Tutorial para Assinaturas Digitais em Linux

André Leon S. Gradvohl, Dr.
gradvohl@unicamp.br

1 Introdução

A assinatura digital de um documento é uma estratégia, baseada em criptografia, para validação da autenticidade e verificação da integridade de um documento. A assinatura digital não deve ser confundida com a assinatura digitalizada. Essa última pode ser facilmente fraudada.

Nesse tutorial, utilizaremos as ferramentas *GNU Privacy Guard* (GPG) e a *Open Secure Sockets Layer* (SSL) *and Transport Layer Security* (TLS) ou simplesmente OpenSSL. Essas ferramentas estão disponíveis na maioria das distribuições Linux.

Para a melhor compreensão dos conceitos desse texto, é importante ter utilizado o tutorial para criptografia. Na Seção 2 a seguir, iniciaremos com o *GNU Privacy Guard*.

2 Assinatura com *GNU Privacy Guard*

As informações sobre o GPG podem ser encontradas em `https://gnupg.org`. Para os exemplos a seguir, utilizaremos como arquivo de origem o `documento.txt`.

2.1 Preparação para essa parte do tutorial

Para se preparar para esse tutorial, é preciso verificar se algumas configurações foram feitas.

i. Verifique se você já tem uma base de chaves criadas com o comando a seguir:

```
gpg --list-keys
```

- Se não tiver uma base de chaves, uma nova base será criada.

ii. Agora, use o comando a seguir para criar suas chaves pública e privada.

```
gpg --gen-key
```

- Forneça as informações solicitadas (Nome, Email)
- Em seguida, confirme as informações e aguarde.
- Depois de concluído, use novamente o comando `gpg --list-keys` para verificar se a chave foi gerada.

2.2 Assinatura de um documento digital com GPG

Agora, vamos realizar a assinatura de um documento digital.

i. Crie o arquivo `documento.txt` com uma frase ou conteúdo que apenas você conheça.

ii. Assine o arquivo com o comando a seguir:

```
gpg -s documento.txt
```

- A opção `-s` assina o arquivo e gera outro arquivo chamado `documento.txt.gpg`.
- Informe uma senha assim que solicitado.
- O arquivo `documento.txt.gpg` gerado está comprimido e tem uma assinatura anexada a si.
- Será solicitada uma senha para criptografia.

2.2.1 Verificação da qualidade da assinatura digital

Para verificar a assinatura de um documento digital, use o comando a seguir:

```
gpg --verify documento.txt.gpg
```

2.3 Descriptografia e verificação da assinatura de um documento digital

Ao receber um documento com assinatura digital usando o GPG, esse arquivo estará comprimido e, portanto, não será possível acessar o conteúdo de forma simples. Assim, antes de ter acesso ao conteúdo do documento, será preciso receber a chave pública do remetente desse documento. A Seção 2.3.1 a seguir mostra o processo para instalar a chave pública do remetente.

2.3.1 Instalação da chave pública do remetente

Vamos supor que seu colega enviou a chave pública no arquivo `chavePublica.asc`. Assim, você deverá adicionar essa chave pública à sua base de chaves com o comando a seguir:

```
gpg --import chavePublica.asc
```

- A opção `--import` informa que o arquivo informado como parâmetro, `chavePublica.asc`, será adicionado à sua base de chaves.

2.4 Obtenção do documento original

Dado que a chave pública está instalada, o próximo passo é a obtenção do arquivo original. Para isso, o comando a seguir deve ser utilizado:

```
gpg  --output ArquivoOriginal.txt --decrypt documento.txt.gpg
```

- A opção `--output` informa o nome do arquivo a ser gerado após a descriptografia e a verificação da assinatura. Nesse exemplo, o nome foi `ArquivoOriginal.txt`.
- A opção `--decrypt` informa o nome do arquivo assinado digitalmente. Nesse exemplo, o nome foi `documento.txt.gpg`.

3 Exercício com GPG

Para realizar esse exercício com GPG, é importante ter alguém com quem você possa interagir ou, se for o caso, utilizar duas contas em máquinas diferentes. Nas duas colunas a seguir, utilizaremos dois personagens: Alice (você) e Bob (seu colega).

A Alice quer enviar um documento assinado digitalmente para o Bob. Por sua vez, Bob precisará da chave pública da Alice para confirmar que o documento está íntegro e foi realmente assinado pela Alice.

Antes de começar o exercício, não esqueça de preparar o ambiente primeiro, de acordo com a Seção 2.1. Isso vale tanto para a Alice, quanto para o Bob.

Comandos executados pela Alice

```
# Passo 1: Armazenamento da chave
#          publica em um arquivo.
gpg --output chavePublicaAlice.asc \
    --armor --export alice@mail.com

# Passo 2: Assinatura do arquivo
#          documento.txt, gerando o
#          arquivo documento.txt.gpg
gpg -s documento.txt

# Passo 3: Envia o arquivo
#          documento.txt.gpg
#          e a chave publica da
#          Alice para o Bob
```

Comandos executados pelo Bob

```
# Passo 1: Recebe o documento e a
#          chave publica da Alice.

# Passo 2: Importa a chave publica
#          da Alice no banco de
#          chaves.
gpg --import chavePublicaAlice.asc

# Passo 3: Verifica a assinatura do
#          arquivo recebido e grava
#          o conteudo em outro
#          arquivo.
gpg --output documento.txt \
    --decrypt documento.txt.gpg
```

4 Assinatura com o OpenSSL

Outra ferramenta disponível em distribuições Linux e que pode ser usada para assinaturas digitais é OpenSSL. Os detalhes sobre o OpenSSL estão em `https://www.openssl.org`. Da mesma forma que fizemos na Seção 2, aqui também vamos utilizar o `documento.txt` como arquivo de origem.

4.1 Preparação para essa parte do tutorial

Para se preparar para esse tutorial, é preciso verificar se algumas configurações foram feitas.

i. Se você ainda não tiver, gere a chave privada com o comando a seguir:

```
openssl genpkey -out privkey.pem -algorithm RSA -pkeyopt rsa_keygen_bits:2048
```

- Nesse comando estamos gerando uma chave privada, usando o algoritmo RSA, com uma chave de 2048 bytes.
- O parâmetro `genpkey` informa que será gerada uma chave privada.
- Após a opção `-out` está o nome do arquivo com a chave privada gerada. Nesse caso, será o arquivo `privkey.pem`.
- Após a opção `-algorithm` está o algoritmo que vamos utilizar e o tamanho da chave privada em bytes (`RSA`). Caso essas informações sejam omitidas, o `openssl` vai usar o RSA e o tamanho 2048 como *default*.
- Depois da opção `-pkeyopt` está o parâmetro que identifica o tamanho da chave `rsa_keygen_bits`. Nesse caso, o tamanho é `2048`.

ii. Agora, use o comando a seguir para criar sua chave pública.

```
openssl rsa -in privkey.pem -outform PEM -pubout -out pubkey.pem
```

- Nesse comando estamos gerando uma chave pública a partir da chave privada.
- O parâmetro `rsa` informa que estamos utilizando o algoritmo RSA.
- Após a opção `-in` está o nome do arquivo que contém chave privada (`privkey.pem`).
- Após a opção `-outform` o formato do arquivo que conterá a chave pública. Nesse caso, usaremos o formato *Privacy Enhanced Mail* (PEM), que é bastante comum em aplicações para *web*.
- A opção `-pubout` indica que apenas a chave pública deve ser mostrada. Sem esse parâmetro, seria mostrada apenas a chave privada.
- Por fim, após a opção `-out` está o nome do arquivo com a chave pública. Nesse caso, o nome é `pubkey.pem`.

Ao final dos passos (i) e (ii), você terá as suas chaves pública (no arquivo `pubkey.pem`) e privada (no arquivo `privkey.pem`).

4.2 Assinatura de um documento digital com OpenSSL

Agora, vamos realizar a assinatura de um documento digital com OpenSSL.

i. Crie o arquivo `documento.txt` com uma frase ou conteúdo que apenas você conheça.

ii. Assine o arquivo com o comando a seguir:

```
openssl dgst -sha256 -sign privkey.pem -out docsigned.sha256 documento.txt
```

- O parâmetro `dgst` informa uma operação de *digest*, isto é, será gerado um código *hash* de um documento para posterior verificação de integridade.
- A opção `-sha256` informa que o algoritmo para a operação de *digest* é o *sha256*.
- Após a opção `-sign` está o nome do arquivo que contém a chave privada (`privkey.pem`), que será utilizado para a assinatura.
- Após a opção `-out` está o nome do arquivo assinado. Nesse caso será o arquivo `docsigned.sha256`.
- O último parâmetro é o arquivo que se deseja assinar. Nesse caso, o arquivo `documento.txt`.

4.2.1 Verificação da assinatura digital

Para verificar se a assinatura foi realizada corretamente, use o comando a seguir:

```
openssl dgst -sha256 -verify pubkey.pem -signature docsigned.sha256 documento.txt
```

- O parâmetro `dgst` informa uma operação de *digest*, isto é, será gerado um código *hash* de um documento para posterior verificação de integridade.
- A opção `-sha256` informa que o algoritmo para a operação de *digest* é o *sha256*.
- A opção `-verify` informa uma operação de verificação de assinatura, usando a chave pública `pubkey.pem`.
- Após a opção `-signature` está o nome do arquivo assinado. Nesse caso será o arquivo `docsigned.sha256`.
- O último parâmetro é o arquivo que se deseja confirmar a assinatura. Nesse caso, o arquivo `documento.txt`.

4.3 Envio do documento e da assinatura

Diferente do GPG, no OpenSSL, o documento e a assinatura estão em arquivos separados. Assim, ao receber ambos os documentos, o destinatário deve executar o comando descrito na Seção 4.2.1, anterior.

5 Exercício com OpenSSL

Para realizar esse exercício com OpenSSL, é importante ter alguém com quem você possa interagir ou, se for o caso, utilizar duas contas em máquinas diferentes.

Nas duas colunas a seguir, utilizaremos dois personagens: Alice (você) e Bob (seu colega). A Alice quer enviar um documento assinado digitalmente para o Bob. Portanto, Alice enviará o documento original (`documento.txt`), a assinatura (`docsigned.sha256`) e a sua chave pública (`privkey.pem`). Por sua vez, Bob precisará da chave pública da Alice para confirmar que o documento está íntegro e foi realmente assinado pela Alice.

Comandos executados pela Alice

```
# Passo 1: Geracao da chave privada.
openssl genpkey -out privkey.pem \
        -algorithm RSA \
        -pkeyopt rsa_keygen_bits:2048

# Passo 2: Geracao da chave publica.
openssl rsa -in privkey.pem \
        -outform PEM -pubout
        -out pubkey.pem

# Passo 3: Assinatura do documento.
openssl dgst -sha256 \
        -sign privkey.pem \
        -out docsigned.sha256 \
        documento.txt

# Passo 4: Envia os arquivos
#          documento.txt,
#          docsigned.sha256,
#          e a chave publica da
#          Alice para o Bob
```

Comandos executados pelo Bob

```
# Passo 1: Recebe o documento
#          original a assinatura,
#          e a chave publica da
#          Alice.

# Passo 2: Verifica se os documentos
#          recebidos estao integros
#          e se a assinatura confere
openssl dgst -sha256 \
      -verify pubkey.pem \
      -signature docsigned.sha256 \
      documento.txt
```

Licença de uso

Para citar este texto, use as informações a seguir:
Gradvohl, A. L. S. Tutorial para Assinaturas Digitais em Linux. Zenodo. DOI: 10.5281/zenodo.6973949, 2022.