Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Engenharia de Software

Meia noite eu te conto: Uma Análise Sobre Arquitetura de um MVP de Software de Gerenciamento de Segredos

Autora: Felipe Campos de Almeida

Orientadora: Profa. Dra. Carla Rocha Aguiar

Brasília, DF

2023

Felipe Campos de Almeida

Meia noite eu te conto: Uma Análise Sobre Arquitetura de um MVP de Software de Gerenciamento de Segredos

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Profa. Dra. Carla Rocha Aguiar

Brasília, DF

2023

Felipe Campos de Almeida

Meia noite eu te conto: Uma Análise Sobre Arquitetura de um MVP de Software de Gerenciamento de Segredos/ Felipe Campos de Almeida. – Brasília, DF, 2023-

53 p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Carla Rocha Aguiar

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2023.

1. Gerenciamento de Segredos. 2. Software de Gerenciamento de Segredos. I. Profa. Dra. Carla Rocha Aguiar. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Meia noite eu te conto: Uma Análise Sobre Arquitetura de um MVP de Software de Gerenciamento de Segredos

CDU 02:141:005.6

Felipe Campos de Almeida

Meia noite eu te conto: Uma Análise Sobre Arquitetura de um MVP de Software de Gerenciamento de Segredos

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 21 de dezembro de 2023:

Profa. Dra. Carla Rocha Aguiar
Orientador

Brasília, DF
2023

Este trabalho é dedicado com todo amor e gratidão à minha mãe, ao meu pai e à minha irmã, cujo apoio incondicional foi a base de tudo que consegui alcançar.

Resumo

A segurança cibernética tem se tornado uma questão de importância crescente à medida que a digitalização das operações empresariais e a conectividade online continuam a se expandir. Nesse contexto, a proteção de informações confidenciais contra ameaças digitais tornou-se um pilar fundamental para a sustentabilidade e o sucesso das organizações.

Neste panorama, os Softwares de Gerenciamento de Segredos (SGS) emergem como ferramentas importantes na salvaguarda de dados sensíveis, como senhas, chaves de API e certificados, contra acessos não autorizados, além de facilitarem a conformidade com as normas de segurança estabelecidas. A popularidade e a necessidade desses softwares têm crescido proporcionalmente aos desafios impostos pelo cenário de ameaças em evolução, levando a um aumento significativo na adoção de práticas de segurança robustas.

O objetivo deste estudo é estabelecer os elementos de uma arquitetura básica e os requisitos essenciais para o desenvolvimento de um Software de Gerenciamento de Segredos (SGS).

Palavras-chave: Gerenciamento de Segredos, Software de Gerenciamento de Segredos, Hashicorp Vault.

Lista de ilustrações

Figura 1 – Autenticação e Autorização 20
Figura 2 – Replicação de instancias 23
Figura 3 – Segmentação da Chave Principal em sub-chaves 24
Figura 4 – Descrição da metodologia 25
Figura 5 – Representação do Modelo C4 27
Figura 6 – Diagrama de Contexto 32
Figura 7 – Diagrama de Contêineres 33

Lista de tabelas

Tabela 1 – Comparação das Características das Ferramentas de Gerenciamento de Segredos. Os dados foram coletados em 07/06/2020. 31
Tabela 2 – Requisitos: Orquestração 34
Tabela 3 – Requisitos: API 34
Tabela 4 – Requisitos: Autenticação e Autorização 35
Tabela 5 – Requisitos: Registro e Auditoria 35
Tabela 6 – Requisitos: Integração 35
Tabela 7 – Requisitos: Banco de Dados 36
Tabela 8 – Requisitos: Comunicação Externa 36

Lista de abreviaturas e siglas

API	Application Programming Interface (Interface de Programação de Aplicação)
REST	Representational State Transfer (Transferência de Estado Representacional)
SGS	Software de Gerenciamento de Segredos
UML	Unified Modeling Language (Linguagem de Modelagem Unificada)

Sumário

1 INTRODUÇÃO 17
1.1 Objetivos 17
1.2 Estrutura de Trabalho 17

2 REFERENCIAL TEÓRICO 19
2.1 O que são segredos? 19
2.2 Estabelecimento de confiança, Autenticação vs. Autorização 19
2.2.1 Dispersão de Segredos 21
2.3 Gerenciamento de Segredos 21
2.4 Recursos e Práticas Comuns de Softwares de Gerenciamento de Segredos (SGSs) 21
2.4.1 Armazenamento de Segredos 21
2.4.2 Mutabilidade 21
2.4.3 Criptografia 22
2.4.4 Lista de Controle de Acesso 22
2.4.5 Auditoria 22
2.4.6 Segredos Dinâmicos 22
2.4.7 Disponibilidade 23
2.4.8 Desbloqueio 23
2.4.9 Limite de Autenticação 24

3 PROPOSTA 25
3.1 Questão de Pesquisa 25
3.2 Metodologia 25
3.2.1 Revisão Teórica e Literária 26
3.2.2 Levantamento de Requisitos 26
3.2.3 Proposta de Arquitetura 26
3.3 Tecnologias 26
3.3.1 Modelo C4 26
3.3.1.1 Contexto 27
3.3.1.2 Contêineres 27
3.3.1.3 Componentes 27
3.3.1.4 Código 28
3.3.2 PlantUML 28

4 RESULTADOS 29

4.1 Qual a forma correta de gerenciar segredos? . . . 29
4.2 Análise de Serviços de Gerenciamento de Segredos . . . 30
4.3 Proposta de Arquitetura . . . 32
4.3.1 Diagrama de Contexto . . . 32
4.3.2 Diagrama de Contêineres . . . 33
4.3.2.1 Descrição dos Contêineres e Requisitos . . . 34
4.3.2.1.1 Orquestração . . . 34
4.3.2.1.2 API . . . 34
4.3.2.1.3 Autenticação e Autorização . . . 34
4.3.2.1.4 Registro e Auditoria . . . 35
4.3.2.1.5 Integração . . . 35
4.3.2.1.6 Banco de Dados . . . 36
4.3.2.1.7 Comunicação Externa . . . 36
4.3.2.2 Descrição das relações entre Contêineres . . . 36
4.3.2.3 API para Orquestração . . . 36
4.3.2.4 Autenticação e Autorização para API . . . 37
4.3.2.5 Autenticação e Autorização para Orquestração . . . 38
4.3.2.6 Comunicação Externa para Orquestração . . . 39
4.3.2.7 Orquestração para Registro e Auditoria . . . 39
4.3.2.8 Orquestração para Integração . . . 40
4.3.2.9 Banco de Dados e Software de Gerenciamento de Segredos . . . 41

5 CONCLUSÃO E TRABALHOS FUTUROS . . . 43

REFERÊNCIAS . . . 45

APÊNDICES 47

APÊNDICE A – CÓDIGOS PLANTUML . . . 49
A.1 Autenticação e Autorização . . . 49
A.2 Diagrama de Contexto . . . 49
A.3 Diagrama de Contêineres . . . 50
A.4 Modelo C4 . . . 51
A.5 Replicação de Instancias . . . 52
A.6 Compartilhamento de Segredos de Shamir . . . 53

1 Introdução

No mundo atual, onde a segurança da informação se tornou um aspecto crítico para a sobrevivência e o sucesso das organizações, a gestão eficaz de segredos - como senhas, chaves de API e certificados - é fundamental. Um Software de Gerenciamento de Segredos (SGS) desempenha um papel vital em proteger esses dados sensíveis contra acessos não autorizados e em garantir a conformidade com as normas de segurança.

A construção de um sistema de gerenciamento de segredos eficaz é uma tarefa complexa e multifacetada, crucial para salvaguardar informações sensíveis em um ambiente digital cada vez mais interconectado. À medida que organizações de todos os tamanhos enfrentam desafios crescentes em proteger dados críticos, a necessidade de uma abordagem sistemática e robusta para a gestão de segredos torna-se primordial.

Este trabalho propõe a condução de uma investigação das práticas atuais no campo, buscando compreender as soluções existentes e as funcionalidades que elas oferecem, afim de entender quais são os pilares arquiteturais de um Software de Gerenciamento de Segredos.

1.1 Objetivos

Este trabalho tem como objetivo apresentar uma visão geral sobre a arquitetura de um Software de Gerenciamento de Segredos. Para que esse resultado seja alcançado, os seguintes objetivos específicos devem ser cumpridos:

- Identificar soluções existentes e suas funcionalidades básicas
- Identificar as práticas mais comuns relacionadas á arquitetura de um Software de Gerenciamento de Segredos
- Definir a arquitetura e requisitos básicos de um Software de Gerenciamento de Segredos

1.2 Estrutura de Trabalho

Este trabalho se apresenta com:

- Um capítulo de referencial teórico, que explica o contexto dos tópicos significativos abordados no estudo e trabalhos relacionados

- Um capítulo de proposta que define a proposta do trabalho, as questões de pesquisa, metodologia e tecnologias utilizadas
- Um capítulo de resultados
- Um capítulo para a conclusão do estudo.

2 Referencial Teórico

2.1 O que são segredos?

À medida que a Engenharia de Software progride, a interligação entre diferentes componentes de software se acentua de maneira constante. A cada ano, novas inovações tecnológicas emergem, e sistemas que há uma década seriam concebidos com base em uma arquitetura monolítica simples, agora ostentam um nível de complexidade substancial no que tange à sua integração com outras aplicações.

Na comunicação entre as diversas aplicações envolvidas em um projeto ou organização, é essencial estabelecer um meio que assegure o acesso seguro apenas entre as partes que devem se comunicar.

Um segredo é um objeto que você deseja proteger contra vazamento para conhecimento público, pois representa algo fornecido por uma parte à outra para autenticar e estabelecer confiança (GATEV, 2021). Esses segredos geralmente consistem em credenciais, como senhas, chaves de API ou outras informações, que concedem autenticação ou autorização a um sistema. O acesso a esses segredos deve ser rigorosamente controlado e auditado para garantir a integridade de sistemas e a segurança de dados confidenciais (BLOMQVIST, 2021).

2.2 Estabelecimento de confiança, Autenticação vs. Autorização

Os conceitos de autenticação e autorização frequentemente são equivocadamente considerados sinônimos, no entanto, eles desempenham funções e propósitos diferentes.

A autenticação determina se a entidade que solicita acesso é quem afirma ser, enquanto a autorização determina quais recursos uma entidade autenticada pode ou não acessar (HE; YANG, 2017). A autenticação acontece primeiro, pois apenas uma entidade conhecida pelo sistema pode ser autorizada a acessar qualquer recurso.

Exemplificando os conceitos, considere o exemplo de uma pessoa que deseja visitar um parque de diversões. Primeiramente, essa pessoa precisa comprovar sua entrada apresentando um ingresso válido na entrada, o que serve como um processo de autenticação, garantindo que ela tenha o direito de acessar o espaço do parque. Essa etapa inicial verifica a identidade do visitante, assegurando que ele adquiriu o direito de entrar no ambiente controlado do parque.

Uma vez dentro do parque, a pessoa se depara com várias atrações, cada uma com

seus próprios requisitos específicos para acesso. Para garantir a segurança dos visitantes, o acesso à roda gigante é restrito a indivíduos com altura superior a 1,5 metros. Isso implica que, mesmo que a pessoa tenha sido autenticada ao entrar no parque, sua autorização para acessar determinadas atrações depende de critérios adicionais, neste caso, a altura. De forma semelhante, o carrossel possui uma restrição de altura máxima de 1,9 metros, visando proteger tanto as crianças quanto os adultos de possíveis perigos ou desconfortos.

Portanto, mesmo estando autenticada para adentrar o parque, a pessoa pode encontrar limitações na sua autorização para usufruir de todas as atrações disponíveis. Esse processo destaca a diferença entre estar autenticado para acessar um local ou sistema e estar autorizado para realizar certas ações ou acessar certos recursos dentro desse ambiente. Essa distinção é crucial para entender como os sistemas de segurança funcionam, não apenas em ambientes físicos como parques de diversões, mas também em contextos digitais, onde a autenticação e a autorização desempenham papéis fundamentais na proteção de informações e na garantia de que apenas usuários autorizados possam acessar determinados dados ou executar certas operações.

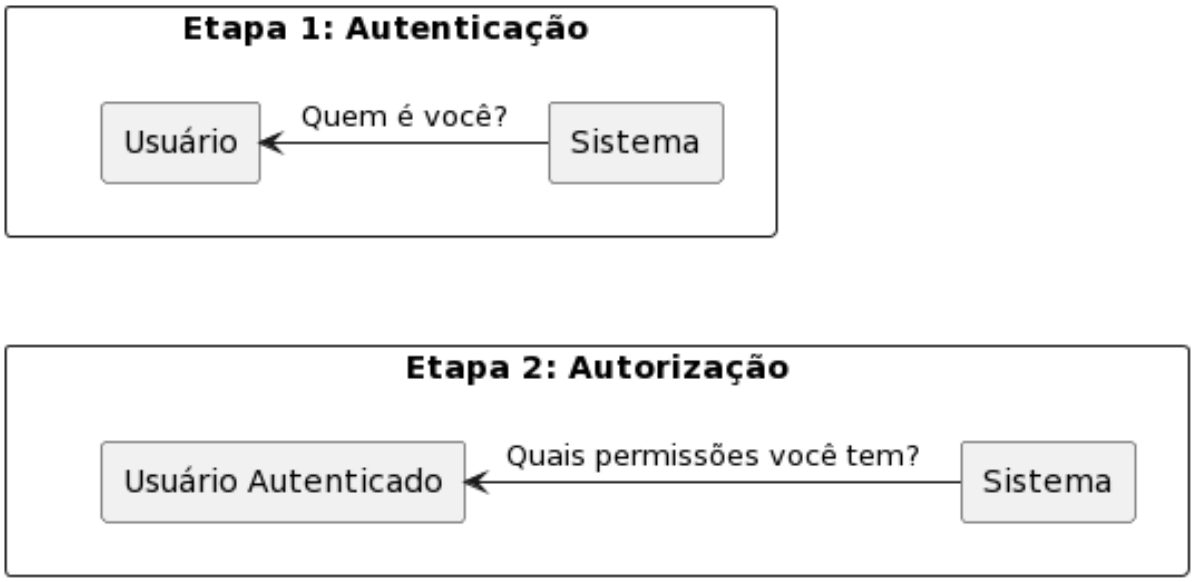

Figura 1 – Autenticação e Autorização

2.2.1 Dispersão de Segredos

O conceito de dispersão de segredos descrito por (HashiCorp Vault, 2018) enfatiza a ideia fundamental de que uma gestão inadequada de informações sensíveis pode resultar em sua disseminação não intencional por vários pontos de uma aplicação ou organização, abrangendo áreas como código-fonte, arquivos de configuração e aplicativos de mensagens utilizados pela equipe de desenvolvimento. Com o tempo, essa propagação inadvertida pode culminar na presença não controlada de informações confidenciais, representando um risco significativo.

2.3 Gerenciamento de Segredos

O Gerenciamento de Segredos tem como propósito organizar o armazenamento e a administração das informações confidenciais utilizadas por outras aplicações, garantindo o acesso exclusivo aos segredos necessários para o funcionamento de cada aplicação.

2.4 Recursos e Práticas Comuns de Softwares de Gerenciamento de Segredos (SGSs)

De acordo com a arquitetura descrita por (HashiCorp Vault, 2018), e os trabalhos de (RAHMAN; BARSHA; MORRISON, 2021), (BASAK et al., 2022) e (BLOMQVIST, 2021), as práticas e recursos mais comuns para este tipo de aplicação são:

2.4.1 Armazenamento de Segredos

A funcionalidade mais básica e central de uma plataforma de gerenciamento de segredos é manter segredos em um lugar seguro. Dessa forma, que na maioria dos casos são guardados por um par chave e valor. O armazenamento de segredos, e consequentemente, seu compartilhamento seguro, remove a necessidade de envio de segredos por meio de e-mails ou qualquer aplicativo de mensagens, como Microsoft Teams ou Slack, onde basta uma conta comprometida para expor dados confidenciais (BASAK et al., 2022).

2.4.2 Mutabilidade

Para assegurar tanto conveniência quanto segurança, é crucial viabilizar a modificação descomplicada dos segredos. Em algumas situações, torna-se imperativo alterar os segredos na plataforma, seja devido a mudanças no banco de dados ou para lidar com um possível comprometimento de um segredo, demandando, assim, a capacidade de realizar modificações.

Conforme discutido por Rahman, Barsha e Morrison (2021) e Basak et al. (2022), mudanças nos segredos devem ocorrer periodicamente para garantir que, mesmo que ocorra um vazamento não detectado, os segredos não possam ser usados para comprometer o sistema no futuro.

2.4.3 Criptografia

O banco de dados do software de gerenciamento de segredos deve ser submetido à criptografia, evitando, assim, o armazenamento de quaisquer segredos em formato legível. Essa abordagem assegura a proteção dos segredos, mesmo diante de possíveis violações de segurança e eventuais comprometimentos do banco de dados do aplicativo.

2.4.4 Lista de Controle de Acesso

As entidades que utilizam o sistema devem passar pelo processo de autenticação e possuir permissões estritamente concedidas para acessar somente os recursos indispensáveis ao cumprimento eficiente de suas atribuições. A divulgação de segredos deve se restringir àqueles que realmente necessitam recebê-los.

2.4.5 Auditoria

A implementação de um sistema de registro se torna essencial para oferecer transparência nas operações ligadas aos segredos na plataforma. É crucial preservar o registro das ações de criação, modificação e acesso aos segredos, a fim de possibilitar a detecção de vazamentos ou usos inadequados dos mesmos.

2.4.6 Segredos Dinâmicos

Intitulados também como segredos efêmeros, este conceito delineado pela (HashiCorp Vault, 2018) abrange a prática de não manter segredos de forma estática. Em vez disso, preconiza-se a geração sob demanda de segredos de uso único para cada cliente, como medida adicional para minimizar as chances de comprometimento de credenciais. Caso um dos segredos seja de alguma maneira exposto, não se torna necessário interromper as operações de todas as aplicações que empregam tal segredo. Esta abordagem permite a criação de um novo segredo sem interrupção. Entretanto, essa abordagem demanda a criação de interfaces específicas para possibilitar a geração e eliminação de credenciais sob demanda em diversas instâncias, como bancos de dados e APIs, entre outros sistemas.

2.4.7 Disponibilidade

Um software de gerenciamento de segredos desempenha um papel fundamental na operação de qualquer organização. Um funcionamento inadequado pode ocasionar disrupções ou falhas em todos os aplicativos que dependem desses segredos. Para garantir a máxima disponibilidade do aplicativo, a abordagem adotada pelo (HashiCorp Vault, 2018) envolve a utilização de múltiplas instâncias espelhadas do aplicativo operando em conjunto. Conforme elucida a (Figura 2) nesse cenário uma instância é designada como líder, direcionando todas as requisições de segredos para ela. A respeito da disponibilidade, também é discutida a prática de replicação por (RAHMAN; BARSHA; MORRISON, 2021).

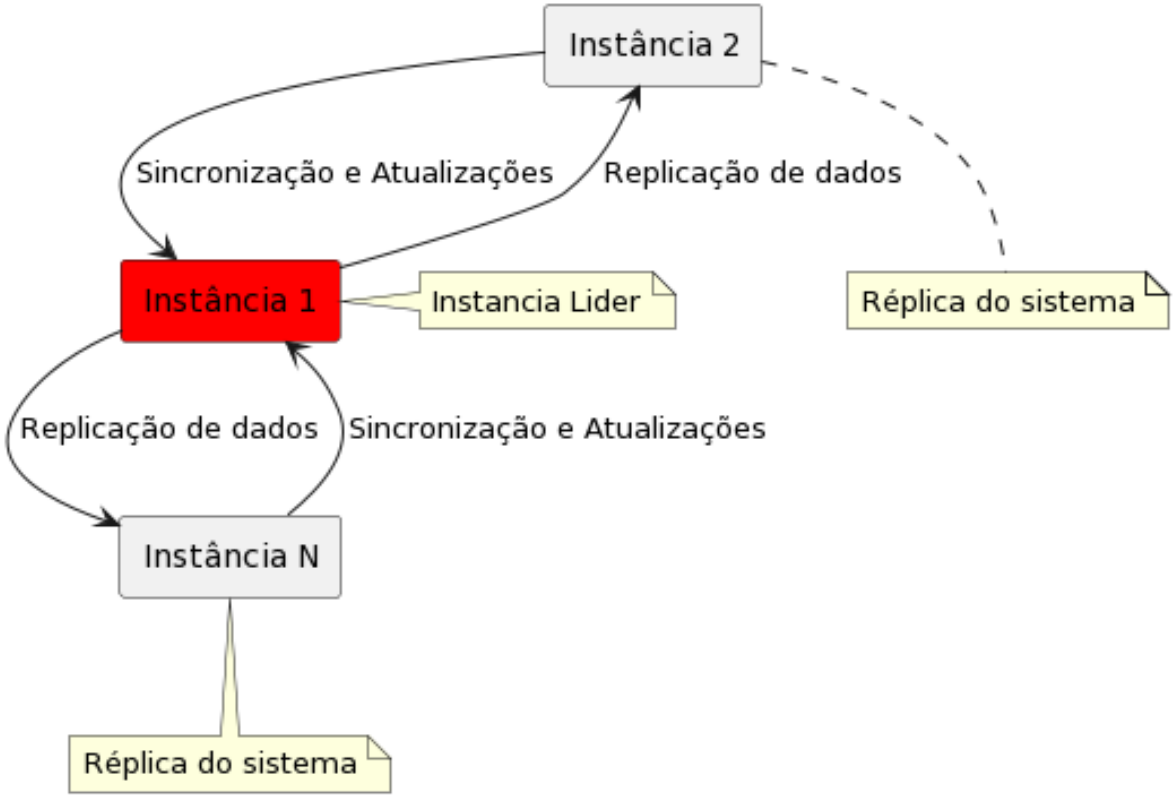

Figura 2 – Replicação de instancias

2.4.8 Desbloqueio

Ao incorporar uma aplicação de gerenciamento de segredos, é imperativo gerar automaticamente uma chave para criptografar os dados confidenciais destinados a serem armazenados no banco de dados - essa chave age como o selo da aplicação. Conhecida como Chave Principal, trata-se de uma chave singular que possibilita o acesso aos dados cifrados. A arquitetura proposta por (HashiCorp Vault, 2018) fundamenta-se no princípio

de compartilhamento secreto formulado por (SHAMIR, 1979) para criar essa chave. Conforme ilustra a (Figura 3) esquema de Compartilhamento Secreto de Shamir segmenta a chave principal em n partes e autoriza a reconstrução da chave por meio da combinação de k sub-chaves, independentemente da ordem. O conhecimento de k - *1* sub-chaves não oferece nenhuma informação acerca da chave principal.

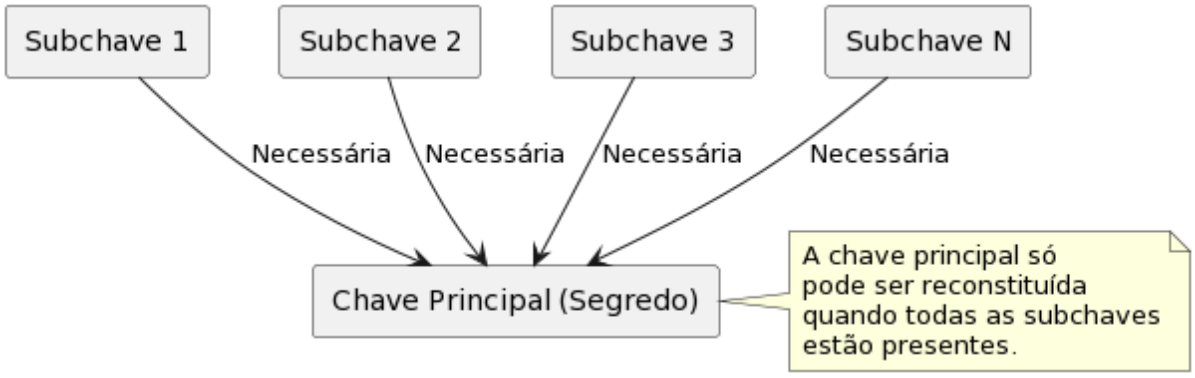

Figura 3 – Segmentação da Chave Principal em sub-chaves

2.4.9 Limite de Autenticação

É de extrema importância evitar a ocorrência de repetidas tentativas de autenticação mal sucedidas. Essa medida visa a prevenção de ataques de força bruta, nos quais invasores tentam adivinhar senhas por meio de inúmeras tentativas. Ao restringir e limitar o número de tentativas mal-sucedidas, o sistema aumenta sua resistência contra ataques desse tipo, assegurando um ambiente mais seguro e protegido para as operações de autenticação.

3 Proposta

Este trabalho propõe descrever os elementos básicos que compõem a arquitetura de um Produto Mínimo Viável de Software de Gerenciamento de Segredos.

3.1 Questão de Pesquisa

Este estudo busca analisar as abordagens possíveis, desafios e soluções mais utilizadas no mercado para o gerenciamento de segredos, com o objetivo de responder à seguinte pergunta de pesquisa:

QP *Qual é a arquitetura básica de um Software de Gerenciamento de Segredos?*

3.2 Metodologia

Neste trabalho será utilizada a metodologia de Pesquisa Exploratória, considerada adequada para investigações em campos pouco explorados. Possui como objetivo principal é familiarizar-se com um fenômeno que deve ser investigado (THEODORSON; THEODORSON, 1970). A escolha dessa metodologia possibilita um entendimento mais aprofundado do ambiente de gerenciamento de segredos e suas complexidades antes de se propor uma arquitetura de sistema. Será adotada uma abordagem qualitativa para a coleta de dados, incluindo revisões de literatura e análise de sistemas de gerenciamento de segredos existentes

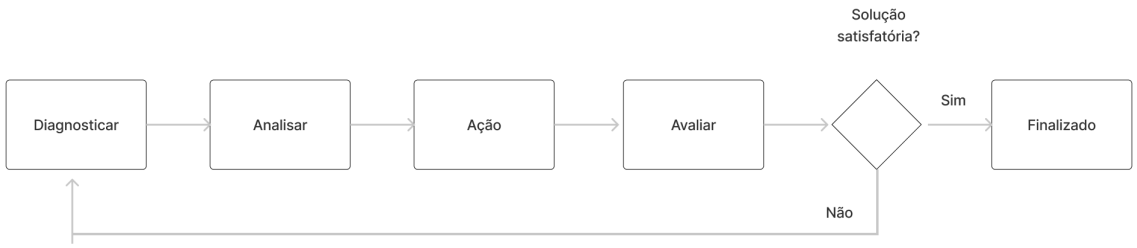

Figura 4 – Descrição da metodologia

3.2.1 Revisão Teórica e Literária

Nesta fase realiza-se uma revisão da literatura existente sobre Sistemas de Gerenciamento de Segredos, com o intuito de identificar os modelos arquiteturais predominantes e os componentes-chave utilizados neste contexto. A revisão aborda conceitos fundamentais, boas práticas, explora métodos tradicionais de gestão de segredos e destaca lacunas existentes nas abordagens utilizadas. Além disso, procede-se à identificação de tecnologias e boas práticas empregadas nos sistemas de gerenciamento de segredos, com especial ênfase em soluções arquitetônicas já implementadas. Este processo visa fornecer uma base sólida para a compreensão do estado atual da área, contribuindo para a fundamentação teórica da proposta arquitetural a ser desenvolvida.

3.2.2 Levantamento de Requisitos

Nesta fase compreende-se e enumera-se as necessidades, bem como os requisitos específicos para um sistema gerenciador de segredos avançado. Realiza-se uma análise abrangente abarcando tanto os requisitos funcionais quanto os não funcionais, proporcionando uma visão completa das demandas essenciais para o funcionamento do sistema.

3.2.3 Proposta de Arquitetura

Nesta fase, baseado nos dados coletados, é desenhada a proposta arquitetural básica para o Sistemas de Gerenciamento de Segredos, englobando componentes-chave, protocolos de segurança e a integração com sistemas externos. Conduz-se uma análise abrangente, discutindo quais os componentes necessários para o funcionamento básico de um Sistema de Gerenciamento de Segredos, utilizando como base os requisitos elicitados.

3.3 Tecnologias

3.3.1 Modelo C4

O Modelo C4 é delineado por quatro visualizações distintas (Contexto, Contêineres, Componentes, e Código) para descrever a arquitetura de software. O modelo C4 não impõe um conjunto rígido de regras na sua notação. Sua principal ênfase está em segmentar o sistema nos quatro níveis descritos e criar diagramas que sejam facilmente compreensíveis, ao invés de estabelecer uma sintaxe uniforme (VáZQUEZ-INGELMO; GARCíA-HOLGADO; GARCíA-PEñALVO, 2020).

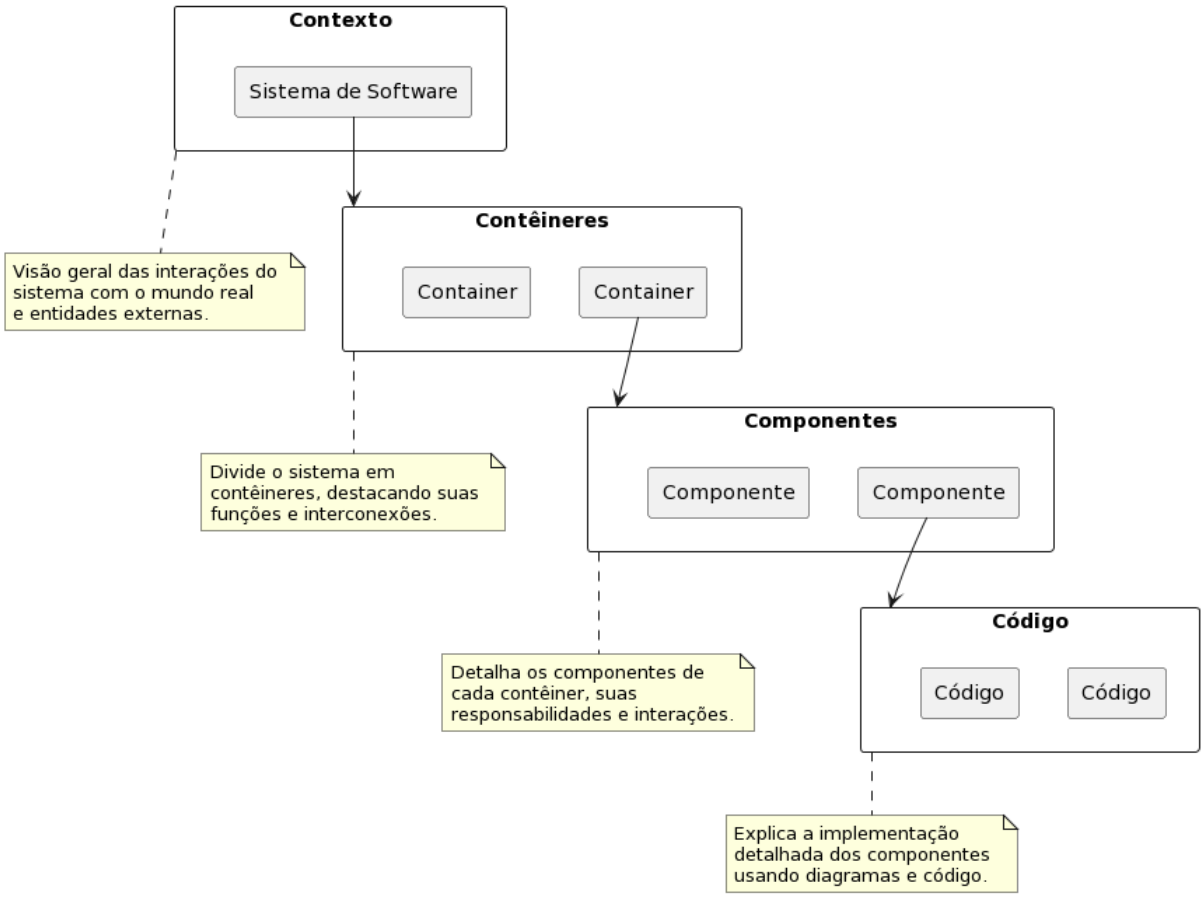

Figura 5 – Representação do Modelo C4

3.3.1.1 Contexto

Esta visualização fornece uma visão de alto nível do sistema. Ela mostra como o sistema em questão se encaixa no mundo real, identificando as várias entidades externas (como pessoas, sistemas e dispositivos externos) com as quais interage. O objetivo é entender as relações entre o sistema e seus usuários ou outros sistemas.

3.3.1.2 Contêineres

Esta camada detalha o sistema em um nível mais alto, dividindo-o em contêineres (aplicações, serviços de dados, etc.). Cada contêiner representa uma aplicação ou um serviço de dados e mostra as responsabilidades de alto nível e como os contêineres interagem entre si.

3.3.1.3 Componentes

Dentro de cada contêiner, o sistema é dividido em componentes. Esta visualização foca na organização e responsabilidades dos componentes dentro de um contêiner,

mostrando como eles interagem entre si para realizar as funções do contêiner. Cada componente representa uma parte do sistema com responsabilidades claramente definidas.

3.3.1.4 Código

A camada de visualização a mais detalhada, mostrando como os componentes são implementados em termos de classes, interfaces e outros artefatos de código. É aqui que os detalhes do design interno de cada componente são explorados, frequentemente utilizando diagramas de classes UML ou diagramas de estrutura semelhantes.

Este trabalho terá como foco as visões de **Contexto** e **Contêineres** do Modelo C4.

3.3.2 PlantUML

PlantUML é uma ferramenta de código aberto e gratuita, que permite a criação de diagramas UML utilizando uma notação de texto simples. Essa ferramenta foi desenvolvida visando a facilidade de atualização da documentação de software, incluindo todos os seus diagramas relevantes, com a conveniência de usar apenas um editor de texto, sem a necessidade de ferramentas adicionais. Isso torna o processo de manutenção da documentação mais ágil e integrado ao fluxo de trabalho de desenvolvimento (POLAJŽER, 2022). O modelo C4 beneficia-se especialmente da natureza flexível do PlantUML. Ao permitir que usuários descrevam estruturas de software e relações entre componentes de forma textual, o PlantUML torna a criação de diagramas do modelo C4 mais acessível e menos propensa a erros, comparada a métodos tradicionais baseados em interfaces gráficas. Esta integração não apenas economiza tempo, mas também garante uma consistência visual nos diagramas, facilitando a compreensão e a comunicação eficaz da arquitetura do software.

4 Resultados

4.1 Qual a forma correta de gerenciar segredos?

Em seu trabalho, (DOTSON, 2019) definiu quatro abordagens razoáveis para gerenciar segredos:

1. Mantenha os segredos apenas no sistema de implantação e controle estritamente o acesso a ele, onde apenas pessoal autorizado pode visualizar ou modificar os segredos.
2. Use um servidor para guardar segredos. O aplicativo implantado deve solicitar os segredos necessários do servidor.
3. Use um servidor para manter os segredos, mas o servidor de implantação recebe um token de uso único e o insere no aplicativo, que solicita os segredos e os mantém na memória.
4. Use o provedor de nuvem para fornecer a identidade e os metadados confiáveis para que os segredos possam ser distribuídos aos aplicativos apropriados.

Considerando o cenário atual da engenharia de software, é importante notar que algumas dessas abordagens apresentam limitações que podem ter impactos prejudiciais no ciclo de desenvolvimento de aplicações:

Abordagens 1 e 4:

1. Inadequação para resolver a questão da utilização de segredos em contextos alternativos, como ambientes de desenvolvimento local, ambientes de teste e ferramentas de compilação e publicação automatizada.
2. Foco exclusivo nos segredos no âmbito da produção ou em provedores de nuvem.

Abordagens 2 e 3:

1. Superam a limitação das abordagens 1 e 4.
2. Proporcionam flexibilidade para empregar segredos distintos de acordo com as demandas específicas.
3. Dinamismo: Adaptação dos segredos de acordo com o contexto de execução da aplicação.

4.2 Análise de Serviços de Gerenciamento de Segredos

A análise dos Serviços de Gerenciamento de Segredos neste trabalho foi baseada no estudo realizado por (KUZMINYKH; GHITA; SHIAELES, 2020). Este estudo avaliou 58 ferramentas de gerenciamento de segredos, com 32 delas sendo de código-fonte aberto e 26 de código fechado. A seleção focou em serviços que atendem aos requisitos de empresas de pequeno e médio porte, fornecendo uma base ideal para elicitar a arquitetura básica de um sistema de gerenciamento de segredos. Os critérios utilizados para a seleção incluem:

- **Disponibilidade de uma API REST:** Essencial para a integração com outras aplicações e sistemas, permitindo uma interação flexível e padronizada.
- **Capacidade de autenticação de usuários:** Fundamental para garantir que apenas usuários autorizados tenham acesso aos segredos, reforçando a segurança do sistema.
- **Política de controle de acesso:** Importante para definir e gerenciar quem pode acessar quais segredos, em que condições, oferecendo uma camada adicional de segurança.
- **Registro de atividades:** Crucial para o monitoramento e auditoria do sistema, permitindo rastrear quem acessou quais segredos e quando.
- **Comunicações seguras:** Garante que todas as transmissões de dados, especialmente os segredos, sejam realizadas de forma segura, protegendo contra interceptações e vazamentos de dados.
- **Possibilidade de backup:** Vital para a recuperação de dados em caso de falhas ou perdas, assegurando a continuidade do negócio e a integridade dos segredos.

A análise concluiu que somente cinco ferramentas atendiam aos requisitos e se qualificavam para um exame mais detalhado, são elas: Cyberark Conjur, Hashicorp Vault, OpenStack Barbican, Pinterest Knox e Square Keywhiz.

Seguindo a escolha das ferramentas, procedeu-se com uma avaliação comparativa, atribuindo pontos a cada Sistema Gerenciador de Segredos (SGS) conforme os critérios específicos estabelecidos:

- **1 Ponto:** Presença de testes unitários no código-fonte, Popularidade na comunidade de desenvolvedores.
- **2 Pontos:** Múltiplos Métodos de Autenticação, Início Automatizado, Documentação Compreensiva, Suporte a Módulo de Segurança de Hardware, Disponibilidade para Uso Comercial, Suporte Técnico para Desenvolvedores.

- **3 Pontos:** Armazenamento Seguro, Registros de Auditoria, Controle de Acesso, Sem Vulnerabilidades Conhecidas, Alto Impacto na Comunidade, Manutenção Ativa e Desenvolvimento.

Tabela 1 – Comparação das Características das Ferramentas de Gerenciamento de Segredos. Os dados foram coletados em 07/06/2020.

Características	Conjur	Vault	Barbican	Knox	Keywhiz
Testes unitários (1p)	Sim	Sim	Sim	Sim	Sim
Popularidade (1p)	Não	Sim	Não	Não	Não
Múltiplos Métodos de Autenticação (2p)	Não	Sim	Sim	Não	Sim
Início Automatizado (2p)	Sim	Não	Sim	Sim	Sim
Documentação Compreensiva (2p)	Sim	Sim	Sim	Não	Não
Suporte a MSH (2p)	Sim	Sim	Sim	Não	Não
Uso Comercial (2p)	Sim	Sim	Sim	Sim	Sim
Suporte Técnico (2p)	Sim	Sim	Sim	Não	Não
Armazenamento Seguro (3p)	Sim	Sim	Sim	Sim	Sim
Registros de Auditoria (3p)	Sim	Sim	Não	Sim	Não
Controle de Acesso (3p)	Sim	Sim	Sim	Sim	Sim
Sem Vulnerabilidades (3p)	Sim	Sim	Não	Não	Não
Impacto na Comunidade (3p)	Sim	Sim	Sim	Sim	Sim
Manutenção Ativa (3p)	Sim	Sim	Sim	Sim	Sim

Ao término da análise, a contagem total de pontos resultou em: Hashicorp Vault com 30 pontos, Cyberark Conjur com 29 pontos, OpenStack Barbican com 25 pontos, Pinterest Knox com 20 pontos e Square Keywhiz com 19 pontos.

Chegou-se à conclusão de que o Hashicorp Vault se destaca como o sistema mais indicado para pequenas organizações (KUZMINYKH; GHITA; SHIAELES, 2020). Portanto, as etapas subsequentes do trabalho foram fundamentadas na arquitetura do Hashicorp Vault.

4.3 Proposta de Arquitetura

4.3.1 Diagrama de Contexto

O Diagrama de Contexto (Figura 6) esboça a interação entre diferentes entidades no âmbito de um Software de Gerenciamento de Segredos. No diagrama, três entidades principais são identificadas: o "Usuário", que pode ser um Administrador, um Desenvolvedor ou outros Serviços; os "Sistemas Externos", representando entidades ou plataformas fora do ambiente do software de gerenciamento de segredos; e o próprio "Software de Gerenciamento de Segredos".

Cada entidade tem um papel específico e vital no ecossistema do software. Os usuários, por exemplo, têm a necessidade de acessar segredos de maneira segura para realizar suas tarefas diárias. O software de gerenciamento de segredos, por sua vez, é responsável por proteger, gerenciar e auditar o acesso a esses segredos, garantindo que apenas usuários autorizados tenham acesso. Por fim, os sistemas externos interagem com o software de gerenciamento de segredos para compartilhar informações de maneira segura e auditável.

O Diagrama de Contexto é crucial para entender como o Software de Gerenciamento de Segredos se encaixa dentro de um ecossistema maior de segurança da informação e gerenciamento de dados.

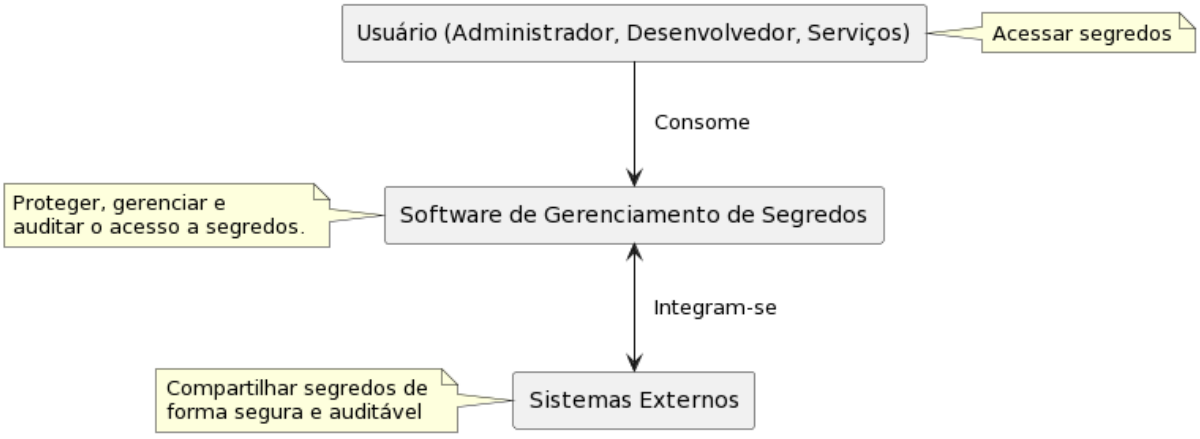

Figura 6 – Diagrama de Contexto

4.3.2 Diagrama de Contêineres

O Diagrama de Contêineres (Figura 7) apresenta a visão das partes que formam um Software de Gerenciamento de Segredos. No centro do sistema está o módulo "Orquestração", representado pela cor verde, representando o núcleo do sistema. Este módulo coordena todas as operações internas e interações com outros componentes. Cada um desses componentes está conectado ao núcleo de orquestração de maneira específica, indicando o fluxo de informações e comandos dentro do sistema. Além disso, o sistema se conecta a um "Banco de Dados", que armazena todas as informações gerenciadas pelo software.

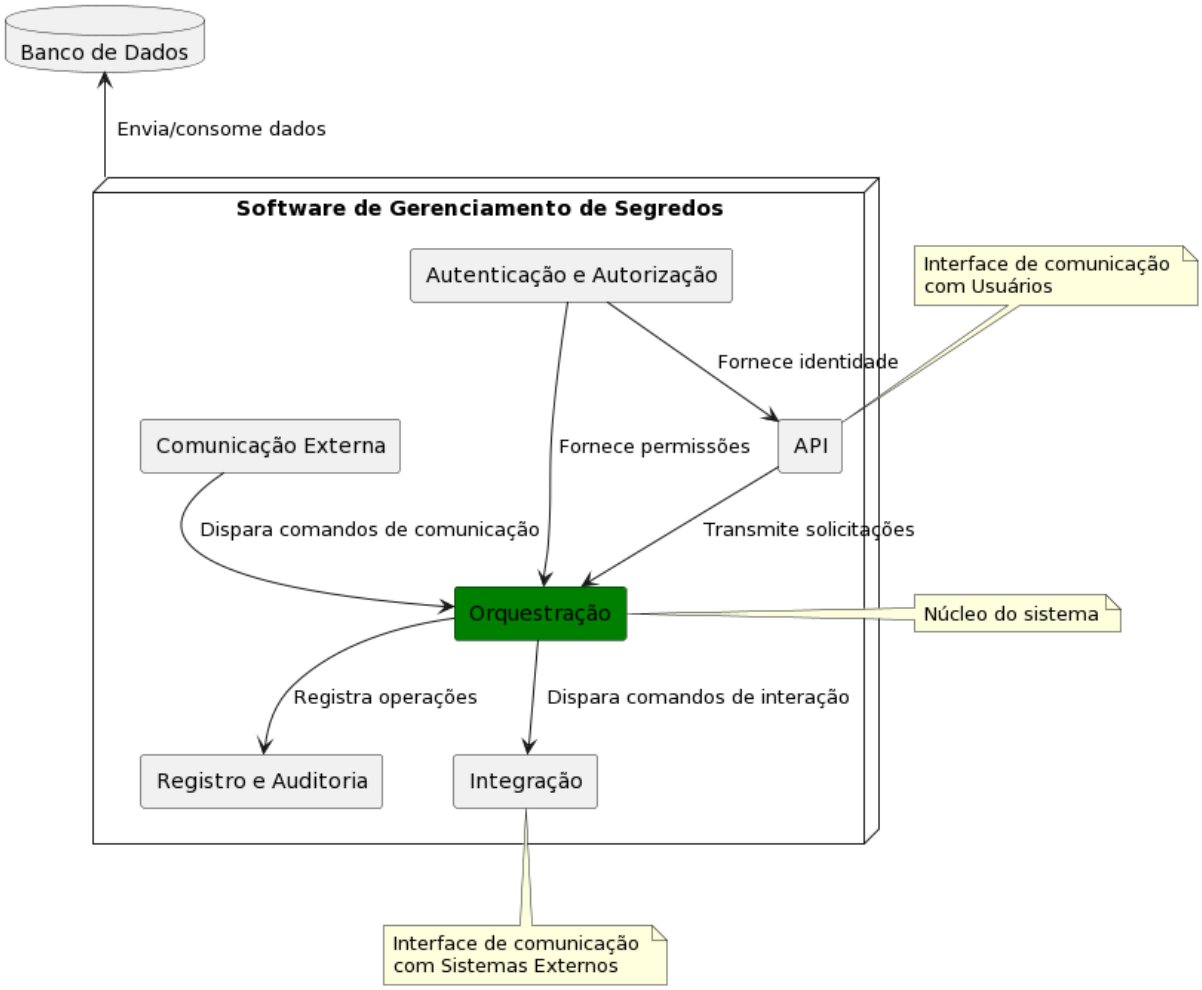

Figura 7 – Diagrama de Contêineres

4.3.2.1 Descrição dos Contêineres e Requisitos

4.3.2.1.1 Orquestração

O contêiner de Orquestração é o núcleo central do sistema. Ele coordena todas as operações e fluxos de trabalho, implementando as regras de negócio. Este contêiner é responsável por tomar decisões lógicas, gerenciando as interações entre os outros contêineres e garantindo que as operações sejam executadas de forma eficiente e segura.

Os requisitos básicos relativos ao contêiner **Orquestração** são:

Requisito	**Descrição**
R1	O módulo deve coordenar fluxos de trabalho complexos e operações entre contêineres.
R2	Implementar um mecanismo de decisão lógica para gerenciar interações e dependências.
R3	Suportar políticas de segurança para operações seguras.

Tabela 2 – Requisitos: Orquestração

4.3.2.1.2 API

Este contêiner gerencia a interface de programação de aplicativos (API) do sistema. Ele expõe funcionalidades para usuários externos e sistemas, facilitando a interação programática com o software de gerenciamento de segredos. A API atua como um ponto de acesso para solicitações externas, encaminhando-as para o Orquestrador para processamento.

Os requisitos básicos relativos ao contêiner **API** são:

Requisito	**Descrição**
R1	Oferecer uma interface clara e documentada para interações programáticas.
R2	Implementar autenticação e autorização nas solicitações de API.
R3	Projetada para alta disponibilidade e escalabilidade.

Tabela 3 – Requisitos: API

4.3.2.1.3 Autenticação e Autorização

Responsável pela autenticação de usuários e serviços, bem como pela gestão de autorizações. Este contêiner verifica as credenciais dos usuários e determina seus níveis de acesso com base em políticas de segurança. Ele garante que somente entidades autorizadas possam acessar ou modificar segredos.

Os requisitos básicos relativos ao contêiner **Autenticação e Autorização** são:

Requisito	**Descrição**
R1	Implementar autenticação robusta e multi-fator.
R2	Gerenciar autorizações com base em funções e políticas de acesso.
R3	Registrar tentativas de acesso para análise de segurança.

Tabela 4 – Requisitos: Autenticação e Autorização

4.3.2.1.4 Registro e Auditoria

O contêiner de Registro e Auditoria é crucial para a conformidade e segurança. Ele registra todas as operações realizadas no sistema, incluindo acesso a segredos, alterações e transações. Esta funcionalidade permite a auditoria posterior das ações e ajuda a identificar e responder a possíveis problemas de segurança.

Os requisitos básicos relativos ao contêiner **Registro e Auditoria** são:

Requisito	**Descrição**
R1	Registrar detalhadamente todas as operações com segredos.
R2	Fornecer ferramentas para análise e auditoria de registros.
R3	Garantir a integridade e imutabilidade dos registros de auditoria.

Tabela 5 – Requisitos: Registro e Auditoria

4.3.2.1.5 Integração

Focado na integração com softwares externos, especialmente aqueles que precisam compartilhar segredos para operar. Este contêiner facilita o compartilhamento seguro de segredos com sistemas de terceiros, gerenciando as interfaces e protocolos necessários para a integração eficaz e segura.

Os requisitos básicos relativos ao contêiner **Integração** são:

Requisito	**Descrição**
R1	Facilitar a integração segura com sistemas externos.
R2	Suportar diversos protocolos e interfaces de comunicação.
R3	Incluir mecanismos de verificação para segurança das integrações.

Tabela 6 – Requisitos: Integração

4.3.2.1.6 Banco de Dados

O contêiner do Banco de Dados é onde os dados são armazenados e gerenciados. Isso inclui segredos (como senhas e chaves de API), registros de auditoria, informações de configuração e qualquer outro dado necessário para as operações do sistema. Ele fornece um armazenamento seguro e eficiente, com recursos para garantir a integridade e a confidencialidade dos dados.

Os requisitos básicos relativos ao contêiner **Banco de Dados** são:

Requisito	**Descrição**
R1	Suportar criptografia de dados em repouso e em trânsito.
R2	Oferecer backup e recuperação de dados.
R3	Otimizar para consultas rápidas mantendo segurança e integridade.

Tabela 7 – Requisitos: Banco de Dados

4.3.2.1.7 Comunicação Externa

Este contêiner é responsável por gerenciar todas as notificações e comunicações externas do sistema para os usuários. Ele lida com o envio de e-mails de confirmação, alertas sobre a expiração de autorizações, notificações de segurança e outras mensagens relevantes. O objetivo deste contêiner é assegurar que os usuários sejam informados de maneira oportuna sobre eventos importantes relacionados ao sistema de gerenciamento de segredos, como mudanças de status, necessidade de ações ou atualizações de segurança. Ele funciona como um intermediário entre o sistema e os usuários finais, mantendo-os informados e engajados com o uso seguro e eficiente do sistema.

Os requisitos básicos relativos ao contêiner **Comunicação Externa** são:

Requisito	**Descrição**
R1	Gerenciar comunicações com usuários finais de forma eficiente.
R2	Assegurar a confidencialidade e integridade das comunicações.
R3	Permitir personalização das mensagens para diferentes grupos de usuários.

Tabela 8 – Requisitos: Comunicação Externa

4.3.2.2 Descrição das relações entre Contêineres

4.3.2.3 API para Orquestração

A interface de programação de aplicativos (API) atua como um intermediário crucial entre os usuários finais e o sistema de gerenciamento de segredos, especificamente o

componente de Orquestração. Quando um usuário envia uma solicitação ao sistema — seja para armazenar um novo segredo, recuperar um segredo existente, ou realizar qualquer outra operação relacionada à gestão de segredos — é a API que recebe inicialmente esta solicitação. Ela é responsável por validar e autenticar a requisição, assegurando que somente pedidos autorizados sejam processados. Após a validação, a API traduz e encaminha essas solicitações ao núcleo central do sistema, o componente de Orquestração.

O Orquestrador, por sua vez, interpreta a solicitação, determina a sequência de ações necessárias para cumpri-la e coordena a execução dessas ações entre os diversos componentes e serviços do sistema. Este processo pode envolver, por exemplo, a interação com o banco de dados para armazenar ou recuperar informações, a aplicação de políticas de segurança, ou a comunicação com outros sistemas para a integração de serviços. O resultado dessa orquestração é então retornado à API, que finaliza o ciclo de solicitação ao entregar a resposta adequada ao usuário. Este fluxo assegura que todas as interações com o sistema sejam gerenciadas de forma centralizada, eficiente e segura, alinhadas com as regras de negócio e políticas de segurança estabelecidas.

4.3.2.4 Autenticação e Autorização para API

O mecanismo de Autenticação e Autorização desempenha um papel fundamental na segurança e controle de acesso dentro do sistema de gerenciamento de segredos, atuando como um guardião que valida a identidade de usuários e sistemas antes de permitir o acesso às funcionalidades expostas pela API. Este processo é essencial para garantir que apenas entidades autenticadas com as devidas permissões possam realizar operações sensíveis, como armazenar, recuperar ou modificar segredos.

Quando uma solicitação é recebida pela API, o primeiro passo é direcioná-la ao componente de Autenticação e Autorização. Aqui, são realizadas duas verificações críticas: primeiro, a autenticação, que confirma a identidade do solicitante por meio de credenciais fornecidas, como nome de usuário e senha, chaves de acesso, ou métodos de autenticação baseados em certificados. Uma vez que a identidade é verificada, o sistema procede à fase de autorização, na qual avalia as permissões do usuário ou sistema solicitante para determinar se ele possui direitos suficientes para executar a ação desejada.

Este componente não apenas valida cada solicitação em termos de identidade e permissões, mas também aplica políticas de segurança detalhadas que governam o acesso aos recursos do sistema. Por exemplo, pode-se restringir o acesso a determinados segredos apenas a usuários com roles específicos ou garantir que certas ações possam ser executadas apenas por sistemas com níveis de confiança apropriados.

Após a conclusão bem-sucedida dessas etapas de autenticação e autorização, a solicitação é então liberada para processamento adicional pela API, que prosseguirá com a execução da operação solicitada. Em caso de falha em qualquer uma dessas verificações,

a solicitação é imediatamente rejeitada, e uma resposta de erro é enviada ao solicitante, indicando a falta de autenticação ou autorização adequada.

Esta abordagem estrita garante que o sistema mantenha a integridade e a confidencialidade dos segredos gerenciados, ao mesmo tempo em que oferece flexibilidade e controle sobre quem pode acessar e gerenciar esses segredos, reforçando a segurança geral do sistema.

4.3.2.5 Autenticação e Autorização para Orquestração

O elo entre os sistemas de Autenticação e Autorização e o componente central de Orquestração é vital para a segurança e a gestão eficaz das operações dentro do sistema de gerenciamento de segredos. Esta conexão assegura que, antes de qualquer operação ser executada pelo Orquestrador, o solicitante — seja um usuário ou um sistema — não apenas seja autenticado, mas também tenha suas permissões cuidadosamente verificadas em relação à tarefa que deseja realizar.

A Autenticação serve como o primeiro nível de segurança, estabelecendo a identidade do solicitante por meio de métodos robustos de verificação de identidade. Uma vez que a autenticação é confirmada, o processo de Autorização entra em cena. Esta etapa é crucial, pois envolve a análise detalhada das permissões do solicitante para assegurar que ele tenha o direito de executar a ação desejada no núcleo do sistema. Isso é feito comparando as solicitações de operação com as políticas de acesso definidas e os privilégios atribuídos ao usuário ou sistema em questão.

Esta verificação de permissões é especialmente importante quando se considera a diversidade e sensibilidade das operações gerenciadas pelo componente de Orquestração. Desde a criação, recuperação, alteração até a exclusão de segredos, cada ação requer um nível apropriado de autoridade para ser executada com segurança. Ao garantir que apenas solicitações autorizadas cheguem ao Orquestrador, o sistema mantém um alto padrão de segurança operacional e de dados, protegendo contra acessos não autorizados ou potencialmente maliciosos.

Além disso, esta abordagem reforça o princípio de menor privilégio, assegurando que usuários e sistemas tenham apenas o nível de acesso necessário para desempenhar suas funções, minimizando assim a superfície de ataque e reduzindo o risco de exposição de dados sensíveis.

Em resumo, a integração dos sistemas de Autenticação e Autorização com o Orquestrador é um mecanismo crítico que fortalece a segurança e a integridade do sistema de gerenciamento de segredos. Ela garante que todas as operações executadas no núcleo do sistema sejam legitimamente autorizadas, sustentando uma governança rigorosa sobre quem pode fazer o quê dentro do ambiente do sistema.

4.3.2.6 Comunicação Externa para Orquestração

A interface de Comunicação Externa serve como um canal vital para o sistema de gerenciamento de segredos interagir com o mundo externo. Este componente é encarregado de gerenciar e disparar comandos de comunicação para o componente central de Orquestração, permitindo assim que o sistema responda dinamicamente a solicitações e eventos originados fora de seus limites internos.

Este processo inicia quando o sistema precisa comunicar-se com entidades externas, como serviços de terceiros, aplicações parceiras, ou sistemas de monitoramento e alerta. Por exemplo, pode ser necessário enviar notificações de segurança, alertas de violação, ou simplesmente transmitir dados de segredos para aplicações autorizadas que dependem dessas informações para operar. Nesse contexto, o componente de Comunicação Externa captura a necessidade de comunicação, formula o comando adequado e o encaminha para o Orquestrador.

O Orquestrador, atuando como o cérebro do sistema, recebe esses comandos de comunicação e determina a melhor maneira de proceder. Dependendo da natureza do comando, ele pode iniciar processos internos para coletar dados necessários, aplicar transformações de segurança, como criptografia ou mascaramento de dados, e finalmente, orquestrar o envio da comunicação ao destino externo adequado. Este processo assegura que todas as comunicações externas sejam conduzidas de forma controlada, segura, e em conformidade com as políticas do sistema.

Além disso, a relação entre Comunicação Externa e Orquestração permite que o sistema seja extensível e adaptável a uma variedade de cenários de comunicação. Ao centralizar a lógica de comunicação no Orquestrador, o sistema pode facilmente integrar-se com novos serviços externos ou adaptar-se a mudanças nos requisitos de comunicação sem necessidade de revisões significativas na arquitetura.

Em resumo, o vínculo entre Comunicação Externa e Orquestração é crucial para a funcionalidade e flexibilidade do sistema de gerenciamento de segredos. Ele não apenas permite que o sistema interaja efetivamente com o ambiente externo, mas também garante que todas as comunicações sejam processadas de forma segura e alinhadas com os objetivos estratégicos e operacionais do sistema.

4.3.2.7 Orquestração para Registro e Auditoria

A integração entre o componente de Orquestração e o sistema de Registro e Auditoria é essencial para garantir a transparência, a segurança e a conformidade do sistema de gerenciamento de segredos. Esta conexão assegura que cada ação executada pelo Orquestrador, o coração do sistema, seja meticulosamente documentada, criando um registro imutável de todas as operações.

Quando o Orquestrador processa qualquer comando — seja ele uma solicitação para criar, acessar, modificar ou excluir um segredo —, não apenas executa a operação requerida, mas também comunica detalhes dessa atividade ao componente de Registro e Auditoria. Esta comunicação inclui informações como a identidade do solicitante, a natureza da operação, a hora exata da sua execução, e o resultado da ação. Estes registros são vitais para uma série de propósitos críticos, incluindo auditorias de segurança, análises forenses, e o cumprimento de regulamentações de proteção de dados.

O sistema de Registro e Auditoria é projetado para capturar, armazenar e proteger esses dados de maneira segura e organizada, facilitando consultas e relatórios. Isto permite que administradores de sistema e auditores revisem as atividades passadas, identifiquem padrões anormais ou suspeitos, e verifiquem a aderência às políticas de segurança estabelecidas. Além disso, em caso de incidentes de segurança, os registros podem fornecer insights valiosos sobre a sequência de eventos, ajudando na rápida resolução de problemas e na implementação de medidas para prevenir futuras ocorrências.

Essa documentação abrangente e sistemática das ações do sistema não apenas reforça a postura de segurança, mas também promove uma cultura de responsabilidade e transparência. Ao garantir que cada operação seja rastreável, o sistema permite uma governança de dados robusta e ajuda a construir confiança entre os usuários e as partes interessadas.

Em resumo, a relação entre Orquestração e Registro e Auditoria é um pilar fundamental na arquitetura do sistema de gerenciamento de segredos, fornecendo os meios para monitorar, revisar e melhorar continuamente a segurança e a eficácia do sistema, enquanto se mantém em conformidade com as exigências legais e normativas.

4.3.2.8 Orquestração para Integração

A conexão entre o mecanismo de Orquestração e o componente de Integração é fundamental para estender as capacidades do sistema de gerenciamento de segredos além de seus limites internos, permitindo uma comunicação eficaz e segura com sistemas externos. Esta interação é crucial para automatizar fluxos de trabalho, sincronizar dados de segredos e aproveitar serviços complementares que enriquecem a funcionalidade e a segurança do sistema.

O Orquestrador, atuando como o cérebro operacional do sistema, identifica quando e como interagir com entidades externas para cumprir requisitos específicos de negócios ou operacionais. Por exemplo, pode ser necessário compartilhar segredos com uma aplicação de nuvem, sincronizar credenciais com um sistema de autenticação federada ou enviar dados de configuração para um serviço de infraestrutura. Para realizar essas ações, o Orquestrador emite comandos direcionados ao componente de Integração.

O componente de Integração, por sua vez, é especializado em estabelecer e gerenciar conexões seguras com sistemas externos. Ele traduz os comandos recebidos do Orquestrador em operações compatíveis com os protocolos e APIs dos sistemas de destino. Além disso, o componente de Integração trata da autenticação e autorização nessas interfaces externas, garantindo que as interações sejam realizadas de forma segura e conforme as políticas de acesso definidas.

Esta camada de Integração também é responsável por adaptar os dados recebidos de sistemas externos para o formato esperado pelo sistema de gerenciamento de segredos, assegurando que a informação integrada seja corretamente processada e armazenada. Essa capacidade de bidirecionalidade na comunicação não apenas amplia o alcance operacional do sistema, mas também permite uma maior flexibilidade e adaptabilidade às mudanças no ambiente tecnológico e nas necessidades de negócios.

Em resumo, a interação entre Orquestração e Integração é uma peça chave na arquitetura do sistema de gerenciamento de segredos, habilitando a execução de uma gama diversificada de funções que dependem da cooperação com sistemas e serviços externos. Por meio desta integração, o sistema consegue não apenas melhorar sua própria segurança e eficiência, mas também contribuir para a segurança e operacionalidade do ecossistema de TI mais amplo.

4.3.2.9 Banco de Dados e Software de Gerenciamento de Segredos

A interação entre o Banco de Dados e o Software de Gerenciamento de Segredos é fundamental para a operação eficaz do sistema, servindo como o repositório central onde todas as informações cruciais são armazenadas e das quais são recuperadas. Este relacionamento simbiótico garante que os segredos, as configurações de sistema, os registros de auditoria e outras informações vitais sejam gerenciados de forma segura, organizada e acessível.

O Banco de Dados atua como uma fonte confiável de verdade para o sistema, mantendo os dados protegidos por meio de camadas robustas de segurança, incluindo criptografia em repouso e em trânsito, controle de acesso baseado em função e mecanismos de autenticação. Ao receber solicitações do Software de Gerenciamento de Segredos, ele processa essas requisições — seja para armazenar novos dados, atualizar informações existentes ou recuperar segredos solicitados — e executa as operações necessárias com eficiência e precisão.

Além de armazenar segredos, o Banco de Dados desempenha um papel crucial na manutenção dos registros de auditoria, que documentam todas as interações com o sistema. Isso não apenas facilita a conformidade com regulamentações e normas de segurança, mas também oferece insights valiosos para análises de segurança, investigações forenses e melhorias contínuas das práticas de gestão de segredos.

A arquitetura do sistema é projetada para garantir que as interações com o Banco de Dados sejam realizadas de maneira segura e eficiente, com o Software de Gerenciamento de Segredos implementando mecanismos de cache, filas e políticas de tentativas para otimizar o desempenho e garantir a integridade dos dados mesmo em condições de alta demanda ou falhas parciais.

5 Conclusão e trabalhos futuros

Este trabalho explorou a arquitetura de um Software de Gerenciamento de Segredos, com foco na visão de contexto e contêineres dentro do modelo C4. Demonstrou-se como uma abordagem sistemática e robusta para o gerenciamento de segredos é crucial em um cenário de segurança cibernética em constante evolução. A pesquisa destacou a importância vital de adotar uma estratégia sistemática e robusta para o gerenciamento de segredos, especialmente considerando o dinamismo e a complexidade crescente do ambiente de segurança cibernética. Ao analisar meticulosamente os Sistemas de Gerenciamento de Segredos frequentemente empregados, assim como revisar literatura pertinente sobre práticas eficazes de gerenciamento de segredos e propor uma nova arquitetura, este trabalho forneceu uma base teórica e prática firme, estabelecendo um ponto de partida para investigações subsequentes.

Futuros trabalhos têm a oportunidade de ampliar a abrangência e a profundidade deste estudo, explorando as demais visões do modelo C4, especificamente as visões de Componentes e Código. Tal expansão não só enriquecerá a compreensão da arquitetura de SGSs, mas também oferecerá uma perspectiva mais holística e detalhada. Ao integrar essas visões adicionais, será possível enfrentar desafios ainda não resolvidos, aprimorar o design arquitetônico e elevar a eficácia e a segurança dos Sistemas de Gerenciamento de Segredos. Essa abordagem integrativa e abrangente promete contribuir significativamente para o desenvolvimento de práticas mais sofisticadas e eficientes no gerenciamento de segredos em um contexto tecnológico em rápida evolução.

Referências

BASAK, S. K. et al. *What are the Practices for Secret Management in Software Artifacts?* 2022. Disponível em: <https://arxiv.org/abs/2208.11280>. Cited 2 times on pages 21 e 22.

BLOMQVIST, M. Secrets management in a multi-cloud kubernetes environment. 2021. Disponível em: <https://www.utupub.fi/bitstream/handle/10024/151776/Secrets_Management_in_a_Multi_Cloud_Kubernetes_Environment_pdf-a.pdf>. Cited 2 times on pages 19 e 21.

DOTSON, C. *Practical Cloud Security: A Guide for Secure Design and Deployment*. O'Reilly Media, Incorporated, 2019. ISBN 9781492037514. Disponível em: <https://www.oreilly.com/library/view/practical-cloud-security/9781492037507/>. Cited on page 29.

GATEV, R. *Introducing Distributed Application Runtime (Dapr): Simplifying Microservices Applications Development Through Proven and Reusable Patterns and Practices*. Berkeley, CA: Apress, 2021. 215–231 p. ISBN 978-1-4842-6998-5. Disponível em: <https://doi.org/10.1007/978-1-4842-6998-5_11>. Cited on page 19.

HashiCorp Vault. *Introduction to HashiCorp Vault*. 2018. Disponível em: <https://www.hashicorp.com/resources/introduction-vault-whiteboard-armon-dadgar>. Cited 3 times on pages 21, 22 e 23.

HE, X.; YANG, X. Authentication and authorization of end user in microservice architecture. *Journal of Physics: Conference Series*, IOP Publishing, v. 910, n. 1, p. 012060, oct 2017. Disponível em: <https://dx.doi.org/10.1088/1742-6596/910/1/012060>. Cited on page 19.

KUZMINYKH, I.; GHITA, B.; SHIAELES, S. Comparative analysis of cryptographic key management systems. In: _____. *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*. Springer International Publishing, 2020. p. 80–94. ISBN 9783030657291. Disponível em: <http://dx.doi.org/10.1007/978-3-030-65729-1_8>. Cited 2 times on pages 30 e 31.

POLAJŽER, A. *Pretvorba diagramov v PlantUML*. Tese (Doutorado) — Univerza v Ljubljani, 2022. Disponível em: <https://repozitorij.uni-lj.si/IzpisGradiva.php?lang=slv&id=143508>. Cited on page 28.

RAHMAN, A.; BARSHA, F. L.; MORRISON, P. Shhh!: 12 practices for secret management in infrastructure as code. In: *2021 IEEE Secure Development Conference (SecDev)*. [s.n.], 2021. p. 56–62. Disponível em: <https://par.nsf.gov/biblio/10343037-shhh-practices-secret-management-infrastructure-code>. Cited 3 times on pages 21, 22 e 23.

SHAMIR, A. How to share a secret. 1979. Disponível em: <https://web.mit.edu/6.857/OldStuff/Fall03/ref/Shamir-HowToShareASecret.pdf>. Cited on page 24.

THEODORSON, G.; THEODORSON, A. *A Modern Dictionary of Sociology*. Methuen, 1970. (A Crowell reference book). ISBN 9780416169409. Disponível em: <https://books.google.com.br/books?id=A8IOAAAAQAAJ>. Cited on page 25.

VáZQUEZ-INGELMO, A.; GARCíA-HOLGADO, A.; GARCíA-PEñALVO, F. J. C4 model in a software engineering subject to ease the comprehension of uml and the software development process. p. 919–924, 2020. Disponível em: <http://repositorio.grial.eu/handle/grial/2041>. Cited on page 26.

Apêndices

APÊNDICE A – Códigos PlantUML

A.1 Autenticação e Autorização

```
@startuml
rectangle "Etapa 1: Autenticação "as r1 {
 rectangle User1 as "Usuário"
 rectangle System1 as "Sistema"

 System1  -left-> User1 : Quem é você?
}

rectangle "Etapa 2: Autorização" as r2 {
 rectangle User2 as "Usuário Autenticado"
 rectangle System2 as "Sistema"


 System2  -left-> User2 : Quais permissões você tem?

}

r1 -[hidden]- r2
@enduml
```

A.2 Diagrama de Contexto

```
@startuml
' Definições de Elementos
rectangle User as "Usuário (Administrador, Desenvolvedor, Serviços)"
rectangle ExternalSystem as "Sistemas Externos"
rectangle System as "Software de Gerenciamento de Segredos"

note right of User: Acessar segredos
note left of System : Proteger, gerenciar e \n
auditar o acesso a segredos.
note left of ExternalSystem : Compartilhar segredos de \n
```

```
forma segura e auditável

' Relacionamentos
User --> System: "   Consome"
System <--> ExternalSystem : "   Integram-se"
@enduml
```

A.3 Diagrama de Contêineres

```
@startuml
!define RECTANGLE rectangle

' Definições de Containers

node "Software de Gerenciamento de Segredos" as sgs {
  RECTANGLE "Orquestração" as core #green
  RECTANGLE "API" as api
  RECTANGLE "Autenticação e Autorização" as auth
  RECTANGLE "Comunicação Externa" as coms
  RECTANGLE "Registro e Auditoria" as logging
  RECTANGLE "Integração" as integration

}

database "Banco de Dados" as db

' Relacionamentos
api --> core : "  Transmite solicitações"
auth --> api : "  Fornece identidade"
auth --> core : " Fornece permissões"
coms --> core : "  Dispara comandos de comunicação"
core --> logging : "Registra operações       "
core --> integration : "  Dispara comandos de interação"

note top of api: Interface de comunicação \ncom Usuários
note bottom of integration: Interface de comunicação \ncom Sistemas Externos
note right of core: Núcleo do sistema

db <-- sgs : "  Envia/consome dados"
```

```
@enduml
```

A.4 Modelo C4

```
@startuml
' Define a camada de Sistema de Software

rectangle "Contexto" as ctx {
    rectangle "Sistema de Software" as sys
}

rectangle "Contêineres" as cont {
    rectangle "Container" as cont_1
    rectangle "Container" as cont_2
}

rectangle "Componentes" as comp {
    rectangle "Componente" as comp_1
    rectangle "Componente" as comp_2
}

rectangle "Código" as cod {
    rectangle "Código" as cod_1
    rectangle "Código" as cod_2
}

ctx -[hidden]- cont
cont -[hidden]- comp
comp -[hidden]- cod

sys --> cont
cont_1 --> comp
comp_1 --> cod


note bottom of [ctx]
  Visão geral das interações do
  sistema com o mundo real
  e entidades externas.
```

```
end note

note bottom of [cont]
  Divide o sistema em
  contêineres, destacando suas
  funções e interconexões.
end note

note bottom of [comp]
  Detalha os componentes de
  cada contêiner, suas
  responsabilidades e interações.
end note

note bottom of [cod]
  Explica a implementação
  detalhada dos componentes
  usando diagramas e código.
end note
@enduml
```

A.5 Replicação de Instancias

```
@startuml
rectangle "Instância 1" as Instance1 #red
rectangle "Instância 2" as Instance2
rectangle "Instância N" as InstanceN

Instance1 --> Instance2 : Replicação de dados
Instance1 --> InstanceN : Replicação de dados

note right of Instance1: Instancia Lider

note bottom of Instance2: Réplica do sistema
note bottom of InstanceN: Réplica do sistema

Instance2 -> Instance1 : Sincronização e Atualizações
InstanceN -> Instance1 : Sincronização e Atualizações
@enduml
```

A.6 Compartilhamento de Segredos de Shamir

```
@startuml
rectangle "Chave Principal (Segredo)" as Secret
rectangle "Subchave 1" as Key1
rectangle "Subchave 2" as Key2
rectangle "Subchave 3" as Key3
rectangle "Subchave N" as KeyN

Key1 -down-> Secret : Necessária
Key2 -down-> Secret : Necessária
Key3 -down-> Secret : Necessária
KeyN -down-> Secret : Necessária

note right of Secret: A chave principal só\npode ser reconstituída\n
quando todas as subchaves\nestão presentes.
@enduml
```