CST em Análise e Desenvolvimento de Sistemas — Prof. Emerson Ribeiro de Mello
SEG786203 – Segurança da Informação — mello@ifsc.edu.br

Laboratório 1.1: GNU Privacy Guard (GPG)

22/10/2024

O GNU Privacy Guard (GPG)[1] é uma ferramenta de criptografia que permite a criação de chaves públicas e privadas para a troca segura de mensagens. Neste laboratório, você irá aprender a criar um par de chaves, exportar a chave pública, importar a chave pública de um colega, criptografar e descriptografar mensagens.

Gerar um par de chaves

Para gerar um par de chaves, execute o comando abaixo:

```
# Gera um par de chaves
gpg --full-generate-key

# Escolha a opção 9 (ECC) sign and encrypt e a curva 1 (Curve 25519)
```

Listar chaves

As chaves são armazenadas em um banco de dados dentro do diretório `~/.gnupg`. Para listar as chaves, execute o comando abaixo:

```
# Lista todas as chaves armazenadas no banco de dados do GPG
gpg --list-keys

# Um exemplo de saída para o comando acima:
/home/juca/.gnupg/pubring.kbx
------------------------
pub   ed25519 2024-10-18 [SC]
      0C8A1F74BAB9AE2C00E616D92D5C437ECD19BE00
uid           [ultimate] Juca Simas <juca@example.org>
sub   cv25519 2024-10-18 [E]

# Para listar somente as chaves privadas
gpg --list-secret-keys
```

Troca de chaves

Para que alguém possa enviar uma mensagem cifrada para você, ou que possa verificar a autenticidade de uma mensagem assinada por você, é necessário que você envie a sua chave pública. Para exportar a chave pública, execute o comando abaixo:

```
# Exporta a chave pública com o ID juca@example.org
gpg --armor --output juca-public.key --export juca@example.org
```

Agora, envie o arquivo `juca-public.key` para um colega. Para importar a chave pública de um colega (Bob), execute o comando abaixo:

```
# Importa a chave pública do colega Bob
gpg --import bob-public.key
```

[1] https://www.gnupg.org/

Cifrar e decifrar arquivos

Para cifrar um arquivo é necessário indicar o destinatário da mensagem (parâmetro `-r`), podendo ser o e-mail ou o ID da chave pública que você importou do seu colega. O arquivo cifrado pode ser representado em formato ASCII (parâmetro `-a`), ou em formato binário. Para cifrar um arquivo, execute o comando abaixo:

```
# Para cifrar o arquivo chamado slides.pdf, irá gerar um arquivo chamado slides.pdf.gpg
gpg -r bob@example.org -e slides.pdf
```

Bob, ao receber o arquivo cifrado, poderá decifrá-lo com o comando abaixo:

```
# Para decifrar o arquivo slides.pdf.gpg, irá gerar um arquivo chamado slides.pdf
gpg --output slides.pdf -d slides.pdf.gpg
```

Assinar e verificar arquivos

A assinatura é armazenada um arquivo com a extensão `.asc`, caso use o parâmetro `-a`. Caso contrário, será gerado um arquivo com a extensão `.gpg`. O arquivo gerado é uma combinação do arquivo original com a assinatura. Para assinar um arquivo, execute o comando abaixo:

```
# Saída: slides.pdf.gpg que consiste no arquivo slides.pdf com a assinatura embutida
gpg -s slide.pdf

# É possível gerar assinaturas que não estão embutidas no arquivo original
gpg -o slides.pdf.sig -a -b slides.pdf
```

Para verificar a autenticidade de um arquivo assinado, execute o comando abaixo:

```
# Verifica a autenticidade do arquivo slides.pdf.gpg. A saída será o arquivo slides.pdf e a
    mensagem Good signature, caso a assinatura seja válida
gpg -v slides.pdf.gpg

# Verifica a autenticidade do arquivo slides.pdf e a assinatura slides.pdf.sig
gpg --verify slides.pdf.sig slides.pdf
```

Cifrar e assinar arquivos

É possível cifrar e assinar um arquivo ao mesmo tempo. Para isso, execute o comando abaixo:

```
# Cifrar o arquivo com a chave pública de Bob e assinar com a chave privada de Juca
gpg -es -r bob@example.org -u juca@example.org slides.pdf

# Decifrar o arquivo e verificar a assinatura
gpg -o slides.pdf -d slides.pdf.gpg
```

Desafio

Siga as instruções no tutorial "*Gerenciar a verificação de assinatura de commit*"[2] para configurar a verificação de assinatura de *commit*, por fim faça um *commit* no seu repositório da lista de exercícios 01 da disciplina de Segurança da Informação para demonstrar que você conseguiu configurar a verificação de assinatura de *commit*.

[2] https://docs.github.com/pt/authentication/managing-commit-signature-verification