Seguridad en entornos Apache

- Hardening Apache -

VI Foro de seguridad RedIRIS

Sergio Castillo Pérez

scastillo@deic.uab.es

Departamento de Ingeniería de la Información y de las Comunicaciones
Universidad Autónoma de Barcelona (UAB)

Índice

Índice

Un poco de historia...

Historia:

- Basado en el servidor httpd 1.3 de NCSA (*National Center for Supercomputing Applications*), escrito por Rob McCool.
- Primera versión publicada en abril de 1995.
- Actualmente tres ramas de desarrollo:
  - 1.3.x
  - 2.0.x
  - 2.2.x

Características de Apache

¿Por qué Apache es diferente a los demás?

Características de Apache

¿Por qué Apache es diferente a los demás?

- Extensible gracias a su arquitectura modular (`http://modules.apache.org/`):
  - `mod_auth`
  - `mod_ssl`
  - `mod_security`
  - `...`

Características de Apache

¿Por qué Apache es diferente a los demás?

- Extensible gracias a su arquitectura modular (`http://modules.apache.org/`):
  - `mod_auth`
  - `mod_ssl`
  - `mod_security`
  - ...
- Soporte de diversos lenguajes para la generación dinámica de páginas web:
  - PHP
  - Python
  - Perl
  - ...

Características de Apache

¿Por qué Apache es diferente a los demás?

- Extensible gracias a su arquitectura modular (`http://modules.apache.org/`):
  - `mod_auth`
  - `mod_ssl`
  - `mod_security`
  - ...
- Soporte de diversos lenguajes para la generación dinámica de páginas web:
  - PHP
  - Python
  - Perl
  - ...
- Altamente configurable.

Características de Apache

¿Por qué Apache es diferente a los demás?

- Extensible gracias a su arquitectura modular (`http://modules.apache.org/`):
  - `mod_auth`
  - `mod_ssl`
  - `mod_security`
  - ...
- Soporte de diversos lenguajes para la generación dinámica de páginas web:
  - PHP
  - Python
  - Perl
  - ...
- Altamente configurable.
- Multiplataforma: GNU/Linux, *BSD, Solaris, MS-Windows, ...

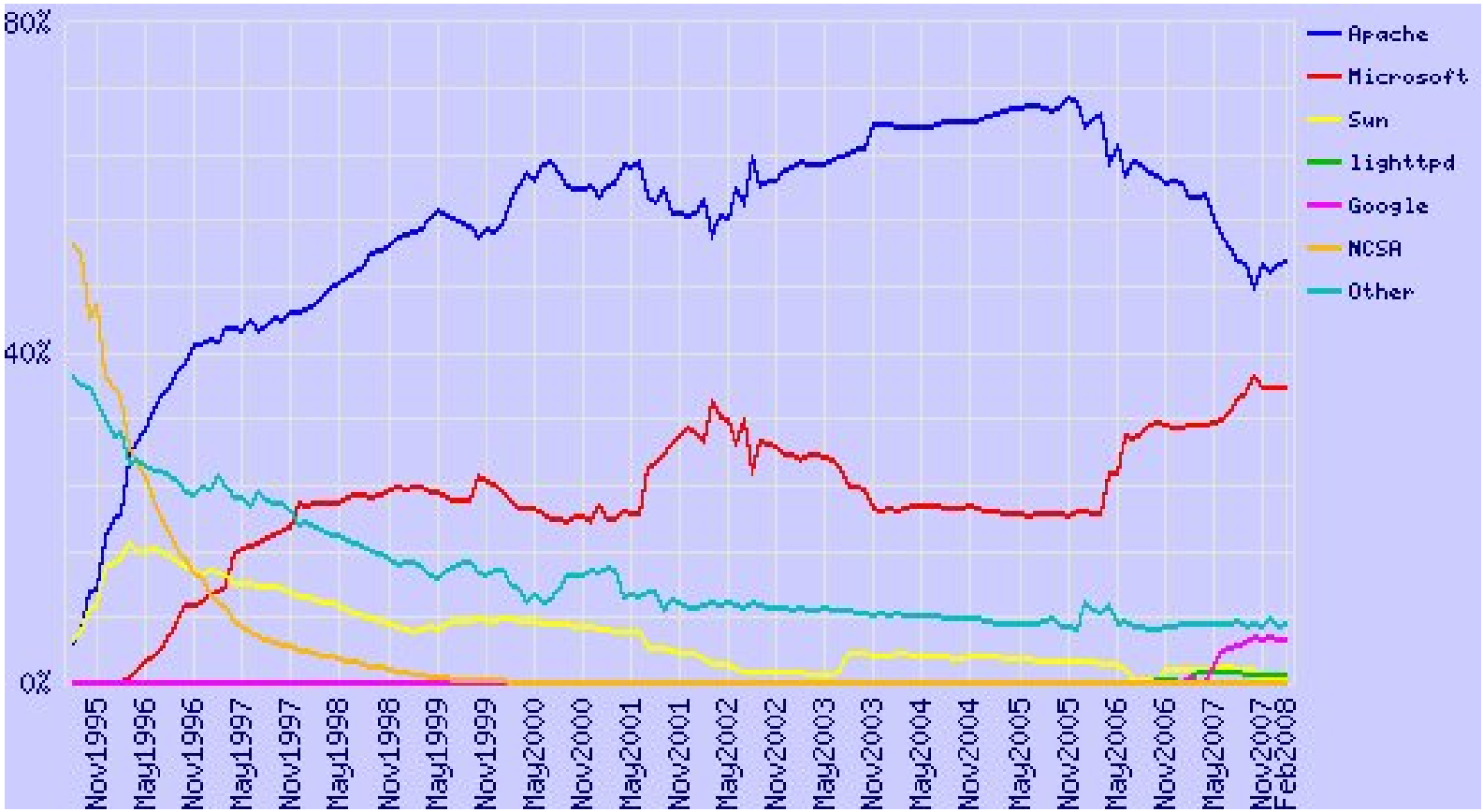
Características de Apache

¿Por qué Apache es diferente a los demás?

- Extensible gracias a su arquitectura modular (`http://modules.apache.org/`):
  - `mod_auth`
  - `mod_ssl`
  - `mod_security`
  - ...
- Soporte de diversos lenguajes para la generación dinámica de páginas web:
  - PHP
  - Python
  - Perl
  - ...
- Altamente configurable.
- Multiplataforma: GNU/Linux, *BSD, Solaris, MS-Windows, ...
- ¡Open Source!

Cuota de mercado

Market Share for Top Servers Across All Domains August 1995 - February 2008*



* Fuente: `http://news.netcraft.com/archives/web_server_survey.html`

Consideración:

- Un porcentaje elevado de administradores no modifica la configuración por defecto.
- Sin embargo, mejorar la seguridad no se centra exclusivamente en crear una buena configuración. Otros aspectos: sistema operativo y red.
- Seguridad en el sistema operativo:
  - *Keep up to date*
  - Mecanismos contra *buffers overflow*: ASLR, NX, Canary, ...
  - Integridad del sistema de ficheros: Tripwire, Aide, ...
  - HIDS
  - ...
- Seguridad en la red:
  - Infraestructura: DMZ
  - NIDS
  - Cortafuegos
  - ...

Índice

Proceso básico de configuración

Medidas básicas:

- Compilación vs. paquetes binarios.
- *Keep up to date*.
- Ejecución de Apache bajo un usuario/grupo específico no privilegiado:
  - `User apache`
  - `Group apache`
- Modificar los permisos de los archivos relacionados con Apache.
- Integrar Apache en entornos jail/chroot (`mod_chroot`).

Proceso básico de configuración

Medidas básicas:

- Deshabilitar módulos innecesarios:
  - `mod_userdir`
  - `mod_info`
  - `mod_status`
  - `mod_include`
  - `mod_autoindex`
  - `mod_{perl,python,php,...}`
- Algunos módulos pueden mejorar la seguridad:
  - `mod_rewrite`
  - `mod_headers`
  - `mod_setenvif`
  - `mod_security`
  - `mod_auth`
  - `mod_ssl`

Proceso básico de configuración

Directivas asociadas a permisos sobre el árbol de directorios:

- `Order, Allow, Deny, Options y AllowOverride`:

```
<Directory />
    Order Deny,Allow
    Deny from all
    Options None
    AllowOverride None
</Directory>
<Directory /var/www/htdocs>
    Order Allow,Deny
    Allow from all
    Options None
    AllowOverride None
</Directory>
```

Proceso básico de configuración

Directivas asociadas a las conexiones:

- `TimeOut, KeepAlive, MaxKeepAliveRequests, KeepAliveTimeout:`

```
# Tiempo de espera de 30 segundos para clientes ``lentos''
TimeOut 30
# Permitir reusar las conexiones para peticiones consecutivas
KeepAlive On
# Permitir hasta 100 peticiones para la misma conexión
MaxKeepAliveRequests 100
# Esperar hasta 10 segundos a la siguiente petición antes
# de cerrar la conexión
KeepAliveTimeout 10
```

Proceso básico de configuración

Directivas asociadas al comportamiento de las instancias:

- Apache 1.x (*prefork*):

```
# Lanzar 5 instancias al iniciarse Apache
StartServers 5
# Mantener 5 servidores listos para gestionar peticiones
MinSpareServers 5
# No mantener más de 10 servidores sin hacer nada
MaxSpareServers 10
# Permitir hasta 150 clientes en total
MaxClients 150
# Permitir hasta 30 peticiones por instancia
MaxRequestsPerChild 30
```

- Apache 2.x: Incorpora el concepto de *multiprocessing modules*:
  - Cada módulo determina una forma distinta de cómo procesar las peticiones.
  - Sólo un módulo activado.
  - Objetivo: optimización para cada sistema operativo.
  - Diversos modelos: *prefork*, *worker*, ...

Índice

Control de acceso

¿Qué mecanismos de control de acceso proporciona Apache?

- Orientado a usuario:
  - Archivos de texto plano (`mod_auth`)
  - Archivos DBM (`mod_auth_dbm`)
  - *Digest* (`mod_auth_digest`)
  - Certificados digitales (`mod_ssl`) → autenticación mutua
- Mediante IP de origen (`mod_access`)
- Según contenido de peticiones (`mod_setenvif` + `mod_access`)
- Uso de lenguajes: PHP, perl, python, . . . Diseño de mecanismos “a medida”.

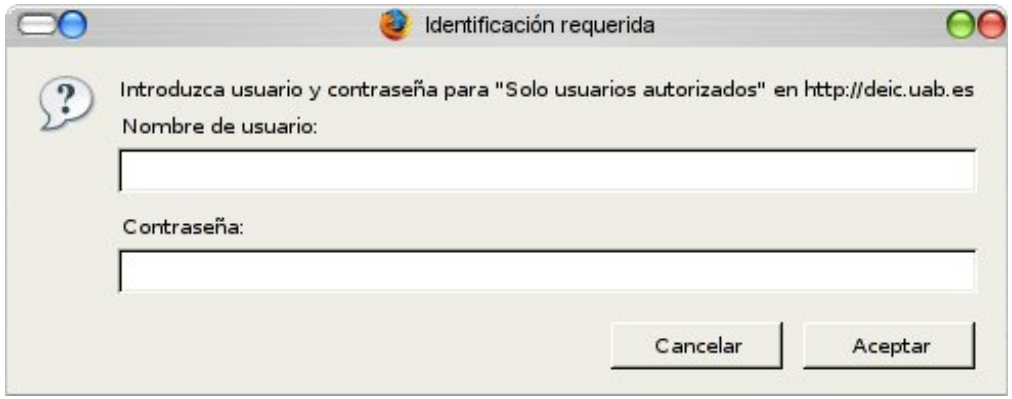
Control de acceso

Ejemplo de control de acceso basado `mod_auth`:

```
user@host ~/public_html $ cat .htaccess
AuthName "Solo usuarios autorizados"
AuthType Basic
AuthUserFile /export/home/user/public_html/.htpasswd
Require valid-user
user@host ~/public_html $ htpasswd -c .htpasswd sergio
New password: ********
Re-type new password: ********
Adding password for user sergio
user@host ~/public_html $ cat .htpasswd
sergio:dZE5vdiBsOjLo
```

Control de acceso

Ejemplo de control de acceso basado en `mod_auth`:



Control de acceso

Ejemplos de control de acceso por IP y por contenido:

```
<Directory /var/www/localhost/htdocs/>
    Order Deny,Allow
    Allow from 127.0.0.0/8 192.168.0.0/24
    Deny from all
</Directory>
```

```
SetEnvIf Request_URI "\.gif$" nogif=y
<Directory />
    Order Deny,Allow
    Deny from env=nogif
</Directory>
```

Control de acceso

¿Es seguro el C.A. basado en login/password?

- Respuesta: no necesariamente, depende de la calidad de las contraseñas.
- La autenticación usando, p.e., `mod_auth`, `mod_auth_dbm` o `mod_auth_digest` puede ser atacada mediante fuerza bruta/diccionarios:

```
user@host ~ $ hydra 192.168.1.1 http-get -L user-dic.txt -P pass-dic.txt -m /
Hydra v5.4 (c) 2006 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2008-03-02 21:44:28
[DATA] 16 tasks, 1 servers, 3020 login tries (l:10/p:302), ~188 tries per task
[DATA] attacking service http-get on port 80
[80][www] host: 192.168.1.1    login: admin   password: crack
[STATUS] attack finished for 192.168.1.1 (waiting for childs to finish)
Hydra (http://www.thc.org) finished at 2008-03-02 21:44:28
```

- Solución:
  - Podemos “mitigar” estos ataques mediante `mod_evasive`.
  - Emplear preferiblemente métodos más robustos (i.e.: certificados digitales).

Índice

Identificación remota y *HTTP Fingerprinting*

¿Qué es el *HTTP Fingerprinting*?

- *HTTP Fingerprinting*: proceso remoto de identificación de servidores web mediante las diferencias existentes en las implementaciones del protocolo HTTP.
- Estrategias de identificación:
  - Cadenas identificativas.
  - Códigos de error (404, 403, ...).
  - Orden en las cabeceras de respuesta.
  - Peticiones HTTP malformadas.
  - ...
- Herramientas:
  - Netcat (`http://nc110.sf.net/`)
  - Nmap (`http://nmap.org/`)
  - Httprint (`http://www.net-square.com/httprint/`)
  - ...

Identificación remota y *HTTP Fingerprinting* - Ejemplos

```
user@host ~ $ nc www.linux-server.com 80
HEAD / HTTP1.0

HTTP/1.1 200 OK
Date: Mon, 24 Mar 2008 17:33:24 GMT
Server: Apache/2.2.8 (Debian) DAV/2 SVN/1.4.6
PHP/5.2.5-3 with Suhosin-Patch mod_ssl/2.2.8 OpenSSL/0.9.8g
Last-Modified: Mon, 18 Feb 2008 19:07:32 GMT
ETag: "462ab-485-4467377cdc500"
Accept-Ranges: bytes
[...]
```

```
user@host ~ $ nc www.win-server.com 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Mon, 24 Mar 2008 04:08:48 GMT
X-Powered-By: ASP.NET
Connection: Keep-Alive
Content-Length: 21210
[...]
```

Identificación remota y *HTTP Fingerprinting* - Ejemplos

```
user@host ~ $ nc www.linux-server.com 80
GET / HTTP/7.0

HTTP/1.1 400 Bad Request
Date: Mon, 24 Mar 2008 19:25:03 GMT
Server: Apache/2.2.8 (Debian) DAV/2 SVN/1.4.6
PHP/5.2.5-3 with Suhosin-Patch mod_ssl/2.2.8 OpenSSL/0.9.8g
Content-Length: 388
Connection: close
Content-Type: text/html; charset=iso-8859-1
[...]
```

```
user@host ~ $ nc www.win-server.com 80
GET / HTTP/7.0

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Mon, 24 Mar 2008 19:25:10 GMT
X-Powered-By: ASP.NET
Connection: Keep-Alive
Content-Length: 21210
[...]
```

Identificación remota y *HTTP Fingerprinting* - Ejemplos

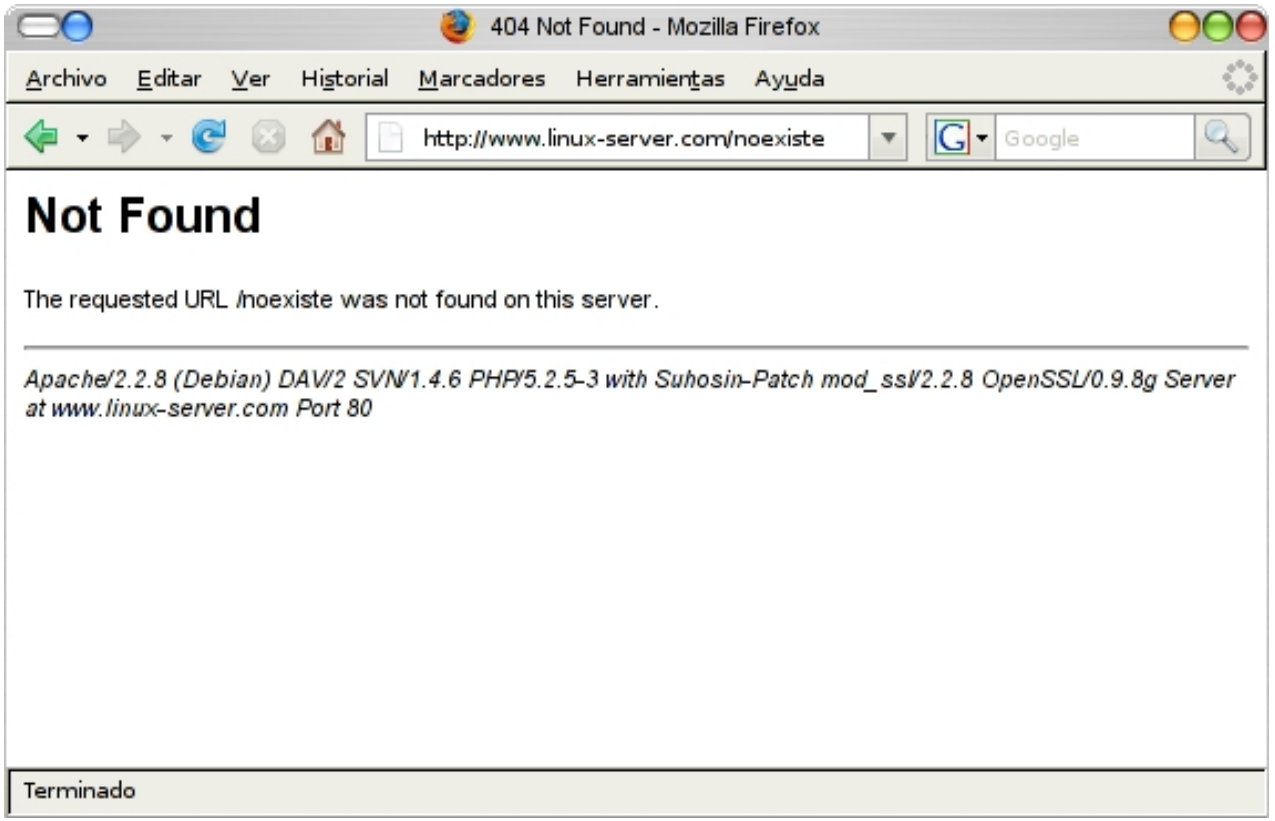
```
user@host ~ $ nmap -A -p80 www.sun-server.com
Starting Nmap 4.53 ( http://insecure.org ) at 2008-03-24 04:44 CET
Interesting ports on www.sun-server.com (192.168.1.1):
PORT   STATE SERVICE VERSION
80/tcp open  http    SunONE WebServer 6.1

Service detection performed. Please report any incorrect results at
http://insecure.org/nmap/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.592 seconds
```

```
user@host ~ $ nmap -A -p80 www.lighttpd-server.com
Starting Nmap 4.53 ( http://insecure.org ) at 2008-03-24 04:50 CET
Interesting ports on www.lighttpd-server.com (192.168.1.2):
PORT   STATE SERVICE VERSION
80/tcp open  http    lighttpd 1.5.0

Service detection performed. Please report any incorrect results at
http://insecure.org/nmap/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.086 seconds
```

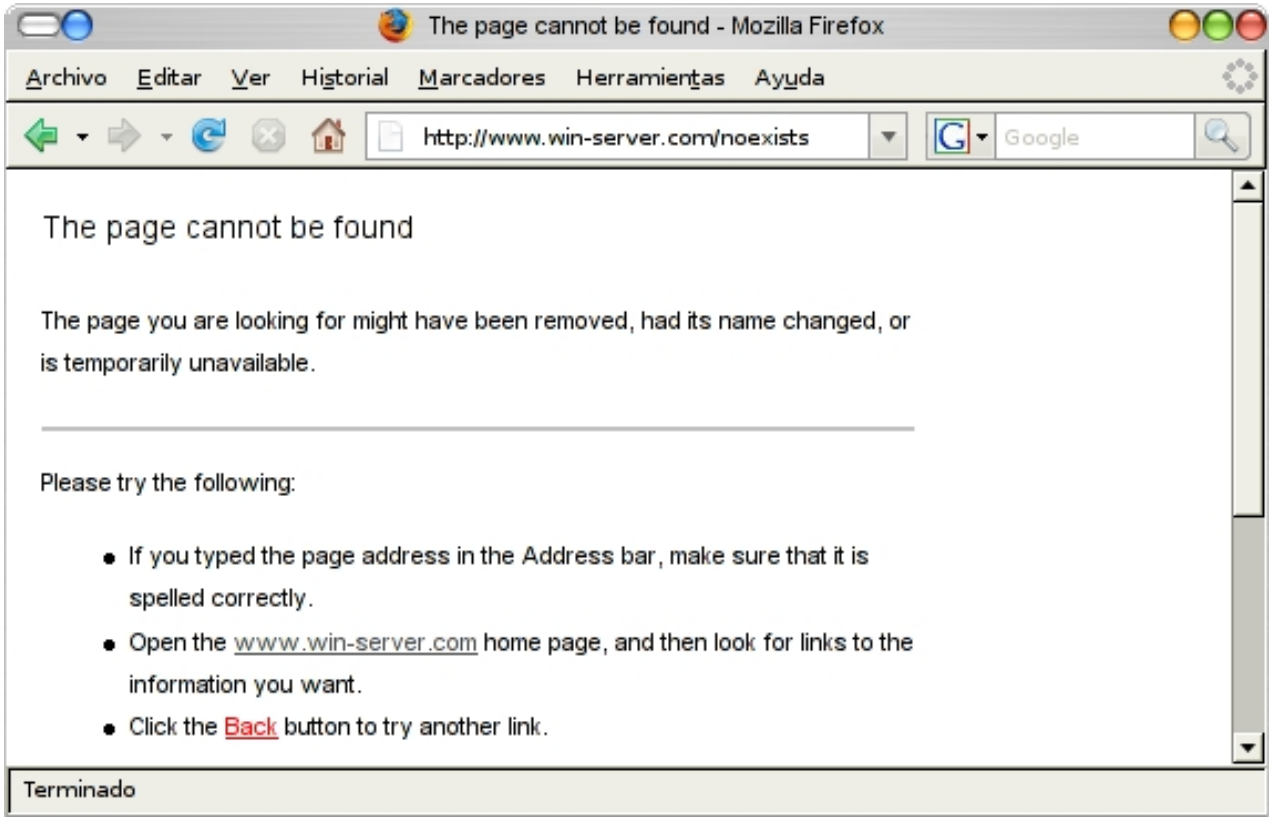
Identificación remota y *HTTP Fingerprinting* - Ejemplos

```
user@host ~/httprint $ ./httprint -h www.netscape-server.com -P0 -s signatures.txt
httprint v0.301 (beta) - web server fingerprinting tool
(c) 2003-2005 net-square solutions pvt. ltd. - see readme.txt
http://net-square.com/httprint/
httprint@net-square.com

Finger Printing on http://192.168.1.3:80/
Finger Printing Completed on http://192.168.1.3:80/
-------------------------------------------------
Host: 192.168.1.3
Derived Signature:
Sun-ONE-Web-Server/6.1
9E431BC86ED3C295811C9DC5811C9DC5050C5D3294DF1BD04276E4BB811C9DC5
7FC8D095AF7A648F2A200B4C4D0ACB9C811C9DC5811C9DC5811C9DC52655F350
FCCC535B811C9DC5FCCC535B811C9DC5E2CE69272576B7696ED3C2959E431BC8
6ED3C295E2CE6922811C9DC5811C9DC5811C9DC56ED3C2956ED3C295E2CE6923
E2CE6923FCCC535F811C9DC5E2CE6927E2CE6920

Banner Reported: Sun-ONE-Web-Server/6.1
Banner Deduced: Netscape-Enterprise/6.0
Score: 119
Confidence: 71.69
------------------------
Scores:
Netscape-Enterprise/6.0: 119 71.69
Microsoft-IIS/6.0: 82 19.47
[...]
```

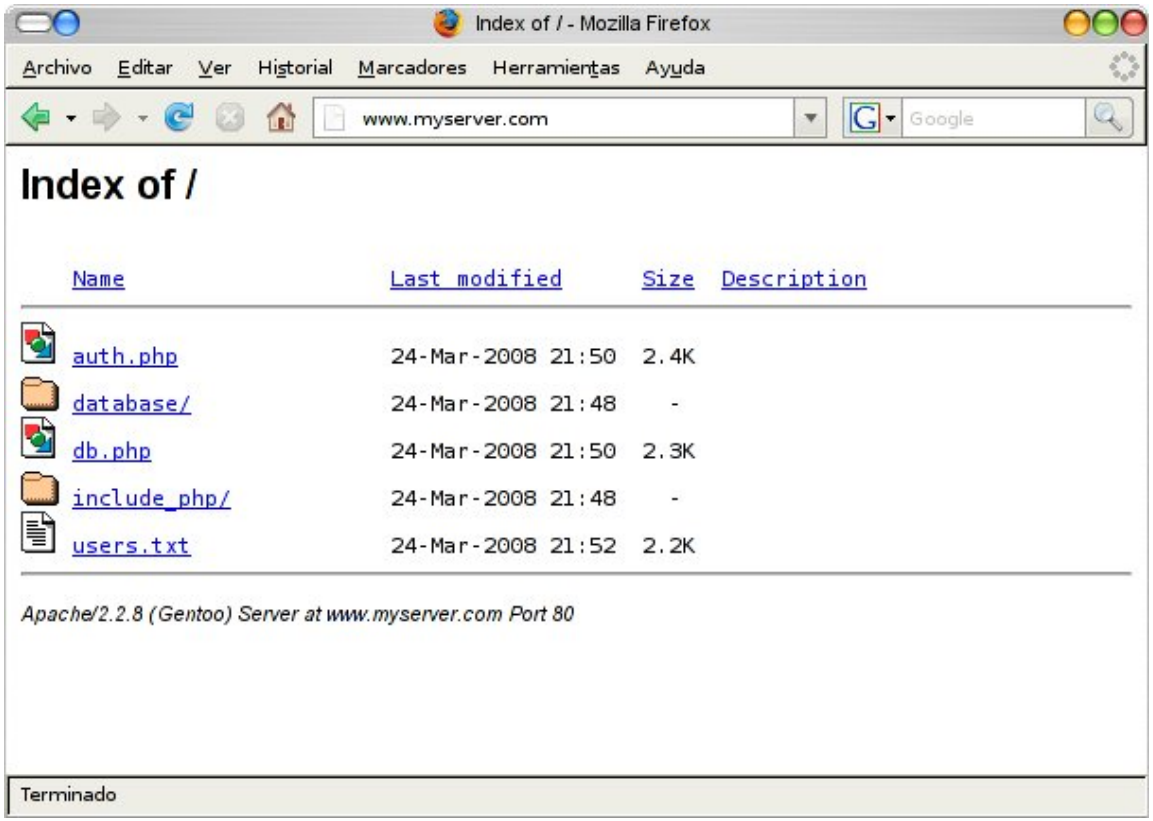
Identificación remota y *HTTP Fingerprinting* - Ejemplos



Identificación remota y *HTTP Fingerprinting* - Ejemplos



Identificación remota y *HTTP Fingerprinting* - Ejemplos



Identificación remota y *HTTP Fingerprinting*

¿Cómo podemos minimizar la identificación remota y el *HTTP Fingerprinting*?

- Directivas `ServerTokens` y `ServerSignature`
- Modificación del código fuente
- `mod_headers`
- `mod_security`
- `mod_setenvif`
- Configuración de respuestas a errores

Identificación remota y *HTTP Fingerprinting*

Directivas `ServerTokens` y `ServerSignature`

- `ServerTokens Prod`: minimiza la información de la cabecera Server a "Apache", eliminado la información de la versión del sistema operativo y los módulos compilados.
- `ServerSignature Off`: elimina la línea de información del servidor añadida en páginas generadas (i.e.: indexes, `mod_status`, `mod_info`, etc...).

Identificación remota y *HTTP Fingerprinting*

Modificación del código fuente (opción 1):

- Cambiar la definición de `AP_SERVER_BASEPRODUCT`:
  - En `httpd.h` para versiones 1.x.
  - En `ap_release.h` para versiones 2.x.

```
1  #define AP_SERVER_BASEVENDOR  "Apache Software Foundation"
2  #define AP_SERVER_BASEPRODUCT "redIRIS Web Server"
3
4  #define AP_SERVER_MAJORVERSION_NUMBER 2
5  #define AP_SERVER_MINORVERSION_NUMBER 0
6  #define AP_SERVER_PATCHLEVEL_NUMBER   63
```

Modificación del código fuente (opción 2):

- Modificar la función `ap_set_version()`:
  - En `http_main.c` para versiones 1.x.

```
1  static void ap_set_version(void)
2  {
3      ap_add_version_component("redIRIS Web Server 1.0");
4      version_locked++;
5  }
```

  - En `core.c` para versiones 2.x.

```
1  static void ap_set_version(apr_pool_t *pconf)
2  {
3      ap_add_version_component(pconf, "redIRIS Web Server 1.0");
4      version_locked++;
5  }
```

Identificación remota y *HTTP Fingerprinting*

Módulo `mod_headers`:

- El módulo `mod_headers` permite alterar las cabeceras enviadas por el servidor.
- En nuestro caso podemos modificar la cabecera *Server* de la siguiente manera:

```
LoadModule headers_module modules/mod_headers.so
Header set Server "redIRIS Web Server"
```

Módulo `mod_security`:

- El módulo `mod_security` también permite modificar las cabeceras enviadas por el servidor.

```
LoadModule sercurity_module modules/mod_security2.so
ServerTokens Full
SecServerSignature "Microsoft-IIS/5.0"
```

Identificación remota y *HTTP Fingerprinting*

Módulo `mod_setenvif`:

- Herramientas sofisticadas de http fingerprinting como *httprint* no son fáciles de "engañar".
- El módulo `mod_setenvif` permite instanciar variables de entorno según concordancia entre expresiones regulares y campos de las peticiones.
- Sintaxis:

```
SetEnvIf Attribute Match_expression Variable=value
```

- Ejemplo:

```
SetEnvIf Request_URI "^/cmd.exe$" log=n
```

Identificación remota y *HTTP Fingerprinting*

Módulo `mod_setenvif`:

- Forma de evadir *httprint*:

```
SetEnvIf Request_Method . BR_http=y
SetEnvIf Request_Method . BR_get=y
SetEnvIf Request_Protocol HTTP\/1\.0$ !BR_http
SetEnvIf Request_Protocol HTTP\/1\.1$ !BR_http
SetEnvIf Request_Method GET !BR_get
SetEnvIf BR_http y BadRequest=y
SetEnvIf BR_get y BadRequest=y

<Directory />
Order Deny,Allow
Deny from env=BadRequest
</Directory>
```

Identificación remota y *HTTP Fingerprinting*

Configuración de respuestas a errores:

- La directiva `ErrorDocument` determina las respuestas ante los errores.
- Tres tipos de respuestas:
  - Texto plano
  - Redirecciones locales
  - Redirecciones externas
- Ejemplos:
  - `ErrorDocument 500 'Mensaje Error'`
  - `ErrorDocument 402 /error402.html`
  - `ErrorDocument 403 http://www.another-server.com`

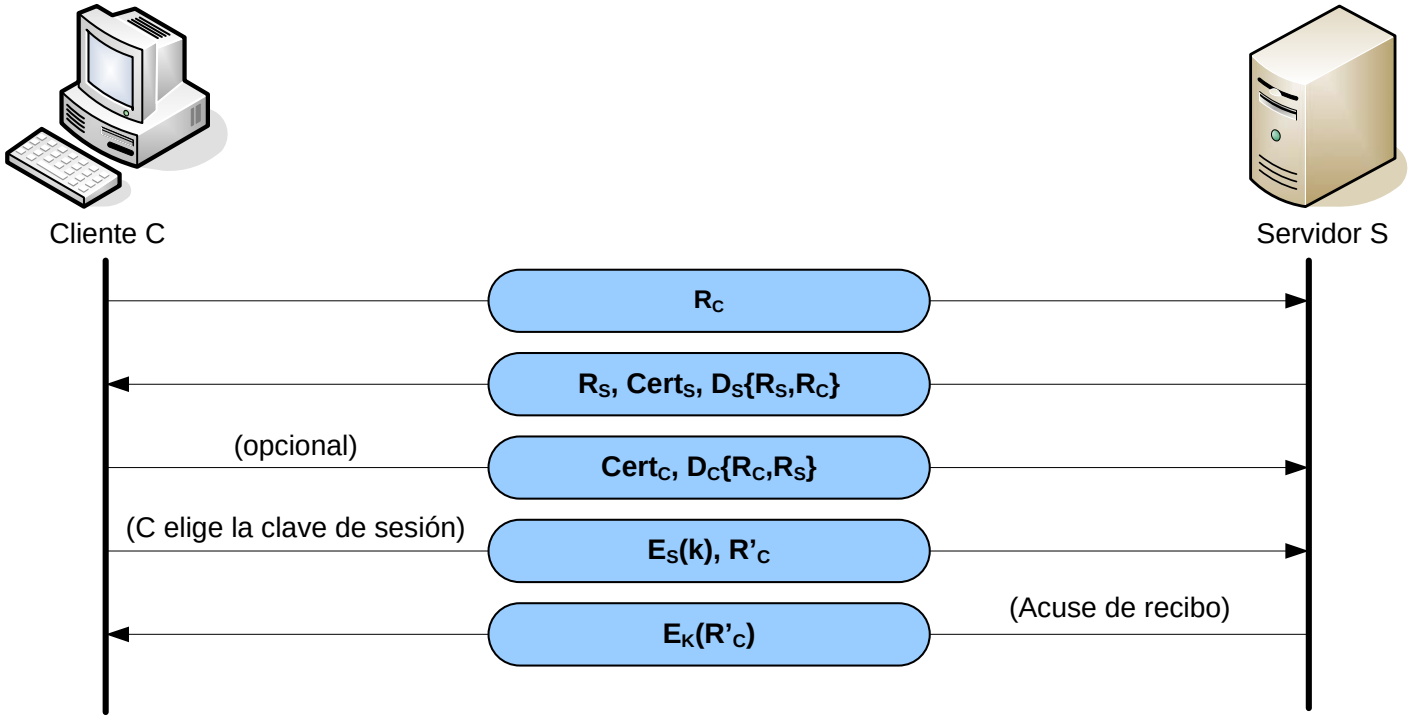
Índice

Características de SSL/TLS

Características de SSL/TLS:

- TLS (*Transport Layer Security*) y su predecesor SSL (*Secure Sockets Layer)* son protocolos criptográficos para el intercambio de datos de forma segura.
- Autenticación basada en criptografía de clave pública:
  - Autenticación mutua
  - Autenticación simple (servidor ante el cliente)
  - Anonimato total
- Cifrado de datos mediante un criptosistema simétrico.
- Integridad de los datos a través de un MAC.
- Implementado entre la capa de aplicación y TCP/IP. Independencia con los protocolos de la capa de aplicación:
  - **HTTP** $\longrightarrow$ **HTTPS**
  - FTP $\longrightarrow$ SFTP
  - POP3 $\longrightarrow$ POP3S
  - ...

SSL/TLS - Protocolo simplificado de handshake



SSL/TLS - Configuración de Apache

Configuración de Apache para soportar SSL/TLS:

```
# Generación de claves:
server ~ # $ openssl genrsa -des3 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
.........++++++
.....................++++++
e is 65537 (0x10001)
Enter pass phrase for server.key: ********
Verifying - Enter pass phrase for server.key: ********

Generating RSA private key, 1024 bit long modulus
.....................................................+++
..............................+++
e is 65537 (0x10001)
```

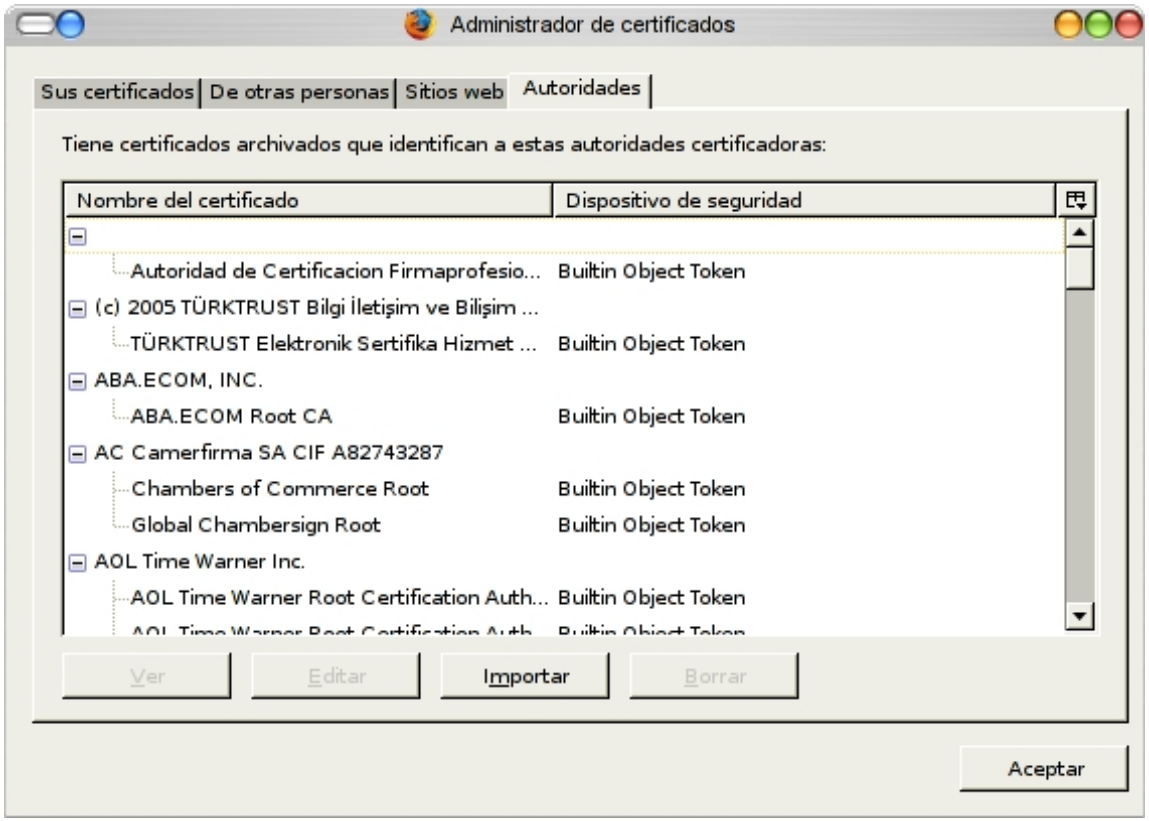
SSL/TLS - Configuración de Apache

```
# Generación del Certificate Signing Request:
server ~ # $ openssl req -new -key server.key -out server.csr
Enter pass phrase for server.key:********
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Barcelona
Locality Name (eg, city) []:Cerdanyola del Valles
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UAB
Organizational Unit Name (eg, section) []:dEIC
Common Name (eg, YOUR name) []:www.myserver.com
Email Address []:admin@myserver.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

# Envío de la petición a una autoridad de certificación:
server ~ # $ mail -s "Petición firma certificado" admin@ac.com < server.csr
```

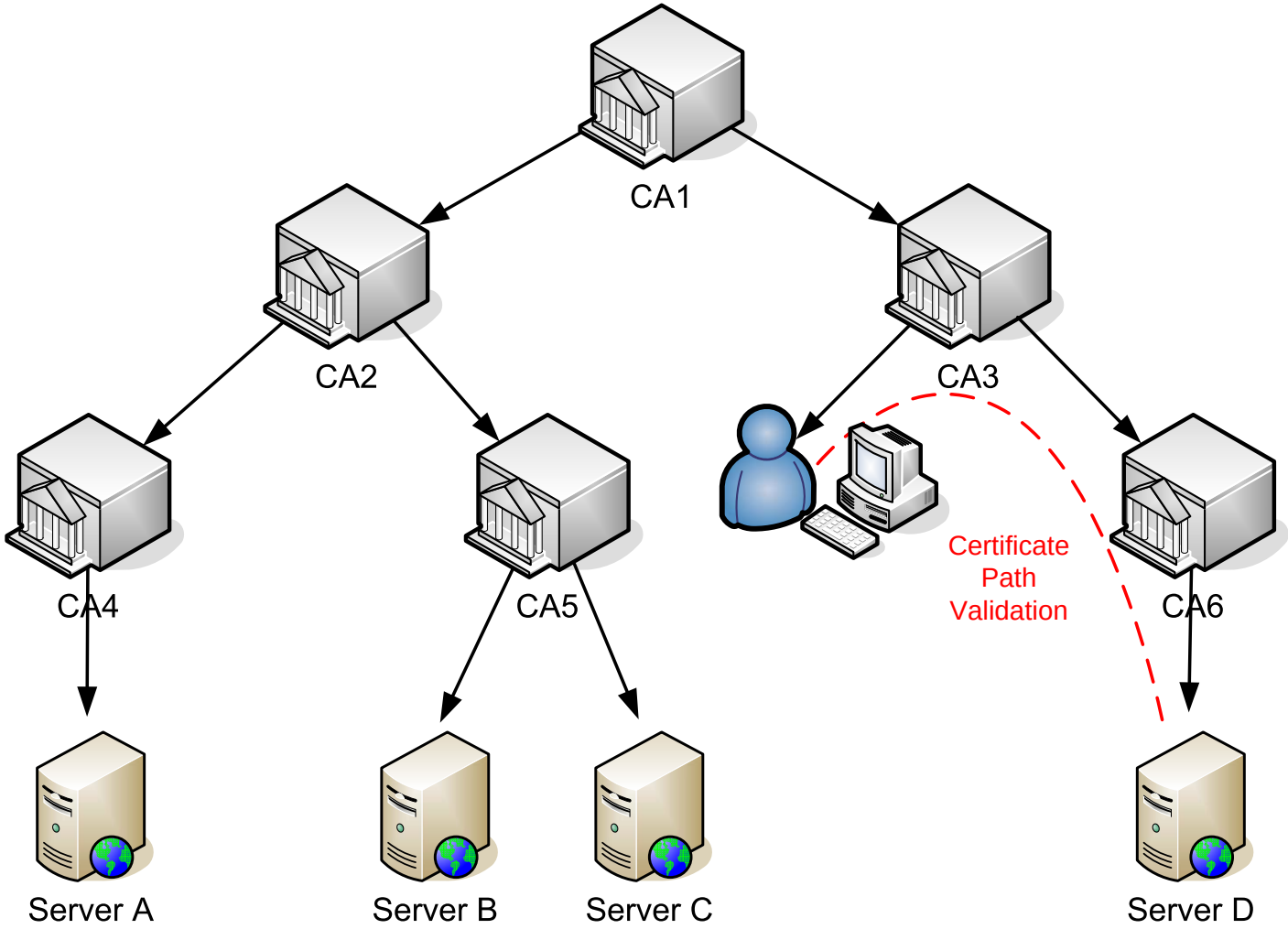
SSL/TLS - Configuración de Apache

```
# Una vez recibido el server.crt firmado...
server ~ # chmod 400 server.crt server.key
server ~ # cp server.crt server.key /etc/apache2/ssl/

# Asegurar que tenemos las directivas en la configuración:
LoadModule ssl_module modules/mod_ssl.so
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key

# Iniciar Apache:
server ~ # /etc/init.d/apache2 start
 * Service apache2 starting
Apache/2.2.8 mod_ssl/2.2.8 (Pass Phrase Dialog)
Some of your private key files are encrypted for security reasons.
In order to read them you have to provide the pass phrases.

Server localhost:443 (RSA)
Enter pass phrase:********

OK: Pass Phrase Dialog successful.
 * Service apache2 started
```

SSL/TLS - Configuración del Cliente



SSL/TLS - Configuración del Cliente



SSL/TLS - ¿Es seguro?

¿Es SSL/TLS un mecanismo 100% seguro?

SSL/TLS - ¿Es seguro?

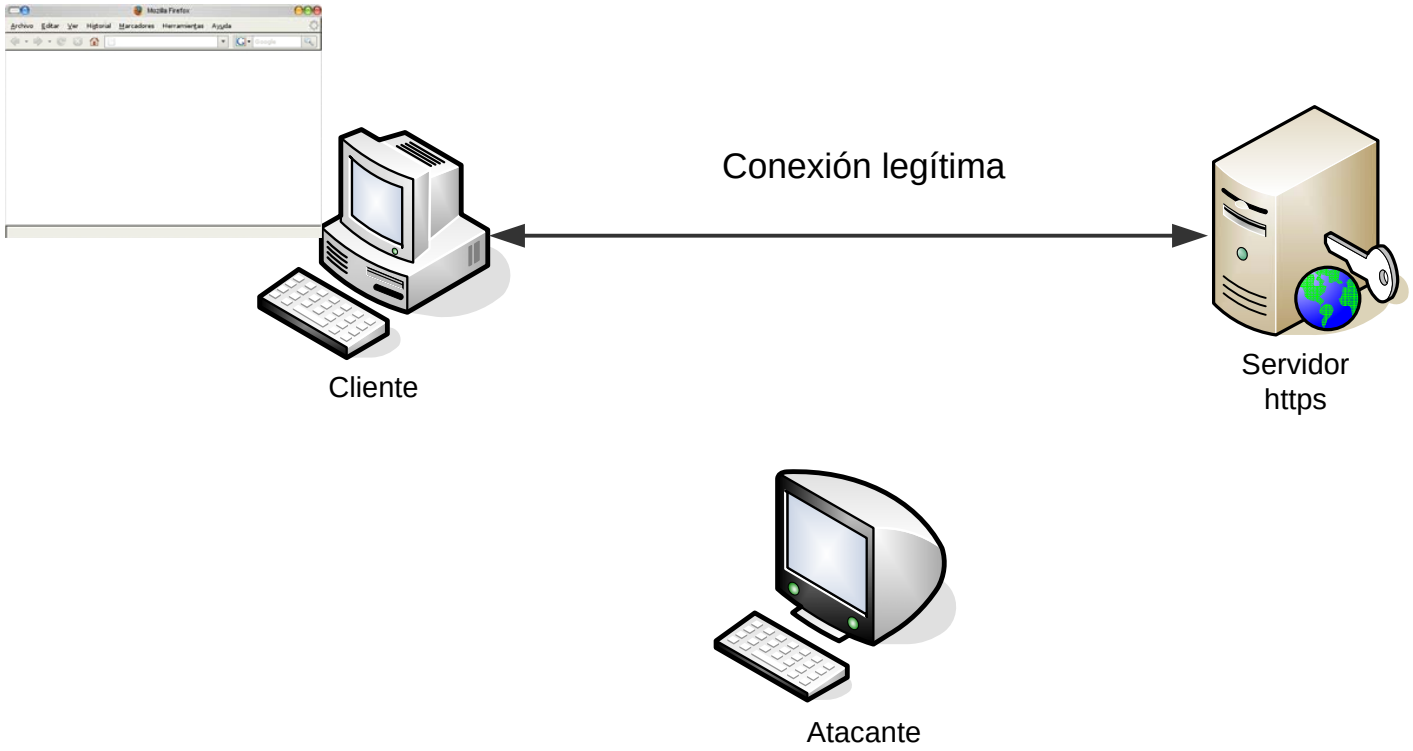
NO

SSL/TLS - ¿Es seguro?

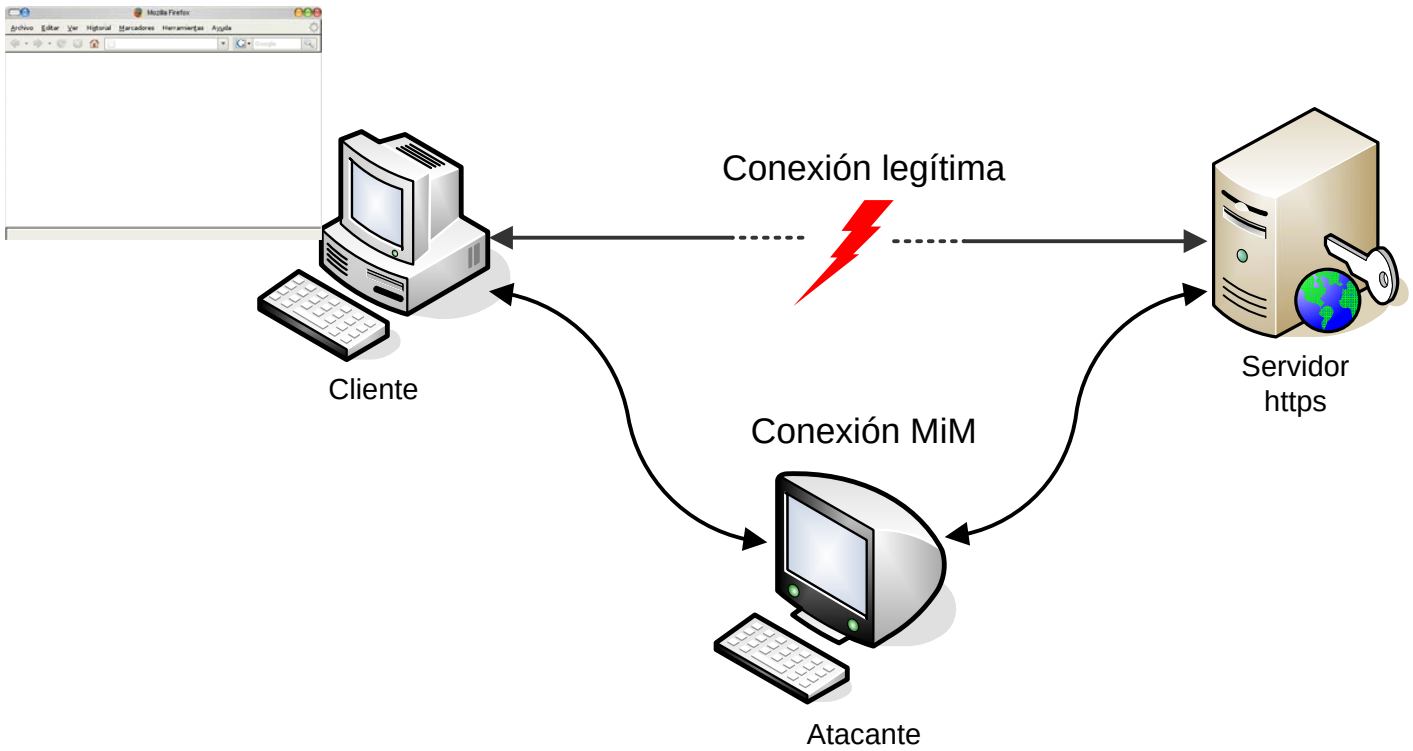
¿Por qué SSL/TLS no es 100% seguro?

- SSL/TLS es susceptible de ataques *Man-In-the-Middle*:
  - *ARP Poisoning*
  - *DNS Spoofing*
  - *ICMP Redirect*
  - *DHCP Spoofing*
  - *Port Stealing*
- ¡Es “especialmente” inseguro si usamos certificados auto-firmados!

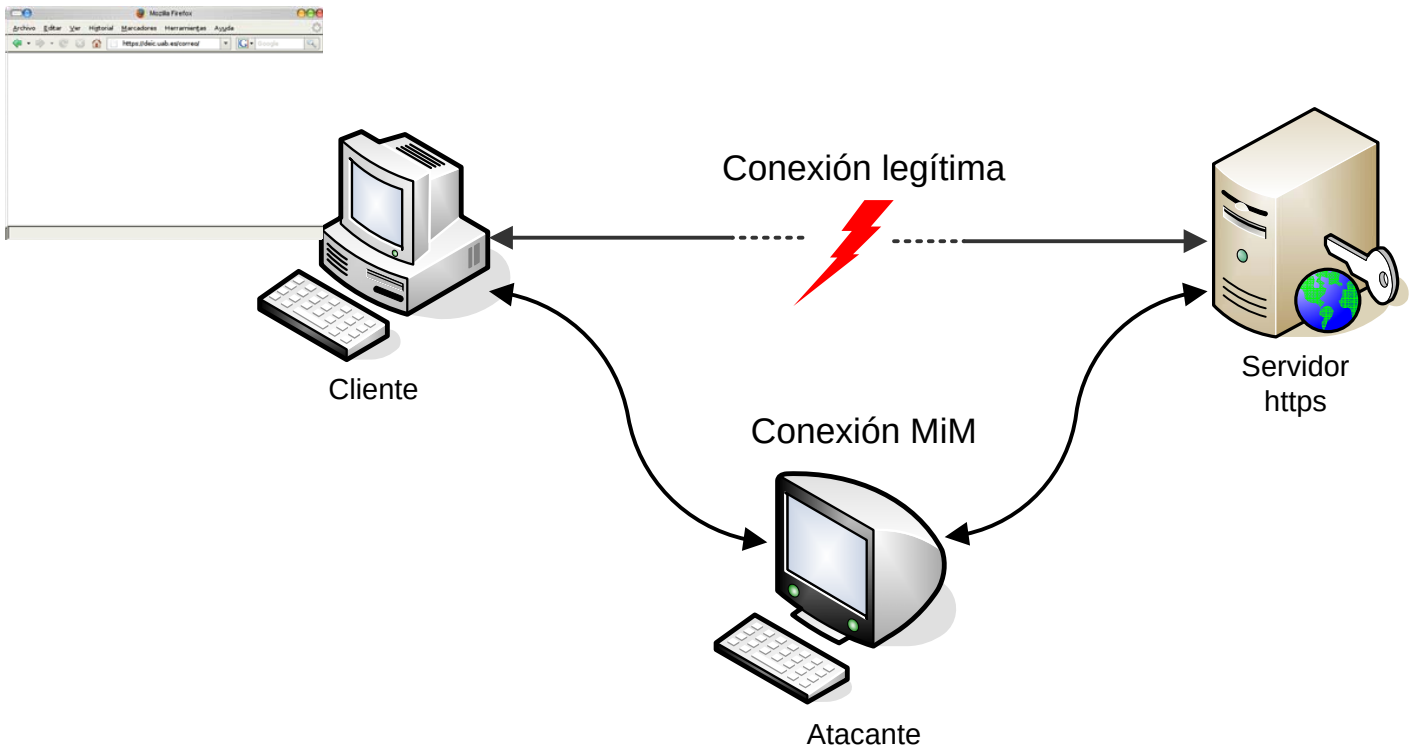
SSL/TLS - Ataques MiM



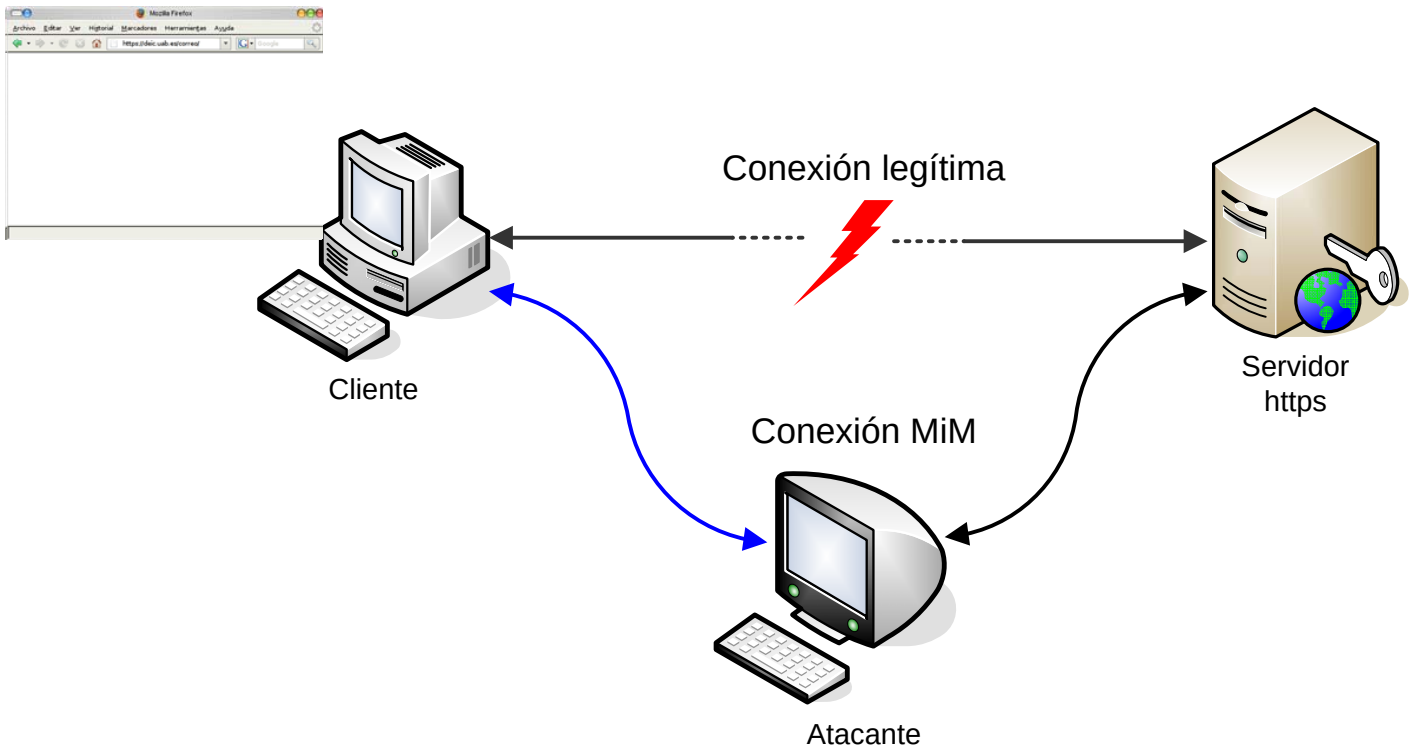
SSL/TLS - Ataques MiM



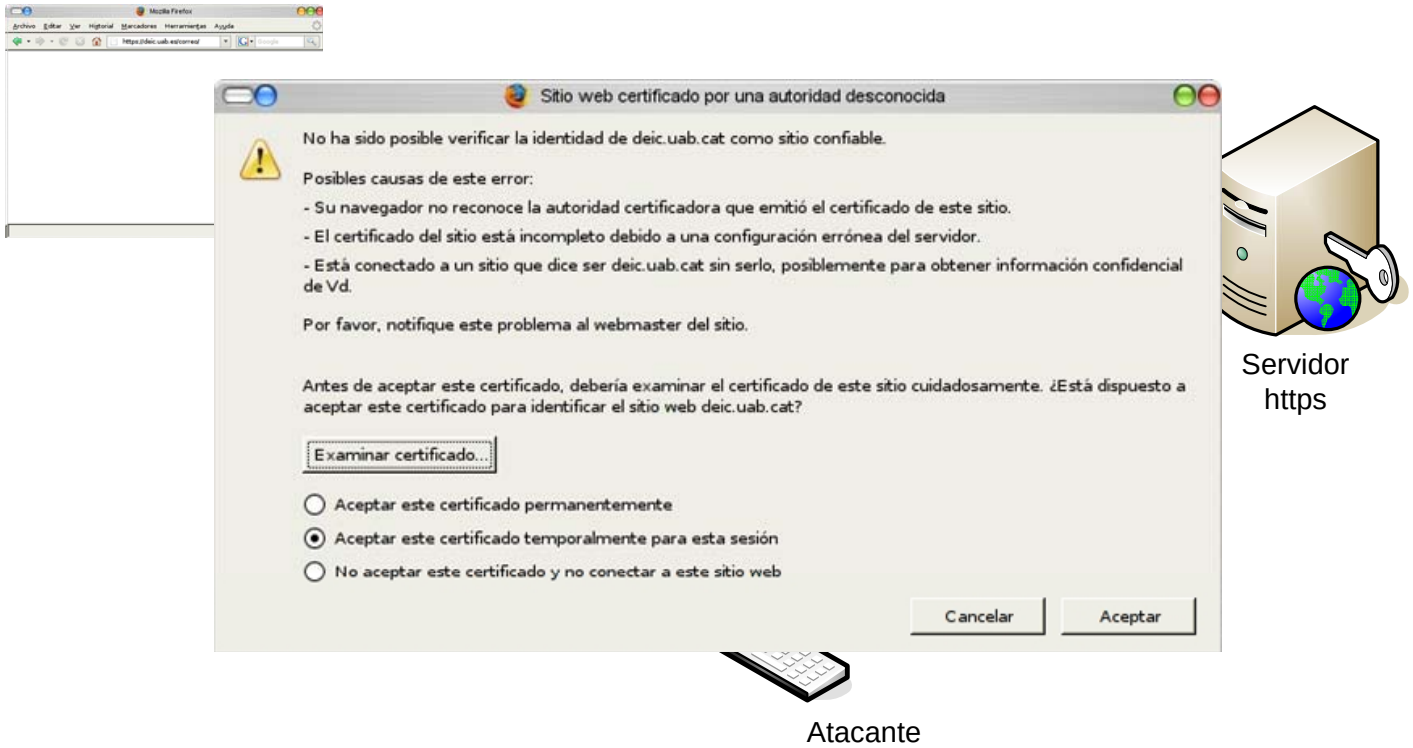
SSL/TLS - Ataques MiM



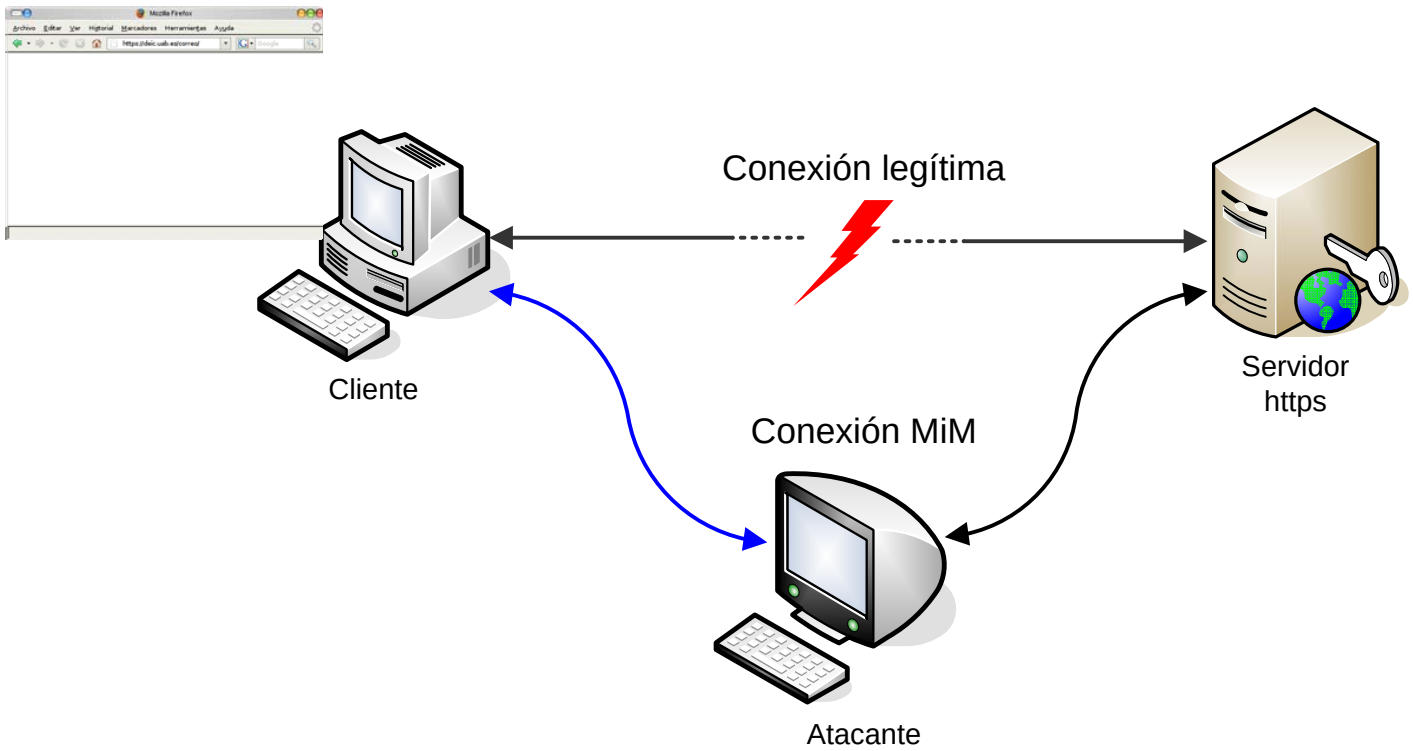
SSL/TLS - Ataques MiM



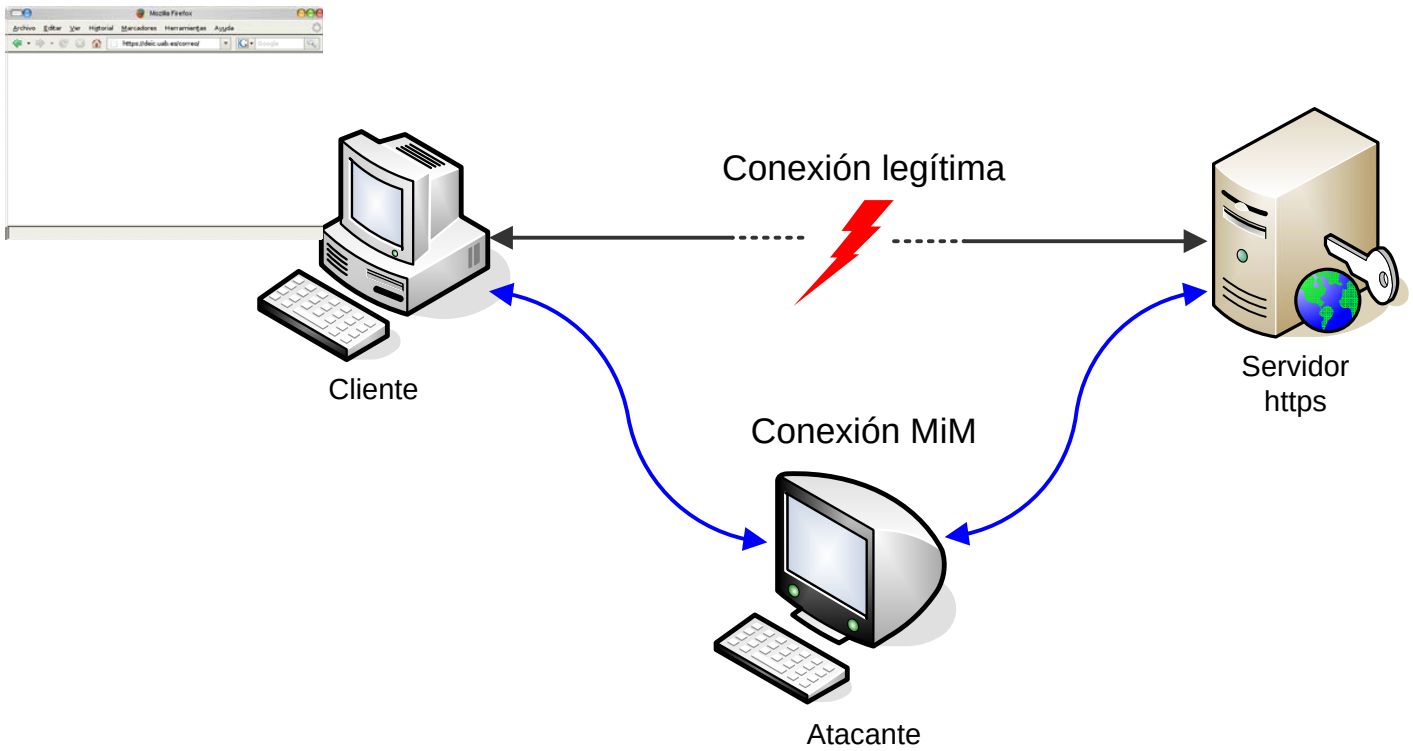
SSL/TLS - Ataques MiM



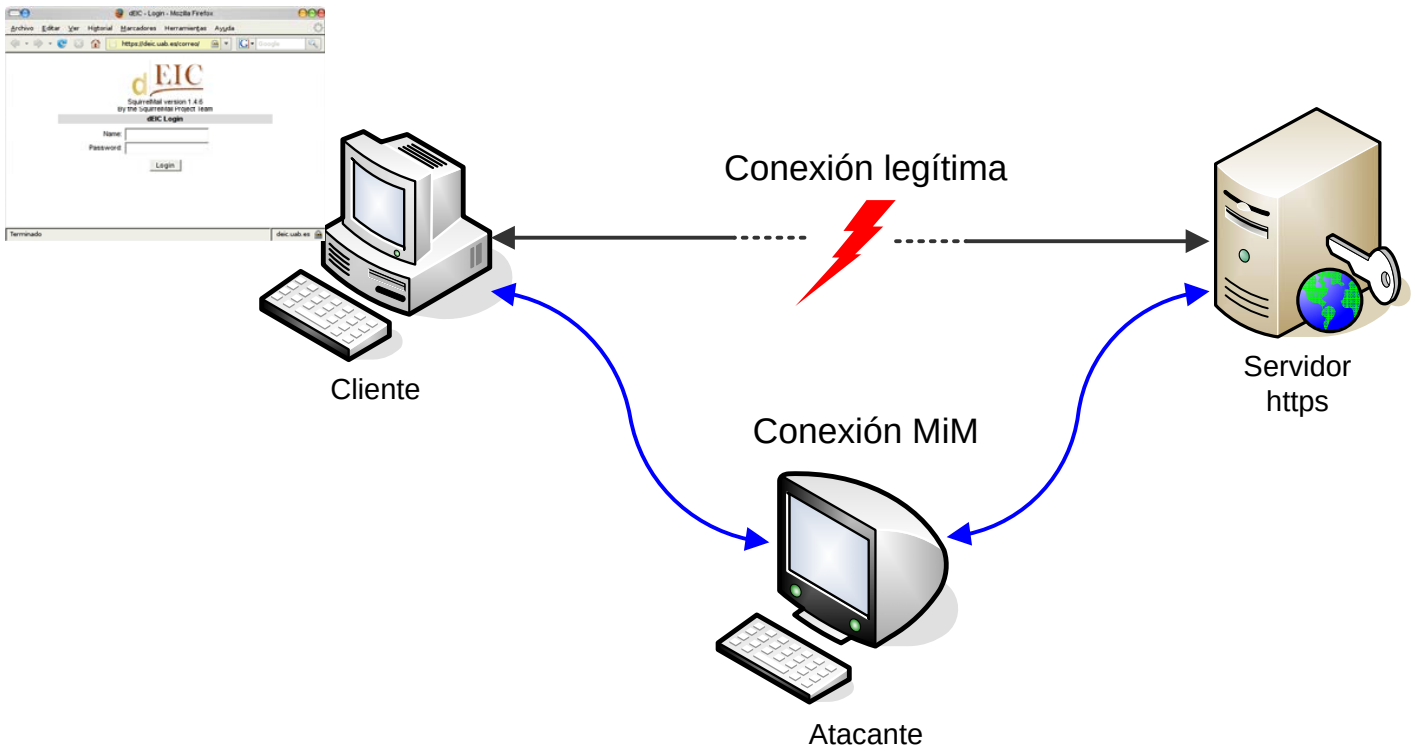
SSL/TLS - Ataques MiM



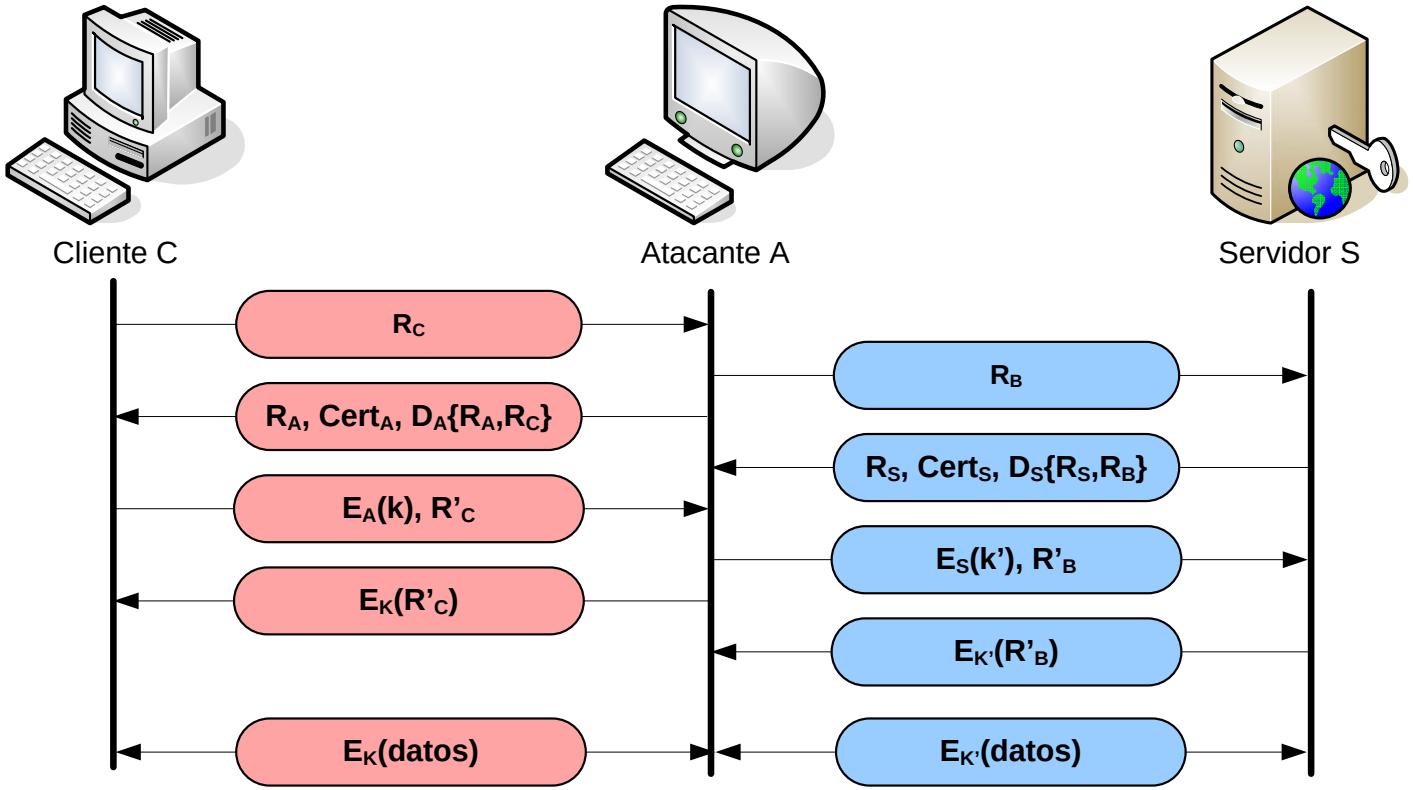
SSL/TLS - Ataques MiM



SSL/TLS - Ataques MiM

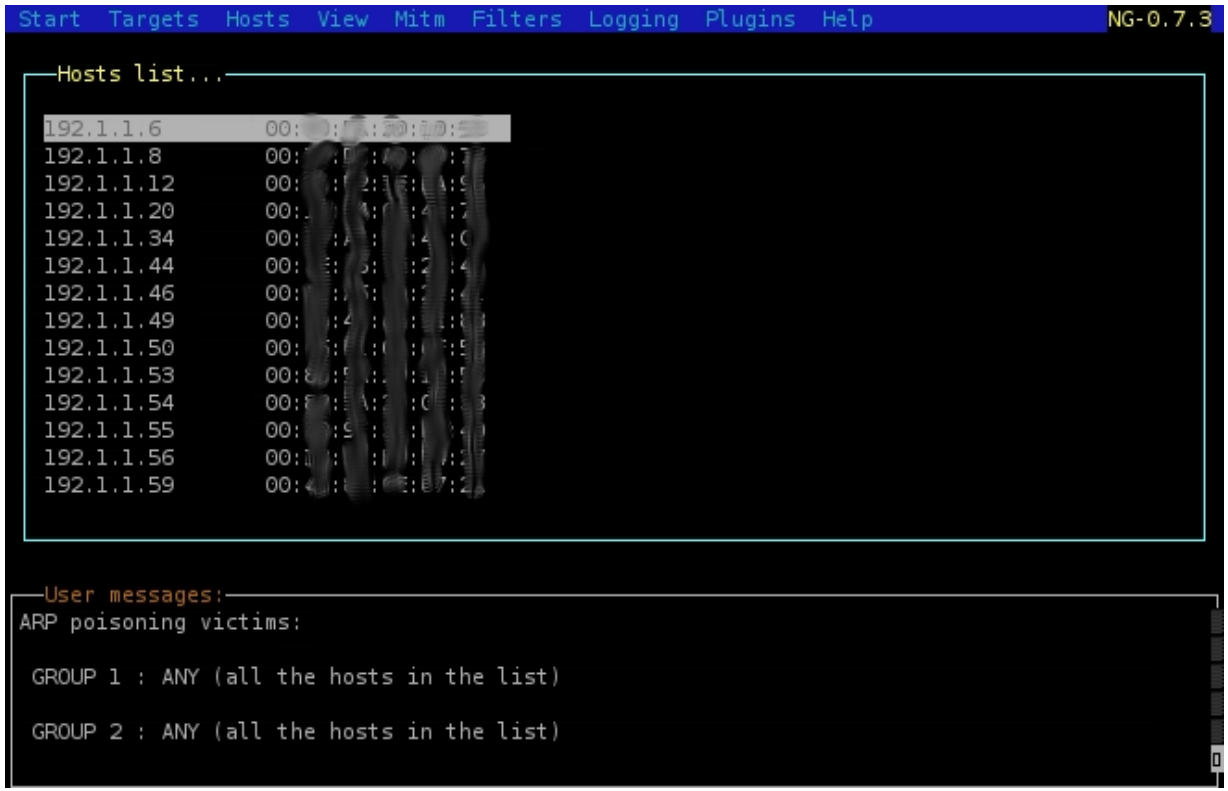


SSL/TLS - Ataques MiM



SSL/TLS - Ejemplo MiM con ettercap

Ejemplo de ataque MiM con ettercap:



SSL/TLS - Ataques MiM

Recomendaciones para mitigar/evitar ataques MiM:

- ¡Formar a los usuarios!
- Procurar no usar certificados auto-firmados.
- MAC *binding* en los *switches*.
- Tablas ARP estáticas (`arp -s`) → difícil de mantener en redes grandes.
- Utilizar herramientas de monitorización de cambios en las tablas ARP:
  - arpwatch (`http://www-nrg.ee.lbl.gov/`)
  - arpalert (`http://www.arpalert.org/`)
  - snort (`http://www.snort.org/`)
- Filtrar paquetes ICMP Redirect.
- Uso de DNSSEC.

Índice

Network DoS

¿Qué hacer ante ataques DoS/DDoS por inundación de tráfico?

- Poca cosa desde el punto de vista del servidor web :-(
- Al menos, configuración adecuada de las directivas relacionadas con conexiones e instancias.
- Diversidad de ataques con características diferentes:
  - *SYN Flood*
  - *ICMP Flood*
  - *UDP Flood*
- Hay que relegar esta tarea al *firewall*. Activación *SYN Cookies*.
- A pesar de esto, afortunadamente existe `mod_evasive` :-)
  (`http://www.zdziarski.com/projects/mod_evasive/`)

Network DoS

Características del módulo `mod_evasive`:

- Pensado para reaccionar ante ataques HTTP DoS o ataques por fuerza bruta.
- Fácil de configurar.
- Capaz de lanzar comandos cuando un DoS es identificado → interacción con otras aplicaciones de seguridad.
- Ejemplos de sus capacidades:
  - Crear una nueva regla en un firewall.
  - Notificaciones por email.
  - Interacción con syslog.

Network DoS

Ejemplo de configuración de `mod_evasive`:

```
DOSHashTableSize    3097
DOSPageCount        10
DOSPageInterval     1
DOSSiteCount        50
DOSSiteInterval     1
DOSBlockingPeriod   10

DOSEmailNotify   "admin@myserver.com"
DOSSystemCommand "iptables -A INPUT -s \%s -p tcp --dport 80 -j DROP"
DOSLogDir        "/var/log/apache/mod_evasive.log"

DOSWhitelist    127.0.0.1
DOSWhitelist    192.168.*.*
```

Índice

Otras consideraciones...

¿Que otros aspectos deberíamos tener en cuenta?

- Configuración de logs mediante `mod_log_config`:
  - Archivos `access_log` y `error_log`
  - Formato de logs mediante la directiva *LogFormat*
  - ¡Deben almacenarse en una máquina remota!
  - Rotación
  - Análisis de forma periódica: logscan, logwatch, ...
- Filtrado a nivel de la capa de aplicación: `mod_security`
- Evitar el *google-hacking* configurando adecuadamente *robots.txt*:

```
User-agent: *     # Aplicable a todos los robots
Disallow: /       # Impide la indexación de todas las páginas
```

Índice

Conclusiones

Conclusiones:

- La mayoría de administradores no adecua la configuración a sus requisitos de seguridad.
- Mejorar la seguridad en Apache no es sólo crear una buena configuración. Otros factores:
  - Sistema operativo
  - Red
  - Formación de usuarios
- Módulos específicos + configuración refinada == **Mayor Seguridad**

Referencias

Diversos autores

Apache documentation

http://httpd.apache.org/docs/

T. Dierks, C. Allen

RF2246 - The TLS Protocol Version 1.0

http://www.ietf.org/rfc/rfc2246.txt

Ivan Ristic

Apache Security

O'Reilly, Marzo 2005, ISBN: 0-596-00724-8.

Ryan C. Barnett

Preventing Web Attacks with Apache

Addison-Wesley, 2006, ISBN: 0-321-32128-6

Tony Mobily

Hardening Apache

Apress, Abril 2004, ISBN: 1-59059-378-2

¡Gracias por su atención!

¿Preguntas?