NetBox-Zabbix Plugin para Gestão e Organização de Data Center

Inês Costa[1,2], Ivo Pereira[1,3] e Daniel Alves de Oliveira[2]

[1] ISEP, Politécnico do Porto, 4249-015 Porto, Portugal
[2] E-goi, 4450-190 Matosinhos, Portugal
[3] INESC TEC, Faculdade de Engenharia, Universidade do Porto, Rua Dr. Roberto Frias, Porto, 4200-465, Portugal
1210814@isep.ipp.pt

Resumo. Este artigo descreve a modernização e centralização do inventário e da monitorização de infraestruturas realizada num estágio curricular de licenciatura. A arquitetura proposta integra o NetBox como sistema de inventário e DCIM (Data Center Infrastructure Management) e o Zabbix como plataforma central de monitorização e alarmística, com autenticação unificada baseada em LDAP (Lightweight Directory Access Protocol). Foi desenvolvido um plugin em Python para sincronização entre o NetBox e o Zabbix; a plataforma de monitorização inclui descoberta automática, templates personalizados, ações corretivas e alertas multicanal, bem como a centralização de tarefas previamente dispersas. A solução foi validada numa prova de conceito em ambiente real de escritório, evidenciando ganhos de visibilidade e eficiência operacional e reduzindo inconsistências entre inventário e monitorização. Embora ainda não esteja implantada no data center principal, a solução encontra-se preparada para escalar; trabalho futuro inclui sincronização em tempo real, novos dashboards e configuração de alta disponibilidade.

Palavras-chave: Gestão de Infraestruturas, Monitorização, Integração de Sistemas, Inventário, Automatização, NetBox, Zabbix, Python.

1 Introdução

O panorama atual evidencia fragmentação entre ferramentas de gestão e monitorização, sem integração nem fonte única de verdade, o que gera inconsistências de inventário, informação desatualizada e maior carga manual. Esta realidade conduz a operações reativas, dificultando a antecipação de incidentes por análise de padrões e aumentando o tempo médio de reparação (MTTR - Mean Time to Repair), com risco acrescido de indisponibilidade, impacto na satisfação dos clientes e na reputação. A dependência de processos manuais limita a escalabilidade e desvia tempo da equipa de iniciativas de maior valor. Face a estes desafios técnicos e riscos de negócio, impõe-se uma solução que privilegie centralização, automação e fiabilidade.

A infraestrutura em estudo suportava e monitorizava serviços críticos (e.g. bases de dados, websites específicos, hardware, testes a portas de rede, entre outros) utilizando

um conjunto de ferramentas de gestão de inventário, endereçamento IP, monitorização e dashboards isoladas. A falta de integração dificultava uma perspetiva atualizada do estado do sistema e aumentava a propensão a erro humano, sobretudo em tarefas de diagnóstico e de resposta a incidentes. Neste cenário, a modernização e consolidação da stack operacional tornaram-se prioritárias.

Apesar de funcionais de forma isolada, as ferramentas existentes apresentavam fragmentação (dados redundantes e desatualizados), subutilização de capacidades (ausência de descoberta/automação coordenada) e baixa escalabilidade organizacional (dificuldade em integrar novos serviços/processos de forma automática). Em particular, a inexistência de um "single source of truth" (SSoT) para inventário e de integrações por API comprometia: (i) a coerência entre inventário e monitorização; (ii) a rapidez na deteção e resposta; e (iii) a rastreabilidade das alterações.

Este projeto teve como objetivos:

1. Centralizar o inventário de recursos (equipamentos, redes, endereços IP, bastidores) e adotar um "source of truth";
2. Unificar a monitorização de ativos e a gestão de alertas;
3. Integrar inventário e monitorização por API, reduzindo inconsistências e esforço manual;
4. Automatizar a descoberta de novos ativos, aplicação de templates, ações corretivas e notificações multicanal;
5. Padronizar tarefas dispersas (e.g. crontabs) e documentação de operação;
6. Validar a abordagem numa prova de conceito em ambiente real, avaliando benefícios e limitações.

O projeto adotou uma abordagem iterativa e incremental, iniciando-se com a avaliação da stack aplicacional e a recolha de informações junto da equipa de Site Reliability Engineering (SRE) para identificar lacunas e redundâncias existentes. Após análise do estado da arte selecionaram-se alternativas open-source que permitiram a centralização das tarefas necessárias - NetBox (inventário/IPAM) e Zabbix (autodescoberta, monitorização e ações corretivas) - promovendo substituição gradual com preservação de dados, integração via APIs (adição automática de hosts, validação cruzada e alarmística centralizada) e consideração do Grafana para reforço de visualização. Em paralelo, a automação (scripts, templates de configurações, atualizações) reduziu a intervenção manual e o risco de erro humano. Foi produzida documentação técnica que facilitou a adaptação de equipa de SRE.

O artigo está organizado da seguinte forma: o capítulo 2 faz uma contextualização do estado da arte, descrevendo trabalhos relacionados; o capítulo 3 apresenta a arquitetura de solução; a implementação e a avaliação são descritas no capítulo 4; e, finalmente, no capítulo 5 são apresentadas as principais conclusões, incluindo limitações e trabalho futuro.

2 Trabalhos relacionados

A literatura recente mostra uma evolução marcada na gestão e monitorização de data centers, impulsionada pela maior complexidade dos sistemas e pela exigência de disponibilidade, eficiência e automação [1]. Estudos de caso evidenciam migrações de stacks fragmentadas para soluções integradas, como a adoção do Zabbix num banco de média dimensão, com ganhos na observabilidade e no tempo de resposta [2], e a transição do projeto Tor de Munin para Prometheus com Grafana, tirando partido de alertas mais avançados (Alertmanager) e de uma linguagem de consulta expressiva (PromQL) [3, 4]. Em paralelo, ferramentas modernas de DCIM, como o NetBox, consolidam a gestão de ativos e IPAM como source of truth e expõem APIs que facilitam integrações com a monitorização (e.g., sincronização NetBox↔Zabbix e enriquecimento de alertas) [5, 6]. Para agregação e visualização unificadas, o Grafana opera como "single pane of glass", integrando métricas de múltiplas origens e reduzindo inconsistências e erros [5, 7]. A automação com Ansible, alinhada com princípios de IaC e práticas DevOps, promove uniformidade de configurações e diminui a propensão a erro humano [8, 9]. No plano arquitetural, surgem propostas avançadas (e.g., Monalytics) que integram monitorização e análise para reduzir latências e custos [10], bem como abordagens distribuídas para IaaS (IaaSMon) com suporte a VMs e integração com plataformas como OpenStack [11]. Adicionalmente, iniciativas de AIOps aplicam aprendizagem automática para previsão de anomalias e otimização operacional [12]. Por fim, a integração da TI com a infraestrutura física, via DCIM e gestão remota, reforça a resiliência em cenários distribuídos e edge [1].

Em síntese, os trabalhos relacionados convergem em três eixos: (i) evolução para soluções de monitorização escaláveis e orientadas a métricas [3, 4]; (ii) centralização da gestão de ativos com suporte a automação e source of truth [5, 6, 8, 9]; e (iii) adoção de arquiteturas proativas/inteligentes (Monalytics, IaaSMon, AIOps) com foco na prevenção e resposta eficiente a falhas [10, 11, 12]. O projeto da empresa em questão posiciona-se neste panorama, aplicando estas práticas para modernizar e automatizar a sua infraestrutura. A escolha da utilização de NetBox (para gestão de inventário e endereçamento IP) e Zabbix (para monitorização centralizada) justifica-se por serem ferramentas open-source e flexíveis, cumprindo os requisitos do projeto.

3 Arquitetura da Solução

A arquitetura é descrita em três níveis complementares, descritos de seguida.

3.1 Arquitetura Lógica de Nível 1 – Vista Lógica Global

A vista lógica de Nível 1 (Fig. 1) apresenta o "Sistema de Gestão e Monitorização de Data Center" como o componente central da arquitetura. Este sistema é responsável por integrar ferramentas díspares através de desenvolvimentos personalizados, nomeadamente um plugin NetBox-Zabbix e scripts de automação, resolvendo assim a fragmentação de ferramentas anterior. O Administrador de TI/SRE interage com este sistema,

que orquestra três serviços externos essenciais: 1) O NetBox, estabelecido como a SSoT para o inventário, onde a solução executa operações de leitura/escrita e expande a interface nativa; 2) O Zabbix, como sistema de monitorização, cuja configuração de hosts e templates é totalmente automatizada pela solução; e 3) O servidor LDAP, que fornece autenticação centralizada e segura para ambas as plataformas. Esta arquitetura garante a transição para uma solução integrada, coesa e robusta para a gestão da infraestrutura.

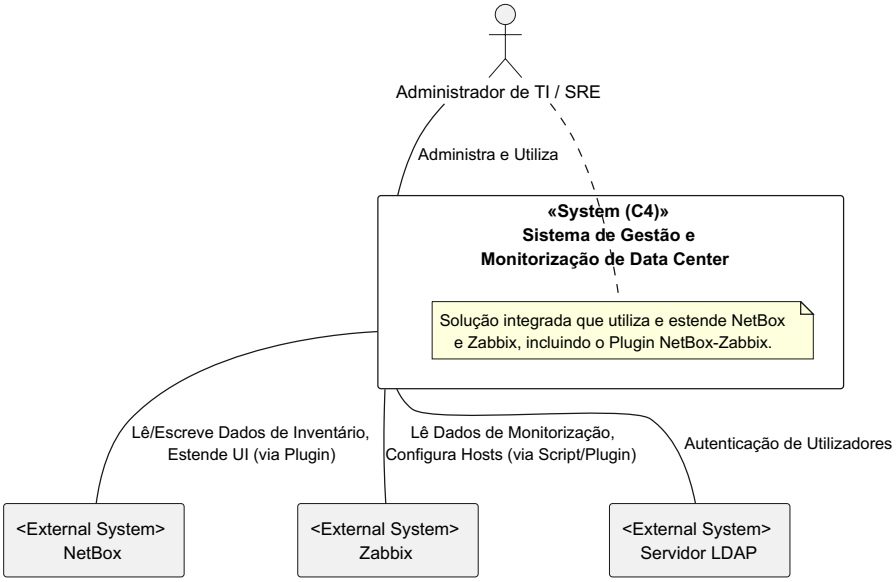

Fig. 1. Vista Lógica de Nível 1

3.2 Arquitetura Lógica de Nível 2 – Decomposição por Camadas/Módulos

A arquitetura lógica de Nível 2 (Fig. 2) decompõe o sistema central em três containers funcionais que promovem uma solução coesa. O primeiro, o NetBox (Sistema de Inventário), atua como a "Fonte da Verdade" (SSoT) e é composto pelo NetBox Core App e pelo Plugin NetBox-Zabbix. Este plugin permite ao Administrador/SRE importar e visualizar hosts descobertos no Zabbix diretamente no NetBox. O segundo container, o Zabbix (Sistema de Monitorização), formado pelo Zabbix Server e Agents, gere a recolha de métricas e alertas, interagindo com o plugin NetBox-Zabbix através da sua API para facilitar a sincronização. Por fim, o Servidor LDAP opera como um sistema externo que centraliza a autenticação de utilizadores para ambas as plataformas, NetBox e Zabbix. Esta arquitetura modular, baseada em APIs e ferramentas open-source, substitui a fragmentação de sistemas anterior por uma solução integrada, flexível e escalável.

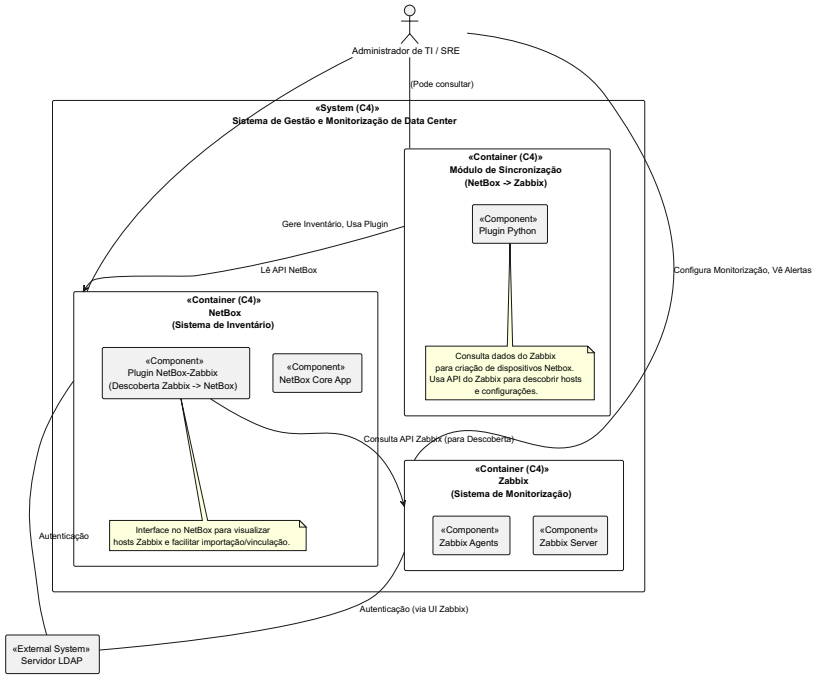

Fig. 2. Vista Lógica de Nível 2

3.3 Arquitetura Lógica de Nível 3 - Design Interno da Integração (Plugin)

A Fig. 3 apresenta a arquitetura interna detalhada do Plugin NetBox-Zabbix, um componente essencial para assegurar a integração e sincronização entre o sistema de inventário NetBox e a monitorização no Zabbix. Este plugin é estruturado de forma modular, com vários componentes internos claramente definidos, cada um com responsabilidades específicas e interações bem estabelecidas.

O ponto central do plugin é o componente Plugin Core (__init__.py), que serve como entrada principal e gere o ciclo de vida do plugin dentro do NetBox Core Framework. Este módulo é responsável pela inicialização e integração geral dos restantes componentes com o NetBox.

A comunicação com o sistema de monitorização é realizada através do Zabbix API Client (api_request.py), que implementa a interface IZabbixAPIAdapter. Este componente gere todas as requisições à API JSON-RPC do Zabbix, garantindo uma comunicação eficiente e estruturada para obter e enviar dados necessários às operações do plugin.

Para representar internamente os hosts descobertos no Zabbix, o plugin usa os Data Models (models.py), nomeadamente a entidade DiscoveredHost. Este modelo de dados, ligado diretamente à base de dados do NetBox através do seu ORM, fornece a estrutura e métodos necessários para manipular os dados relacionados com os hosts a serem sincronizados.

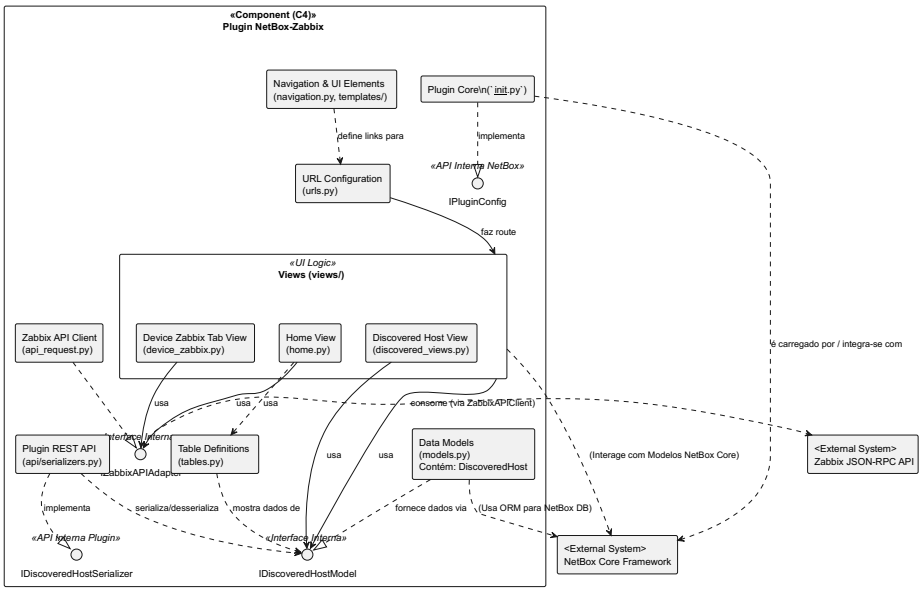

Fig. 3. Vista Lógica de Nível 3 – NetBox-Zabbix Plugin

A interação direta com o utilizador acontece através das Views (views/). Estas são responsáveis pela lógica da interface, incluindo a página principal (Home View), a página de visualização e importação de hosts descobertos (Discovered Host View), e uma aba específica nos detalhes dos dispositivos NetBox para apresentar informações provenientes do Zabbix (Device Zabbix Tab View). As Views recolhem os dados dos Data Models, utilizam o Zabbix API Client para obter informações externas e são disponibilizadas ao utilizador através da configuração de URLs (urls.py).

A configuração dos endpoints da aplicação é feita pelo componente URL Configuration (urls.py), que mapeia claramente cada URL para as Views correspondentes. Já o agrupamento e apresentação visual da interface são suportados pelo componente de Navigation & UI Elements (navigation.py, templates/), que define a forma como o plugin aparece no NetBox e fornece os templates necessários para exibição na interface web.

Os dados exibidos nas interfaces são organizados pelo módulo de Table Definitions (tables.py), estruturando claramente a apresentação das informações provenientes dos Data Models em tabelas visuais consistentes.

Finalmente, o plugin disponibiliza uma interface programática através do componente Plugin REST API (api/serializers.py). Este permite a serialização e desserialização dos dados relacionados com os hosts descobertos, facilitando interações futuras com outros sistemas ou componentes internos do NetBox.

O fluxo principal de interação dentro do plugin inclui duas funcionalidades fundamentais. Primeiro, a descoberta e importação de hosts do Zabbix para o NetBox: o utilizador acede à Discovered Host View, o plugin utiliza o Zabbix API Client para obter

a lista atualizada de hosts disponíveis no Zabbix, e permite ao utilizador importá-los diretamente para o inventário do NetBox através dos Data Models. Segundo, a navegação direta entre NetBox e Zabbix: quando o utilizador consulta detalhes de um dispositivo, uma aba específica permite visualizar rapidamente informações relevantes provenientes do Zabbix sem necessidade de trocar manualmente entre interfaces.

Em resumo, a modularidade desta arquitetura garante flexibilidade e facilidade na manutenção do plugin.

3.4 Componentes do NetBox-Zabbix Plugin

O Diagrama de Componentes (Fig. 4) ilustra a arquitetura modular do NetBox-Zabbix Plugin. O Plugin Core atua como orquestrador central, interagindo com o NetBox Core Framework. A comunicação externa é gerida por um Zabbix API Client dedicado, que implementa uma interface (IZabbixAPIAdapter) para abstrair as chamadas à API JSON-RPC do Zabbix. O módulo Data Models utiliza o ORM do Django para persistir dados na base de dados do NetBox, com destaque para o modelo DiscoveredHost. A lógica de interface (Views) fornece os pontos de interação do utilizador, incluindo a Home View (descoberta), a Discovered Host View (gestão/importação) e a Device Zabbix Tab View (integrada na página do dispositivo NetBox). Módulos auxiliares de URL Configuration e Navigation & UI Elements asseguram a integração visual. Adicionalmente, uma Plugin REST API interna expõe os dados programaticamente, garantindo a modularidade e extensibilidade da solução.

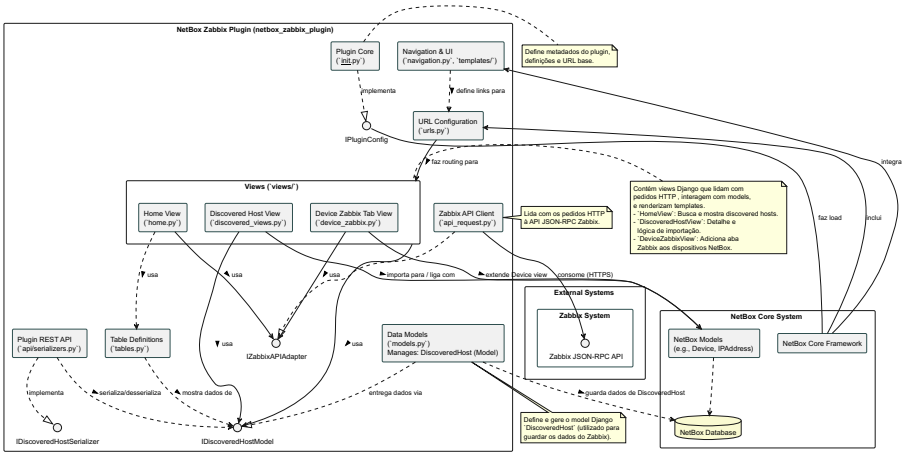

Fig. 4. Diagrama de Componentes – NetBox-Zabbix Plugin

4 Implementação e Avaliação da Solução

Este capítulo descreve em detalhe a implementação da solução definida anteriormente, abordando as tecnologias utilizadas, configuração dos sistemas, integração entre componentes e a validação da solução, com foco na centralização da informação, automatização, escalabilidade e facilidade de manutenção.

4.1 Descrição da Implementação

A implementação da arquitetura foi realizada utilizando uma stack tecnológica open-source composta por NetBox, Zabbix, Python, LDAP (FreeIPA) e NGINX.

- NetBox (Inventário e DCIM): Implementado como "Fonte da Verdade" (SSoT) num container Proxmox. A estruturação do inventário foi realizada por inserção manual e complementada por um plugin de migração (Python) desenvolvido para importar dados de rede do sistema legado LibreNMS.
- Integração NetBox-Zabbix: Foi desenvolvido um plugin Python personalizado que utiliza as APIs JSON-RPC e REST para garantir a consistência de dados. O plugin permite a importação de hosts do Zabbix para o NetBox e a visualização de hiperligações diretas para os dashboards Zabbix a partir da interface do NetBox.
- Zabbix (Monitorização Centralizada): Substituiu um conjunto de ferramentas fragmentadas (Monit, Munin, etc.). A implementação destacou-se pela automação do deployment do Zabbix Agent 2 através de scripts Bash, que configuram os agentes para auto registo, monitorização de systemd e execução de ações corretivas remotas. A solução utiliza LLD para descoberta automática e ações corretivas para remediação de incidentes (e.g.: reinício de serviços).
- Autenticação e Acesso (LDAP e NGINX): O acesso às interfaces do NetBox e Zabbix foi unificado através de integração com o FreeIPA (LDAP). O NGINX foi utilizado como reverse proxy para ambas as aplicações, configurado com HTTPS e regras de segurança.
- Suporte Legacy: Foi garantida a compatibilidade com sistemas operativos mais antigos (e.g.: CentOS 6) e desenvolvidos templates de monitorização específicos para serviços baseados em SysV Init.

4.2 Testes e Validação

Embora não tenha sido implementada uma suíte de testes automatizada, a solução foi submetida a uma validação funcional pragmática em ambiente operacional, focada em quatro pilares:

- Validação da Precisão das Métricas: Verificação manual dos itens de monitorização do Zabbix contra a execução de comandos nativos nos hosts (e.g.: df -h). As falhas detetadas foram validadas de forma cruzada com logs de sistema e com as ferramentas legadas (M/Monit) para assegurar a fiabilidade e prevenir falsos positivos.
- Validação do Ciclo de Alerta e Remediação: Simulação controlada de falhas, incluindo a paragem intencional de serviços (e.g.: systemctl stop nginx) e o esgotamento de recursos (usando stress e fallocate). Validou-se a ativação de triggers, a entrega de notificações (SMS/Chat) e a execução de ações corretivas automáticas (e.g.: reinício de serviço).
- Validação da Integração (Plugin NetBox-Zabbix): Testes focados no fluxo de sincronização, confirmando a comunicação API, a listagem de hosts descobertos na interface do NetBox, a importação correta de dados e o funcionamento das hiperligações de acesso rápido ao Zabbix.

- Validação da Autenticação Centralizada (LDAP): Confirmação de que o acesso às interfaces do NetBox e Zabbix era corretamente gerido pelo diretório LDAP, validando logins bem-sucedidos e a aplicação correta do mapeamento de grupos para os perfis de permissão internos de cada aplicação.

4.3 Avaliação da Solução

A avaliação da solução demonstrou ganhos operacionais e técnicos significativos, validando a abordagem de integração. Os principais resultados incluem:

- Centralização do Inventário: A implementação do NetBox como SSoT eliminou a dispersão e redundância de dados, assegurando um inventário de ativos físicos e lógicos com elevada precisão.
- Automação de Processos: O plugin de integração NetBox-Zabbix automatizou o ciclo de vida dos hosts (criação/atualização) entre o inventário e a monitorização. Adicionalmente, as ações corretivas automáticas no Zabbix reduziram o tempo de resposta a incidentes (MTTR) e a dependência de intervenção manual.
- Monitorização e Resposta a Incidentes: O Zabbix centralizou a monitorização, proporcionando uma plataforma unificada para deteção de falhas, triggers personalizadas e mecanismos de alerta proativos (chat/SMS), minimizando o tempo de inatividade.
- Reforço da Segurança: A integração com um servidor LDAP centralizado para ambas as plataformas (NetBox e Zabbix) unificou a gestão de acessos e permissões, simplificando a administração e reforçando a conformidade com as políticas de segurança.
- Manutenção e Escalabilidade: A adoção de ferramentas open-source robustas e o design modular do plugin garantem a sustentabilidade da solução e facilitam a extensibilidade futura, como a implementação de sincronização automática de ativos.

5 Conclusões

Este trabalho detalhou a implementação de uma solução integrada de gestão de infraestrutura, unificando o NetBox como SSoT para inventário e o Zabbix para monitorização centralizada. Os objetivos do projeto foram alcançados: o NetBox centralizou o inventário de ativos e um plugin NetBox-Zabbix foi desenvolvido para facilitar a migração de dados. A plataforma Zabbix consolidou a monitorização, a alarmística e as ações corretivas automáticas, absorvendo scripts e cronjobs anteriormente fragmentados.

As principais limitações identificadas incluem a ausência de sincronização em tempo real entre as plataformas, a falta de uma configuração de alta disponibilidade (HA) e a inexistência de testes formais. Como trabalho futuro, propõe-se a evolução do plugin para suportar sincronização em tempo real, a implementação de HA e a expansão da solução para o ambiente de data center, acompanhada pelo desenvolvimento de uma suíte de testes formais, que possibilitem uma avaliação robusta.

Referências

1. Data Center Dynamics: The resurgence of DCIM: Navigating the future of data center management.
   Disponível em: https://www.datacenterdynamics.com/en/opinions/the-resurgence-of-dcim-navigating-the-future-of-data-center-management/
2. Zabbix Blog: Zabbix migration in a mid-sized bank environment. Disponível em: https://blog.zabbix.com/zabbix-migration-in-a-mid-sized-bank-environment/13040/
3. anarcat (blog): Replacing Smokeping with Prometheus. Disponível em: https://anarc.at/blog/2020-06-04-replacing-smokeping-prometheus/
4. StackShare: Munin vs Prometheus — What are the differences? Disponível em: https://stackshare.io/stackups/munin-vs-prometheus
5. NetBox Labs: Is a Network Source of Truth Essential for Automation? Disponível em: https://netboxlabs.com/blog/do-you-need-a-source-of-truth-for-network-automation-not-at-first/
6. Zabbix Blog: NetBox as Home CMDB and Integrated with Zabbix. Disponível em: https://blog.zabbix.com/netbox-as-home-cmdb-and-integrated-with-zabbix/29324/
7. DZone: Introduction to Grafana, Prometheus, and Zabbix. Disponível em: https://dzone.com/articles/introduction-to-grafana-prometheus-and-zabbix
8. Syed Asif: Ansible and NetBox — Ansible for Network Automation. Disponível em: https://medium.com/@sydasif78/ansible-and-netbox-896f14d991d5
9. Red Hat: What is Infrastructure as Code (IaC)? Disponível em: https://www.redhat.com/en/topics/automation/what-is-infrastructure-as-code-iac
10. Wang, C., Schwan, K., Talwar, V., Eisenhauer, G., Hu, L., Wolf, M.: A Flexible Architecture Integrating Monitoring and Analytics for Managing Large-Scale Data Centers. In: 8th IEEE International Conference on Autonomic Computing (ICAC 2011), pp. 141–150. IEEE (2011)
11. Gutierrez-Aguado, J., Alcaraz Calero, J.M., Diaz Villanueva, W.: IaaSMon: Monitoring Architecture for Public Cloud Computing Data Centers. *Journal of Grid Computing* 14(2), 283–297 (2016). https://doi.org/10.1007/s10723-015-9357-4
12. Dong, W.: AIOps Architecture in Data Center Site Infrastructure Monitoring. *Computational Intelligence and Neuroscience* 2022, 1988990 (2022). https://doi.org/10.1155/2022/1988990