**GIT DEBAIXO DOS PANOS**

* Tópicos fundamentais para entender o funcionamento do Git

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Conceitos que permeiam o Git.

Texto

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Isso é importante, pois esse identificador é único. Identificando um arquivo de forma segura e rápida.

Interface gráfica do usuário

Descrição gerada automaticamente

Utilizando novamente o comando “echo”, porém com os argumentos complementares para jogar o que eu printaria por um algoritmo de encriptação (do “openssl”, “sha1”).

Texto

Descrição gerada automaticamente

Por exemplo, criei o arquivo “BEJITA.txt”, gerei uma hash com a primeira versão do arquivo, e depois alterei apenas uma letra, mudando completamente a hash.

* Objetos internos do Git

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

3 objetos internos que são responsáveis pelo versionamento do código.

Texto

Descrição gerada automaticamente

Fazendo o esquema de passar uma string para uma hash, podemos utilizar a função “hash-object” do Git, em conjunto com a “flag” “—stdin” (a função espera receber um arquivo, e estamos enviando um texto, isso é o que queremos que a função compreenda). Logo, passando nossa string por essa função, temos a devolução de um “SHA1”.

Porém observe que passando diretamente pelo openssl, sem utilizar o Git isto é, é gerado outro tipo de chave.

Por quê?

Interface gráfica do usuário, Aplicativo, PowerPoint

Descrição gerada automaticamente

Esses objetos específicos do Git, são manipulados através de objetos próprio. Os arquivos ficam armazenados no objeto Blob. Esse objeto contém metadados, então objeto “Blob” terá: o tipo do objeto (“Blob”), o tamanho desse arquivo, o \0, e o conteúdo propriamente dito.

Texto

Descrição gerada automaticamente

Refazendo a situação anterior, mas agora repassando os metadados para o segundo método, é possível observar que a mesma string gerada pelo Git, foi gerada também fora dele. Em suma, o Git também guarda metadados e por isso, gera uma “SHA1” diferente.

Interface gráfica do usuário, Aplicativo, PowerPoint

Descrição gerada automaticamente

As “trees” armazenam e apontam para blobs e outras trees (Afinal, diretórios podem conter outros diretórios, de forma recursiva). A tree contém metadados também, como pode ser vista na imagem. Interessantemente, o blob só armazena o SHA1, enquanto a tree armazena o nome do arquivo. Além disso, monta a estrutura de onde estão localizados os arquivos. As trees possuem seu próprio SHA1 que se relaciona aos seus metadados, então se um arquivo (logo, seu SHA1), for alterado, o SHA1 da tree também é alterado (toda a leitura, estrutura e encriptação do objeto, mesmo).

Diagrama

Descrição gerada automaticamente

Observe como uma tree funciona de maneira básica. A tree é um objeto que encapsula esse comportamento de diretórios, apontando para outras trees e blobs, e logo, para arquivos.

Interface gráfica do usuário

Descrição gerada automaticamente

O mais importante, ele vai juntar tudo e dar sentido a alteração. Aponta para: trees, aponta para um commit realizado antes dele (parente), aponta para um autor, aponta para uma mensagem (esses dois últimos é o que dão a ideia de um sentido, a mensagem serve para explicar o que é esse commit). Também leva um timestamp (a hora em que foi criada).

Um commit também possui um SHA1, significa que a alteração de um arquivo, também afeta corrente acima. Isso mostra, de forma segura, que um commit é diferenciado. Otimo para montar linha do tempo, e evitar que atores maliciosos ajam sobre o código, sem que a ação fique obvia.

Diagrama

Descrição gerada automaticamente

* Chaves SSH e Tokens

1. Chave SSH

Grosso modo é uma forma de estabelecer uma conexão segura e encriptada entre duas máquinas. Como por exemplo: máquina local -> Servidor do Github, utilizando duas chaves: uma pública e uma privada. Nossa chave pública poderá ser enviada diretamente para o servidor juntamente com o código. De certa forma, é como se a nossa máquina fosse considerada confiável e que possamos enviar código sem necessitar nem de autenticação manual.

Para isso, é possível fazer um processo: (verificar o vídeo Chave SSH e Token)

**\*\* Nota: comando “cat id\_25519.pub” (por exemplo) é um comando especial que quando usado no BASH permite que vejamos o conteúdo de um arquivo. Quando usado no arquivo da chave pública, é possível verificar o conteúdo do arquivo.**

**\*\* Nota 2: “eval $(ssh-agent -s)” (faz a inicialização do agente que preciso para postar o próximo comando) e “ssh-add id\_ed25519” (repassando chave privada para um programa que trabalhará com reconhecimento da chave pública que mandei para o github).**

* **Verificar o documento de texto “SSH e Tokens” <**