



Seguridad Multinivel en servidores web

OWASP Spain 2008

Presentación: Luís Calero Martín

Desarrollo: Hugo Vázquez Caramés, Luís Calero

Fuentes: Argus Systems, IBM, NSA, CC, TSEC, RSBAC.





Empresa

- Empresa de nicho dedicada a la seguridad telemática.
- Reducida pero selecta cartera de clientes.
- Importamos conocimiento y tecnología: EEUU, Alemania, Suiza...
- Trabajos de I+D en seguridad desde 1999.



Método Pentest

- Seleccionamos y gestionamos RRHH de la más alta calidad para llevar a cabo proyectos de seguridad.
- Tiger Teams bajo demanda formados por expertos provenientes de dentro y fuera de nuestras fronteras.
- Ningún equipo simultanea dos proyectos de envergadura



Valor Pentest

- Experiencia trabajando con entidades financieras, aseguradoras, sector energético, multinacionales, defensa, AAPP, etc.
- Formaciones completamente a medida del cliente: MLS, programación de exploits, web hacking avanzado...
- Muy ágiles en la planificación y ejecución de proyectos de seguridad.

Repercusión Pentest

Fallos en un cortafuegos anunciado como 'inhackable'

Una vulnerabilidad en OWA
Permite obtener credenciales

Una nueva vulnerabilidad en OWA (Outlook Web Acces) puede ser utilizada por un atacante remoto para obtener las credenciales del usuario que abre un mensaje en dicho



ComputerWeekly.com

Logged Out

IT Management Security

Friday 10 October 2003

Spanish experts set up hacker competition

Two Spanish security experts participate in a competition to production server.

Telefónica parchea su 'proxy' para subsanar un fallo de seguridad

MAL
Los investigadores españoles Hugo Vázquez y Toni Cortés han detectado una importante vulnerabilidad en el programa Inktoni Traffic-Server, en los *proxies* de de Telefónica y otros proveedores del mundo.



15/05/06 Banca Electrónica. Nuevos Vectores de Ataque

Este artículo ha sido facilitado por Hugo Vázquez Caramés, autor de los métodos de 'hacking' utilizados por los señores de Banca Electrónica, concluye la susceptibilidad de estos métodos a verse comprometidos por una serie de vectores de ataque.

E-CERT certifica la validez de este tipo de ataques.

Puede ver el artículo completo aquí



Technical Report

Number 653



Computer Laboratory

cisco Security Advisory: IOS HTTP Server Command Injection Vulnerability

Current ID: 68322
Library ID: cisco-sa-20051201-http
u/wwww.cisco.com/warp/public/707/cisco-sa-20051201-http.shtml
vision 1.0
Public Release 2005 December 01 2100 UTC (GMT)

Anonymity and traceability in cyberspace

The Register

'Itm hacking challenge' product is flawed
By JFF Layton
Published: 2005-09-04 14:00:00
Although the 'hacking challenge' consumer security device is not foolproof, Spanish video hackers claim

EL PAIS

En français: Cisco et Checkpoint font vulnérabilités dans leurs "firewall"
Ils ont été trouvés sur deux produits "leader" du marché sur le produit Firewall-1 de Checkpoint.

English version
In the case of Firewall-1 it is an early stage to define all possible usage of that vulnerability. No information is available from the vendor. But using the vulnerability found by Hugo Vázquez Caramés from Grupo ADD, it seems to be possible to have the firewall being not responding anymore to certain remote services among which the famous SecurRemote used by mobile users for remote connections through Internet.

Résumé en français
Dans le cas de Firewall-1 il est en cours pour déterminer l'étendue de cette vulnérabilité. Pour le moment il n'y a pas d'information du vendeur. Mais en la découverte de Hugo Vázquez Caramés du groupe ADD, il semble être possible de rendre indisponible un ensemble de services distants parmi lesquels le fameux SecurRemote qui permet à des utilisateurs mobiles de se connecter avec leurs entreprises.





Presentación *Hugo Vázquez*

- **CEO** de Pentest
- Responsabilidades: Dep. I+D, Evaluación de soluciones de seguridad, gestión de Tiger Teams.
- Áreas Tecnológicas de Interés: Seguridad del S.O., Sistemas de Autenticación,...
- Aficiones: Volar, el mar,...



Presentación *Luís Calero*

- **Director Técnico** de Pentest
- Responsabilidades: Desarrollo y defensa de proyectos en las TI, gestión de *Tiger Teams*.
- Áreas Tecnológicas de Interés: Comunicaciones, Seguridad en servidores, Sistemas de seguridad perimetral...
- Aficiones: Deportes de aventura en montaña y la robótica...



Agradecimientos

- **Innovative Security Systems -Argus Systems-**
Por descubrirnos el mundo de los sistemas MLS.
- **NSA**
Por desarrollar una solución MLS integrada en el kernel de Linux (2.6).
- **Comunidad en general**
Por comenzar desarrollos –RSBAC,...- orientados a sistemas MLS/MAC cada vez más usables y eficientes

(I) ¿Qué sabe Pentest de MLS?

- En el año **2000** miembros que ahora conforman **Pentest** se interesan por sistemas MLS y comienzan estudiarlos.
- En el año **2001** se hace público un informe espectacular que describe un hack a un sistema MLS.
- El informe de LSD -“Last Stage of Delirium”- deja constancia de la extrema dificultad que conlleva hackear un sistema MLS:

Necesidad de trabajar en “ring 0”

(II) ¿Qué sabe Pentest de MLS ?

- **2002, 2003** Evaluación del mercado de los MLS.
- En el **2005** Pentest inicia contactos con *Argus Systems*.
- En el **2006** Pentest recibe formación específica de sistemas MLS en Suiza, y se convierte en uno de las 18 empresas del mundo capaces de operar **Pitbull**, un sistema MLS utilizado por el D.o.D. de EEUU.
- En el **2007** IBM hace efectivo el derecho de integrar el sistema MLS de Argus en **AIX 6**.
- En el **2008** Pentest es la única empresa de España – y una de las pocas del mundo- con conocimientos para operar el MLS de AIX 6.



Introducción

- El objetivo de esta presentación es dar a conocer las posibilidades de los sistemas MLS en entornos web.
- Gran parte de nuestra experiencia se basa en un software comercial (*Pitbull Foundation*)
- La presentación se apoya en dicho software para los ejemplos, pero se basa en estándares –TSEC, CC,...-
- Se presentan iniciativas del mundo *Open Source* u otros fabricantes



Temas de discusión

- ¿Qué es un sistema MLS?.
- Tecnologías existentes.
- El mercado de los MLS.
- MLS y web.
- MLS:
 - Rentabilidad frente a otras soluciones.



(I) ¿Qué es un sistema MLS?

- De la Wikipedia: “(...) *application of a computer system to process information with **different sensitivities** (i.e. classified information at different security levels), permit simultaneous access by users with different security clearances and needs-to-know, and prevent users from obtaining access to information for which they lack authorization (...)*”

http://en.wikipedia.org/wiki/Multilevel_security



(II) ¿Qué es un sistema MLS?

- Historicamente certificaciones via TSEC (*rainbow series*).
- Actualmente CC.
- No es perfecto, se le conocen problemas: *sanitization, covert channels...*
 - Aún así es extremadamente **robusto**



(III) ¿Qué es un sistema MLS?

- Etiquetado de datos:
 - (SECRET, TOP SECRET, PUBLIC...)
- Generalmente se apoyan en *Mandatory Access Control*, *Role Based Access Control*, *Domain Based Access Control*, etc.
- Al final, un sistema solo puede certificarse como MLS **si el KERNEL es MLS!**



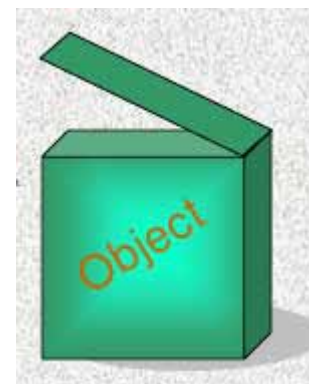
(I) Objetos y sujetos

- Todo en UNIX es un **objeto** o un **sujeto**.
 - Los sujetos llevan a cabo acciones.
- En UNIX, todos los sujetos son procesos.



(II) Objetos y sujetos

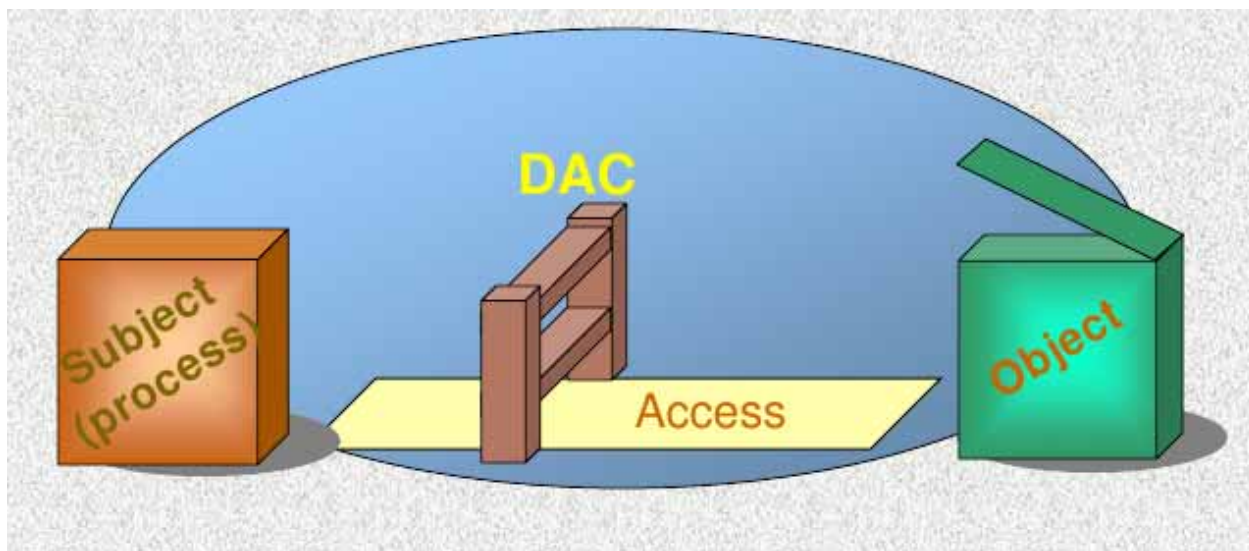
Los sujetos actúan sobre objetos





(I) DAC

- “Discretionary Access Control”
- Bajo control del propio usuario:
el usuario “decide” con quien comparte la información.





(II) DAC

- Controla la lectura, escritura y ejecución
- El propietario de un fichero puede dar acceso a terceros

Basado en:

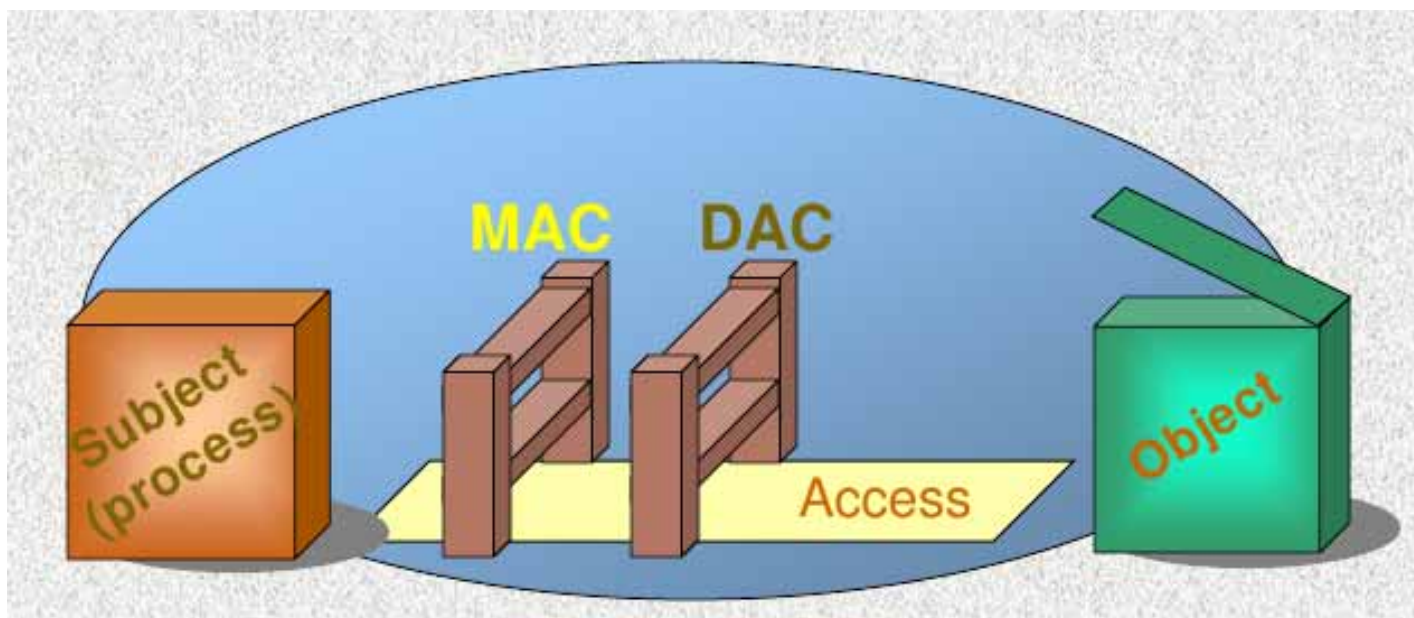
Bits de permisos de UNIX (r/w/x)

Identificadores de usuario y grupo (UID, GID)



(I) MAC

- Mandatory Access Control.
- Controlado por el sistema.





(II) MAC

- Controla la lectura, escritura y ejecución
- El sistema controla el acceso,
tal y como hayan definido los administradores
- El propietario de un fichero no puede cambiar sus permisos MAC
- El propietario del fichero no controla a quien le da acceso



MAC en un sistema MLS

- Basado en etiquetas de sensibilidad (SL's)
 - SLs indican el nivel de sensibilidad
 - Cualquier sujeto y objeto tiene una SL asociada
 - El sistema compara las etiquetas de los objetos y los sujetos

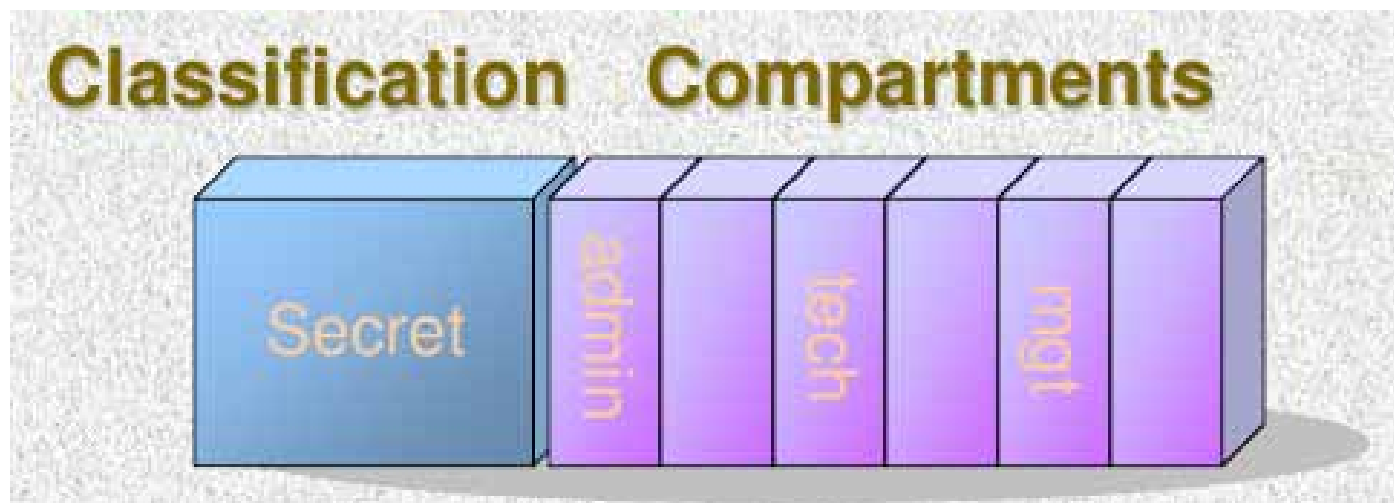
(Las reglas de comparación determinan el acceso)

 - Etiquetas de nivel superior dominan a etiquetas de nivel inferior



Etiquetas de sensibilidad

- Nivel de seguridad: clasificación
p.e., SECRET, PUBLIC
- Cero o más compartimentos
p.e., admin, technical, etc.





Clasificación

- Orden jerárquico:
 - La información sólo fluye de niveles inferiores a superiores.
- Las clasificaciones pueden personalizarse para cada sistema.



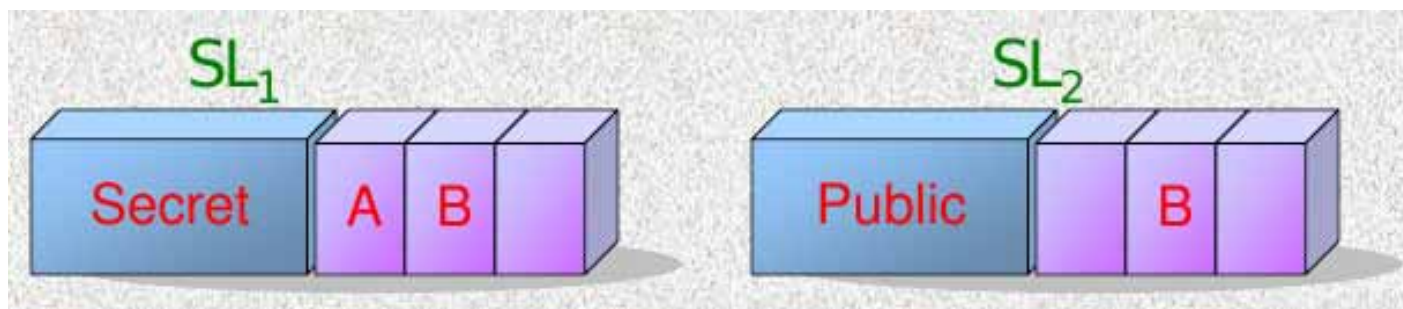


Dominancia

- SL_1 domina a SL_2 si:

Clasificación de SL_1 \geq clasificación de SL_2
y

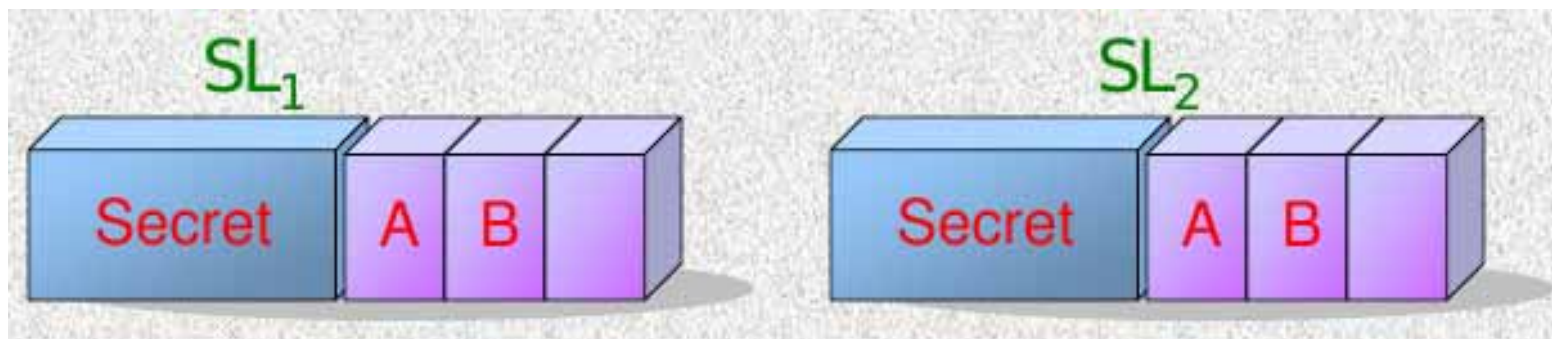
Los compartimentos de SL_1 incluyen a los compartimentos de SL_2





Igualdad

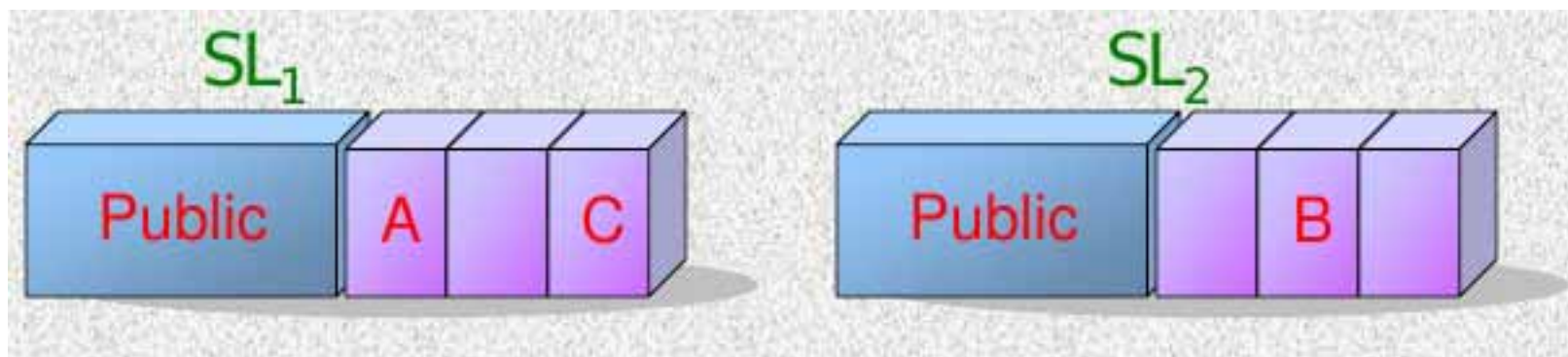
- La igualdad es un caso especial de dominancia.
- **Clasificaciones y compartimentos iguales.**
- SL's iguales se dominan recíprocamente.





SL's Disjuntas

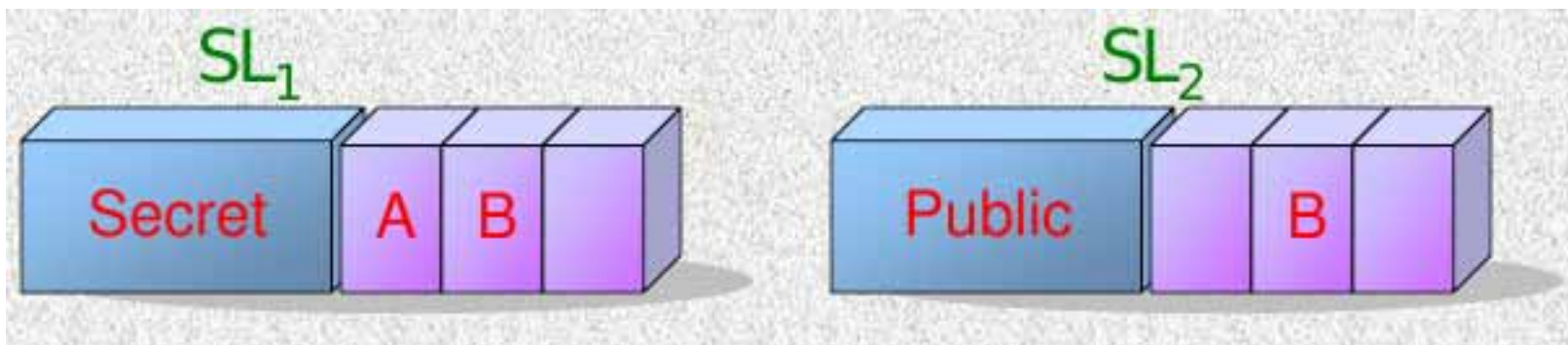
- Si no hay dominancia, las SL's son **disjuntas** o no comparables.





Dominancia estricta

- SL_1 domina a SL_2 si:
 SL_1 **domina a** SL_2
 y
 SL_1 **es diferente a** SL_2





MAC y las SL's

To **read**, process SL must dominate file SL

To **write**, process SL must equal file SL

To **execute**, process SL must dominate file SL (same as for read access)

If SLs disjoint, no access allowed

♦ Remember: DAC access also required



(I) MAC y las SL's

| Process SL | Dominance relationship | File SL | Access allowed |
|---------------|---------------------------|------------|----------------------|
| TS A | <input type="text"/> | SEC | <input type="text"/> |
| TS | <input type="text"/> | SEC A | <input type="text"/> |
| PUB A | <input type="text"/> | CON A | <input type="text"/> |



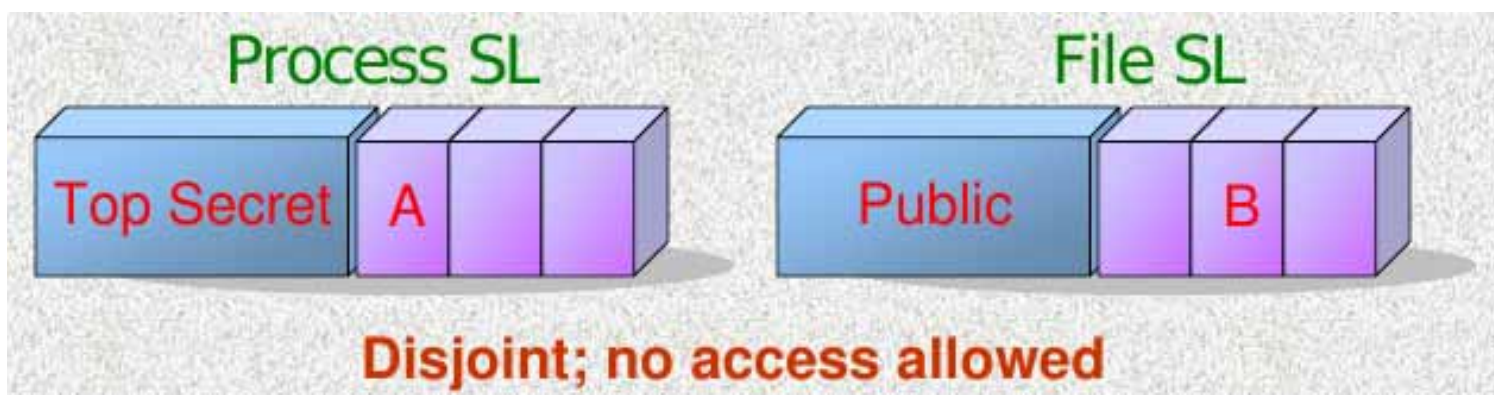
(II) MAC y las SL's

| Process SL | Dominance relationship | File SL | Access allowed |
|---------------|---------------------------|------------|-------------------|
| TS A | dominates | SEC | R X |
| TS | disjoint | SEC A | -none- |
| PUB A | dominated by | CON A | -none- |



Compartimentos

- Tan importantes como las clasificaciones.
- Procesos con altas clasificaciones **NO** pueden acceder a ficheros de distintos compartimentos.





Ejemplos de dominancia

Classes: PUB < CON < SEC < TS

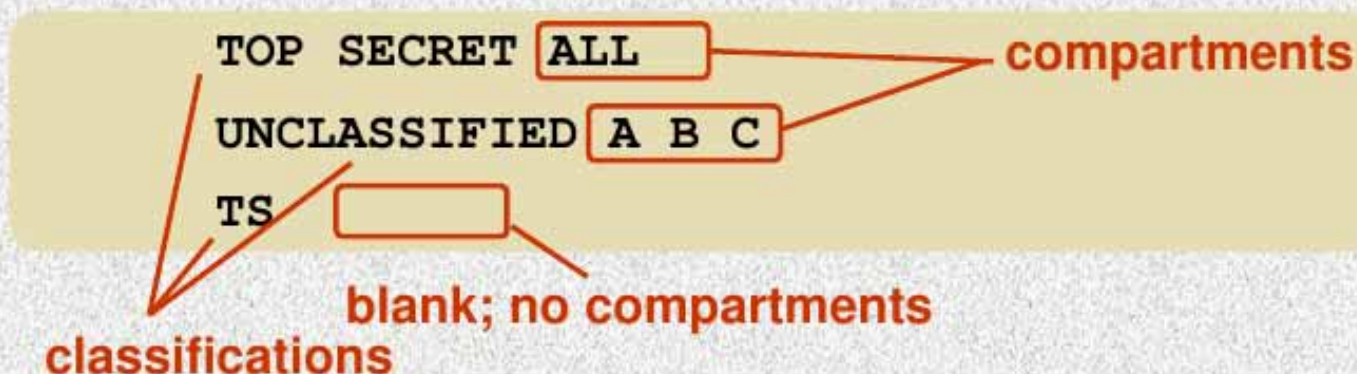
Compartments: A, B, C

| Label 1 | Label 2 | L1 dom L2 ? |
|-------------|---------|-------------|
| PUB | PUB | Yes |
| CON : A B C | SEC : A | No |
| TS : A | TS | Yes |
| SEC : A | SEC : B | No |
| SEC : A B | CON : B | Yes |
| TS : A B | PUB : C | No |



(I) Notación

- ◆ Displayed on system as class, space, compartments (space-separated)





(II) Notación

- ♦ May be displayed in text as class, colon, compartments (space separated)





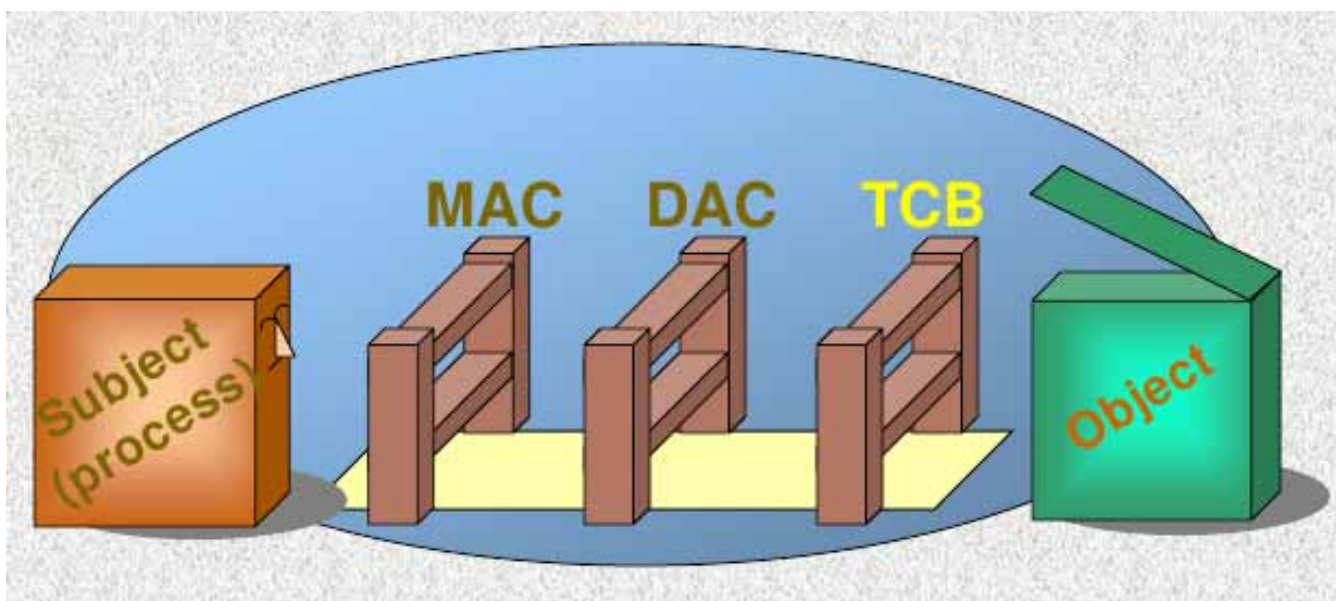
Otros conceptos

- Nivel de “visibilidad de un proceso”
Max., Min., Efectivo...
- Etiquetas de Sistema
(System High, System Low,...)
- SL's de directorios, Herencia de SL's
- RIPS0, CIPS0...
- Rangos de compartimentos...
- Directorios multinivel,...
- Mandatory Integrity Control, etc...



(I) TCB

- Trusted Computing Base
- El sistema se “autoprotege”





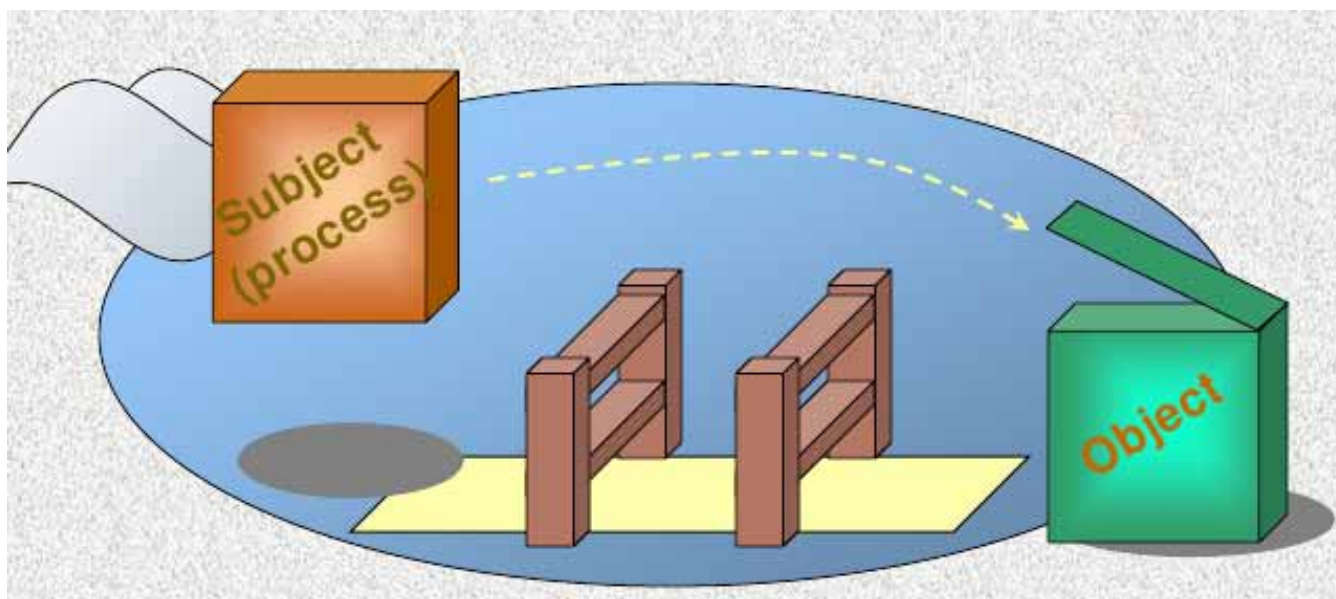
(II) TCB

- El “Trusted Computing Base” es el *hardware*, *firmware* y *software* que refuerza la seguridad del sistema
- Típicamente: Kernel, programas setuid, etc...



(I) Privilegios

- Permiten que los procesos hagan **bypass** de la seguridad.





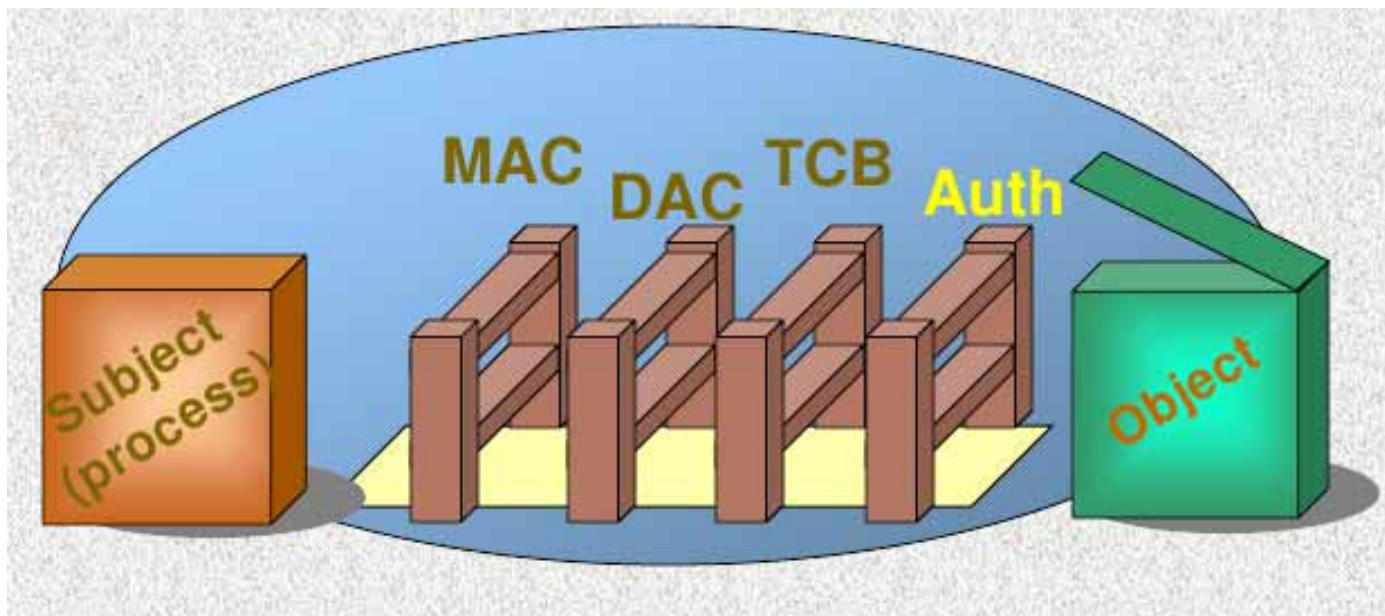
(II) Privilegios

- Cada privilegio es un riesgo potencial.
- Principio del “mínimo privilegio”: sólo permitir lo mínimo para ser funcional.
- Un sistema **MLS/MAC** puede tener hasta **más de 100 privilegios**.
- Un sistema UNIX tradicional, sólo uno: “root”.
root puede hacer lo que quiera en el sistema.



(I) Autorizaciones

- Controlados por un rol





(II) Autorizaciones

- A una cuenta se le pueden dar autorizaciones para garantizar el acceso a programas.
- Define una serie de acciones que se le permite hacer al usuario.



Ejemplo Autorizaciones

- **ISSO** Information Systems Security Officer
 - Mantiene la política de seguridad.
- **SA** System Administrator
 - Mantenimiento de cuentas de usuario.
 - Instalación de software.
- **SO** System Operator
 - Backups.
 - Gestión impresora.
 - Apagado del sistema.



Privilegios vs. Autorizaciones

- **Privilegio**

- Atributos de un proceso.
- Permite que un proceso bypasssee la seguridad.
- Restricciones.

- **Autorización**

- Asociados al *user ID*.
- Permite la ejecución de programas restringidos, otorga funcionalidad “extra” a un programa, etc.



Objetivo final en MLS

- Sistema compartimentalizado.
- Procesos con mínimos privilegios.
- Usuarios con mínimas autorizaciones.



Tecnologías de “apoyo” en sistemas MLS

- Multitud de variantes y modos de implementar MLS.
 - MLS se suele apoyar en distintas tecnologías:
 - RBAC.
 - RSBAC.
 - DBAC.
- etc.



RBAC

- Role Based Access Control.
- Usado por AIX6 (Pitbull Foundation), Solaris, SELinux, GRsec (no es MLS),...
- Se definen roles.
 - Cada rol tiene asociados una serie de privilegios.
 - Todos los usuarios del sistema tienen un rol.



RSBAC

- Rule Set Based Access Control.
- Open Source Access Control (GPL).
- Disponible para Linux.
- Modular (Security Modules).
- Cada módulo se encarga de un aspecto concreto de la seguridad:
 - AUTH (autenticación).
 - UM (User Management).
 - DAZ (Dakuzo Malware Scanner).
 - MAC...
- Extremadamente flexible.



DBAC

- Domain Based Access Control.
- Una implementación simplificada de un sistema MAC.
- Los dominios se pueden aplicar a usuarios, ficheros, procesos y *networking*...
- Usado por Pitbull LX (no es MLS).



DBAC: Ejemplo

- Al fichero “X” se le asigna el dominio sys(-w-).
- Este dominio no limita la lectura o ejecución.
- Si un proceso quiere escribir en el fichero “X” debe pertenecer al dominio “sys” y con permisos de escritura: (-w-), (rw-) o (rwx).



El mercado de los MLS

- Algunos ejemplos de MLS:
 - Solaris 10 (nativo)
 - Solaris 10 + Pitbull
 - AIX 6
 - SELinux
 - RSBAC
- etc.



Solaris 10 (nativo)

- Tecnología evolución de Trusted Solaris.
- Pros: ?
- Contras: propietario, penaliza rendimiento.
- Gestión...?



Solaris 10 + Pitbull

- Pros:
 - Tecnología probada durante más de 10 años (Credit Suisse, D.o.D EEUU).
- Contras:
 - Propietario, gestión “no trivial”.



AIX 6

- Sistema MLS adquirido a Argus Systems.
- Pros:
 - Estable y robusto basado en tecnología probada.
- Contras:
 - Propietario, gestión “no trivial”



SELinux

- Desarrollado por la *NSA*.
- Pros:
 - GPL, integrado en kernel 2.6, empleado en distintas distribuciones: RedHat, EnGarde,...
- Contras:
 - Requiere de interfaz de gestión, políticas estáticas – etiquetado durante el arranque- y poco intuitivas.



RSBAC

- Referencia : <http://www.rsbac.org>
- Pros:
 - GPL, modular, muy flexible.
- Contras:
 - Gestión “no trivial”, reducido grupo de desarrollo/soporte.



(I) MLS y web

- Los sistemas **MLS** pueden mejorar y facilitar el despliegue seguro de servicios públicos, p.e. **WEB**.
- El **etiquetado de datos** permite clasificar las distintas entidades del sistema en función de la sensibilidad.



(II) MLS y web

- Tradicionalmente:
Aislamiento mediante chroot.
- Actualmente:
Implementa MAC sin aprovechar las capacidades MLS.
- En el futuro:
MLS 100% funcional en WEB.



¿ Porqué usar MLS en WEB?

- Algunos entornos chroot se pueden romper.
- Las políticas MAC sin MLS permiten fugas de información.
- Permite abstraer la seguridad de un sistema fácilmente mediante la clasificación de sus activos.



(I) Problemática asociada a MLS en WEB

- Un sólo binario que controlar: httpd
- Múltiples usuarios (clientes http)
- Múltiples activos de información
(Páginas web, BBDD)
- A ciertos activos de información –BBDD- no se les puede aplicar MLS con sencillez...



(II) Problemática asociada a MLS en WEB

- Las BBDD o un servidor web son un único objeto.
- A un objeto sólo puede aplicarse una etiqueta de seguridad.
- Una BBDD o un servidor web puede tener acceso información de distintos niveles...



Una solución

- Distintos servidores web y distintas BBDD.
- Cada par “web + BBDD” dentro de un nivel de sensibilidad.
- Los usuarios se acreditan y son redirigidos al servidor web + BBDD que corresponde a su nivel.
- “Algo” gestiona las operaciones entre entidades de distintos niveles.



Estudio de una solución comercial

- Elementos principales de la solución:

Security Gate

Secure CGI Module

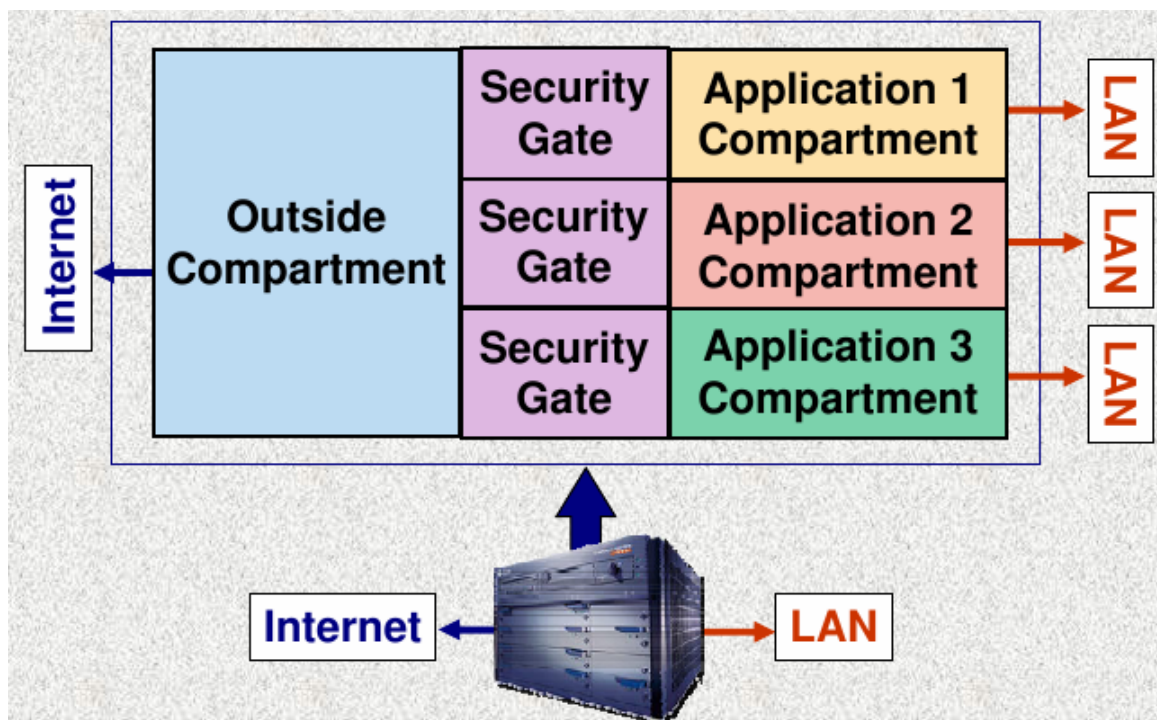
Secure Communications Enforcer

Secure Authentication Module



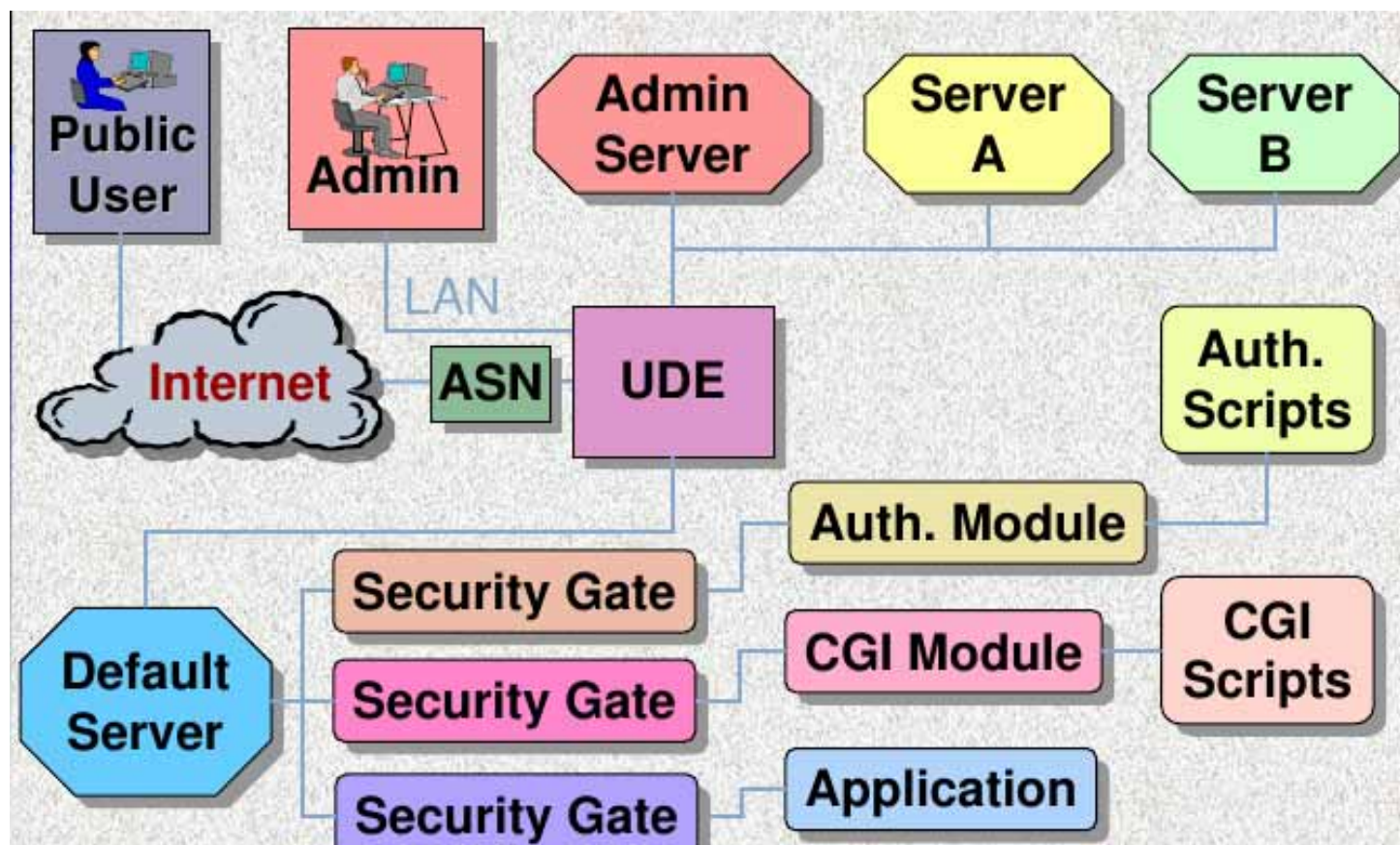
(I) MLS en web: diseño

- Los usuarios se autentifican a través de las “Security Gates” y son redirigidos a la aplicación correspondiente.

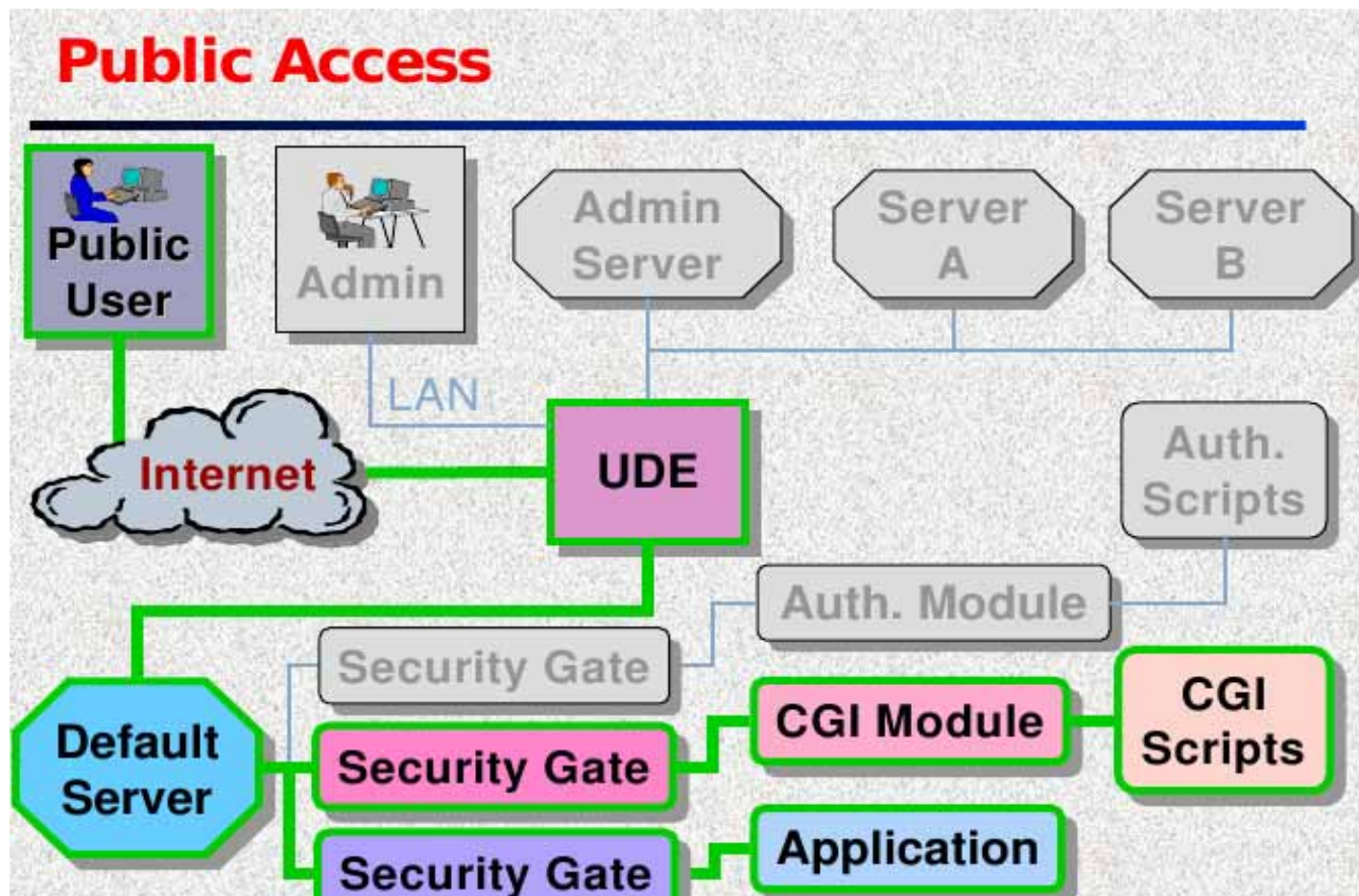




(II) MLS en web: diseño

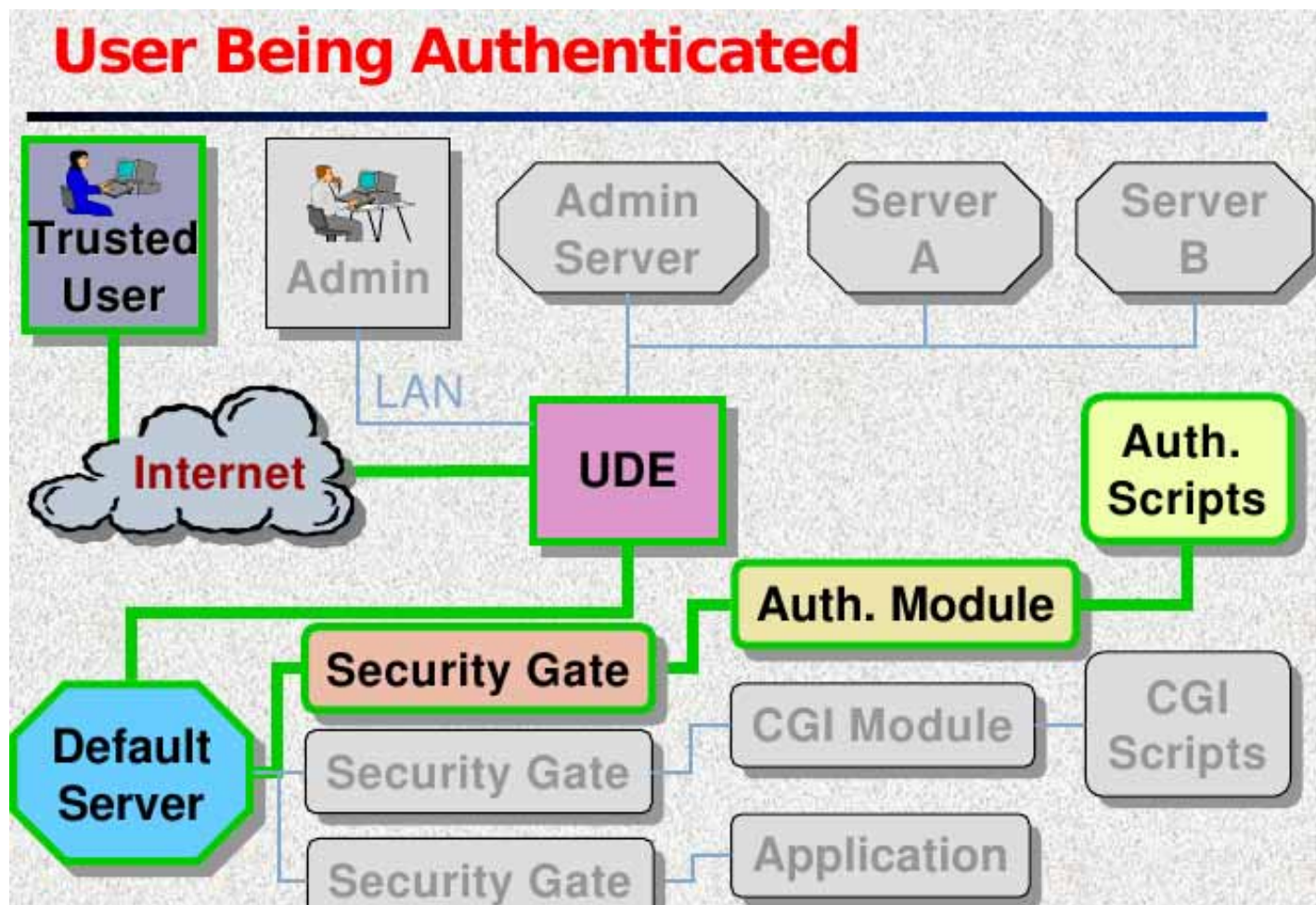


(I) MLS en web funcionando



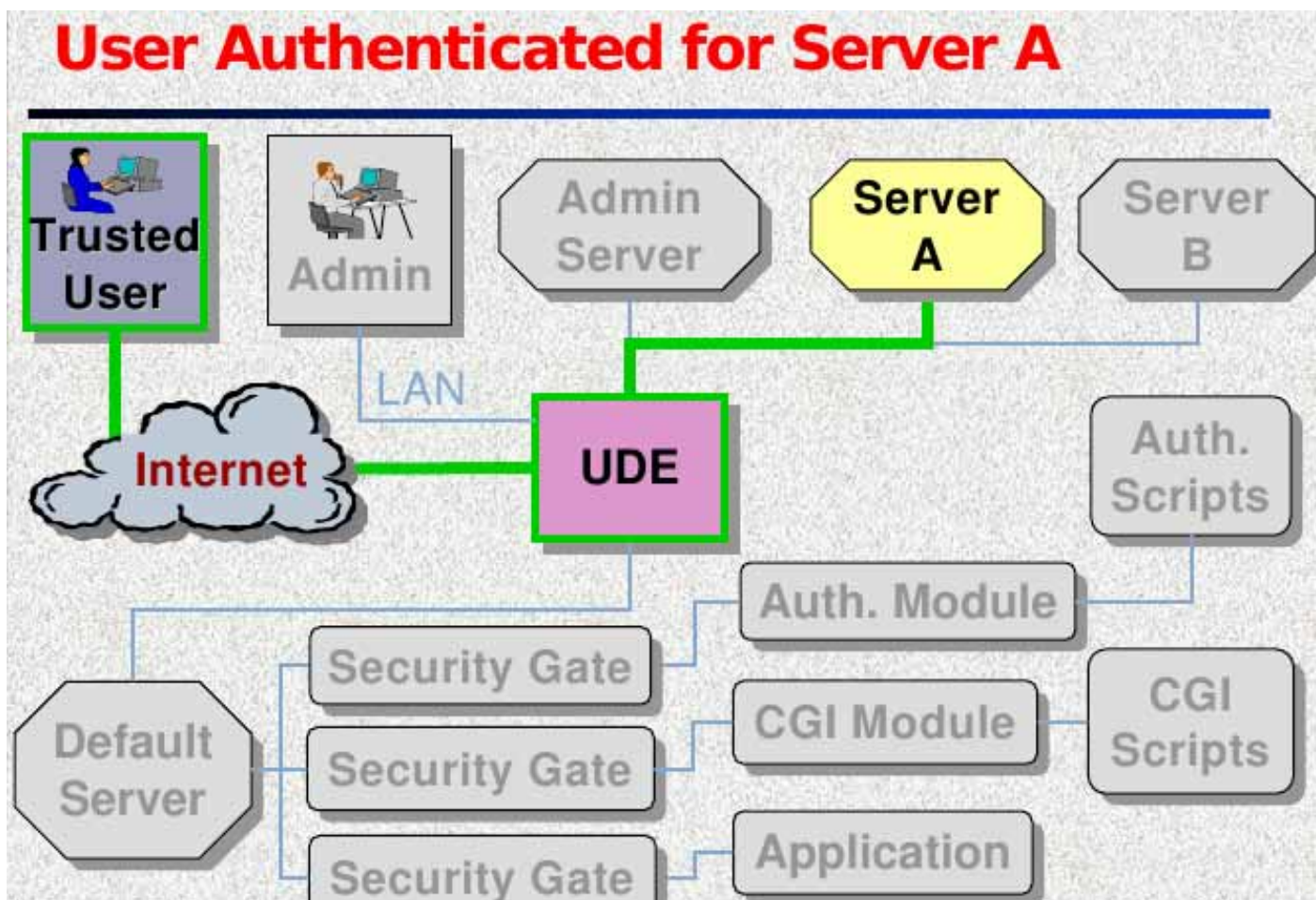


(II) MLS en web funcionando



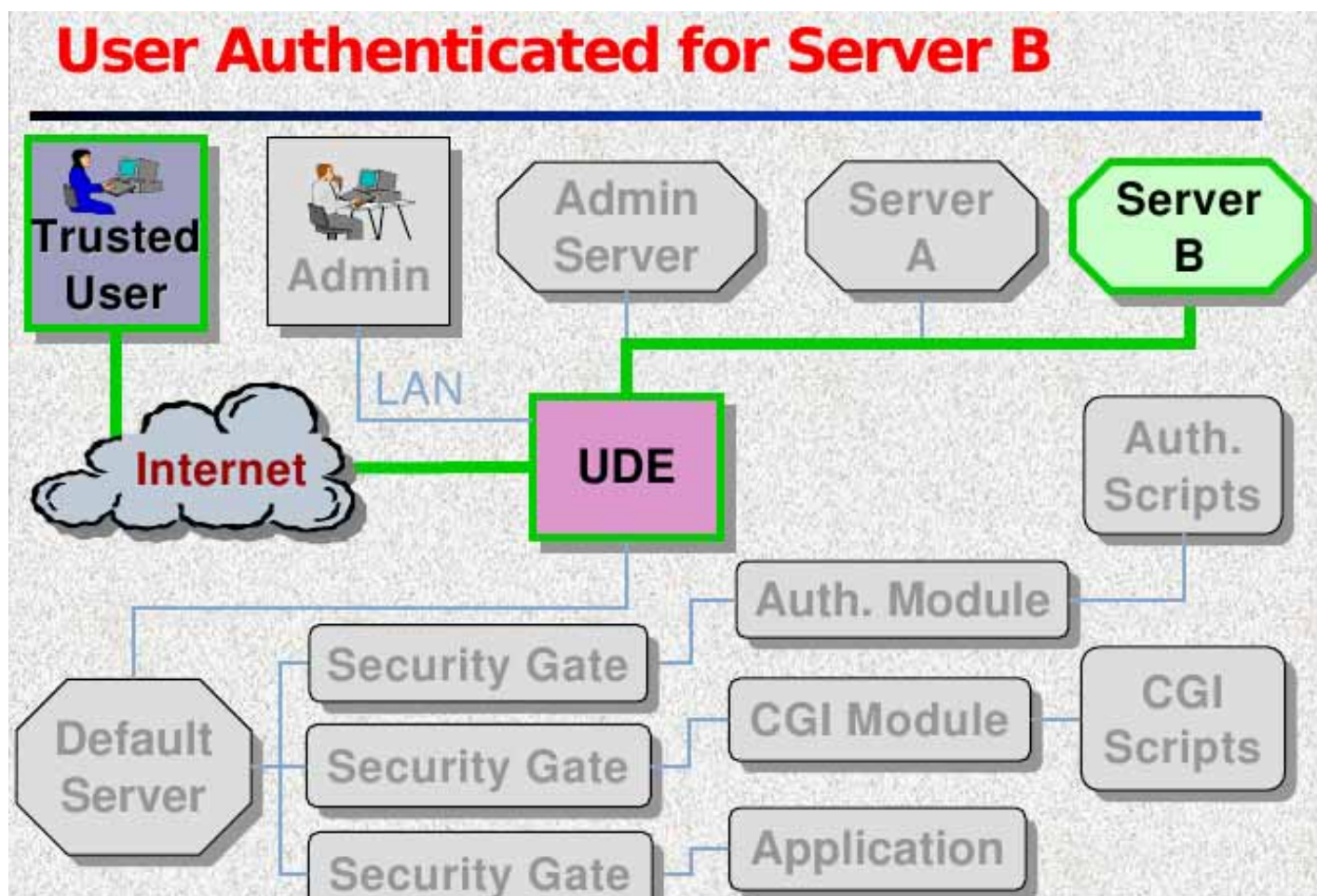


(III) MLS en web funcionando



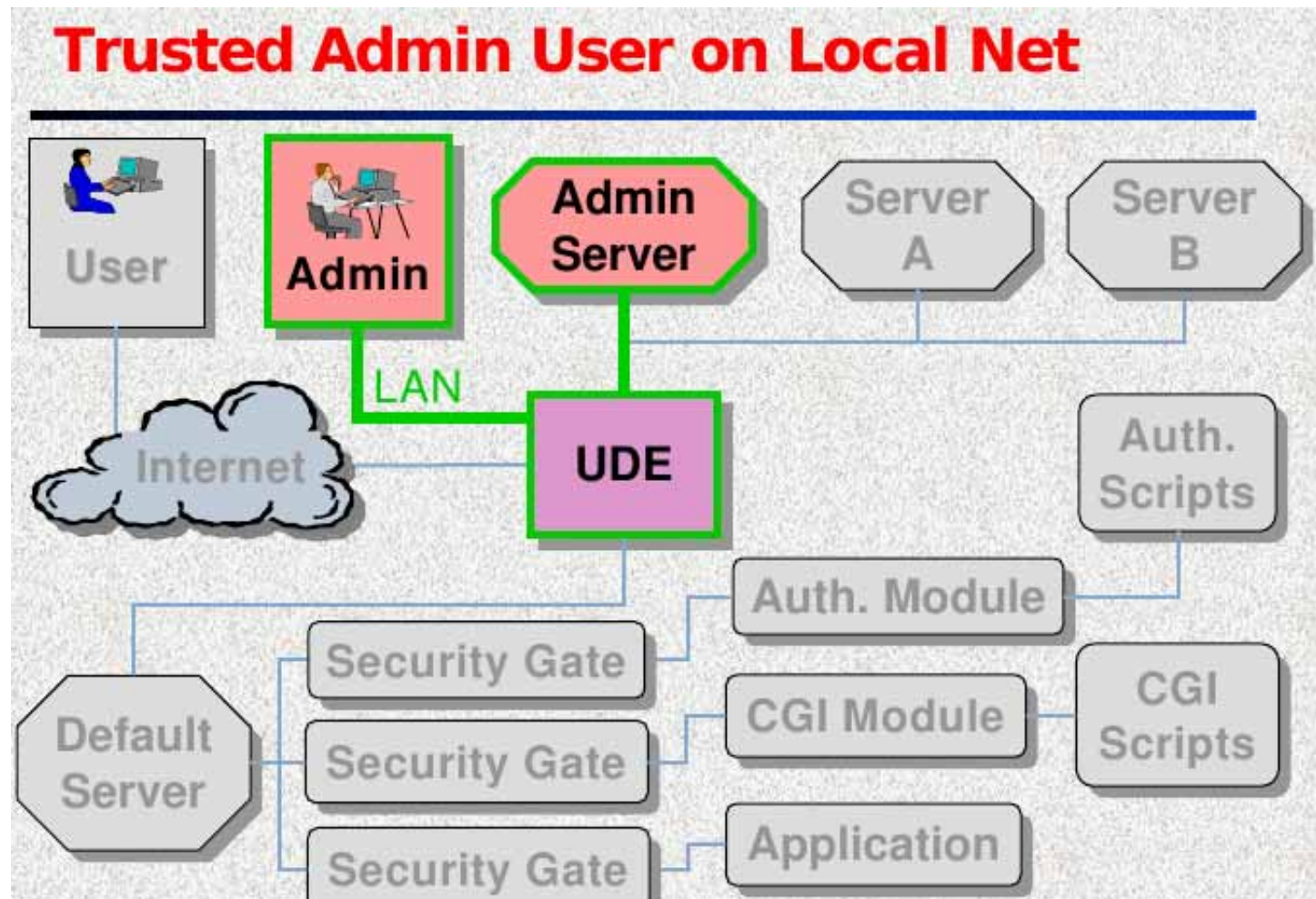


(IV) MLS en web funcionando





(V) MLS en web funcionando



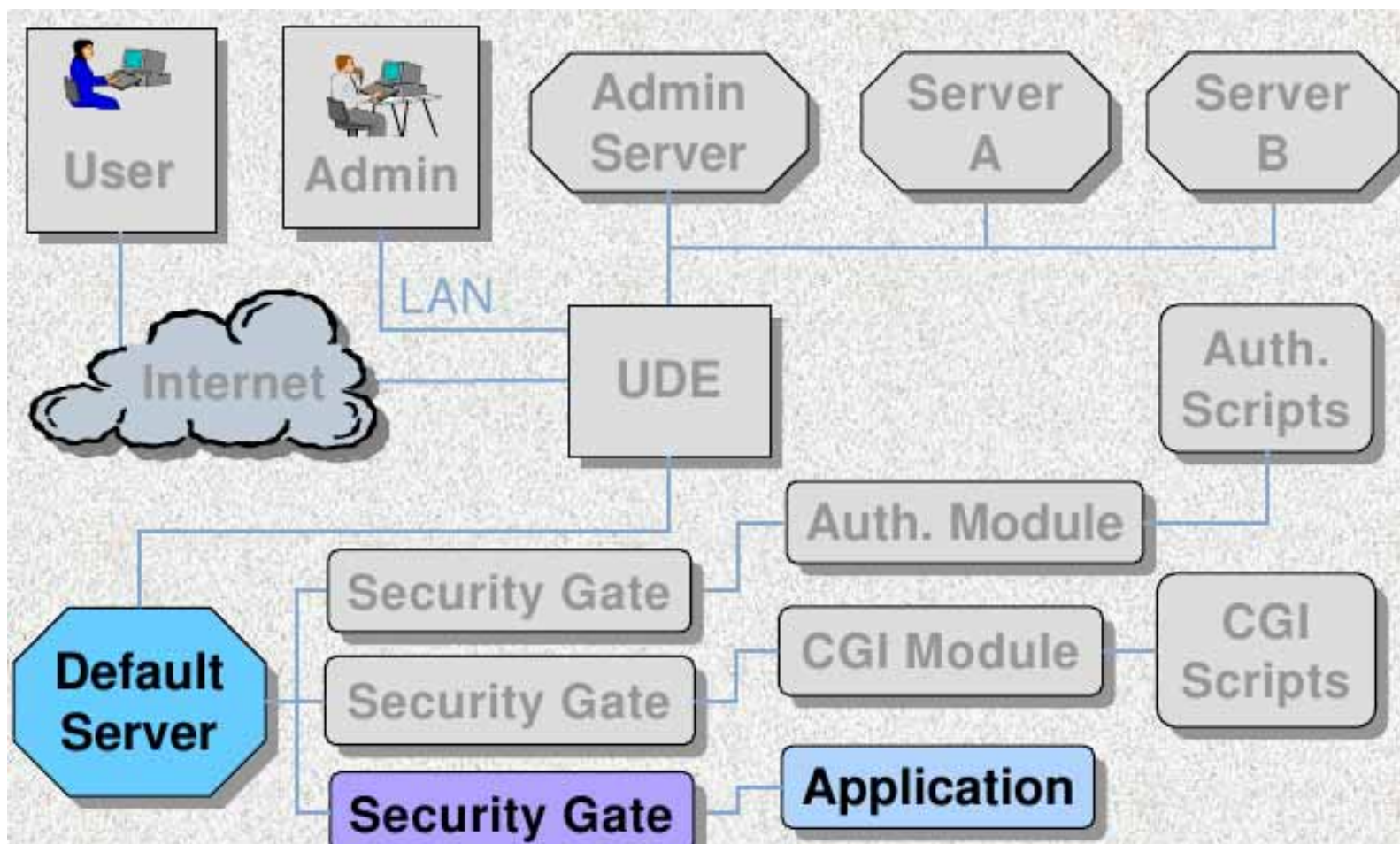


Security Gates

- Permite comunicación entre dos procesos con *etiquetas disjuntas*.
- Sólo permite comunicación “uno-a-uno” –entre procesos-
- Generalmente un servidor tendrá distintas “Security Gates” funcionando al mismo tiempo.

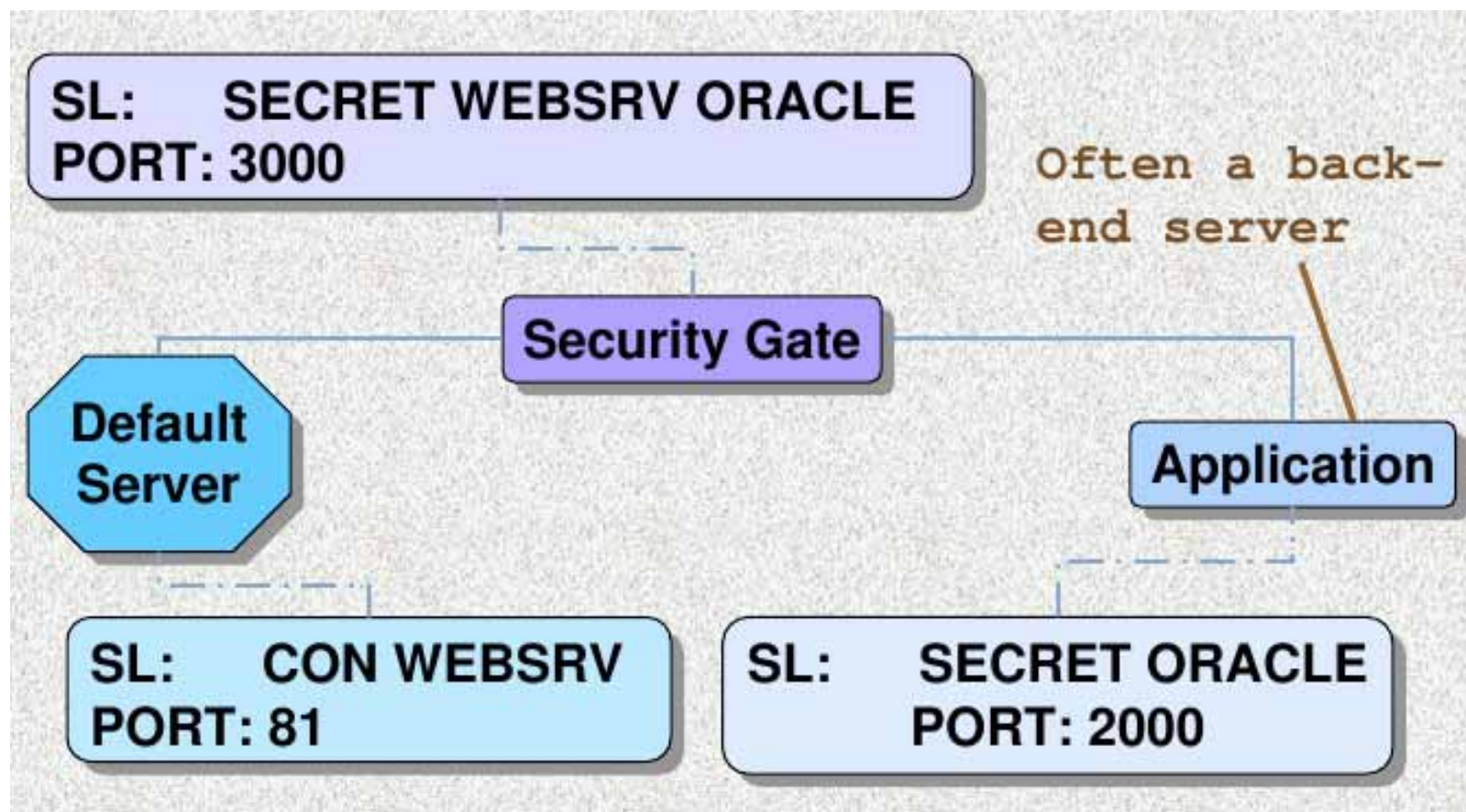


(I) Security Gates: ejemplo





(II) Security Gates: ejemplo





Security Gates: notas

- Las SL's de las *Security Gates* deben dominar a las SL's de los servidores.
- Tienen una configuración independiente de la del servidor web.



Security Gates. Ejemplo de configuración

```
<security_gate_oracle>
MODULE          SG
FE_SL           CON WEBSRV
#
BE_SL           SECRET ORACLE
BE_HOSTNAME     localhost
BE_PORT         2000
#
# continued...
#
```



Security Gates. Ejemplo de configuración

```
# ...continued
#
SG_HOSTNAME localhost
SG_PORT      3000
SG_SL        SECRET WEBSRV ORACLE
SG_MODE      Stream
SG_PIDFILE   /PATH/log/sg.sql.pid
SG_LOGFILE   /PATH/log/sg.sql.log
#
</security_gate_oracle>
```

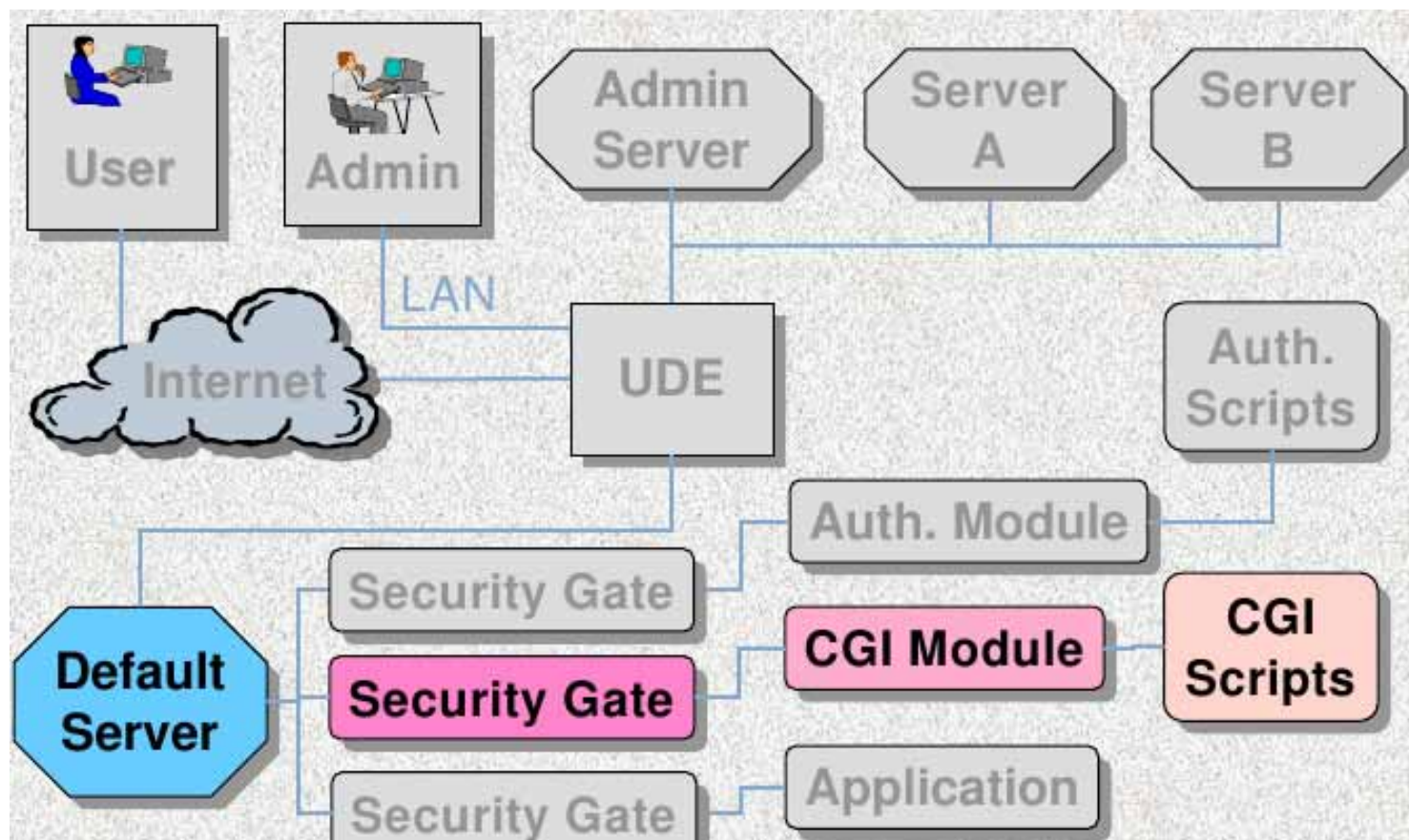



Secure CGI Module (CGId)

- Ejecución segura de las CGI's.
- Comunicación entre servidor web y las CGI's via *Security Gates*.
- Aisla la funcionalidad de las CGI's de la del propio servidor web.
- Partición virtual entre los programas CGI y el propio servidor.



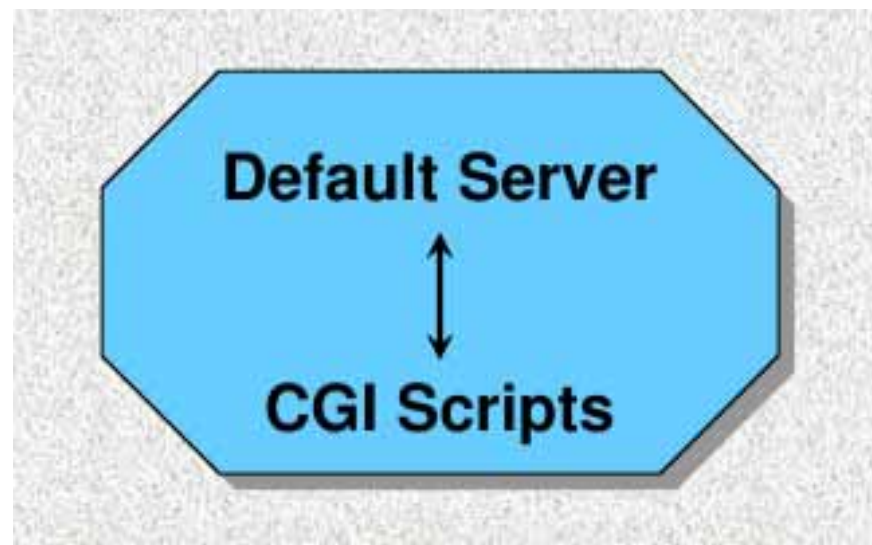
Secure CGIId: ejemplo





Antes de CGIId

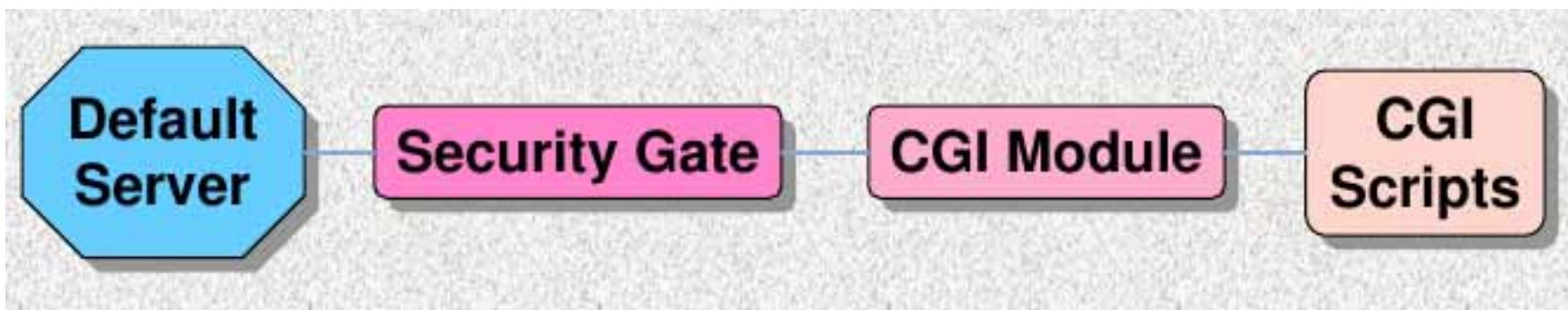
- Servidor y *scripts* en el mismo compartimento.
- El servidor se comunica directamente con los scripts.
- Hackea el servidor y hackea los scripts...





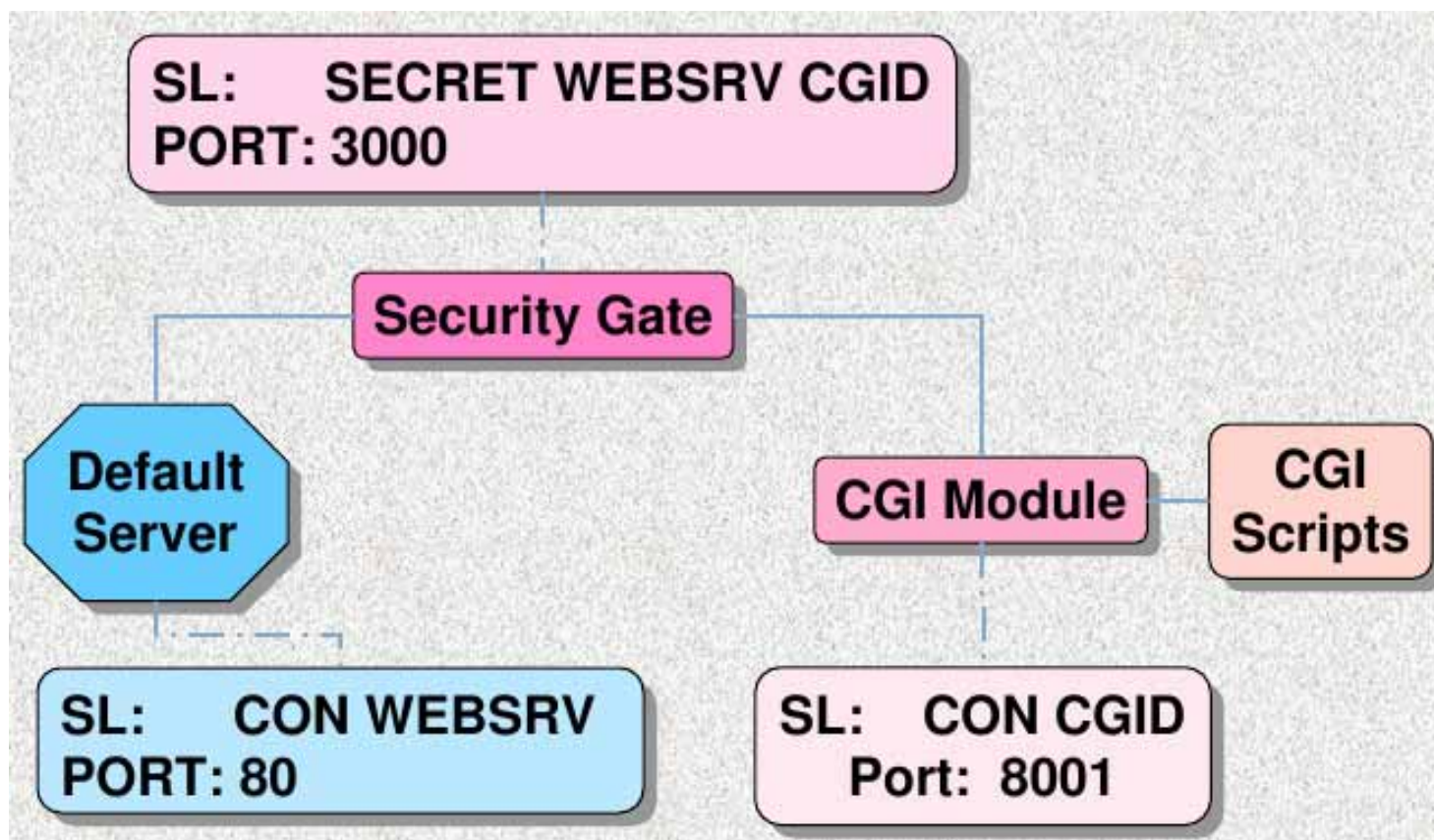
Después de CGId

- Servidor y scripts en distintos compartimentos.





Ejemplo CGIId





Ejemplo config. CGId

```
<cgid>  
HOST      localhost  
PORT      /PATH/sockets/8001  
DOCUMENT_ROOT /PATH/cgid  
PIDFILE    /PATH/log/cgid.pid  
LOGFILE    /PATH/log/cgid.log  
</cgid>
```



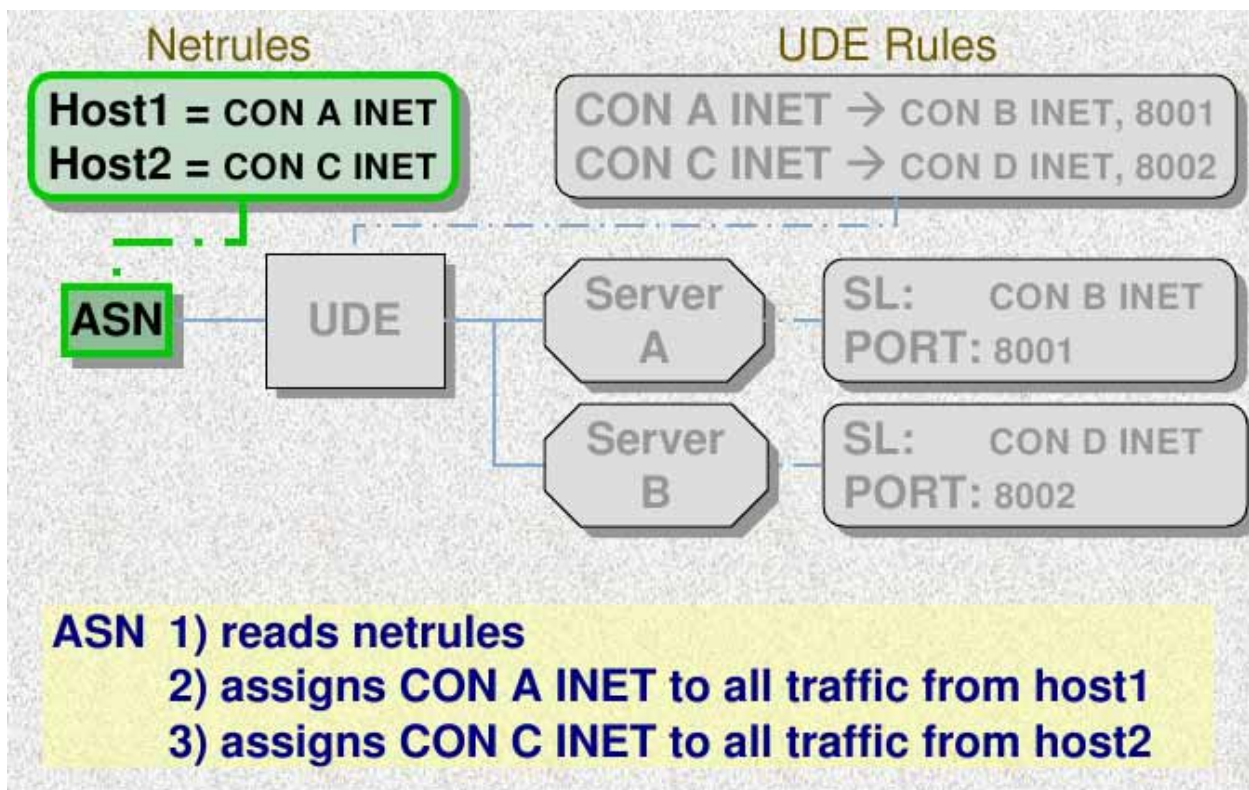
Secure Communications Enforcer (UDE)

- UDE –de Upgrade/Downgrade Enforcer-.
- Permite comunicación entre dos o más procesos con SL's disjuntas.
- Sólo permite comunicaciones “many-to-many”.
- Redirige el trafico en función de:
 - Reglas del fichero de configuración.
 - SL's de la petición (definido por la ASN).
 - URL de la petición.
 - Sesión (cookies,...)



Secure Communications Enforcer (UDE): ejemplo

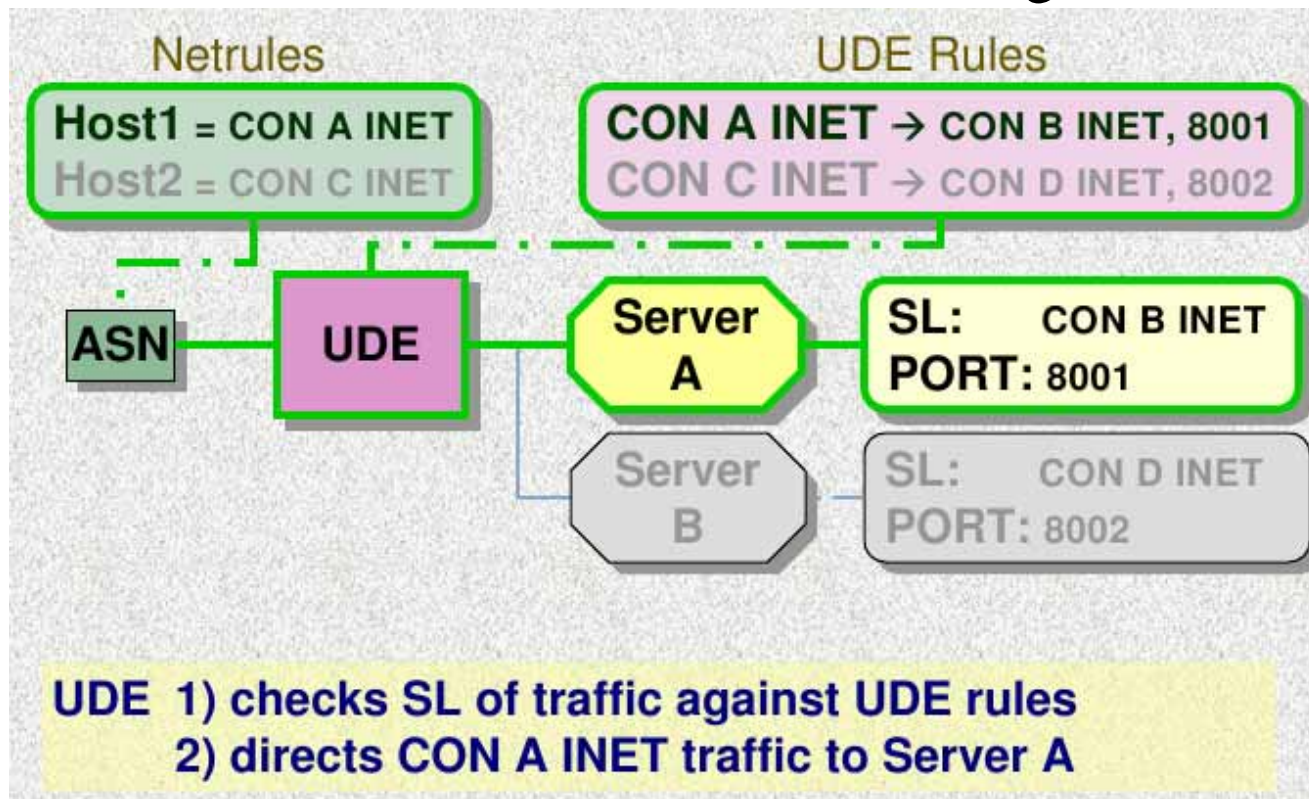
*ASN= Advanced Secure Networking





Secure Communications Enforcer (UDE): ejemplo

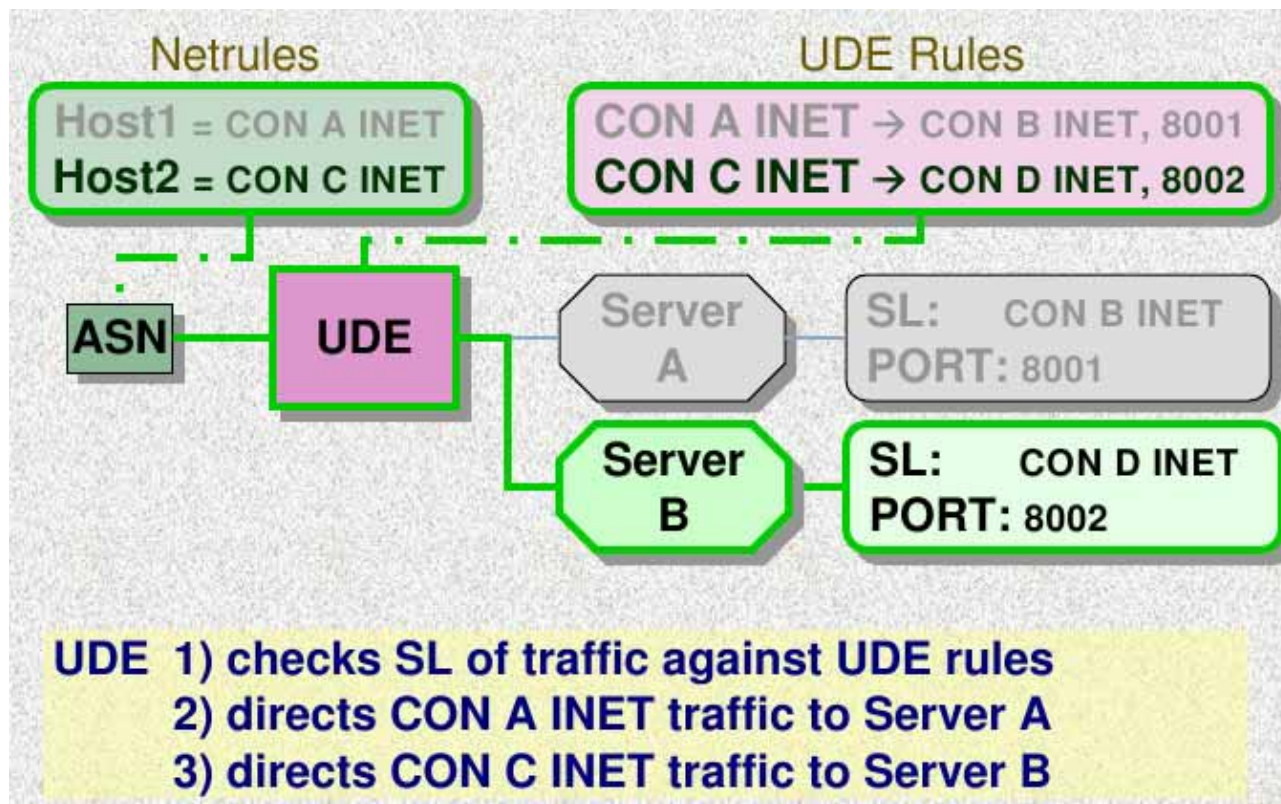
*ASN= Advanced Secure Networking





Secure Communications Enforcer (UDE): ejemplo

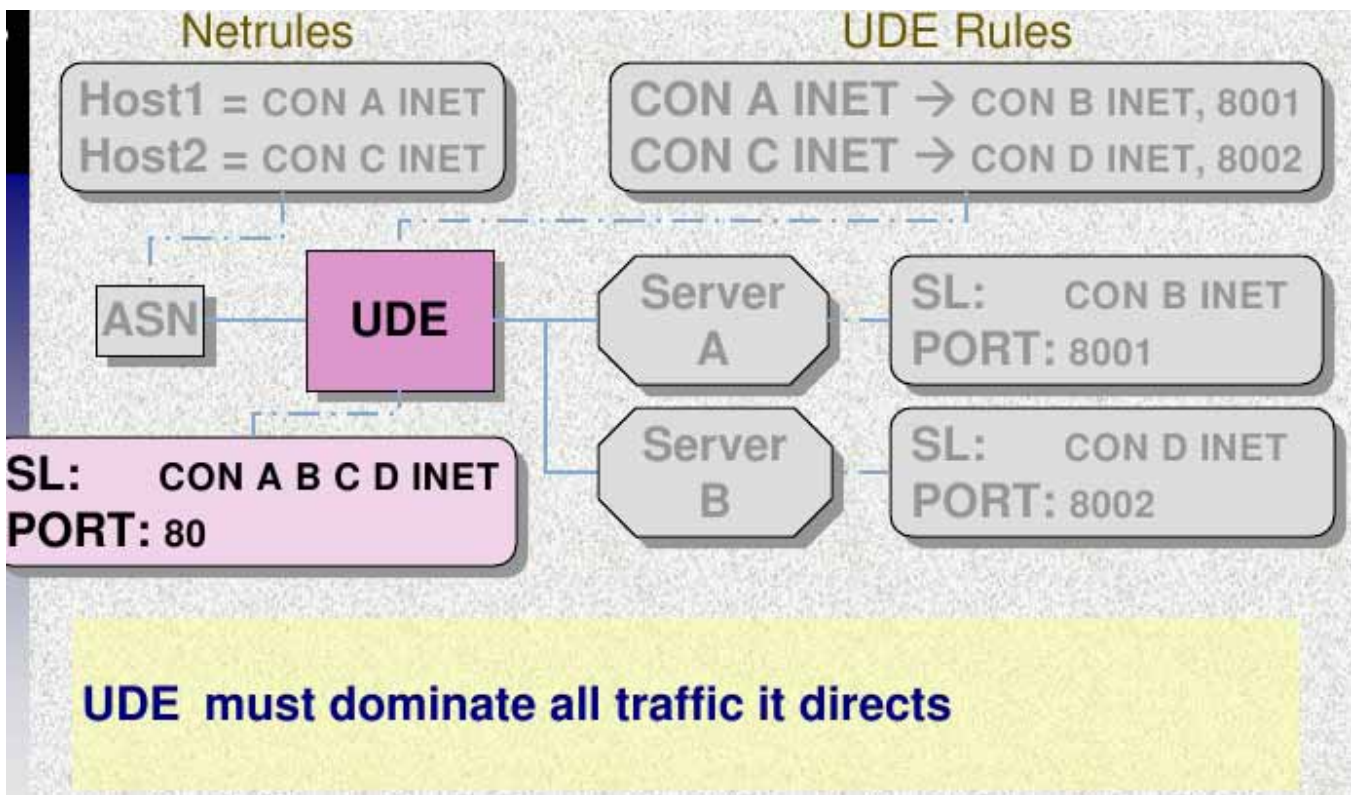
*ASN= Advanced Secure Networking





Secure Communications Enforcer (UDE): ejemplo

*ASN= Advanced Secure Networking





Secure Communications Enforcer (UDE) - Notas

- El UDE se suele situar en al frente de todo (puerto=80).
- La SL del UDE debe dominar las SL's de todo el *frontend* y el *backend*.
- UDE puede recibir información de todo el mundo, pero no puede modificar nada.

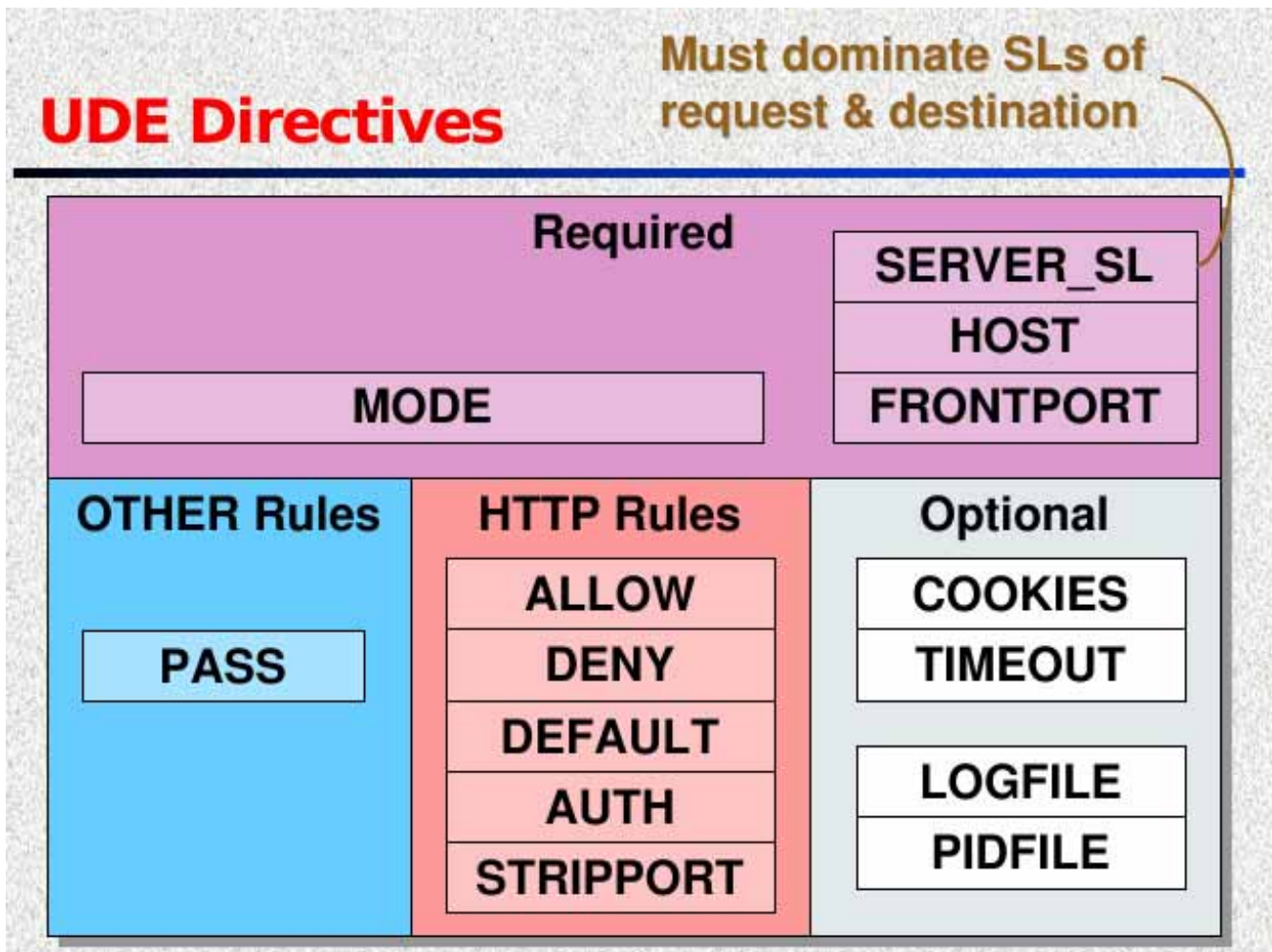


UDE - Reglas

- Determinan las decisiones de enrutado.
- Si la regla encaja, redirige petición.
- Si la regla no encaja, deniega.
- “Pass Rules”
 - Sólo en “Other mode”.
 - Redirige basado sólo en las SL's



UDE – Directivas de configuración





UDE – Ejemplo config

```
<UDENFORCER>  
SERVER_SL "TOP SECRET ALL"  
HOST      localhost  
FRONTPORT 80  
MODE      OTHER  
PASS      "CON A INET" localhost 8001  
          "CON B INET"  
PASS      "CON C INET" localhost 8002  
          "CON D INET"  
</UDENFORCER>
```



UDE – Ejemplo config

```
<UDENFORCER>  
SERVER_SL  "TOP SECRET ALL"  
FRONTPORT  80  
MODE       HTTP  
COOKIES     ON  
TIMEOUT     -20  
LOGFILE     /PATH/log/ude.log  
PIDFILE     /PATH/log/ude.pid  
#  
# continued...
```



UDE – Ejemplo config

```
# ...continued  
AUTH      localhost 9004 "CONF B INET"  
AUTH      localhost 9005 "TOP SECRET ALL"  
DEFAULT   "PUBLIC INET" localhost 9001  
          "CONFIDENTIAL A INET"  
  
ALLOW     / "UNCLASSIFIED A INET" localhost  
          9002 "CON A INET"  
  
DENY      /cgi-bin "UNCLASSIFIED A INET"  
STRIPPORT localhost 8001 localhost 127.0.0.1  
          www.mycompany.com  
  
</UDENFORCER>
```

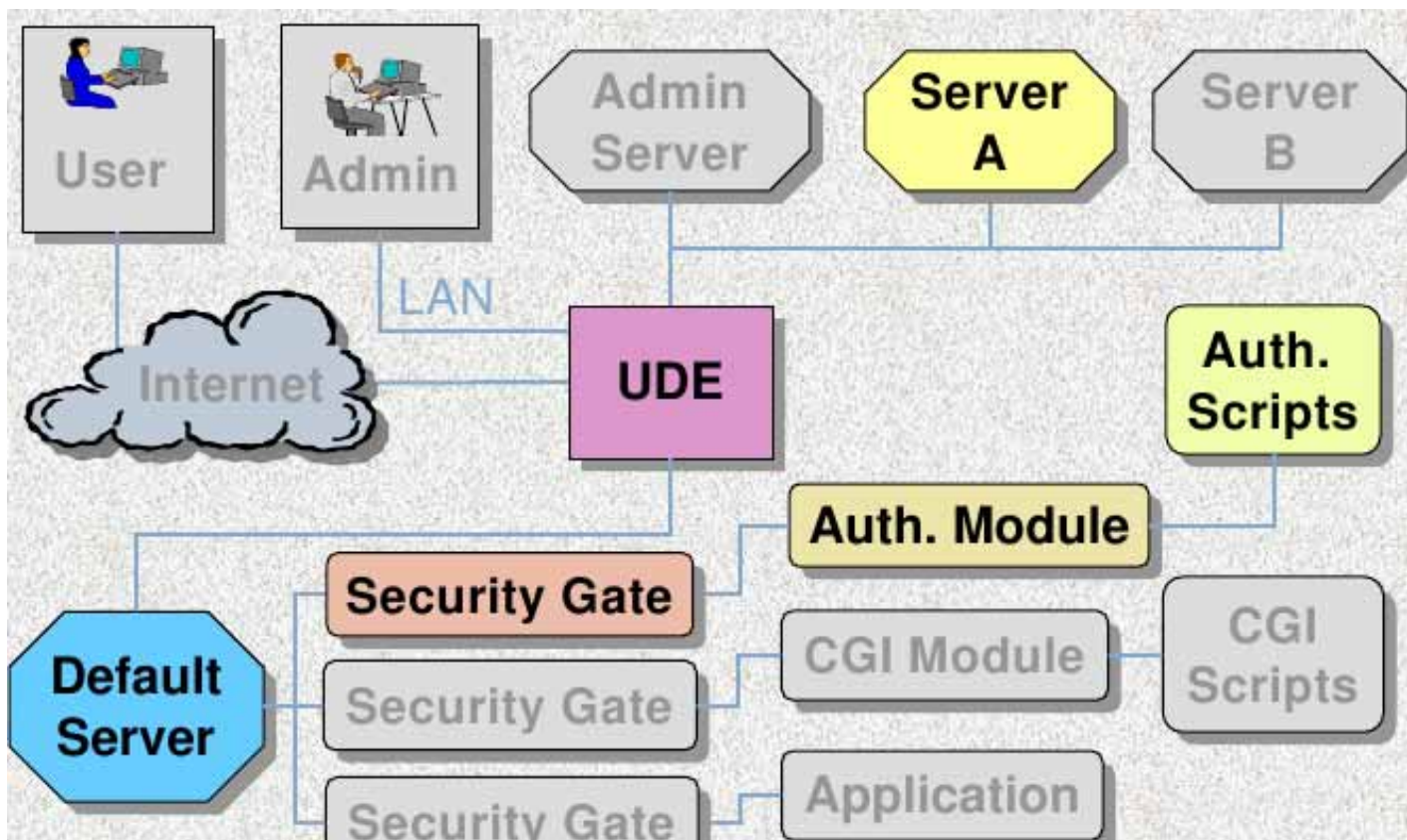


Secure Authentication Module

- Comunicación entre servidor y los programas de autenticación via *Security Gate*.
- Soporta CUALQUIER programa de autenticación (DNle, OTP,...)
- Si la autenticación tiene éxito, se setea cookie y se redirige la petición al sitio especificado por el UDE.

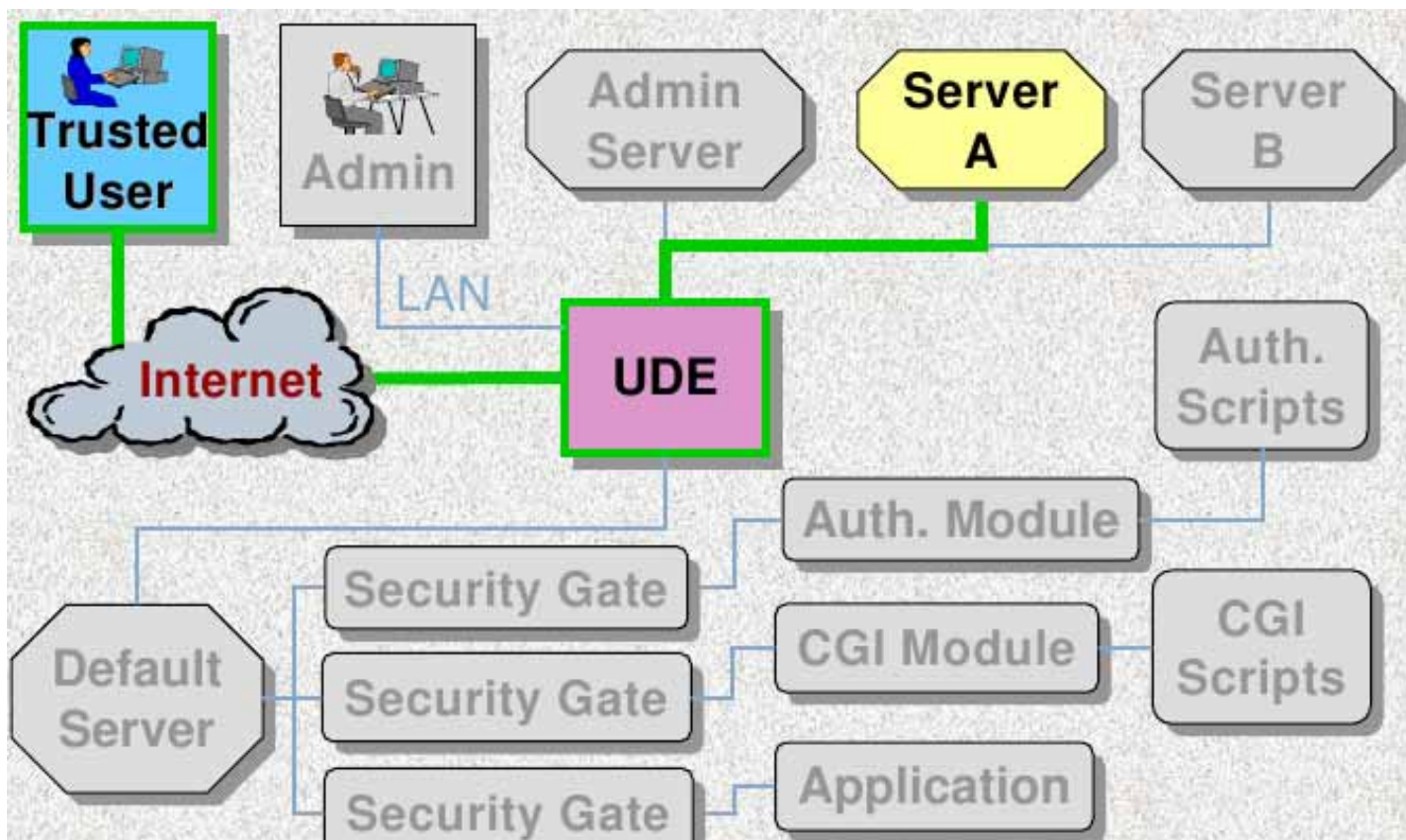


(I) Secure Authentication Module Ejemplo



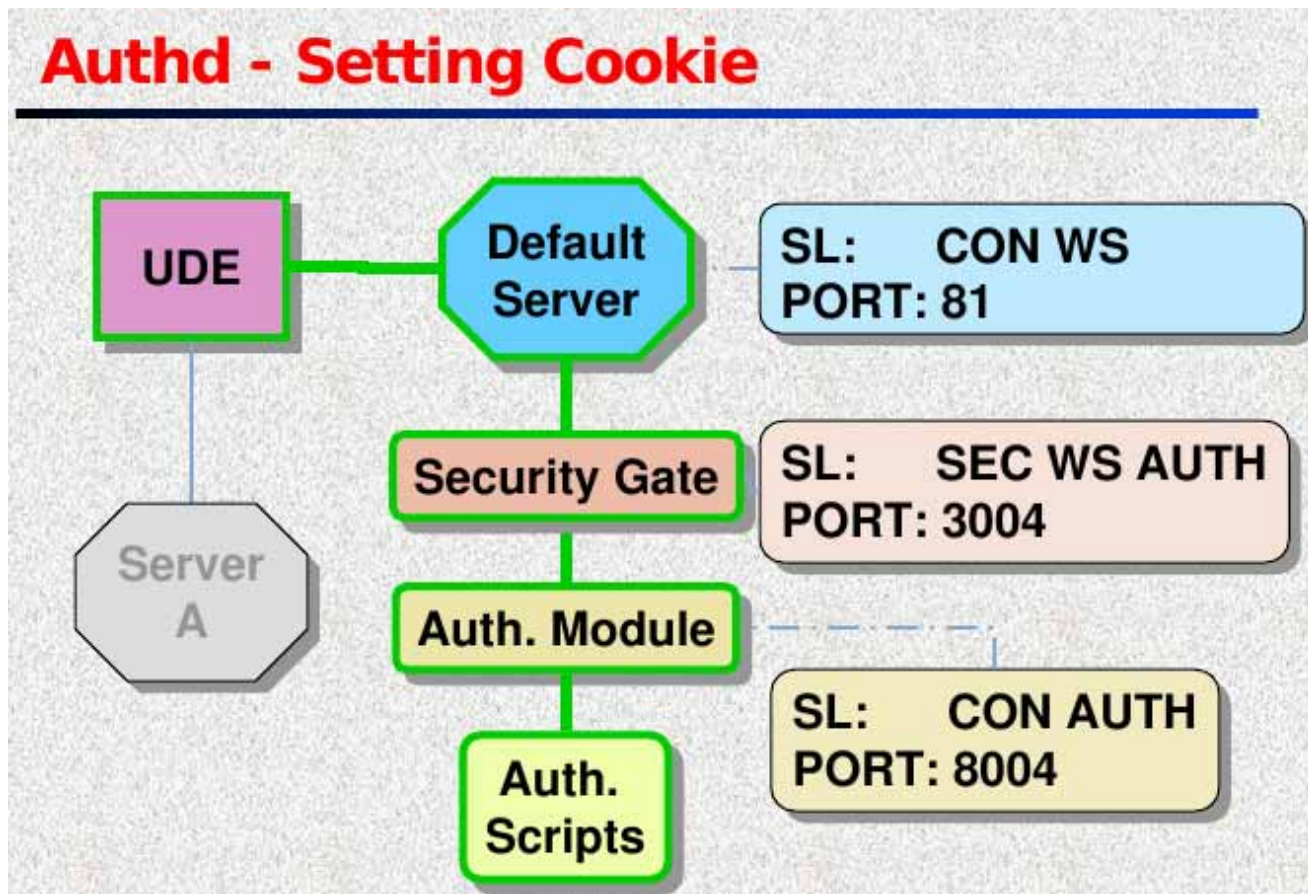


(II) Secure Authentication Module Ejemplo



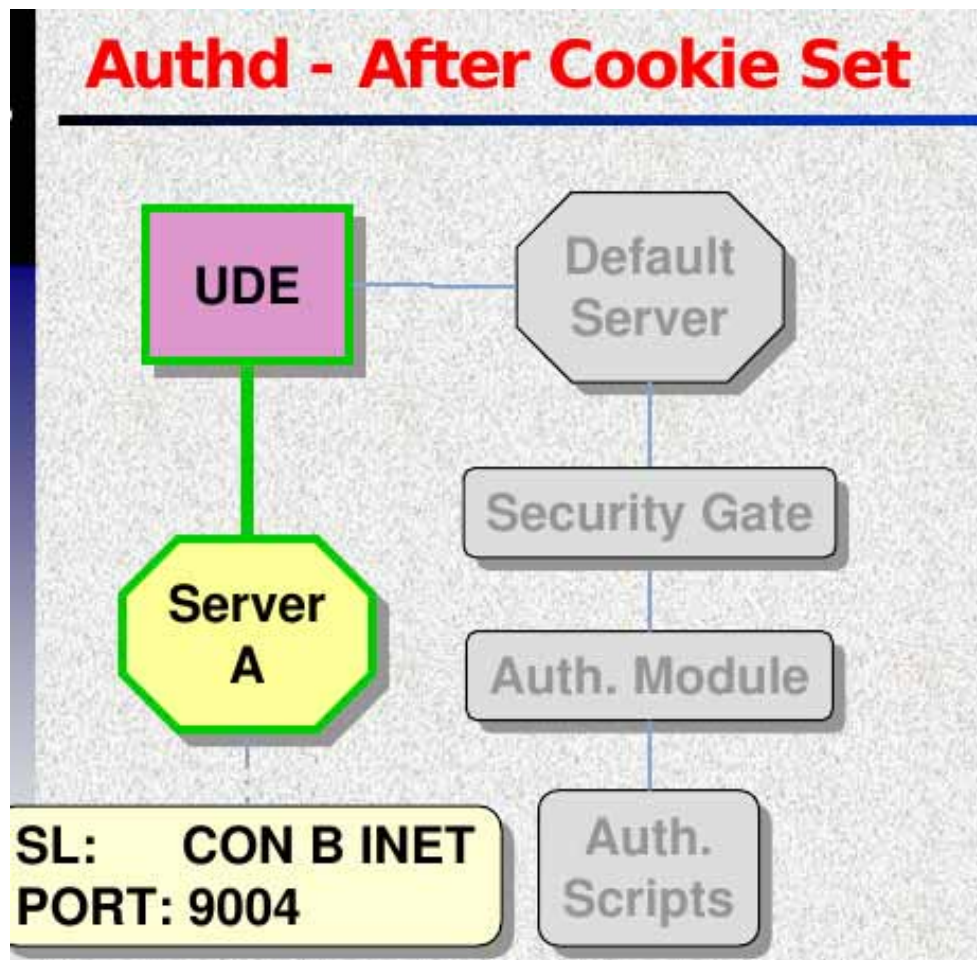


(III) Secure Authentication Module Ejemplo



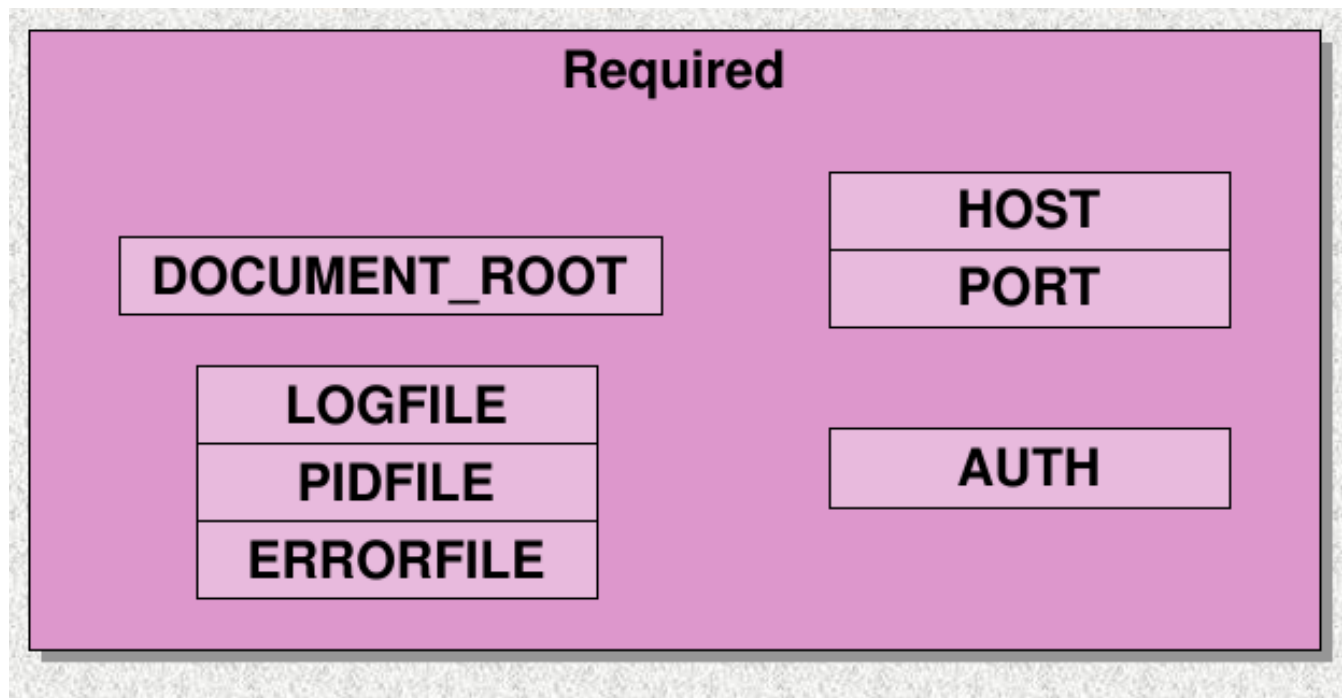


(IV) Secure Authentication Module Ejemplo





Secure Authentication Module -Directivas





Secure Authentication Module significado reglas

- Especifica el programa que debe ejecutar.
- Especifica el servidor al que redirigir después de una autenticación exitosa.
- Debe tener una regla que “encaje” en la configuración de UDE.



(I) Secure Authentication Module ejemplo config

```
<authd>
PORT      /PATH/sockets/authd.socket
HOST      localhost
DOCUMENT_ROOT  /PATH/authd
ERRORFILE  /PATH/bin/error.auth
PIDFILE    /PATH/log/authd.pid
LOGFILE    /PATH/log/authd.log
#
#  continued...
```



(II) Secure Authentication Module ejemplo config

```
# ...continued
#
AUTH      checkB.auth    localhost    9004
           "CONFIDENTIAL B INET"
           https://www.widgets.com
AUTH      checkTS.auth  localhost    9005
           "TOP SECRET ALL"
           https://www.widgets.com
</authd>
```



Ventajas de sistemas MLS vs Máquinas Virtuales

- Consume muchos menos recursos.
- Mecanismos integrados para controlar el flujo de información.
- Más barato.



Escenario 1: Máquinas Virtuales

- N° de máquinas = $2 \times N^{\circ}$ sujetos + $2 \times N^{\circ}$ Objetos (mínimo)
- Mantenimiento de muchas máquinas virtuales...



Escenario 2: MLS

- N° de máquinas = 1
- Mantenimiento de 1 máquina.



Día a día: escenario ejemplo

- Por ej.:
 - PÚBLICO, CLIENTES, DESARROLLO, GESTIÓN.
- 4 niveles de sensibilidad.
- No se desea flujo de información entre los 4 activos.



Día a día: actualizaciones

- Entorno convencional
4 máquinas físicas/virtuales mínimo = 4 sistemas operativos.
- MLS
1 máquina física = 1 sistema operativo



Día a día: buscar información

- Entorno convencional
busca en la máquina de GESTION, la de CLIENTES, la de DESARROLLO, la PÚBLICA...
- MLS
busca en 1 sistema



Día a día: transferir/compartir información

- Entorno convencional
A través de la red, lento, sin control.
- MLS
Localmente, rápido, controlado.



Día a día: redundancia

- Entorno convencional
 $4 \text{ sistemas} \times 2 = 8 \text{ sistemas}$
- MLS
 $1 \text{ sistema} \times 2 = 2 \text{ sistemas}$



Día a día: puntos de fallo hardware

- Entorno convencional
4 sistemas = 4 puntos de fallo hardware
- MLS
1 sistema = 1 punto de fallo hardware



Día a día: desarrollo web

- Entorno convencional

Si se desea homogeneidad obliga a mantener mismas versiones de BBDD, de intérpretes de lenguaje web, etc...

- MLS

Trivial: 1 versión de servidor web/BBDD + 1 versión de intérprete en un sólo sistema.



Día a día: gestión general

- Entorno convencional
4 sistemas = 4 contraseñas de root, 4 de ...
Relaciones de confianza...? Peligro!
- MLS
1 sistema = contraseñas estándar de un
UNIX + roles (3 o 4)



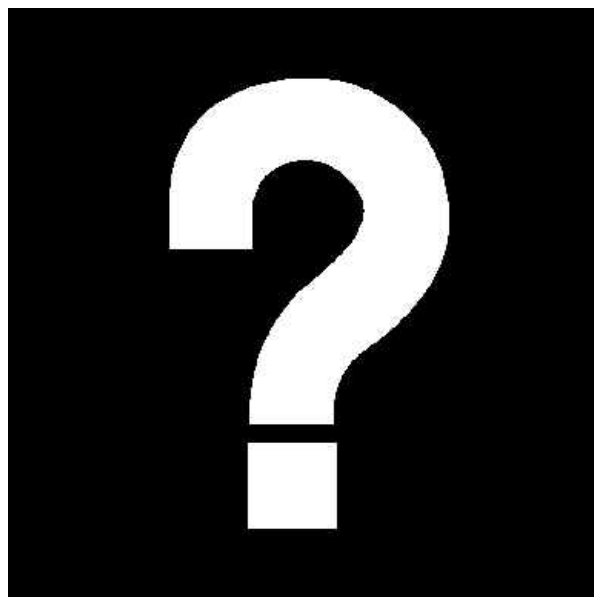
Día a día: si llega el fatídico día... FORENSICS

- Entorno convencional
4 sistemas = 4 análisis forenses
- MLS
1 sistema = 1 análisis forenses



MITOS sobre MLS

- MLS es solo para entornos críticos





MITOS sobre MLS

- **MLS es sólo para entornos críticos**

FALSO

- Distribuciones de sobremesa –Fedora- ya usan MLS.

*Aunque **no** la aprovechen 100%*

- Compartimentalizar el sistema NO implica información crítica.



MITOS sobre MLS

- MLS es sólo para corporaciones de gran poder adquisitivo.





MITOS sobre MLS

- MLS es sólo para corporaciones de gran poder adquisitivo

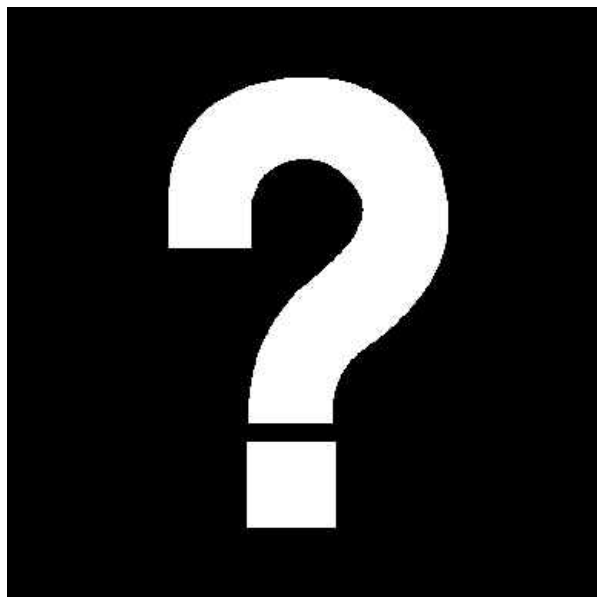
FALSO

- La virtualización o la diversificación de hardware y sistemas operativos requiere más recursos.
- Un sistema MLS permite ahorrar siempre (licencias, hardware, etc)



MITOS sobre MLS

- La gestión de un sistema MLS requiere de un gran *know-how*.





MITOS sobre MLS

- La gestión de un sistema MLS requiere de un gran know-how

FALSO

- Lo requiere su diseño e implementación, **NO** su posterior gestión
- Por otro lado, siempre es mas simple administrar 1 sistema que N servidores, N aplicaciones, N....



(I) Conclusiones

- Los sistemas MLS son muy útiles para clasificar y compartimentalizar aplicaciones y activos de información.
- Pueden apoyarse en MAC, RBAC, DBAC, etc
- Permiten encapsular servicios web con elegancia y robustez.



(II) Conclusiones

- Las principales diferencias entre los sistemas MLS es la implementación particular que cada fabricante hace de los estándares.
- Existen soluciones GPL y soluciones de pago
- A día de hoy, ninguna solución MLS es trivial. Requiere formación, pues rompe con los conceptos clásicos de UNIX.



Gracias

- Turno de preguntas -

Seguridad Multinivel en servidores web

OWASP Spain 2008