**TEMA:** Segurança da informação

**SUBTEMA:** Vazamentos de arquivos e códigos em plataformas de versionamento de código

**PROBLEMA:** Como reduzir os danos causados por vazamentos em plataformas de versionamento de código?

**OBJETIVO GERAL:** Elaborar um modelo de versionamento de código que reduza os danos causados por vazamentos

**JUSTIFICATIVA DO TRABALHO**: Tentar garantir a segurança de arquivos e códigos privados.

**OBJETIVOS ESPECÍFICOS**

* Pesquisar como e porquê foram feitos os primeiros protocolos de segurança/criptografia
* Entender o quê são versionadores de códigos
* Comparar o funcionamento de dois dos principais versionadores de códigos
* Expor casos relevantes de vazamentos de códigos e segredos
* Desenvolver uma solução para a proteção de arquivos

======================================================================

# Requisitos

* O sistema deve permitir o upload e download de arquivos
* O sistema deve gerar uma chave de criptografia simétrica para cada repositório criado.
* O sistema deve armazenar a chave de criptografia do repositório no computador local do dono.
* O processo de criação e armazenamento da chave de criptografia do repositório deve ser feito localmente.
* O sistema deve criptografar a chave simétrica do repositório com a chave pública do usuário que desejar fazer download dele.

# Descrições de Usabilidade

## Simulando um caso de uso pra exemplificar os processos do sistema:

### Pré-condições

Ao criar um usuário no site, ele terá para si um nickname e uma a Chave Pública (visível para todos, sejam eles usuários ou o próprio servidor), e uma Chave Privada (armazenada no seu PC local, que só ele conhece). Isso se chama ***Criptografia Assimétrica: qualquer coisa fechada com uma chave só pode ser aberta pela outra.***

### Criar um repositório

O usuário cria um repositório, e junto com ele cria uma chave simétrica de criptografia (vamos chamar de chave R). Essa chave, por ser simétrica, é responsável tanto por “fechar” quanto por “abrir” os arquivos. Todos os arquivos deste repositório vão ser “fechados” com a chave R quando forem upados pro servidor. A chave R só vai estar armazenada no PC local do usuário que criou o repositório ou dos colaboradores, então mesmo que alguem hackeie a conta do dono do repositório e veja os arquivos, não vai conseguir acessar os arquivos.

Ao criar um repositório, o software na máquina do usuário estabelece uma **conexão criptografada de ponta a ponta** com o servidor. Ele pega a chave pública do servidor, criptografa a chave R do repositório com ela, e envia ela criptografada para o servidor. O servidor então descriptografa a chave R do repositório com a sua própria chave privada (pois ela foi criptografada com a sua própria chave pública; criptografia assimétrica) e armazena essa chave. Essa chave R será o fator de autenticação para permitir ou não que um usuário acesse, faça download e upload dos arquivos daquele repositório.

### Convidar colaboradores

O dono decide compartilhar o repositório com um colaborador. Ele criptografa a chave R com a chave pública do novo colaborador. O servidor envia a chave R, criptografada, até o colaborador, que no seu PC local, descriptografa a chave R com a sua própria chave privada.

### Acessar e interagir com o repositório

Para acessar o repositório, seja o dono ou qualquer colaborador: ao entrar na página do repositório, o aplicativo solicitaria a chave R do repositório. O usuário faz upload da chave e envia, e se a chave for a correta, ele agora possui permissão para mexer com os arquivos do repositório.

O colaborador baixa os arquivos do servidor, criptografados com a chave R, chave essa que ele possui agora, mas que obviamente foi enviada conscientemente pelo dono do repositório. Assim, o colaborador consegue “abrir” os arquivos com a chave R, que também vai ficar armazenada no seu PC local.

⁠Quando for upar esses arquivos modificados para o servidor, o colaborador fecha eles com a Chave R, e envia pro servidor.

Processo se repete para a adição de outros colaboradores.

# Descrições de Tela

* Tela de login
* Tela com os seus repositórios criados
* Visualização em árvore dos arquivos do repositório
* Histórico de commits com nome dos arquivos modificados, data, hora e usuário que commitou
* Tela de adição de colaborador
* Notificação de “você foi adicionado como colaborador”, e explicando que o usuário precisa instalar uma versão local do software pra conseguir extrair a chave do repositório

# Resultados: primeiro protótipo

Funções desenvolvidas:

* Cadastro de usuário
* Login
* Tela inicial com visualização / criação de repositórios
* Criação de repositório
* Visualização de repositórios do usuário
* Envio de commit (selecionando uma pasta, que pode ter outras pastas dentro dela).
* Visualização dos arquivos de um repositório
* Filtro da visualização dos arquivos por commit
* Download dos arquivos de um commit de um repositório (baixados na pasta Downloads dentro do diretório do próprio projeto)
* Exclusão de repositório com exclusão em cascata de seus commits, diretórios e arquivos