Configuración da seguridade no servidor Web

Índice

1. Configuración da seguridade no servidor web

1.1 Control de acceso

O control de acceso en Apache permite evitar ou permitir o acceso de determinados computadores.

Nas versións previas á 2.4, dito control de acceso facíase empregando as directivas `Order`, `Allow` e `Deny`. Pero estas directivas, a pesares de que aínda son recoñecidas na versión 2.4, xa non deben ser empregadas, posto que desaparecerán en futuras versións.

1.1.1 Directiva Require

Permite indicar se unha petición ten acceso ao recurso solicitado en función da súa IP, enderezo simbólico (dominio), usuario autenticado, pertenza a un grupo de usuarios, etc.

- `Require all granted`
  O acceso permítese sempre.
- `Require all denied`
  O acceso non se permite nunca.
- `Require ip ip_ou_rango [ip_ou_rango] ...`
  O acceso permítese desde esa IP.
- `Require host enderezo_simbólico`
  O acceso permítese desde ese enderezo simbólico.
- `Require user usuario1 [usuario2] ...`
  O acceso se permite aos usuarios especificados.
- `Require group grupo1 [grupo2] ...`
  O acceso se permite aos grupos especificados.
- Cada una das opcións da directiva Require pode ser negada empregando a opción `not`.

```
Require not ip 192.168.205
```

- Se varias directivas Require se empregan nunha mesma sección sen estar contidas nun contedor de autorización, o acceso se permite se se cumpre algunha delas (equivalente a `RequireAny`).

Contedores de autorización

Estes contedores poden aniñarse entre eles para indicar condicións máis complexas.

RequireAll

Para que se permita o acceso ao recurso, deben cumprise todas as condicións indicadas por cláusulas `Require` que contén.

RequireAny

Para que se permita o acceso ao recurso, debe cumprise algunha das condicións indicadas por cláusulas `Require` que contén.

RequireNone

Para que se permita o acceso ao recurso, non debe cumprise ningunha das condicións indicadas por cláusulas `Require` que contén.

Exemplos

- Permitir o acceso desde todas as IP menos a 10.252.46.165:

```
<RequireAll>
    Require all granted
    Require not ip 10.252.46.165
</RequireAll>
```

- Permitir o acceso a todos menos aos hosts de `xunta.gal`

```
<RequireAll>
    Require all granted
    Require not xunta.gal
</RequireAll>
```

- Permitir o acceso desde todas as IP menos a 10.252.46.165 e a todos os hosts menos aos de `xunta.gal`:

```
<RequireAll>
    Require all granted
    Require not ip 192.168.205
    Require not host xunta.gal
</RequireAll>
```

- Permitir o acceso ao directorio `/www/interno` ben aos hosts pertencentes a `xunta.gal` ou os que están na rede `192.168.0.0/24`.

```
<Directory "/www/interno">
   <RequireAny>
      Require host xunta.gal
      Require ip 192.168.0
   </RequireAny>
</Directory>
```

1.2 Autorización e autenticación

Para configurar o servidor Apache para que sexa capaz de autenticar aos usuarios e verificar a autorización do mesmo ao recurso solicitado, é necesario realizar as seguintes accións:

- Crear un ficheiro con usuarios.
- Crear un ficheiro con grupos (se é necesario).
- Definir as directivas no ficheiro de configuración.

É necesario que tanto os ficheiros de usuarios como os de grupos, se atopen almacenados fóra dos directorios publicados, para que desta forma ninguén poida descargalos.

O control de acceso pode facerse para todos os ficheiros do sitio web ou para algúns ficheiros (mediante a directiva `<Files>`) ou directorios (mediante a directiva `<Directory>`).

Autenticación Basic e Digest

A diferenza entre ambas é que no modo de autenticación Basic, os contrasinais envíanse en claro, mentres que en Digest os contrasinais envíanse cifrados.

O módulo de Apache que controla a autenticación Basic é `auth_basic` e está habilitado por defecto, pola contra, o que controla a autenticación Digest é `auth_digest` e está deshabilitado por defecto.

Creación de usuarios

Para crear o ficheiro utilizamos o comando `htpasswd` (se a autenticación é Basic) ou `htdigest` (se a autenticación é Digest). A primeira vez que o utilicemos temos que poñerlle a opción `-c`, para que cre o arquivo. Crearíamos o usuario da seguinte maneira:

```
htpasswd [-c] /etc/apache2/contr_basic xiana
```

Para `htdigest` temos que indicar, ademais, o dominio ao que pertence o usuario:

```
htdigest [-c] /etc/apache2/contr_digest profes xiana
```

Para empregar a autenticación por usuarios, debe estar habilitado o módulo `authz_user`, que o está por defecto.

Creación de grupos

Créase un ficheiro cun grupo por liña coa seguinte sintaxe:

```
nomeGrupo: usuario1 usuario2 usuario3 ...
```

Para empregar a autenticación por usuarios, debe estar habilitado o módulo `authz_groupfile`, que o está por defecto.

Directivas de restrición de acceso

Directiva AuthType

Indicaremos o tipo de autenticación Basic ou Digest

Directiva AuthName

Indica a área de influencia da autenticación. A cadea indicada nesta directiva é a que aparecerá no cadro de diálogo que se mostra ao usuario. En autenticación Digest, indica o dominio ao que ha de pertencer o usuario.

Directiva AuthUserFile

Indica a ruta ao ficheiro no que se almacenan os usuarios e contrasinais.

Directiva AuthGroupFile

Indica a ruta ao ficheiro no que se almacenan os grupos.

Directiva Require

Indica que usuarios ou grupos teñen acceso á área. Se se indica `valid-user` significa que se permitirá o acceso a calquera dos usuarios que aparezan no ficheiro.

```
Require user profesor1 profesor2
Require group profes
```

Exemplo

- Permitir o acceso a calquera usuario válido con autenticación Basic:

```
AuthType basic
AuthName "Zona privada IES Aller"
AuthUserFile "/etc/apache2/contr_basic"
Require valid-user
```

- Permitir o acceso ao usuario `xiana` con autenticación Digest:

```
AuthType Digest
AuthName "profes"
AuthBasicProvider file
AuthUserFile "/etc/apache2/contr_digest"
Require user xiana
```

- Permitir o acceso ao grupo `profesores` con autenticación Basic:

```
AuthType Basic
AuthName "Para invitados"
AuthBasicProvider file
AuthUserFile "/etc/apache2/contr_basic"
AuthGroupFile "/etc/apache2/grupos"
Require group profesores
```

Exemplo con control de acceso

- Permitir o acceso ao directorio `/www/docsadmin` ben ao usuario `superadmin` ou ben aos usuarios do grupo `admins` que accedan desde a rede `192.168.0.0/24`.

```
<Directory "/www/docsadmin">
   <RequireAny>
      Require user superadmin
      <RequireAll>
         Require group admins
         Require ip 192.168.0
      </RequireAll>
   </RequireAny>
</Directory>
```

1.3 Arquivos .htaccess

Os ficheiros `.htaccess` son ficheiros de configuración distribuída. A súa utilidade é incluír directivas que afecten unicamente ao directorio no que están situados e, por extensión, aos seus subdirectorios.

Recoméndase evitar o uso destes ficheiros sempre que sexa posible, xa que a procura dos mesmos a través dos directorios retarda moito o servidor Apache e pode constituír un problema de seguridade. É dicir, sempre que a persoa que vai configurar o devandito directorio teña acceso aos ficheiros de configuración globais, debe situar nestes as directivas relativas a ese directorio (dentro dun bloque `Directory`).

Hai situacións, como por exemplo aquelas nas que varios usuarios empregan un mesmo servidor Apache para os seus sitios web e ditos usuarios non teñen acceso aos ficheiros globais de configuración, nas que a única solución é o emprego de ficheiros `.htaccess`.

Cando está permitido o uso de arquivos `.htaccess` e estes se modifican, non é necesario o reinicio do servidor Apache. De feito, se o usuario tivese permiso para levar a cabo este reinicio, non debería modificar a configuración en ditos arquivos, senón no arquivo principal de configuración do servidor.

1.3.1 Directiva AccessFileName

Utilízase para modificar o nome dos ficheiros de configuración distribuídos (por defecto, `.htaccess`). Exemplo:

```
AccessFileName .config
```

Isto faría que a configuración dun directorio se situase en ficheiros chamados `.config`.

1.3.2 Directiva AllowOverride

Indica que directivas declaradas nos ficheiros `.htaccess` deben sobrescribir ás declaradas nos ficheiros de configuración globais. Os seus valores poden ser, entre outros:

- `None`: Os ficheiros `.htaccess` son ignorados completamente.
- `All`: Todas as directivas declaradas nos `.htaccess` sobrescriben ás declaradas nos ficheiros globais.
- `AuthConfig`: Permite o uso de directivas de autorización (`AuthGroupFile`, `AuthName`, `AuthType`, `AuthUserFile`, `Require`, etc.).
- `Indexes`: Permite o uso de directivas que controlan a indexación de directorios (por exemplo, `DirectoryIndex`).
- `Options`: Permite directivas que controlan características específicas do directorio (por exemplo, `Options`).

1.4 Criptografía

A criptografía é a ciencia que estuda os algoritmos de cifrado e descifrado da información.

A información orixinal que debe protexerse denomínase *texto plano* (texto en claro). O cifrado é o proceso de converter o texto plano nun texto imposible de ler chamado *texto cifrado* ou *criptograma*. Para obter un texto cifrado, aplícase un algoritmo de cifrado, utili-

zando unha clave, ao texto plano:

Para recuperar a mensaxe orixinal, collerase o criptograma (texto cifrado), a clave e aplicarase o algoritmo de descifrado:

Os principais obxectivos que busca a criptografía son:

- Confidencialidade: a información revélase unicamente aos usuarios autorizados.
- Integridade: garante a exactitude da información contra a alteración, perda ou destrución da información.
- Autenticación: comprobación da identidade.
- Non repudio (irrenunciabilidade ou vinculación): vincula un documento ou transacción a unha persoa ou equipo.

Dependendo do número de claves utilizadas polos algoritmos de cifrado/descifrado, existen dous tipos de métodos criptográficos:

- Criptografía de clave simétrica, que emprega unha única clave.
- Criptografía de clave asimétrica, que emprega dúas claves.

Existen sistema chamados de criptografía híbrida, que empregan os dous sistemas (simétrico e asimétrico) na procura dun mellor rendemento.

1.4.1 Criptografía de clave simétrica

Úsase unha única clave para cifrar e descifrar mensaxes. Unha vez que as dúas partes que se van a comunicar posúen a mesma clave, o remitente cifra unha mensaxe con ela, envíao ao destinatario e este descífrao coa mesma.

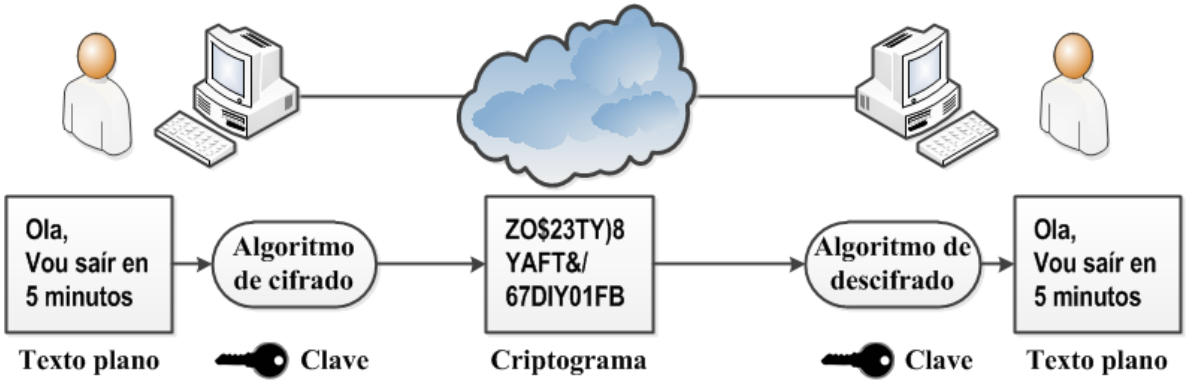


Fig. Criptografía de clave simétrica

Un problema que teñen os sistemas de clave simétrica é o intercambio de claves; xa que, antes de poder enviar información, os participantes teñen que intercambiarse a clave. Se a clave cae en malas mans toda comunicación que use esa clave comprometida non será segura; xa que, persoas non autorizadas poderán acceder aos contidos transmitidos.

1.4.2 Criptografía de clave asimétrica

Os sistemas criptográficos de clave simétrica teñen o problema da distribución da clave entre os participantes da comunicación. Nos sistemas de clave asimétrica non existe este problema e cada participante ten un parella de claves propias:

- Clave privada:
  - O seu propietario debe mantela en segredo a toda costa.
  - Permite cifrar mensaxes (que poderán ser descifrados coa clave pública).
  - Permite descifrar mensaxes cifrados coa clave pública do propietario.
- Clave pública:
  - Debe repartirse a todos os usuarios cos que se queira comunicar.
  - Úsase para enviar mensaxes cifradas ao propietario, que só el poderá descifrar usando a súa clave privada.
  - Úsase para descifrar mensaxes enviadas polo propietario e cifrados coa clave privada.

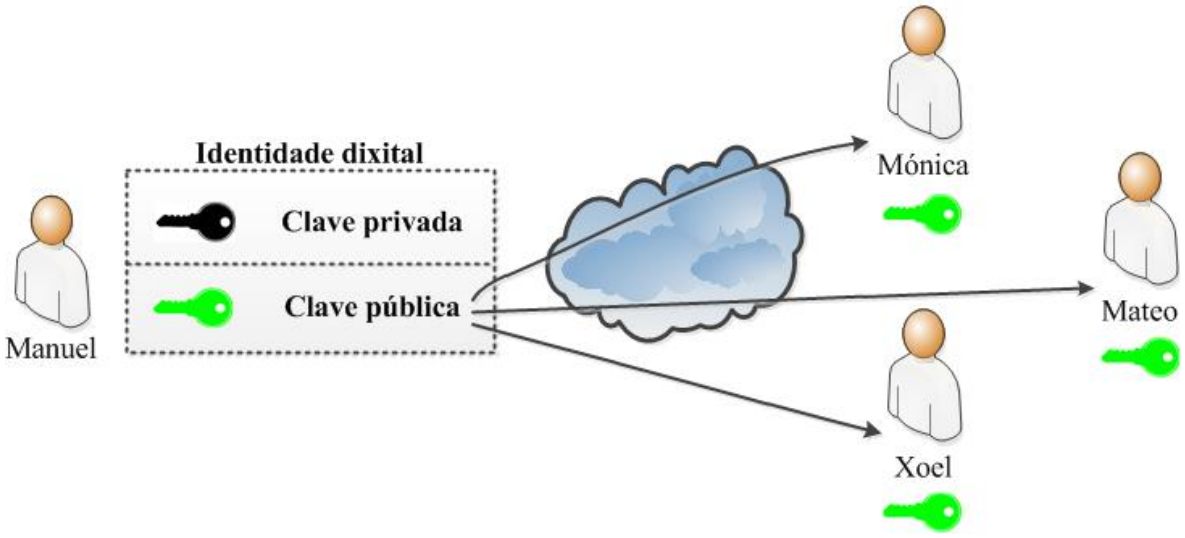


Fig. Criptografía de clave asimétrica

Na seguinte imaxe pode verse o que sucede se Mónica envía unha mensaxe a Manuel cifrándoa coa clave pública de Manuel. O que consegue é que únicamente Manuel a poida descifrar usando a súa clave privada (a de Manuel). Ninguén, agás o que teña a clave privada de Manuel poderá descifrar a mensaxe; polo tanto, conseguiuse a confidencialidade.

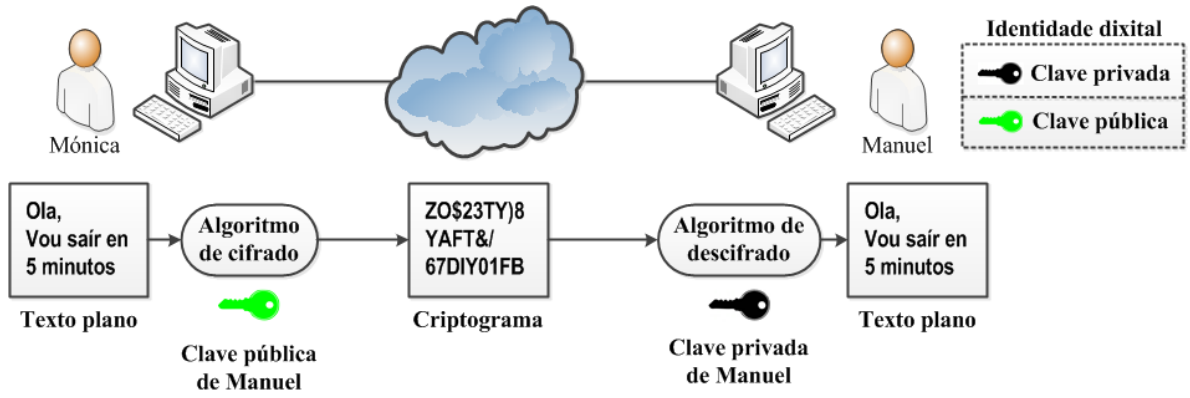


Fig. Confidencialidade usando criptografía de clave asimétrica

Agora Manuel envía unha mensaxe cifrada coa súa clave privada a Mónica. Mónica, usando a clave pública de Manuel, será capaz de descifrar a mensaxe e terá a seguridade de que a mensaxe foi enviada por Manuel; xa que, ninguén agás Manuel posúe a clave privada (de Manuel). Conséguese a identificación e a vinculación (non repudio) do remitente.

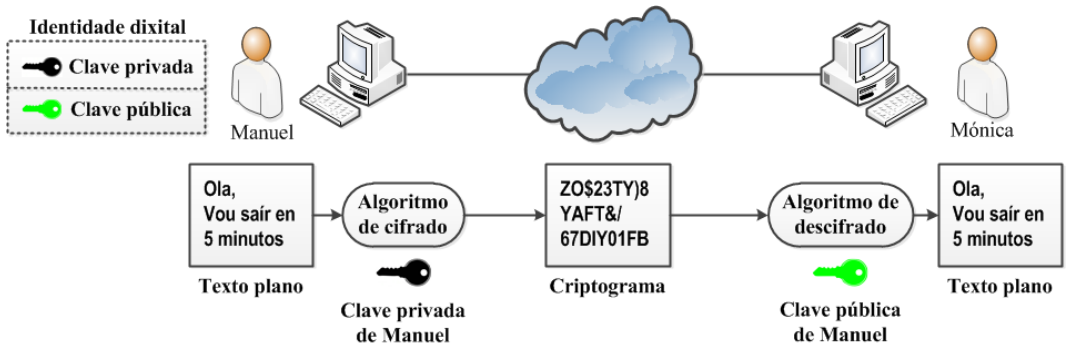


Fig. Identificación e vinculación usando criptografía de clave asimétrica

Na situación onde Manuel e Mónica queren comunicarse buscando á vez confidencialidade, identificación e vinculación:

- Mensaxes de Manuel a Mónica:
  - Manuel cifra a mensaxe coa súa clave privada e despois a cifra coa clave pública de Mónica.
  - A mensaxe dobremente cifrada envíase a Mónica.
  - Unha vez que lle chega a mensaxe a Mónica, esta descifra a mensaxe coa súa clave privada e o resultado procede a descifralo coa clave pública de Manuel. Unha vez aplicado este procedemento, Mónica recupera a mensaxe orixinal de Manuel.

  O que se consegue con este sistema é:

- Ao cifrar en primeiro lugar coa clave privada do remitente, o destinatario poderá estar seguro que a mensaxe realmente procede do remitente; e polo tanto, garántese a identificación e vinculación do remitente.
- Ao cifrar en segundo lugar coa clave pública do destinatario, asegúrase que unicamente o destinatario pode ver o contido da mensaxe (grazas a que a única clave que pode descifrar a mensaxe é a privada do destinatario); e polo tanto, garántese a confidencialidade do envío.

Na seguinte imaxe pode verse o proceso no envío de mensaxes de Manuel a Mónica:

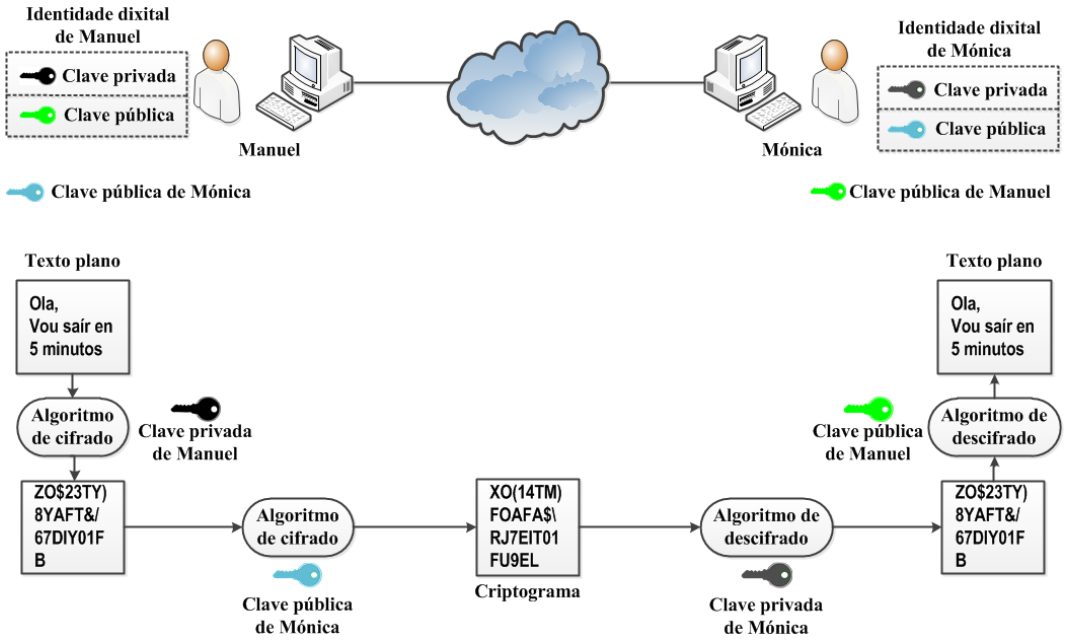


Fig. Confidencialidade, identificación e vinculación usando criptografía de clave asimétrica

Os métodos criptográficos garanten:

- Que esa parella de claves só se poida xerar unha vez, de modo que se pode asumir que non é posible que dúas persoas obtivesen casualmente a mesma parella de claves.
- Que a partir da clave pública non se pode deducir nada da clave privada.
- Que o que cifra a clave pública só pode ser descifrado coa privada e o que cifra a clave privada só o descifra a pública.

1.5 Sistemas criptográficos híbridos

Debido a que os sistemas de cifrado asimétricos son computacionalmente máis custosos que os simétricos e que estes últimos teñen o problema de intercambio de claves, inventáronse os sistemas de criptografía híbrida. Nestes sistemas híbridos, ao comezo da comunicación úsase un sistema de claves asimétricas para intercambiar de forma segura unha clave simétrica de sesión, que se usará para cifrar o resto da comunicación de forma máis eficiente.

Na seguinte figura pode verse un resumo do proceso:

- Paso 1: Manuel envía a clave pública a Mónica.
- Paso 2: Mónica xera unha clave de sesión que é cifrada usando a clave pública de Manuel e a envía a Manuel. Este procede a descifrala usando a súa clave privada.
- Paso 3: Unha vez que os dous participantes da comunicación xa coñecen a clave de sesión, poden proceder a enviar mensaxes cifradas e descifralas usando a mesma clave (criptografía simétrica).

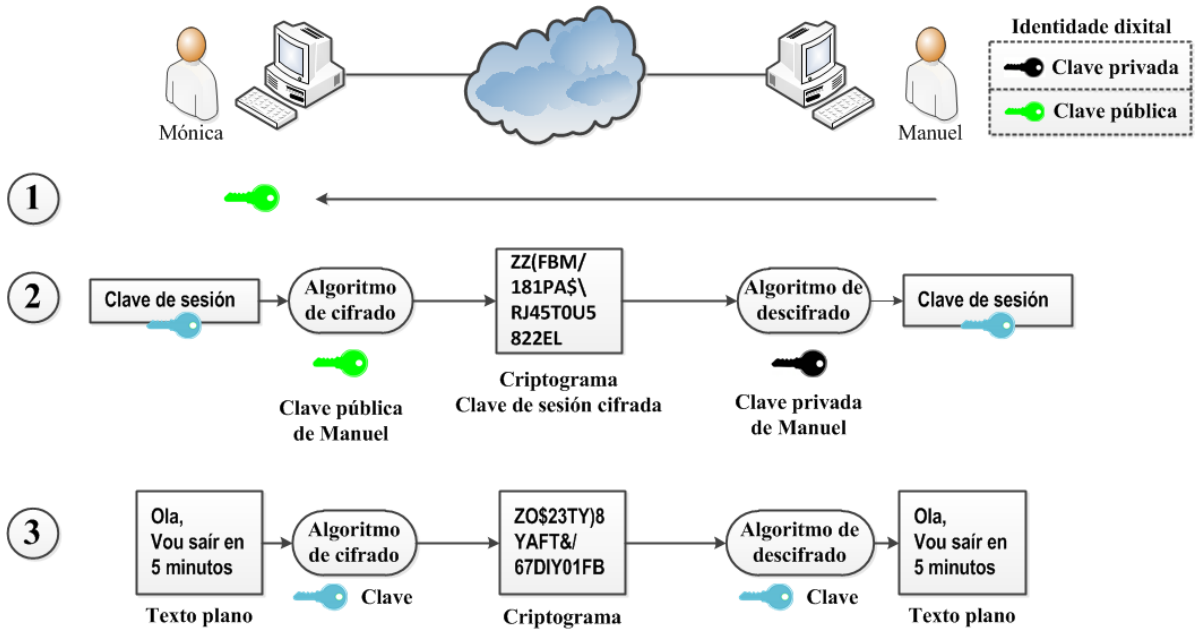
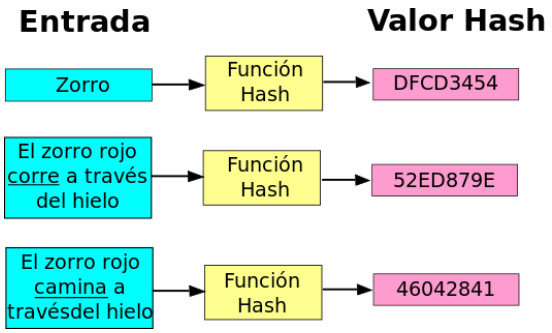


Fig. Criptografía híbrida

1.5.1 Funcións Hash

Unha función hash é unha función matemática que converte un conxunto de datos noutro, normalmente nunha cadea de lonxitude fixa. O valor devolto pola función hash chámase resumo, checksum, suma hash ou hash. As funcións hash tamén son coñecidas como funcións resumo ou funcións digest. A seguinte imaxe exemplifica o funcionamento dunha función hash:



Funcionamento das funcións hash. Wikipedia. Licenza CC BY-SA 3.0.

Unha función hash funciona nunha soa dirección; e polo tanto, é imposible calcular os datos orixinais a partir do valor resumo. Ao aplicar unha función hash a un documento ou ficheiro o valor resumo é unha cadea que identifica inequivocamente o contido. Este feito fai que as funcións hash se usen para:

- Comprobación da integridade dun ficheiro: as funcións hash permiten saber se un ficheiro sufriu modificacións; xa que, un cambio no contido do ficheiro tradúcese nun valor resumo diferente.
- Identificación de ficheiros: como o hash depende do contido do arquivo é posible identificar arquivos con independencia do seu nome e/ou ubicación.
- Almacenamento de contrasinais: en vez de gardar o contrasinal dos usuarios, gárdase o hash do contrasinal. Cando un usuario quere autenticarse indicando o seu contrasinal, calcúlase o hash e compárase co valor do hash almacenado.

1.5.2 Certificados dixitais

Un certificado dixital é un documento electrónico, mediante o cal unha entidade (chamada de forma xenérica terceiro de confianza) garante o vínculo entra a identidade dun suxeito e unha clave pública (e polo tanto coa súa privada correspondente). Basicamente, é un documento que asocia unha identidade real cunha identidade dixital emitido por unha organización que se encargou de verificar esa asociación.

Un certificado dixital contén xeralmente información do tipo:

- Datos do usuario: nome, apelidos, enderezo de correo electrónico, organización á que pertence, posto ou cargo profesional, etc.
- Clave pública, lonxitude e algoritmo usado na súa xeración.
- Datos da entidade que emitiu a certificado.
- Validez.
- Uso/s para os que foi rexistrada/autorizada a clave.

A firma dixital permite asegurar a integridade do certificado e posibilitar as comprobacións de que non se realizaron cambios dende a súa emisión. Para asinar o certificado utilízase outra parella de claves xeradas con esta finalidade e asociadas á entidade certificadora.

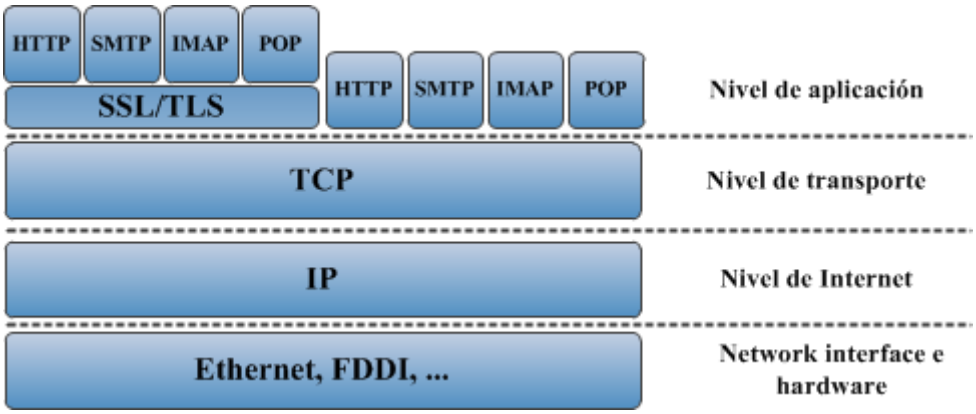
Para verificar a integridade dun certificado, calcúlase o hash e compróbase co valor da firma (previo descifrado coa clave pública da entidade).

1.6 Privacidade e autenticidade nas transmisións web

Desde o punto de vista de seguridade das comunicacións entre o servidor e o cliente, o protocolo HTTP apenas contempla os tres requirimentos necesarios para ser considerado seguro:

- Autenticación: O cliente non ten forma de comprobar que o servidor ao que se conecta é realmente ao que quería conectarse. O servidor só pode comprobar a identidade do cliente mediante usuario e clave, método débil (sobre todo se as comunicacións van en claro) e mediante IP/nome de máquina de orixe.
- Privacidade: O protocolo HTTP viaxa en claro pola rede polo que o contido e os datos de usuario e clave poden ser facilmente capturados por terceiras persoas que se atopen nalgún punto entre o servidor e o cliente.
- Integridade: Aínda que o protocolo HTTP usa o protocolo de transporte TCP, que garante que toda a información chega íntegra entre cliente e servidor, ningún dos dous protocolos evita que un terceiro poida interpoñerse entre cliente e servidor substituíndo as mensaxes dun e outro a vontade.

Aquí é onde entran en xogo SSL (Secure Sockets Layer) e TLS (Transport Layer Security). SSL/TLS son protocolos criptográficos que permiten ter comunicacións seguras. Tanto SSL coma TLS son protocolos de nivel de aplicación que a través do uso de certificados e criptografía híbrida permiten establecer canles de comunicación seguras polas que poden circular paquetes de datos doutros protocolos de nivel de aplicación. Esta característica é moi interesante; xa que permite que sobre canles seguros SSL/TLS viaxen datos de protocolos non seguros sen necesidade de modificalos, por exemplo, HTTP (HTTPS).



O módulo `mod_ssl` pon a disposición do servidor web Apache, usando a biblioteca OpenSSL, os protocolos SSL (v2 e v3) e TLS (considerado como SSLv3.1), que realizan o labor de autenticación (de clientes e servidores), cifrado das transmisións e integridade do seu contido:

- Autenticación: mediante certificados dixitais, que están asinados por unha autoridade certificadora (CA) recoñecida por ambos os extremos, que identifican inequivocamente ao seu posuidor.
- Privacidade: mediante o uso de algoritmos de cifrado, toda comunicación entre cliente e servidor é só lexible para estes.
- Integridade: mediante o uso de algoritmos de resumo ou hash impídese que esta sexa modificada de forma inadvertida

1.7 HTTPS en Apache

É posible configurar Apache para que sirva contidos seguros empregando HTTPS, para elo, é preciso instalar o módulo `mod_ssl`.

1.7.1 Xeración de clave e certificado

Para a xeración da clave e o certificado, empregamos as ferramentas ofrecidas pola libraría OpenSSL. Dita libraría está instalada en Ubuntu, para empregala en Windows, debemos empregar o executable `C:\Apache24\bin\openssl.exe`.

Xeración de clave

```
openssl genrea -out clave.key 2048
```

Que indica que se creará unha clave de 2048 bits.

Solicitude de certificado

```
openssl req -new -key clave.key -out solicitude.csr
```

Con este comando se xera unha solicitude de certificado a partir da clave privada indicada, cos datos que se nos irán solicitando (os que rematan en [] son opcionais). Esta solicitude é a que se debe enviar á autoridade de certificación.

Xeración de certificado autofirmado

Para facer probas, queremos firmar o noso propio certificado, para o que empregamos o comando:

```
openssl x509 -req -days 365 -in solicitude.csr -singkey clave.key -out certificado.crt
```

Co que se xerará un certificado válido por 365 días, a partires da solicitude `solicitude.csr` e firmado coa clave `clave.key`.

1.7.2 Configuración dun sitio virtual seguro

O proceso de creación dun sitio virtual seguro é o mesmo que para a creación dun sitio virtual, coa diferenza de que, no ficheiro de configuración do sitio virtual hai que incluír a maiores as liñas indicadas en cor vermella:

```
<IfModule mod_ssl.c>
 <VirtualHost *:443>
        ServerName sitio-seguro.com
        DocumentRoot /var/www/sitio-seguro

        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/certificado.crt
        SSLCertificateKeyFile /etc/ssl/private/clave.key
 </VirtualHost>
</IfModule>
```

Directivas

- `SSLEngine`. Cando se lle da valor `on`, indica que se habilita SSL para ese sitio.
- `SSLCertificateFile`. Indica a localización do certificado.
- `SSLCertificateKeyFile`. Indica a localización da clave privada.