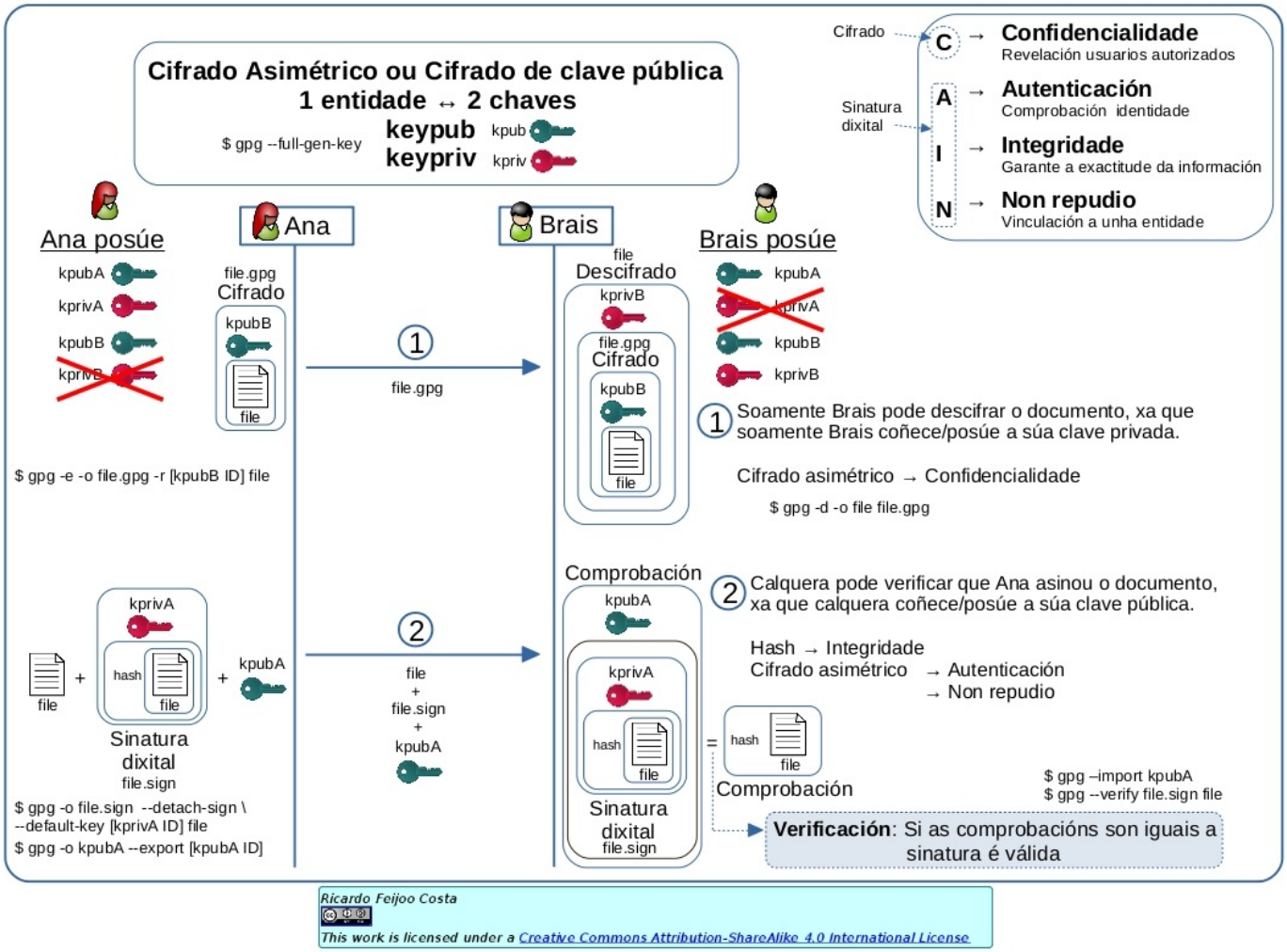
Práctica BRS
CAIN - gpg

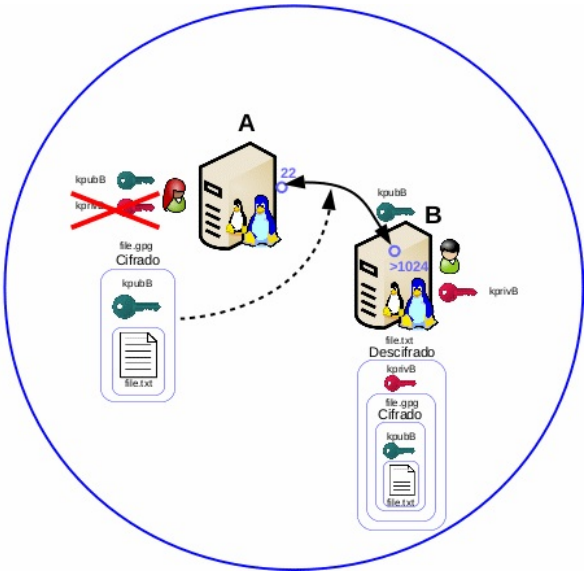


ESCENARIO

Máquinas virtuais ou físicas:
RAM ≤ 2048MB CPU ≤ 2 PAE/NX habilitado
Rede: 192.168.120.0
BIOS: Permite arranque dispositivo extraíble: CD/DVD, USB

Máquina A:
Servidor SSH
IP/MS: 192.168.120.100/24
SO: Kali Live amd64
Usuarios: Ana e Brais

Máquina B:
Cliente SSH
IP/MS: 192.168.120.101/24
SO: Live GNU/Linux
Usuario: Brais



LIMITACIÓN DE RESPONSABILIDADE O autor do presente documento declina calquera responsabilidade asociada ao uso incorrecto e/ou malicioso que puidese realizarse coa información exposta no mesmo. Por tanto, non se fai responsable en ningún caso, nin pode ser considerado legalmente responsable en ningún caso, das consecuencias que poidan derivarse da información contida nel ou que esté enlazada dende ou hacia el, incluíndo os posibles erros e información incorrecta existentes, información difamatoria, así como das consecuencias que se poidan derivar sobre a súa aplicación en sistemas de información reais e/ou virtuais. Este documento foi xerado para uso didáctico e debe ser empregado en contornas privadas e virtuais controladas co permiso correspondente do administrador desas contornas.

NOTAS:

- **CAIN**: Confidencialidade, Autenticación, Integridade, Non repudio
- **gpg**
- **OpenPGP**
- **Philip Zimmermann**

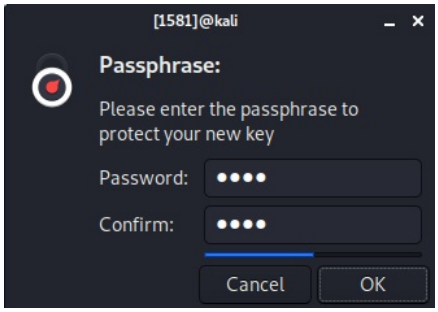
Práctica

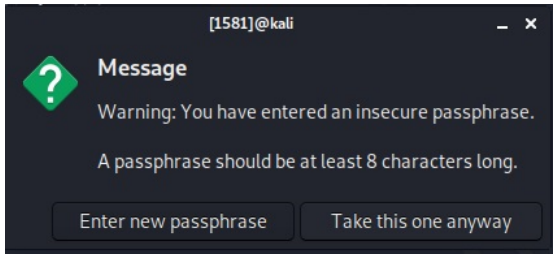
Configurar cifrado asimétrico

1. Arrancar coa distro Kali Live amd64. Abrir un terminal e executar:
   kali@kali:~$ gpg --full-gen-key #Crear un par de chaves: pública e privada a través de diálogos nun menú. Escollemos:
   1 → Para poder cifrar e asinar
   4096 → Cifrado de 4096bits
   2y → Validez de 2 anos
   y → Confirmamos con y (yes) xa que estamos de acordo co elixido
   User Kali → Nome real do usuario
   kali@kali.local → Email do usuario
   live amd64 → Comentario
   O → Ok, estamos de acordo.

```
kali@kali:~$ gpg --full-gen-key
gpg (GnuPG) 2.2.17; Copyright (C) 2019 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/kali/.gnupg' created
gpg: keybox '/home/kali/.gnupg/pubring.kbx' created
Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 2y
Key expires at Tue 22 Nov 2022 06:22:06 PM UTC
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: User Kali
Email address: kali@kali.local
Comment: live amd64
You selected this USER-ID:
    "User Kali (live amd64) <kali@kali.local>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
```

   Passphrase → Pomos como contrasinal: 1234 e repetimos o contrasinal. O sistema avisa que o contrasinal non é seguro, pero confirmamos que queremos ese contrasinal





```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/kali/.gnupg/trustdb.gpg: trustdb created
gpg: key 4F90C79614E37874 marked as ultimately trusted
gpg: directory '/home/kali/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/kali/.gnupg/openpgp-revocs.d/862F21E03338AE413923D10A4F90
C79614E37874.rev'
public and secret key created and signed.

pub   rsa4096 2020-11-22 [SC] [expires: 2022-11-22]
      862F21E03338AE413923D10A4F90C79614E37874
uid                      User Kali (live amd64) <kali@kali.local>
sub   rsa4096 2020-11-22 [E] [expires: 2022-11-22]
```

kali@kali:~$ ls -lahtr $HOME/.gnupg #Executar o comando ls dentro do cartafol de traballo do usuario ($HOME=/home/kali) coas opcións -l, -a, -h, -t e -r. A opción -l permite amosar de forma extendida o atopado (tipo de ficheiro, permisos, propietarios...), a opción -h engade unha letra indicativa de tamaño, tal como M para megabytes binarios (`mebibytes'), a cada tamaño. A opción -t clasifica polo tempo de modificación (o `mtime' no inodo) en vez de alfabeticamente, cos ficheros máis recientes en primeiro lugar. A opción -r clasifica en orde inversa. Polo tanto, o comando lista ficheiros e directorios do directorio /home/kali/.gnupg (que garda o anel de chaves) amosando de abaixo hacia arriba os máis recentes e en formato de lectura de tamaño máis amigable para as persoas (K, M, G...)

```
kali@kali:~$ ls -lahtr .gnupg/
total 12K
drwxr-xr-x 14 kali kali  440 Nov 22 18:51 ..
-rw-------  1 kali kali   32 Nov 22 18:51 pubring.kbx~
drwx------  2 kali kali   80 Nov 22 18:57 private-keys-v1.d
-rw-r--r--  1 kali kali 2.5K Nov 22 18:57 pubring.kbx
-rw-------  1 kali kali 1.3K Nov 22 18:57 trustdb.gpg
drwx------  2 kali kali   60 Nov 22 18:57 openpgp-revocs.d
drwx------  4 kali kali  140 Nov 22 18:57 .
kali@kali:~$
```

kali@kali:~$ gpg -k #Listar as chaves públicas gardadas no anel de chaves

```
kali@kali:~$ gpg -k
/home/kali/.gnupg/pubring.kbx
------------------------------
pub   rsa4096 2020-11-22 [SC] [expires: 2022-11-22]
      862F21E03338AE413923D10A4F90C79614E37874
uid           [ultimate] User Kali (live amd64) <kali@kali.local>
sub   rsa4096 2020-11-22 [E] [expires: 2022-11-22]

kali@kali:~$
```

kali@kali:~$ gpg -K #Listar as chaves privadas gardadas no anel de chaves

```
kali@kali:~$ gpg -K
/home/kali/.gnupg/pubring.kbx
------------------------------
sec   rsa4096 2020-11-22 [SC] [expires: 2022-11-22]
      862F21E03338AE413923D10A4F90C79614E37874
uid           [ultimate] User Kali (live amd64) <kali@kali.local>
ssb   rsa4096 2020-11-22 [E] [expires: 2022-11-22]

kali@kali:~$
```

Cifrar un arquivo

2. Cifrar un arquivo

kali@kali:~$ echo 1234 > file.txt #Crear o ficheiro file.txt co contido 1234
kali@kali:~$ cat file.txt #Ver o contido do ficheiro file.txt
kali@kali:~$ gpg -e -o file.gpg -r kali@kali.local file.txt #Cifrar coa clave pública do usuario identificado co email kali@kali.local o ficheiro file.txt. O ficheiro cifrado terá o nome file.gpg
kali@kali:~$ cat file.gpg #Ver o contido do ficheiro file.gpg
kali@kali:~$ md5sum file.gpg #Crear hash MD5 do ficheiro file.gpg

kali@kali:~$ gpg -e -o file2.gpg -r kali@kali.local file.txt #Cifrar coa clave pública do usuario identificado co email kali@kali.local o ficheiro file.txt. O ficheiro cifrado terá o nome file2.gpg
kali@kali:~$ cat file2.gpg #Ver o contido do ficheiro file2.gpg
kali@kali:~$ md5sum file2.gpg #Crear hash MD5 do ficheiro file2.gpg.

3. Compara os "hash" dos ficheiros file.gpg e file2.gpg anteriores. Que acontece? Por que?

Os "hash" son distintos: A diferenza entre os dous ficheiros .gpg, a pesar de ter o mesmo contido orixinal, débese principalmente á natureza aleatoria do cifrado GPG. Imaxina isto: Cifrar un ficheiro con GPG é como envolver un paquete nunha caixa moi segura. Cada vez que envolves o paquete, usas unha combinación diferente de cadeados e chaves únicas. Mesmo se o paquete interior (o ficheiro orixinal) é o mesmo, a caixa final (o ficheiro .gpg) será diferente debido a esas combinacións únicas.
Por que pasa isto?

- **Clave simétrica aleatoria:** GPG xera unha clave secreta aleatoria cada vez que cifra un ficheiro. Esta clave emprégase para cifrar o contido e logo se cifra coa clave pública do destinatario. Como esta clave secreta cambia en cada cifrado, o resultado final tamén cambia.
- **Metadatos:** Ademais do contido cifrado, GPG inclúe información adicional como a data e hora de cifrado, o algoritmo usado, etc. Esta información, que pode variar entre cifrados, tamén contribúe ás diferenzas entre os ficheiros .gpg.

En resumo:

- A variación nos ficheiros .gpg é unha característica desexable do cifrado GPG. Garante que cada ficheiro cifrado sexa único e difícil de descifrar para alguén que non teña a clave privada correspondente. Aínda que os ficheiros cifrados parezan diferentes, o contido orixinal pode ser recuperado de forma exacta ao descifralos coa clave correcta.
- A aleatoriedade no proceso de cifrado GPG débese en gran parte ao uso de vectores de inicialización (IV) aleatorios en os algoritmos de cifrado en bloque. Estes IVs aseguran que o cifrado dun bloque de datos sexa diferente cada vez, mesmo se o bloque de datos en si é idéntico. Ademais, a inclusión de un hash da mensaxe (HMAC) no ficheiro cifrado proporciona unha garantía adicional de integridade.

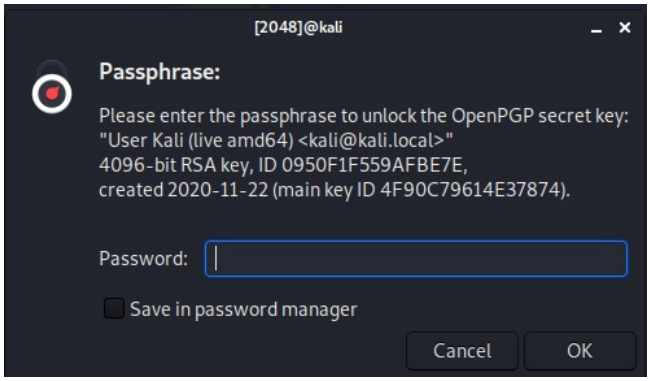
Descifrar un arquivo

4. Descifrar un arquivo

kali@kali:~$ mkdir descifrados #Crear o directorio descifrados

kali@kali:~$ gpg -d -o descifrados/file-descifrado.txt file.gpg #Descifrar coa chave privada do usuario co cal se cifrou o arquivo mediante a súa chave pública. Como este usuario tiña configurada o passphrase solicítase. O ficheiro descifrado será gardado en descifrados/file-descifrado.txt.

```
kali@kali:~$ mkdir descifrados
kali@kali:~$ gpg -d -o descifrados/file-descifrado.txt file.gpg
```



kali@kali:~$ cat descifrados/file-descifrado.txt #Ver o contido do ficheiro descifrados/file-descifrado.txt

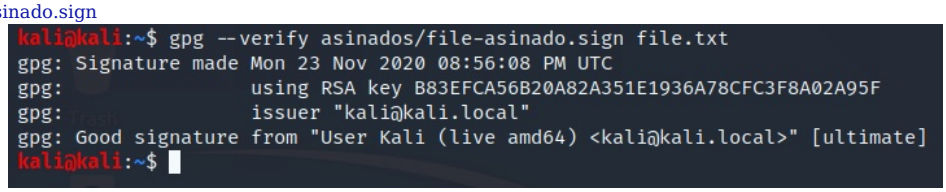
```
kali@kali:~$ gpg -d -o descifrados/file-descifrado.txt file.gpg
gpg: encrypted with 4096-bit RSA key, ID 0950F1F559AFBE7E, created 2020-11-22
      "User Kali (live amd64) <kali@kali.local>"
kali@kali:~$ cat descifrados/file-descifrado.txt
1234
kali@kali:~$
```

Asinar un arquivo

5. Asinar un arquivo
   kali@kali:~$ mkdir asinados #Crear o directorio asinados
   kali@kali:~$ gpg -o asinados/file-asinado.sign --detach-sign --default-key kali@kali.local file.txt #Asinar coa chave privada do usuario kali. Como este usuario tiña configurada o passphrase solicítase, a non ser que xa fora solicitado durante os últimos 10 minutos. O ficheiro asinado será gardado en asinados/file-asinado.sign

Verificar a sinatura dun arquivo

6. Verificar a sinatura dun arquivo
   kali@kali:~$ gpg --verify asinados/file-asinado.sign file.txt #Verificar a sinatura do ficheiro file.txt mediante o ficheiro asinado file-asinado.sign



Resolver

7. Realiza o cifrado e descifrado dun ficheiro empregando 2 entidades usuarios: Ana e Brais. Para iso:
   - Crea os usuarios Ana e Brais

```
root@192.168.120.100# useradd -m -d /home/ana -s /bin/bash -p $(mkpasswd -m sha-512 abc123.) ana
root@192.168.120.100# useradd -m -d /home/brais -s /bin/bash -p $(mkpasswd -m sha-512 abc123.) brais
root@192.168.120.101# useradd -m -d /home/brais -s /bin/bash -p $(mkpasswd -m sha-512 abc123.) brais
```

   - Crea o par de chaves para o usuario Brais: kpubB e kprivB.

```
brais@192.168.120.101:~$ gpg --full-gen-key
```

   - Brais envía a súa chave pública (kpubB) ao usuario Ana mediante conexión SSH.

```
brais@192.168.120.101:~$ gpg -o kpubB --export brais@brais.local
brais@192.168.120.101:~$ scp -P 22 kpubB brais@192.168.120.100:/tmp
```

   - Ana crea un ficheiro e cifra ese ficheiro coa chave pública de Brais (kpubB).

```
ana@192.168.120.100:~$ gpg --import /tmp/kpubB
ana@192.168.120.100:~$ echo 'Serás capaz de ler isto...' > /tmp/file.txt
ana@192.168.120.100:~$ gpg -e -o /tmp/file.gpg -r brais@brais.local /tmp/file.txt
```

   - Ana fai chegar ese ficheiro cifrado a Brais. Por exemplo, Ana copia ese ficheiro cifrado en /tmp e Brais mediante conexión SSH recolle o ficheiro.

```
brais@192.168.120.101:~$ scp -P 22 brais@192.168.120.100:/tmp/file.gpg .
```

   - Brais a través da súa chave privada (kprivB) descifra ese ficheiro.

```
brais@192.168.120.101:~$ gpg -d -o ./file.txt ./file.gpg
brais@192.168.120.101:~$ cat file.txt
```

8. Realiza o cifrado e descifrado dun ficheiro empregando 2 entidades usuarios: Ana e Brais. Ademais Ana debe asinar o arquivo cifrado. Para iso:
   - Realiza o apartado anterior.
   - Crea o par de chaves para o usuario Ana: kpubA e kprivA.

```
ana@192.168.120.100:~$ gpg --full-gen-key
```

   - Ana asina o arquivo file.gpg

```
ana@192.168.120.100:~$ gpg -o /tmp/file.sign --detach-sign --default-key ana@ana.local /tmp/file.gpg
```

   - Ana deixa a súa chave pública, xunto co ficheiro cifrado e o ficheiro cifrado asinado en /tmp para que Brais poida copialos mediante conexión SSH.

```
ana@192.168.120.100:~$ gpg -o /tmp/kpubA --export ana@ana.local
```

   - Brais copia a chave pública de Ana (kpubA) e os ficheiros cifrado (file.gpg) e asinado (file.sign) mediante conexión SSH.

```
brais@192.168.120.101:~$ scp -P 22 brais@192.168.120.100:/tmp/kpubA .
brais@192.168.120.101:~$ scp -P 22 brais@192.168.120.100:/tmp/file.gpg .
brais@192.168.120.101:~$ scp -P 22 brais@192.168.120.100:/tmp/file.sign .
```

   - Brais importa a chave pública de Ana e comproba a sinatura do ficheiro file.gpg mediante file.sign.

```
brais@192.168.120.101:~$ gpg --import kpubA
brais@192.168.120.101:~$ gpg --verify file.sign file.gpg
```

   - Brais unha vez comprobada a sinatura descifra o ficheiro file.gpg a través da súa chave privada (kprivB).

```
brais@192.168.120.101:~$ gpg -d -o ./file-ok.txt ./file.gpg
brais@192.168.120.101:~$ cat file-ok.txt
```

Ricardo Feijoo Costa