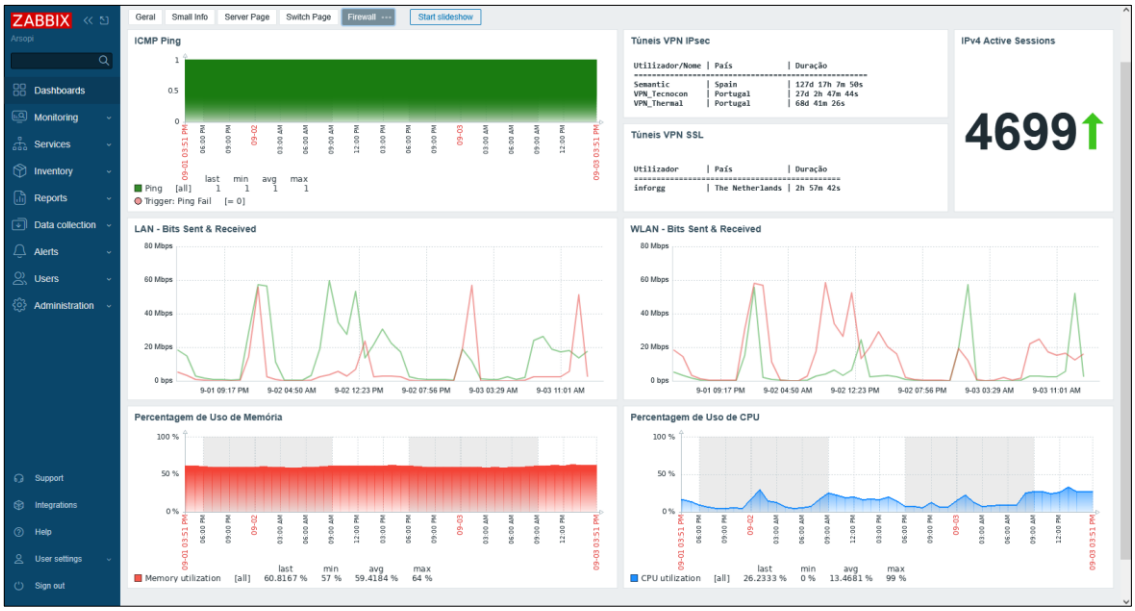


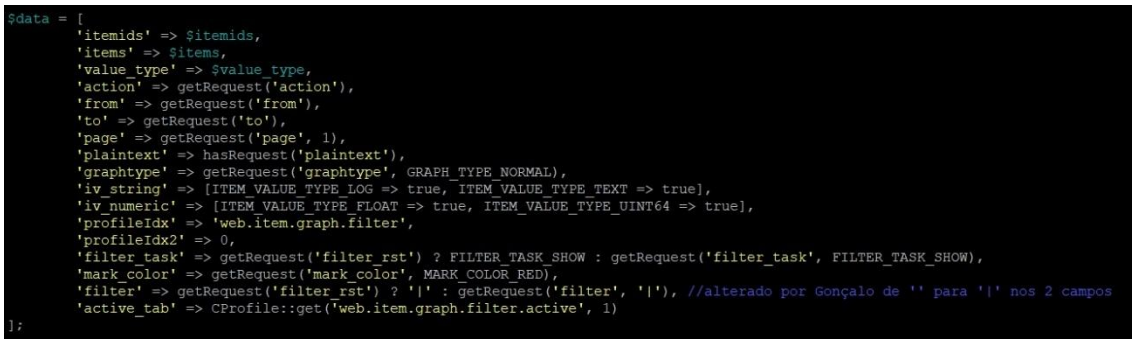
Figura 44 – Demonstração da página “Firewall” do *dashboard* “Arsopi”

3.6.6 Outros desenvolvimentos

No decorrer do processo de desenvolvimento da solução, foram identificadas necessidades adicionais e surgiram diversos desafios que exigiram resolução, com o objetivo de tornar a ferramenta mais intuitiva, eficiente e de fácil utilização. Para responder a estas exigências, foi necessário proceder a alguns desenvolvimentos complementares, ainda que de menor dimensão, de forma a colmatar as lacunas encontradas. Embora o Zabbix constitua uma plataforma robusta e abrangente, disponibilizando de forma acessível os mecanismos necessários para uma monitorização eficaz da rede, subsistem limitações inerentes, tanto ao nível da lógica de funcionamento do sistema como no que respeita à sua componente visual.

3.6.6.1 Pesquisa automática

Uma das dificuldades identificadas durante a utilização da ferramenta ocorreu no momento de análise dos alertas gerados após períodos de inatividade, como, por exemplo, ao regressar de um fim de semana. Ao aceder ao histórico de determinados problemas, verificava-se a existência de múltiplas instâncias vazias para determinados itens. Tal situação ocorria, nomeadamente, em itens como "Erros de Event Viewer", em que poderia ter sido registado um erro algumas horas antes, mas, devido à configuração de atualização a cada cinco minutos, surgiam diversas instâncias no registo ao longo do mesmo intervalo temporal. Para mitigar esta limitação, optou-se por automatizar a pesquisa utilizando o caractere "|", dado que, sempre que são registados erros, que constituem a informação efetivamente relevante para a análise, as colunas são separadas por este símbolo. Para a implementação desta solução, foi necessária a edição do código do ficheiro "/usr/share/zabbix/ui/history.php", de modo a substituir a pesquisa por cadeia vazia pela pesquisa com base no caractere "|", conforme ilustrado na Figura 45.



```
$data = [
        'itemids' => $itemids,
        'items' => $items,
        'value_type' => $value_type,
        'action' => getRequest('action'),
        'from' => getRequest('from'),
        'to' => getRequest('to'),
        'page' => getRequest('page', 1),
        'plaintext' => hasRequest('plaintext'),
        'graphtype' => getRequest('graphtype', GRAPH_TYPE_NORMAL),
        'iv_string' => [ITEM_VALUE_TYPE_LOG => true, ITEM_VALUE_TYPE_TEXT => true],
        'iv_numeric' => [ITEM_VALUE_TYPE_FLOAT => true, ITEM_VALUE_TYPE_UINT64 => true],
        'profileIdx' => 'web.item.graph.filter',
        'profileIdx2' => 0,
        'filter_task' => getRequest('filter_rst') ? FILTER_TASK_SHOW : getRequest('filter_task', FILTER_TASK_SHOW),
        'mark_color' => getRequest('mark_color', MARK_COLOR_RED),
        'filter' => getRequest('filter_rst') ? '|' : getRequest('filter', '|'), //alterado por Gonçalo de '' para '|' nos 2 campos
        'active_tab' => CProfile::get('web.item.graph.filter.active', 1)
];
```

Figura 45 – Demonstração de um excerto de código da alteração no ficheiro "history.php"

Após esta alteração, passou a ser mais fácil e rápido analisar os problemas que surgem de alertas recebidos, uma vez que apenas mostra as instâncias com erros, omitindo informação que não nos é útil.

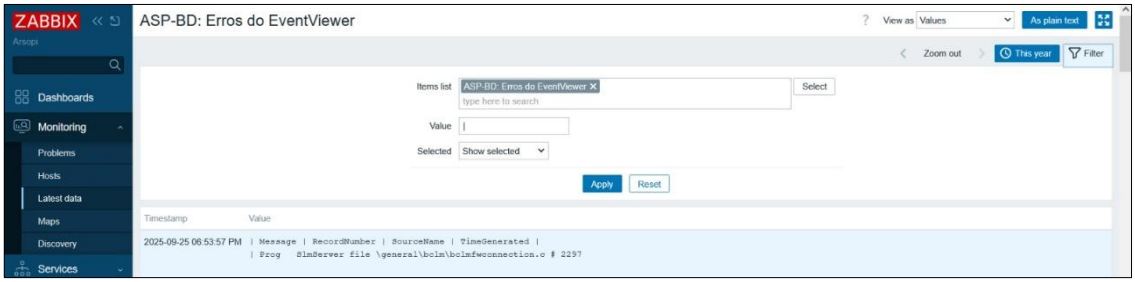


Figura 46 – Demonstração de exemplo após a alteração feita no filtro

3.6.6.2 Remoção de espaçamento entre *widgets*

Com o intuito de otimizar a visualização de uma das páginas do *dashboard*, foi necessário proceder à remoção dos espaçamentos pré-definidos entre *widgets*. O Zabbix estabelece, por defeito, uma margem entre estes elementos, a qual, em determinadas circunstâncias, resulta na ocupação de espaço desnecessário e pode comprometer a leitura imediata da informação apresentada. Tal situação revela-se particularmente relevante no contexto de um *dashboard* cuja principal finalidade consiste em proporcionar uma análise rápida e intuitiva dos dados. Para ultrapassar esta limitação, desenvolveu-se uma função específica no ficheiro "/usr/share/zabbix/ui/js/functions.js", que permitiu ajustar o comportamento de visualização de acordo com as necessidades do projeto.

```
function removePaddingFromContainers(){
if (window.location.href.includes("dashboardid=344") &&
window.location.href.includes("page=2")) {
        const dashboard = document.querySelector('.dashboard-
grid');
        dashboard.classList.add('custom-dashboard-grid');
        const widgets = document.querySelectorAll('.dashboard-
grid-widget-container');
        widgets.forEach(widget=> {
              widget.classList.add('custom-container');
                    const header =
widget.querySelector('.dashboard-grid-widget-header');
                    if (header) {
                          header.classList.add('custom-
header');
                    }
              const contents =
widget.querySelector('.dashboard-grid-widget-contents');
              if (contents){
                    const body =
contents.querySelector('.dashboard-grid-widget-body');
                    if (body){
                          const label =
body.querySelector('.widget-label');
                          const labelTest =
document.querySelector('.widget-label');
                          if(label){
                                const labelText =
label.textContent || label.innerText;
                                if
(labelText.includes('Switch')) {

 widget.classList.remove('custom-container');

 widget.classList.add('custom-container2');
                                }
                          }
                    }
              }
        });
 }
}
```

Código 33 – Função “removePaddingFromContainers”

As classes utilizadas neste código foram criadas anteriormente no ficheiro “/usr/share/zabbix/ui/assets/styles/blue-theme.css” de forma que o *padding* utilizado fosse removido, tal como explica a função, passando de um *dashboard* como o da figura 47 para o *dashboard* mostrado na figura 39.

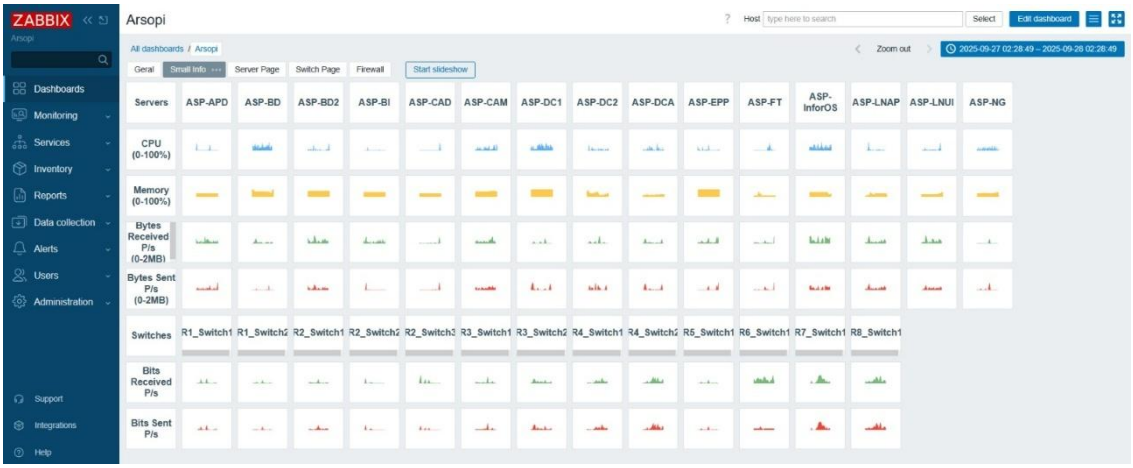


Figura 47 – Demonstração da página “Small Info” do *dashboard* “Arsopi” antes das alterações visuais

3.6.6.3 Interfaces com nome de dispositivo

Outra modificação implementada consistiu na alteração da nomenclatura das interfaces dos *switches*, substituindo designações genéricas, como “Interface 1”, por identificadores mais específicos, como “PC-1”. Esta alteração, solicitada pela entidade supervisora, teve como objetivo facilitar a identificação de picos de tráfego de rede, permitindo, através da associação direta entre os gráficos gerados e a designação da interface, determinar com maior rapidez a origem do incidente e, consequentemente, adotar as medidas corretivas adequadas. Por motivos de segurança, o código desenvolvido para esta modificação não é apresentado; contudo, o resultado obtido encontra-se parcialmente ilustrado na Figura 43.

3.6.6.4 Remover *widgets* desnecessários

A modificação realizada consistiu na implementação de um mecanismo que permite mostrar ou ocultar os *widgets* da página “Switch Page” do *dashboard* “Arsopi”, em função do *host* selecionado, conforme detalhado no ponto 3.6.5.1.4. Para alcançar este objetivo, foi desenvolvida uma função responsável por controlar a visibilidade dos diferentes *widgets* e proceder ao seu reposicionamento no *dashboard*, de forma a garantir uma apresentação mais clara e uma utilização eficiente do espaço disponível.

```
function removeUselessWidgets() {
    let count = 0;

    setTimeout(function() {
        let widgets = document.querySelectorAll('.dashboard-grid-widget');

        widgets.forEach(widget => {
           widget.style.visibility = "visible";
        });

        widgets.forEach(widget => {
            let noDataDiv = widget.querySelector('.no-data-message');
           let title = widget.querySelector('.dashboard-grid-widget-container .dashboard-grid-
widget-header h4');
            if (window.location.href.includes("dashboardid=344") &&
window.location.href.includes("page=3")) {
                if (noDataDiv) {
                            widget.style.visibility = "hidden";
                } else {
                    count++;
                    widget.style.left = (count % 2 === 0) ? "50%" : "0%";
                    widget.style.top = `${Math.floor((count - 1) / 2) * 350}px`;
                }
            } else if (window.location.href.includes("dashboardid=344") &&
window.location.href.includes("page=4")) {
                const host = document.querySelector('input[name="dashboard_hostid"]');
                if(!host || host == ''){
                    if(widget.style.width == '50%' || widget.style.width == '16.6667%'){
                                widget.style.visibility = "hidden";
                    }
                } else {
                            console.log(host);
                    if(widget.style.width == '33.3333%' || widget.style.width == '25%' ||
noDataDiv){
                                widget.style.visibility = "hidden";
                            } else if((host.value === "10730" &&
title.textContent.includes("Interface"))){
                                            widget.style.visibility = "hidden";
                    } else {
                        count++;
                        if(count == 1){
                            widget.style.left = "0%";
                            widget.style.top = "0px";
                        } else if(count == 2){
                            widget.style.left = "50%";
                            widget.style.top = "0px";
                        } else if(count > 2){
                            if ((count - 2) % 6 === 1) {
                                widget.style.left = "0%";
                            } else if ((count - 2) % 6 === 2) {
                                widget.style.left = "16.6667%";
                            } else if ((count - 2) % 6 === 3) {
                                widget.style.left = "33.3333%";
                            } else if ((count - 2) % 6 === 4) {
                                widget.style.left = "50%";
                            } else if ((count - 2) % 6 === 5) {
                                widget.style.left = "66.6667%";
                            } else if ((count - 2) % 6 === 0) {
                                widget.style.left = "83.3333%";
                            }
                            widget.style.top = `${(Math.floor((count - 3) / 6) * 210) + 350}px`;
                        }
                    }
                }
            }
        });
    }, 1);
}
```

Código 34 – Função “removeUselessWidgets”

3.6.6.5 Selecionar *host*

Foi desenvolvida uma função que possibilita a seleção de um *host* através de um simples clique sobre a respetiva *label*. Esta funcionalidade visa essencialmente melhorar a rapidez de acesso e a usabilidade, permitindo que o utilizador final, ao pretender consultar informação relativa a um determinado *host*, seja automaticamente direcionado para o *dashboard* correspondente, onde são apresentadas as informações específicas desse dispositivo. Esta melhoria revela-se particularmente útil na página "Small Info", uma vez que esta disponibiliza uma visão geral dos dispositivos mais relevantes e, quando necessário, possibilita a consulta detalhada de cada um deles mediante a interação direta com o respetivo título.

```
function selectHost(){
  const host = document.querySelector('input[name="dashboard_hostid"]');
  if(window.location.href.includes("dashboardid=344") &&
window.location.href.includes("page=4") && (!host)) {
          const dashboard = document.querySelector('.dashboard-grid');
          const headers = dashboard.querySelectorAll('.dashboard-grid-widget-header');
          headers.forEach(header=> {
                  const title = header.querySelector('h4');
                  if (!title) return;

                  // Adiciona estilo de cursor e hover
                  title.style.cursor = 'pointer';
                  title.style.transition = 'color 0.2s';

                  title.addEventListener('mouseover', () => {
                          title.style.color = '#007bff';
                          title.style.textDecoration = 'underline';
                  });

                  title.addEventListener('mouseout', () => {
                          title.style.color = '';
                          title.style.textDecoration = '';
                  });

                  // Adiciona comportamento de clique
                  title.addEventListener('click', () => {
                          const switchName = title.innerText.trim();
                          let hostId;
                          if(switchName.includes("R1_Switch1")){
                                  hostId = '10691';
                          } else if(switchName.includes("R1_Switch2")){
                                  hostId = '10692';
                          } else if(switchName.includes("R2_Switch1")){
                                  hostId = '10693';
                          } else if(switchName.includes("R2_Switch2")){
                                  hostId = '10694';
                          } else if(switchName.includes("R2_Switch3")){
                                  hostId = '10695';
                          } else if(switchName.includes("R3_Switch1")){
                                  hostId = '10696';
                          } else if(switchName.includes("R3_Switch2")){
                                  hostId = '10697';
                          } else if(switchName.includes("R4_Switch1")){
                                  hostId = '10698';
                          } else if(switchName.includes("R4_Switch2")){
                                  hostId = '10730';
                          } else if(switchName.includes("R5_Switch1")){
                                  hostId = '10699';
                          } else if(switchName.includes("R6_Switch1")){
                                  hostId = '10700';
                          } else if(switchName.includes("R7_Switch1")){
                                  hostId = '10701';
                          } else if(switchName.includes("R8_Switch1")){
                                  hostId = '10702';
                          }
                          if (hostId) {
                                  // Redireciona para o mesmo dashboard, com o host
filtrado
                                  window.location.href = "https://ln-
netmon.arsopi.pt/zabbix.php?action=dashboard.view&dashboardid=344&hostid=" + hostId +
"&from=now-2d&to=now&page=4";
                          } else {
                                  alert("Host não reconhecido: " + switchName);
                          }
                  });
          });
  }
}
```

Código 35 – Função “selectHost”

3.6.6.6 Criação de *widget* “label”

Conforme já referido, um dos desenvolvimentos realizados consistiu na criação, de raíz, de um novo *widget*, designado como “label”. A necessidade deste *widget* surgiu da intenção de disponibilizar, em determinadas situações, um *widget* que consiste apenas num título que, ao ser clicado, pudesse redirecionar o utilizador para outra página ou funcionalidade, conforme descrito no ponto anterior. A implementação deste *widget* foi realizada com base no tutorial apresentado na secção de módulos da documentação oficial do Zabbix, motivo pelo qual não se considerou necessário proceder à sua documentação detalhada (Zabbix, s.d.).

4 Avaliação da solução implementada

4.1 Casos reais de utilização

A implementação da ferramenta de monitorização revelou-se determinante para o aumento da eficiência operacional da equipa de IT, proporcionando maior facilidade de deteção de incidentes, maior proatividade e uma redução significativa no tempo de resposta a falhas ou anomalias. Além da rapidez na identificação, a solução permitiu ainda atuar de forma preventiva em diversas situações, mitigando riscos antes que se transformassem em problemas reais para os utilizadores.

Um exemplo prático é a monitorização do espaço em disco nos servidores. A geração de alertas sempre que a percentagem de espaço livre atinge valores críticos possibilita que a equipa atribua recursos adicionais ou realize limpezas preventivas, evitando indisponibilidades. De forma semelhante, a monitorização da utilização do processador permite detetar de imediato processos ou aplicações que estejam a consumir recursos de forma inesperada, possibilitando uma intervenção rápida e eficaz.

Outro caso de utilização relevante está relacionado com os erros registados no “Event Viewer” dos sistemas Windows. Anteriormente, era necessário proceder diariamente a uma verificação manual dos principais servidores, uma tarefa que consumia bastante tempo da equipa. Com a nova solução, os erros são identificados automaticamente e notificados em tempo real, permitindo uma resposta imediata e libertando recursos humanos para outras atividades de maior valor.

A monitorização de alterações de *password* também se revelou particularmente útil. Frequentemente, após a alteração da palavra-passe, alguns serviços Microsoft exigem novo processo de autenticação, o que pode originar falhas até que a atualização seja realizada. O alerta gerado pelo sistema permite à equipa identificar rapidamente estas situações, proceder às atualizações necessárias e minimizar o tempo de inatividade do utilizador, evitando interrupções prolongadas no seu trabalho.

Também a monitorização de serviços essenciais demonstrou o seu valor. Exemplos incluem serviços de antivírus (como o Trend Micro) ou serviços críticos de bases de dados (SQL), CAD ou CAM. O alerta imediato da indisponibilidade de qualquer um destes serviços permite uma ação proativa, muitas vezes antes mesmo de os utilizadores reportarem a ocorrência.

Outro cenário recorrente é a deteção de falhas de conectividade (*ping*) em dispositivos de rede, como *switches*, servidores ou pontos de acesso. Nestas situações, a ferramenta notifica de imediato a equipa, que pode intervir rapidamente, reduzindo substancialmente o tempo médio de resposta e mitigando o impacto para os utilizadores finais.

Por fim, a monitorização contínua permite ainda identificar padrões de tráfego anómalos na rede, como picos simultâneos em múltiplos *switches*, ajudando a antecipar problemas de desempenho e a proceder a ajustes preventivos na infraestrutura.

No conjunto, estes são alguns dos inúmeros exemplos que demonstram que a adoção da solução de monitorização não só aumentou a capacidade de reação da equipa, como também reforçou a vertente preventiva, contribuindo para uma gestão mais eficiente e resiliente da infraestrutura tecnológica da organização.

4.2 Feedback da equipa de IT

Como se pôde verificar nos exemplos descritos anteriormente, a introdução da ferramenta de monitorização resultou numa redução significativa da carga de trabalho manual da equipa de IT, enquanto aumentou a eficiência e a proatividade na deteção e resolução de incidentes. Tarefas que anteriormente exigiam verificações manuais frequentes ou tempo de diagnóstico prolongado passaram a ser executadas de forma automatizada e em tempo real, permitindo à equipa concentrar-se em atividades de maior valor estratégico.

Esta melhoria operacional reflete-se no feedback bastante positivo por parte da equipa, que reconhece na solução implementada um aliado fundamental na gestão diária da infraestrutura. A facilidade acrescida na deteção de problemas, a rapidez de resposta e a capacidade de atuar preventivamente são apontados como fatores determinantes para a satisfação geral com a ferramenta.

Importa ainda referir que a solução não é encarada como um sistema estático, mas sim como um projeto em constante evolução. A equipa de IT tem procurado explorar continuamente novas funcionalidades, configurar alertas adicionais e integrar diferentes métricas, levando a ferramenta ao limite das suas capacidades. Este esforço contínuo traduz-se numa utilização cada vez mais otimizada e adaptada às necessidades específicas da organização, contribuindo para uma análise mais completa e para uma monitorização cada vez mais eficaz da rede corporativa.

4.3 Futuras melhorias

A solução atualmente implementada cumpre todos os requisitos inicialmente definidos, não existindo, neste momento, nenhuma necessidade identificada que não esteja a ser satisfeita. Ainda assim, a equipa reconhece que existem oportunidades de melhoria que podem contribuir para tornar o sistema mais intuitivo e abrangente.

Um dos aspetos a otimizar prende-se com a apresentação de alguns itens monitorizados, nomeadamente os alertas provenientes do "Event Viewer". Neste caso, a informação é devolvida em formato extenso e pouco prático, pelo que se pretende desenvolver mecanismos de transformação do texto em mensagens mais curtas e diretas, facilitando a leitura e análise imediata por parte da equipa técnica.

Em paralelo, encontra-se em curso um estudo para avaliar de forma mais aprofundada o potencial da monitorização de *logs* do "Event Viewer". Embora, atualmente, estejam a ser recolhidos apenas os eventos mais críticos relacionados com a segurança ao nível de utilizadores e grupos do domínio, existe margem para explorar muitas outras categorias de eventos que podem enriquecer a análise e reforçar a segurança da infraestrutura.

Adicionalmente, foi identificada a necessidade de realizar uma monitorização mais detalhada dos registos (*logs*) dos *switches*, motivada por um incidente recente que poderia ter sido evitado ou mitigado caso existisse um acompanhamento mais rigoroso deste tipo de informação. Neste sentido, esta melhoria encontra-se já em fase de desenvolvimento, prevendo-se a sua implementação num futuro próximo.

Outra vertente importante de evolução consiste na extensão da solução às restantes empresas do grupo. Este processo já está em execução e contempla a definição de uma lista de

dispositivos a monitorizar, bem como das métricas a considerar em cada caso. Paralelamente, estão a ser estudados os mecanismos de atribuição de permissões adequadas aos utilizadores de cada empresa, garantindo um acesso controlado e alinhado com as responsabilidades de cada equipa.

Por fim, foram ainda identificadas possíveis melhorias relacionadas com a visualização de dados nos *dashboards*. Apesar de o Zabbix disponibilizar uma ampla gama de *widgets* e opções gráficas, algumas necessidades específicas da organização exigem ajustes ao código da ferramenta. Já foram implementadas pequenas alterações nesse sentido, enquanto não são disponibilizadas, de forma nativa, novas opções que permitam personalizações mais completas.

Em suma, apesar de a solução atual cumprir integralmente os objetivos estabelecidos, a equipa mantém uma postura de evolução contínua, procurando otimizar a usabilidade, ampliar o âmbito da monitorização e melhorar a forma como a informação é apresentada, de modo a garantir que o sistema acompanha o crescimento e as necessidades futuras da organização.

5 Conclusão

A implementação de uma solução de monitorização na Arsopi surgiu da necessidade de reforçar a segurança, disponibilidade e eficiência operacional da infraestrutura tecnológica. Após uma análise comparativa de diferentes ferramentas, concluiu-se que o Zabbix reunia as condições ideais, quer pela sua versatilidade e robustez, quer pela sua capacidade de integração com diversos sistemas e dispositivos.

A metodologia seguida permitiu não apenas validar a adequação da solução, mas também adaptá-la ao contexto específico da organização. Foram configuradas métricas abrangentes para servidores, dispositivos de rede e *firewall*, recorrendo a diferentes métodos de recolha de dados (SNMP, WMI, *scripts* externos e APIs), e criados *dashboards* personalizados que proporcionam uma visão clara e centralizada do estado da infraestrutura. Paralelamente, foram aplicadas boas práticas de segurança, assegurando a proteção da solução contra potenciais ameaças.

Os resultados obtidos confirmam o impacto positivo da ferramenta: a equipa de IT passou a dispor de maior capacidade de deteção precoce de incidentes, reduziu significativamente o tempo de resposta e ganhou em proatividade na gestão da rede. Casos concretos, como a deteção de falhas de serviços críticos, erros do Event Viewer ou anomalias de tráfego em dispositivos de rede, ilustram a relevância prática da solução. O *feedback* da equipa foi globalmente positivo, reforçando a perceção de que o Zabbix contribuiu para uma gestão mais eficiente e segura da infraestrutura.

Apesar da abrangência da implementação, identificaram-se algumas oportunidades de melhoria, nomeadamente no aperfeiçoamento das métricas do Event Viewer, na evolução dos *dashboards* e na extensão da solução às restantes empresas do grupo, assegurando uniformidade e escalabilidade da monitorização.

Em síntese, este trabalho demonstrou que uma solução de monitorização bem configurada constitui uma ferramenta estratégica para a administração de sistemas e para a segurança da informação. A implementação do Zabbix na Arsopi representou um passo significativo no

fortalecimento da infraestrutura tecnológica, contribuindo para a sua resiliência e para uma maior capacidade de resposta a desafios futuro

Referências

Alia Abdi, A. & Boubeghel, R., 2024. *Comparison of Different Monitoring and Alerting Tools in IT Infrastructure,* s.l.: s.n.

BasuMallick, C., 2023. *Spiceworks.* [Online]
Available at: https://www.spiceworks.com/tech/networking/articles/distributed-component-object-model-dcom/https://www.spiceworks.com/tech/networking/articles/distributed-component-object-model-dcom/
[Acedido em 3 janeiro 2025].

Canzari, M., Di Carlo, M., Dolci, M. & Smareglia, R., 2019. *Using Nagios to monitor the Telescope Manager (TM) of the Square Kilometre Array (SKA),* s.l.: s.n.

Case, J., Fedor, M., Schoffstall, M. & Davin, J., 1990. *A Simple Network Management Protocol (SNMP),* s.l.: RFC Editor.

Checkmk, 2023. *Basic information on the installation of Checkmk.* [Online]
Available at: https://docs.checkmk.com/latest/en/install_packages.html
[Acedido em 5 janeiro 2025].

Checkmk, 2025. *Release Notes.* [Online]
Available at: https://docs.checkmk.com/latest/en/release_notes.html
[Acedido em 5 janeiro 2025].

Checkmk, 2025. *The user interface.* [Online]
Available at: https://docs.checkmk.com/latest/en/user_interface.html
[Acedido em 4 janeiro 2025].

Checkmk, s.d. *About Us.* [Online]
Available at: https://checkmk.com/company/about-us
[Acedido em 4 janeiro 2025].

Checkmk, s.d. *Pricing.* [Online]
Available at: https://checkmk.com/pricing?currency=EUR
[Acedido em 3 janeiro 2025].

Cisco, s.d. *What is Network Monitoring?.* [Online]
Available at: https://www.cisco.com/c/en_uk/solutions/automation/what-is-network-monitoring.html
[Acedido em 3 janeiro 2025].

Cloudflare, s.d. *O que é o Protocolo de internet?.* [Online]
Available at: https://www.cloudflare.com/pt-br/learning/network-layer/internet-protocol/
[Acedido em 2 janeiro 2025].

Cloudflare, s.d. *What is the Internet Control Message Protocol (ICMP) ?.* [Online]
Available at: https://www.cloudflare.com/learning/ddos/glossary/internet-control-message-protocol-icmp/
[Acedido em 1 agosto 2025].

dos Santos Souza Silva, A., Oliveira dos Santos, M. J. & Almeida de Oliveira, R., 2024. Network management study with Zabbix Server. *Brazilian Journal of Technology,* p. 24.

Fortinet, s.d. *Simple Network Management Protocol.* [Online]
Available at: https://www.fortinet.com/br/resources/cyberglossary/simple-network-management-protocol
[Acedido em 2 janeiro 2025].

Fortra, 2025. *What is SNMP? How SNMP Works.* [Online]
Available at: https://www.fortra.com/blog/what-snmp-how-snmp-works
[Acedido em 5 janeiro 2025].

Gonçalves Coelho, J. et al., 2020. Network Monitoring Tools: Enabling the Management of Network Assets, Ensuring High Availability. *International Journal of Development Research,* 10(06), pp. 37020-37026.

Guo, F., Chen, C. & Li , K., 2024. Research on Zabbix Monitoring System for Large-scale Smart Campus Network from a Distributed Perspective. *Journal of Electrical Systems,* pp. 631-648.

Harrington, D., Presuhn, R. & Wijnen, B., 2002. *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks,* s.l.: RFC Editor.

IBM, 2025. *Remote Procedure Call.* [Online]
Available at: https://www.ibm.com/docs/en/aix/7.3?topic=concepts-remote-procedure-call
[Acedido em 3 janeiro 2025].

Johnson, R. & Elizabeth, N. E., 2017. *Research Gate.* [Online]
Available at: https://www.researchgate.net/profile/N-Elizabeth/publication/323347624_Network%27s_server_monitoring_and_analysis_using_Nagios/links/5b2a2261aca27209f3752a30/Networks-server-monitoring-and-analysis-using-Nagios.pdf
[Acedido em 4 janeiro 2025].

Koskinen, M., 2024. *Integrating open-source computer and network monitoring software to an automation supervision system,* s.l.: s.n.

Kumar, A., 2024. *Research Gate.* [Online]
Available at: https://www.researchgate.net/profile/Ashish-Kumar-348/publication/394041372_THE_NEW_INTELLIGENT_MONITORING_SOLUTIONS_IN_UNIX_NAGIOS_ZABBIX_AND_BEYOND/links/68861e74f8031739e60916fe/THE-NEW-INTELLIGENT-MONITORING-SOLUTIONS-IN-UNIX-NAGIOS-ZABBIX-AND-BEYOND.
[Acedido em 5 janeiro 2025].

Kumar, S., 2025. *Research Gate.* [Online]
Available at:
https://www.researchgate.net/publication/393356670_Linux_Server_Monitoring_and_Uptime_optimization_in_Healthcare_IT_Review_of_Nagios_Zabbix_and_Custom_Scripts
[Acedido em 5 janeiro 2025].

Lavagnoli, E. & Gonçalves, J. A., 2024. *Monitoramento de hosts através do software Zabbix,* São Paulo: s.n.

Microsoft, 2021. *Microsoft Learn.* [Online]
Available at: https://learn.microsoft.com/en-us/windows/win32/wmisdk/about-wmi
[Acedido em 3 janeiro 2025].

Microsoft, 2025. *Microsoft Learn.* [Online]
Available at: https://learn.microsoft.com/pt-br/windows-server/identity/ad-ds/plan/appendix-l--events-to-monitor
[Acedido em 8 setembro 2025].

Nagios, s.d. *About Nagios Core.* [Online]
Available at:
https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/about.html
[Acedido em 5 janeiro 2025].

Nagios, s.d. *Nagios Core.* [Online]
Available at: https://www.nagios.org/projects/nagios-core/
[Acedido em 3 janeiro 2025].

Nagios, s.d. *Quickstart Installation Guides.* [Online]
Available at:
https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/quickstart.html
[Acedido em 5 janeiro 2025].

Nagios, s.d. *The Nagios Story.* [Online]
Available at: https://www.nagios.org/story/
[Acedido em 2 janeiro 2025].

Ogogo, W. L., 2021. Real-Time Monitoring of Network Devices: Its Effectiveness in Enhacning Network Security. *East African Journal of Information Technology,* pp. 1-6.

Phiri, M. & Mfupa, P., 2016. Network Monitoring System. *International Journal of Novel Research in Computer Science and Software Engineering,* 3(2), pp. 55-68.

serverfault, 2024. *Which permissions/rights does a user need to have WMI access on remote machines?.* [Online]
Available at: https://serverfault.com/questions/28520/which-permissions-rights-does-a-user-need-to-have-wmi-access-on-remote-machines
[Acedido em 27 janeiro 2025].

VMWare, s.d. *Network Monitoring.* [Online]
Available at: https://www.vmware.com/topics/network-monitoring
[Acedido em 2 janeiro 2025].

Zabbix, s.d. *About Us.* [Online]
Available at: https://www.zabbix.com/about
[Acedido em 4 janeiro 2025].

Zabbix, s.d. *Create a widget (tutorial).* [Online]
Available at:
https://www.zabbix.com/documentation/current/en/devel/modules/tutorials/widget
[Acedido em 8 junho 2025].

Zabbix, s.d. *Get Zabbix.* [Online]
Available at:
https://www.zabbix.com/download?zabbix=7.2&os_distribution=ubuntu&os_version=24.04&components=server_frontend_agent&db=mysql&ws=apache
[Acedido em 20 janeiro 2025].

Zabbix, s.d. *Quickstart.* [Online]
Available at: https://www.zabbix.com/documentation/7.2/en/manual/quickstart
[Acedido em 17 janeiro 2025].

Zabbix, s.d. *Web interface installation.* [Online]
Available at: https://www.zabbix.com/documentation/7.2/en/manual/installation/frontend
[Acedido em 20 janeiro 2025].

Zabbix, s.d. *Zabbix appliance.* [Online]
Available at: https://www.zabbix.com/documentation/7.2/en/manual/appliance
[Acedido em 4 janeiro 2025].

Zabbix, s.d. *Zabbix features.* [Online]
Available at: https://www.zabbix.com/documentation/7.2/en/manual/introduction/features
[Acedido em 3 janeiro 2025].

Zabbix, s.d. *Zabbix overview.* [Online]
Available at: https://www.zabbix.com/documentation/7.2/en/manual/introduction/overview
[Acedido em 4 janeiro 2025].