Sistemas Distribuídos

Anti-Ransomware

João Rolo

Tomás Ferreira

Martim Costa

GitHub: <https://github.com/tolojo/Anti-Ransomware-SD>

1. **Descrição do Problema**

Uma modificação não autorizada de um sistema de ficheiros é algo que se deseja evitar, seja devido a ataques de ransomware ou outra causa.

O que pretendemos com este trabalho é criar um sistema que seja tolerante a falhas, eficaz e escalável.

O que nos levou a escolher este tema é o facto do projeto da unidade curricular Project Factory trabalhar com dados que são sensíveis e que não podem ser alterados no sistema por terceiros, desta forma iremos implementar o nosso próprio sistema de anti-ransomware, de forma a ter um maior entendimento de como este tipo de software funciona.

1. **Casos de Uso**

* Autenticação no sistema;
* Alteração de ficheiros por um utilizador autorizado;
* Alteração de ficheiros por um utilizador não autorizado;
* Deteção e recuperação de ficheiros corrompidos.

1. **Solução a Implementar**

Todos os ficheiros vão estar guardados num servidor que é replicado (dois servidores ativos em simultâneo), cuja alteração só será permitida a utilizadores autenticados. Estes ficheiros vão ser encriptados com uma Hash Function (SHA-256), isto consiste em correr a função SHA-256 pelos ficheiros contidos na base de dados, o que vai resultar num mapeamento, por ficheiro para um ficheiro resultante de 256 bits, e a chave resultante da função vai ser guardada num servidor que não é acedido pelos utilizadores, juntamente com uma cópia do ficheiro que está contido no banco de dados dos clientes.

A utilização desta função de hashing vai possibilitar que seja verificado se um ficheiro foi alterado ou não, uma vez que o resultado do hashing não muda se o input for o mesmo, logo se o ficheiro for alterado poderá ser detetada essa alteração e depois poderá ser reposto.

A função SHA-256 é uma variante do SHA-2 que consiste em transformar um tamanho arbitrário de dados e mapeá-lo para dados de um tamanho físico o que irá servir para verificar a integridade dos ficheiros.

Para o nosso sistema, será utilizado um mecanismo manual, através da utilização de um cliente web, que necessitará de autenticação para utilizar (2FA), para fazer a verificação se algum ficheiro foi alterado de forma indevida, utilizando o mecanismo referido acima, fazendo a comparação da hash gerada do ficheiro que se encontra no servidor dos clientes com a hash guardada na base de dados de back-up. Se for verificada alguma alteração não autorizada de ficheiros, será então reposto o ficheiro com a versão devida.

1. **Enquadramento nas Áreas da Unidade Curricular**

Em termos da disciplina de Sistemas Distribuídos é disponibilizada uma divisão da carga de trabalho por parte dos servidores, tal como das bases de dados, aumentando assim a eficácia e tornando-a mais tolerante a falhas.

Um Sistema Distribuído vai permitir que, na ocorrência de algum erro ou atualização nos servidores, não seja posta em risco a operacionalidade do sistema.

Do ponto de vista de sincronismo do sistema é necessário saber que dados são necessários manter e quais são necessários alterar quando estes estão em concorrência.

Relativamente à segurança dos ficheiros é indispensável existir uma forma detetar ficheiros corrompidos ou indevidamente alterados, como também uma forma de recuperar os mesmos de forma a manter a integridade do sistema. Assim a autenticação 2 fatores será crucial para garantir que apenas utilizadores autenticados possam alterar os dados dos ficheiros.

1. **Requisitos Técnicos**

O sistema irá ter 1 servidor de comunicação com o cliente, 2 servidores que se encarregam da encriptação dos ficheiros e armazenar a cópia de segurança dos ficheiros e 1 servidor de load balancer (Nginx) que vai redirecionar os pacotes para um dos servidores ligados.

1. **Planeamento**

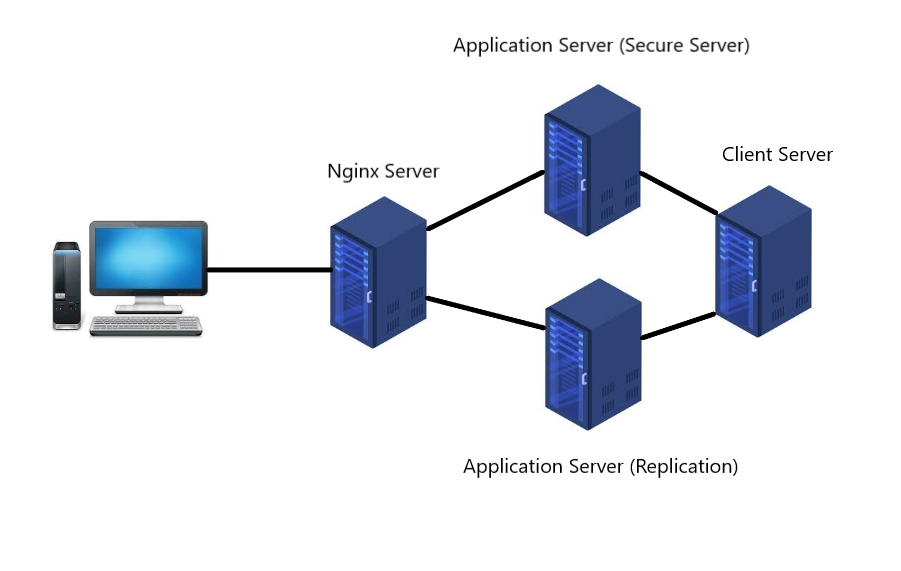
Uma imagem com texto

Descrição gerada automaticamente

1. **Arquitetura da Solução**

A arquitetura do sistema vai consistir num cliente web para proceder a ligação com os servidores de segurança, que vão ser responsáveis pela verificação da integridade dos ficheiros e reposição dos mesmos e que vão guardar as Hash Keys dos ficheiros e as cópias de segurança dos ficheiros contidos no servidor do cliente e o servidor de Nginx vai ser responsável por fazer o load balancing do sistema, ou seja, vai verificar qual dos servidores se encontra online. No caso de estarem ambos online vai escolher de forma aleatória a qual se vai ligar e se só estiver um servidor online ele vai conectar-se a esse servidor.

Os servidores necessitam de ter no mínimo 1GB de memória RAM e 10GB de memória em disco.



1. **Métodos**
2. **SHA Converter**
3. **SHA Comparision**
4. **Download File**
5. **Tecnologias**

**** ** f**

1. **Bibliografia**

<https://www.n-able.com/blog/sha-256-encryption>

<https://www.notion.so/product>