

Sistemas Operativos
2º Curso
Doble Grado en Ingeniería Informática y Matemáticas

Tema 6:

Mecanismos de seguridad



José Antonio Gómez Hernández, 2020.

Contenidos


- ▷ Objetivos de protección y seguridad.
- ▷ Autenticación.
- ▷ Mecanismos de autorización.



Bibliografía

- ▷ W. Stallings, *Operating Systems: Internals and Design Principles, 8th Edition*, Pearson, 2014.
 - Capitulo 14, Apartado 14.1 “Conceptos de Seguridad del Computador”, 14.2 “Amenazas, ataques y bienes”, y 14.4 “Resumen de software malicioso”
 - Capítulo 15, Apartados 15.1 y 15.2 sobre “Autenticación” y “Control de acceso”, 15.3 “Detección de intrusiones” y 15.4 “Defensa frente al malware”.





1.

Objetivos de protección y seguridad

Definición de conceptos básicos sobre seguridad

Conceptos básicos

- ▷ Seguridad y protección tratan de controlar el uso no autorizado y acceso a recursos hardware/software de un computador.
- ▷ Violaciones potenciales de la seguridad:
 - Obtención no autorizada de información
 - Modificación no autorizada información
 - Denegación no autorizada de servicio a un usuario autorizado, por medios:
 - Internos (caída del sistema, sobrecarga del sistema, etc.)
 - Externos (corte eléctrico, fuego, etc,).



Seguridad interna y externa

- ▷ La seguridad de un sistema de computador se puede dividir en:
 - *Externa o física* - trata con la regulación de acceso al sistema, que incluye la máquina física (hardware, discos, cintas, fuente de alimentación, aire acondicionado, terminales, consola, etc). Las cuestiones relativa a ella son más simples y de naturaleza administrativa, p. ej. un vigilante, códigos secretos en la puerta, sistema anti-incendios, etc.



Seguridad interna/externa (ii)

- *Interna* - trata con el acceso y uso a la información y hardware almacenado en el sistema. Esta es más compleja y sutil, y será el núcleo del tema.
- ▷ Entre la seguridad externa y la interna esta la **autenticación**, mediante la cual un usuario se identifica para poder acceder al hardware y software del sistema.



Seguridad y protección

- ▷ La definición de estos términos no es exacta y depende de quién la utilice.
- ▷ Para unos:
 - Protección son los mecanismo.
 - Seguridad son las políticas
- ▷ Para otros:
 - Seguridad es un termino más amplio que protección, y trata más de aspectos externos al sistema; mientras que protección trata con los aspectos de la seguridad que conciernen al SO.



Objetivos o propiedades

- ▷ **Integridad** – modificación sólo por personal autorizado.
 - ▷ **Confidencialidad** – acceso sólo por personal autorizado.
 - ▷ **Disponibilidad** – acceso continuo del personal autorizado.
 - ▷ **No repudiación** – incapacidad para rechazar propiedad
 - ▷ **Autenticidad** – verificación de la identidad
 - ▷ ...
- Triada
CIA



Principios de diseño de sistemas seguros

- ▷ **Economía** - un mecanismo de protección debe ser económico de desarrollar y usar. Su inclusión en el sistema no debe provocar costo o sobrecarga sustancial.
- ▷ **Mediación completa** - el diseño de un sistema completamente seguro requiere que cada solicitud de acceso a un objeto deba ser comprobada para su autorización.
- ▷ **Diseño abierto** - un mecanismo de protección no debe dejar su integridad en el desconocimiento de posibles atacantes relativo al propio mecanismo de protección.



Principios de diseño (ii)

- ▷ **Separación de privilegios** - un mecanismo de protección que requiere dos llaves para obtener acceso a un objeto protegido es más robusto y flexible que uno que utiliza una.
- ▷ **Menor privilegio** - un sujeto debe tener los derechos de acceso mínimos que son suficientes para completar su tarea. También se conoce este principio como “need-to-know”.



Principios de diseño (y iii)

- ▷ **Mecanismo del mínimo común** - la porción de un mecanismo común a más de un usuario debe ser minimizada. Cualquier acoplamiento entre usuarios representa un camino de información potencial entre ellos y por tanto un ataque potencial a su seguridad.
- ▷ **Aceptabilidad** - un mecanismo de protección debe ser fácil de usar, sino no será aceptado y por tanto, no se utilizará por los usuarios.
- ▷ **Fallo seguro por defecto** - el caso por defecto debe significar la falta de acceso, dado que es lo más seguro.

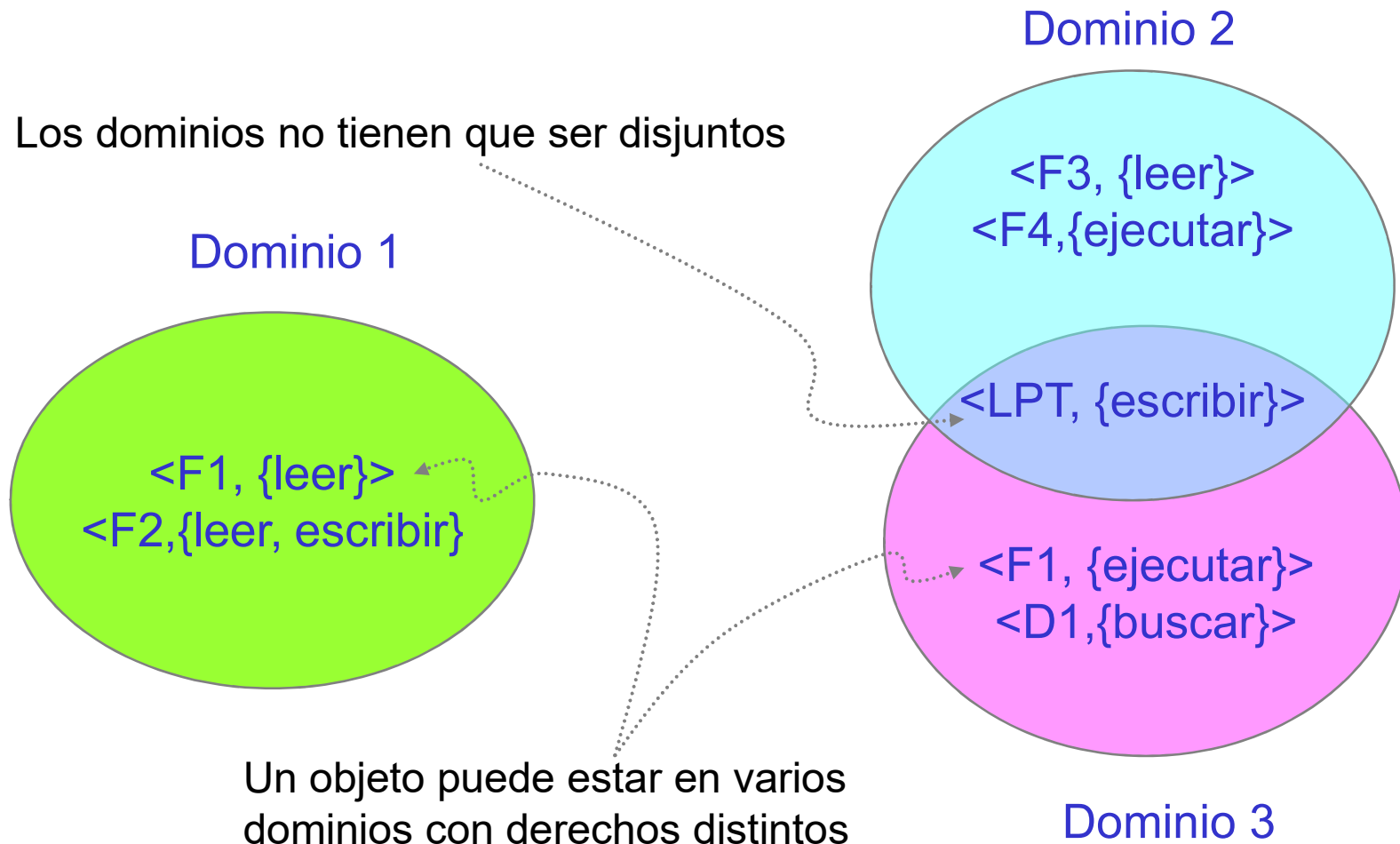


Dominios de protección

- ▷ Podemos ver un computador como una colección de procesos y objetos (recursos).
- ▷ Los objetos, tienen un nombre único que los diferencia de otros, y son accedidos a través de un conjunto definido de operaciones - son esencialmente un TAD.
- ▷ La habilidad para ejecutar una operación sobre un objeto es un derecho de acceso.
- ▷ Un **dominio de protección** define un conjunto de objetos y una colección de derechos de acceso `<nombre_objeto, conjunto_derechos>`.
- ▷ Un proceso se ejecuta en un dominio de protección.



Dominios de protección: ejemplo



Amenazas y ataques

- ▷ **Amenaza** (*threat*) es el conjunto de circunstancias que tienen el potencial de causar daños o pérdidas. Implica una motivación, un actor, y posibilidad de daño.
- ▷ **Vulnerabilidad** es una debilidad en el sistema (procedimiento, diseño o implementación) que se puede aprovechar para causar daño.
- ▷ Un humano (*intruder* o *insider*) o sistema que aprovecha una vulnerabilidad realiza un **ataque** (*attack*) al sistema.



Vector de ataque

- ▷ Definimos como **vector** (o superficie) **de ataque** al método que utiliza una amenaza para atacar a un sistema (la red, el control de acceso, etc.).
- ▷ **Ataque/amenaza del día cero** es el que se produce por una vulnerabilidad desconocida y que los desarrolladores han tenido cero días para arreglarla.
- ▷ **Contramedida** (o control) son las acciones, dispositivos, procedimientos o técnicas que eliminan o reducen una vulnerabilidad.



Software malicioso (*malware*)

- ▷ Denominamos ***malware*** al software diseñado para provocar daño o usar los recursos de un computador objetivo.
- ▷ Existe una gran variedad:
 - Puertas traseras (*backdoors*)
 - Virus
 - Troyanos
 - *Ransomware*
 - *Exploits*
 - *Rookits*
 - . . .





2.

Autenticación

¿Cómo identificamos a los usuarios?

Autenticación

- ▷ **Autenticación** es el proceso de identificar a un alguien, cuenta con dos fases (RFC 4949):
 - **Identificación** – presentar un identificador al sistema de seguridad.
 - **Verificación** – presentar o generar información de autenticación que corrobora la unión entre la entidad y el identificador.



Control de acceso

- ▷ **Control de acceso** es la colección de mecanismos que permiten gestionar un sistema para ejercitar una influencia directa o restrictiva sobre el comportamiento, uso y contenido de un sistema.
- ▷ Permite específicamente gestionar lo que un usuario puede hacer, a que recursos puede acceder, y que operaciones puede realizar en el sistema. Reduce la superficie de ataque al limitar las interacciones del atacante con el sistema.

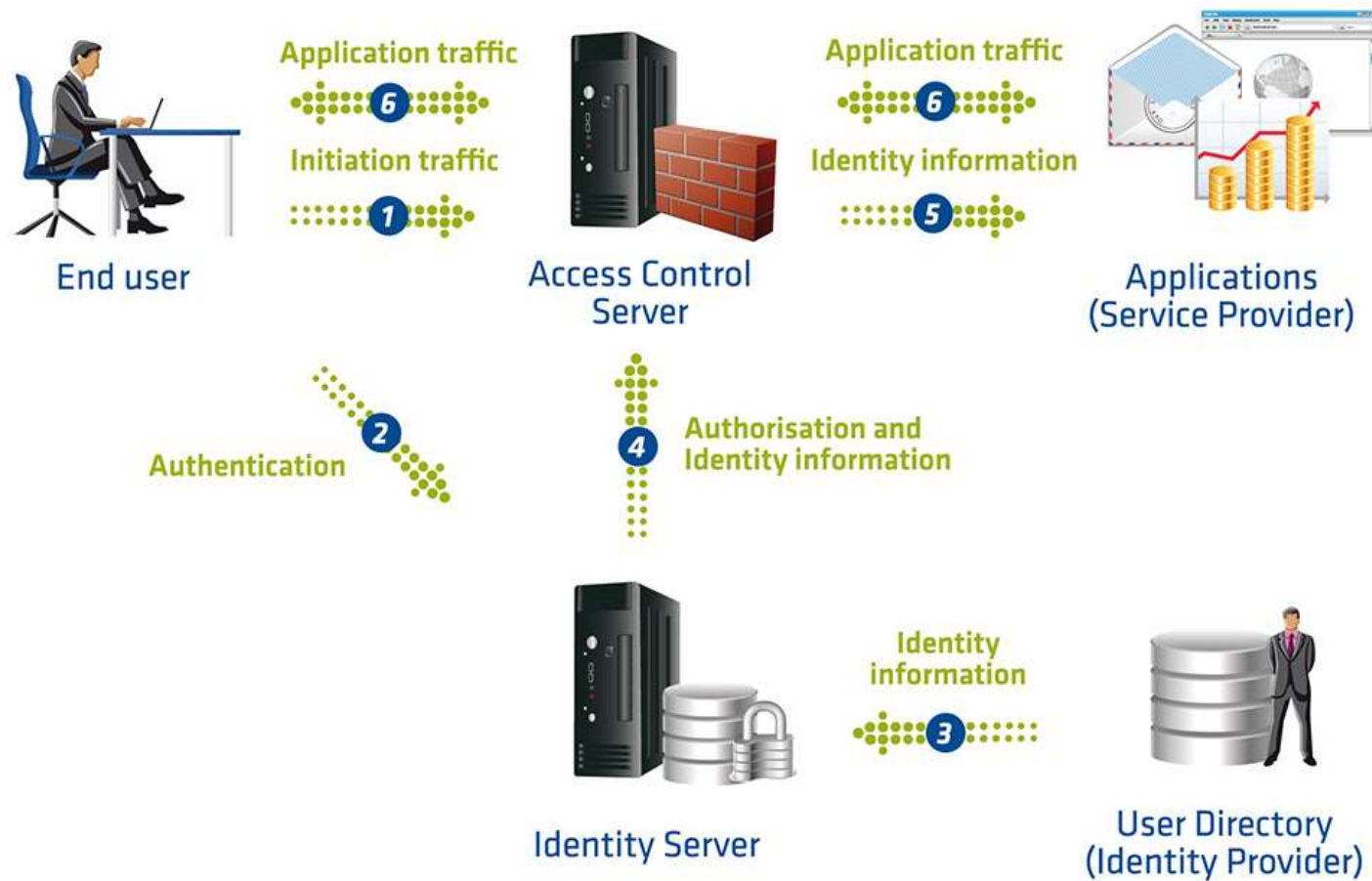


Identificación

- ▷ Gestión de identidades: termino amplio para incluir el uso de diferentes productos para identificar, autenticar y autorizar a los usuarios mediante medios automatizados.
- ▷ **Identificación** – método de establecer la identidad del sujeto (usuario, programa o proceso)
 - Utilización de nombres de usuario y otra información pública.
 - Conocer los requisitos de identificación de componente.



Relación de conceptos



Explicación de la Figura

- 1) La petición inicial se redirige al servicio responsable de la identificación.
- 2) El servicio de identificación se activa para autenticar al usuario.
- 3) El servicio de identificación realiza la autenticación (en sistemas en red puede mantener en caché la información de identidad) en base a las reglas establecidas por el administrador de identidades.
- 4) El servicio de identificación transmite la información de identidad y autorización al servicio de control de acceso.
- 5) El servicio de control de acceso transmite la información de identificación que necesita cada sistema. La decisión de autorización es realizada por el servidor de control de acceso o la aplicación basándose en la información de identificación.
- 6) El usuario puede acceder a las aplicaciones autorizadas mientras el servicio de control de acceso actúa como proxy. El proceso es transparente al usuario.



Autenticación: medios

- ▷ El sistema de protección depende de la habilidad para identificar a los programas/procesos de cada usuario.
- ▷ La autenticación descansa en alguna combinación de los siguientes elementos:
 - Posesión de objeto (llave o tarjeta).
 - Conocimiento del usuario (login/password).
 - Atributos del usuario (huella dactilar, firma, patrón de retina, etc.).
- ▷ **Autenticación multifactor (MFA)** – uso de dos o más factores básicos usados conjuntamente. La más conocida la **autenticación en dos fases (2FA)**.





3.

Mecanismo de autorización

¿Cómo autorizar el acceso de los usuarios a los recursos?

Modelo de la Matriz de Acceso

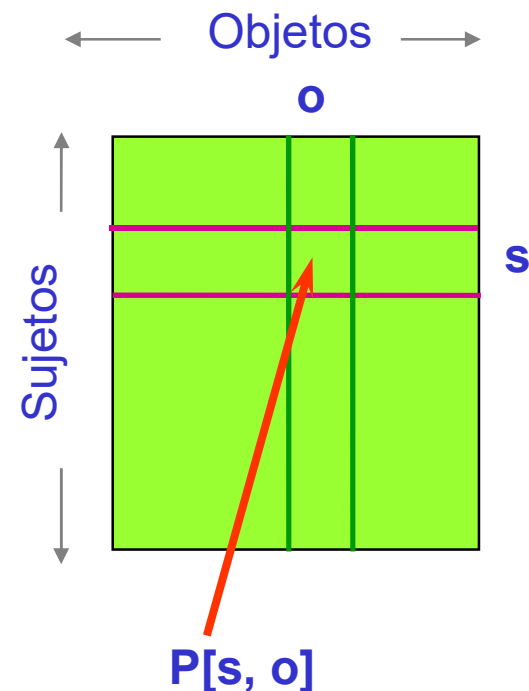
▷ Consta de tres componentes:

- **Objetos** - un conjunto finito de entidades, O , a las cuales debemos controlar el acceso. P. ej. un archivo.
- **Sujetos** - un conjunto finito de entidades, S , que acceden a los objetos actuales. P.ej, un proceso. ¡ Ojo !: $S \square O$.
- **Derechos genéricos** - un conjunto finito de derechos genéricos, $R = \{r_1, r_2, r_3, \dots, r_n\}$, da varios derechos de acceso que los sujetos pueden tener sobre objetos. P. ej., lectura, escritura, etc.



Estado de protección

- ▷ El estado de protección de un sistema esta representado por la tripleta (S, O, P) donde S es el conjunto de sujetos, O el conjunto de objetos, y P la matriz de acceso con una fila por sujeto y una columna por objeto
- ▷ La matriz de acceso P es en sí misma un objeto protegido.
- ▷ $R \supset P[s,o]$ =derechos de acceso que un sujeto s tiene sobre un objetos o.



Ejemplo de matriz de acceso

	S ₁	S ₂	S ₃	F ₁	F ₂	D ₁	D ₂
S ₁	Control	Propiedad Bloqueo Desbloqueo	Propiedad Control	Leer Escribir	Leer Escribir	Buscar	Propiedad
S ₂	Bloquear Desbloquear	Control		Propiedad	Modificar	Propiedad	Buscar
S ₃			Control	Borrar	Propiedad Ejecutar		



Imponer política de seguridad

- ▷ Se impone una política de seguridad validando cada acceso del usuario con los derechos de acceso apropiados. Se realiza por un **monitor** de la forma:
 - 1) Sujeto solicita un acceso α a un objeto o .
 - 2) El sistema de protección presenta la triple (s, α, o) al monitor de o .
 - 3) El monitor mira los derechos de acceso: si $\alpha \in \square P[s, o]$, permite el acceso; en caso contrario, no.
- ▷ El modelo de la matriz de acceso es muy popular debido a su simplicidad, estructura elegante, y docilidad de implementación.



Implementaciones de la matriz de acceso

- ▷ La matriz de acceso no es muy densa => la implementación directa es ineficiente en almacenamiento.
- ▷ Ganamos eficiencia si la descomponemos en:
 - **Capacidades** - asignamos los derechos de acceso (fila) a sus respectivos objetos. Una fila puede colapsarse borrando las entradas nulas.
 - **Listas de control de acceso (ACL)** - asignamos a cada objeto los derechos de acceso de varios sujetos (columna) al objeto. Podemos borrar entradas nulas.
 - Existe un tercer método, denominado **lock-key**, que es una combinación de ambos enfoques.

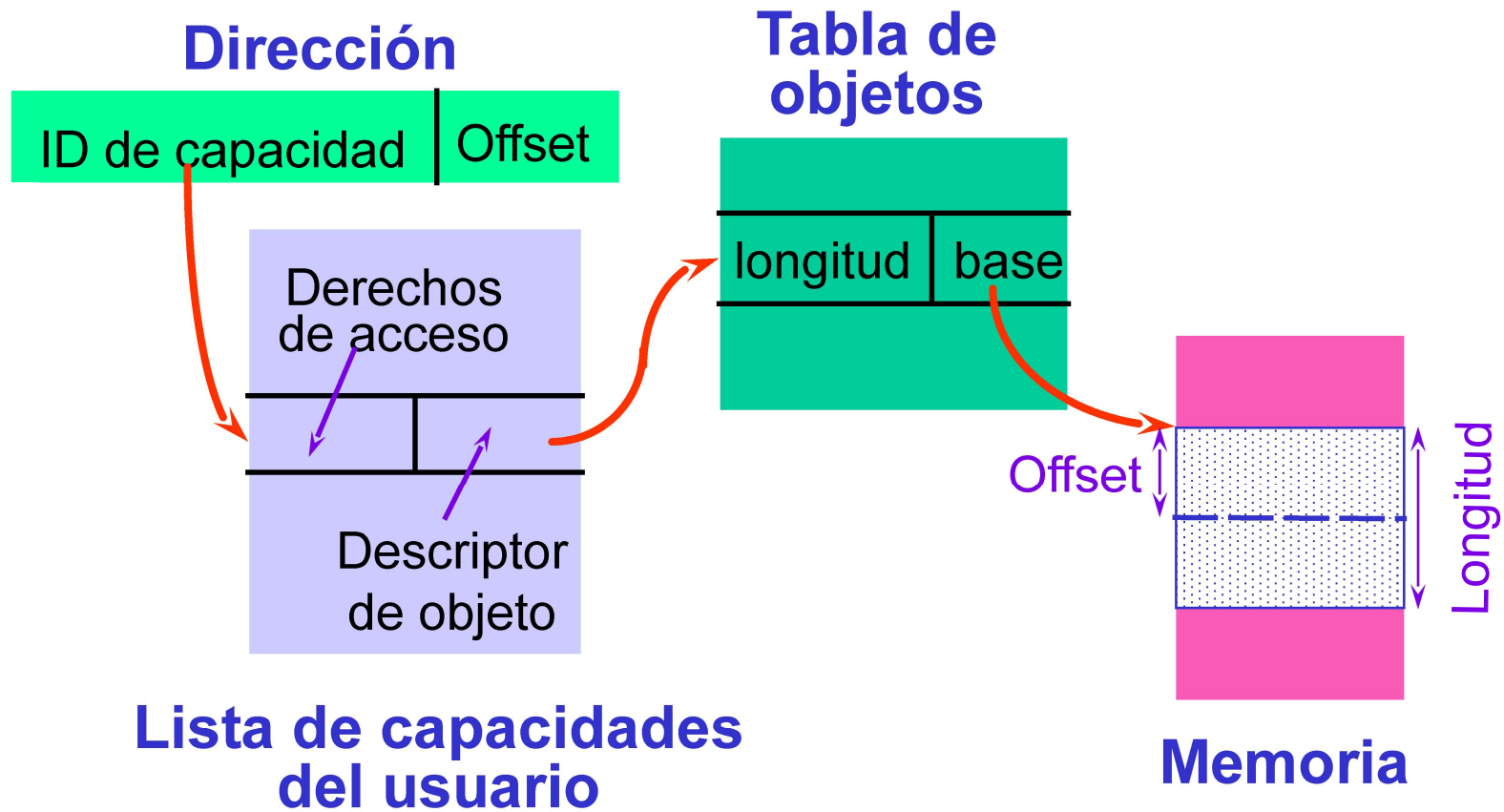


Capacidades (capabilities)

- ▷ Cada sujeto s tiene asignada la lista de tuplas $(o, P[s, o])$ para todos los objetos o a los que puede acceder. Cada tupla se denomina capacidad. Si s tiene una capacidad $(o, P[s, o])$, entonces está autorizado a acceder a o como especifica $P[s, o]$.
- ▷ Una capacidad tiene dos campos:
 - Un identificador o descriptor de objeto
 - Lista de derechos de acceso

Descriptor de objeto	Derechos de acceso
descriptor_objeto_i	lectura, ejecución, etc.

Direccionamiento con capacidades



Direccionamiento basado en capacidades: características

- ▷ Las dos características más relevantes son:
 - Reubicabilidad –podemos reubicar un objeto en memoria sin tener que cambiar las capacidades que lo referencia (solo cambiar la base en la tabla de objetos)
 - Compartición - varios programas pueden compartir un objeto (programa/datos) con diferentes nombres (descriptores de objetos)
- ▷ Observación: reubicabilidad y compartición se consiguen con la indirección introducida con la tabla de objetos. Pero esto, ¡ya lo hemos visto otras veces!



Capacidades: implementación

- ▷ Como mecanismo de direccionamiento, las capacidades pueden estar empotrada en los programas y estructuras de datos del usuario.
- ▷ Sin embargo, para mantener la capacidades libres de falsificación, un usuario no debe ser capaz de acceder (leer, modificar, o construir) a una capacidad.
- ▷ Tenemos que establecer métodos para implementarlas de forma segura.



Capacidades: implementación

(ii)

- ▷ Dos formas de implementar capacidades:
- **Enfoque etiquetado** - se añaden un(os) bit(s) a cada palabra de memoria y registros; estos están activos si la información es una capacidad. Existen instrucciones privilegiadas para manipularlos.
 - **Enfoque particionado** - capacidades y datos ordinarios se almacenan separadamente. Cada objeto tiene dos segmentos, también el procesador tiene dos juegos de registros. Los usuarios no pueden manipular los segmentos y registros que contienen capacidades.



Capacidades: ventajas

- ▷ **Eficiencia** para comprobar la validez de acceso; una acceso es implícitamente válido si se tiene una capacidad.
- ▷ **Simplicidad** debido a la correspondencia natural entre la propiedades estructurales de la capacidad y las propiedades semánticas de las variables de direccionamiento.
- ▷ **Flexibilidad** dado que permiten al usuario definir ciertos parámetros. P. ej. un usuario puede definir cual de sus direcciones contiene capacidades.



Capacidades: desventajas

- ▷ **Control de propagación** - ciertas aplicaciones necesitan por seguridad y contabilidad controlar la copia de capacidades de un usuario a otro. Por ello, se introduce un bit de copia o contador de profundidad.
- ▷ **Revisión** - es difícil conocer los sujetos que tienen acceso a un objeto
- ▷ La **revocación**, o retirada, de derechos es difícil de implementar.
- ▷ Es necesario un **mecanismo de recolección** de basura para borrar objetos no referenciados por una capacidad, por ejemplo, tener un creador de objetos.



Listas de Control de Acceso

- ▷ Cada objeto o tiene asociada una lista de pares $(s, P[s, o])$ para los sujetos s con acceso al él.
- ▷ Una solicitud de acceso α de un sujeto s a un objeto o , se realiza de la siguiente manera:
 - El sistema busca el ACL de o en busca de una entrada $(s, \square \Phi)$ para el sujeto s .
 - Si existe entrada $(s, \square \square \Phi)$ para s , se comprueba si la solicitud realizada esta permitida, es decir, $\alpha \in \Phi$
 - Si esta permitido el acceso, se ejecuta la solicitud. En caso contrario se genera un error.



Características de los ACLs

- ▷ El principal inconveniente de las ACL's es su **eficiencia** pobre de ejecución, ya que se realiza la búsqueda en la lista para cada acceso a un objeto protegido.
- ▷ Ventajas:
 - La **revocación** de derechos a un sujeto es simple, rápida y eficiente. Se realiza eliminando al sujeto de la lista.
 - Fácil **revisión** de un acceso (examinar la lista de un objeto). Sin embargo, es difícil determinar a que objetos tiene acceso un sujeto.



ACLs: implementación

- ▷ Dos son los puntos más importantes:
 - Eficiencia de ejecución - Solución: almacenar en un **registro sombra** los derechos de acceso de s extraídos de la lista en el primer acceso a o; a partir de ahora, actúa como una capacidad. Implicaciones negativas en la revocación de derechos.
 - Eficiencia de almacenamiento - ACL es grande por el nº de usuarios y tipos de acceso. Los usuarios se puede resumir estableciendo **grupos de protección**. Los tipos de acceso, se simplifican limitándolos y asignando a cada uno un bit de un vector.



ACLs: propagación de derechos

- ▷ Dos métodos de control de la propagación:
 - **Autocontrol** - el propietario del objeto tiene un derecho especial de acceso mediante el que puede cambiar el ACL. Desventaja: el control esta centralizado en un proceso, el creador del objeto.
 - **Control jerárquico** - al crear un objeto, su propietario especifica los procesos con derechos para modificar el ACL del objeto. Los procesos se agrupan en una jerarquía y un proceso puede modificar el ACL asociado con los procesos pertenecientes a ella.



Modelos de control de acceso

- ▷ En el diseño y construcción del SO se debe elegir el modelo de control de acceso a usar:
 - *Control de Acceso Discrecional (DAC)* – es discrecional en el sentido un sujeto puede pasar algunos de sus permisos sobre un objeto a cualquier otro sujeto.
 - *Control de Acceso Obligatorio (MAC)* – el SO restringe la habilidad de un sujeto de realizar ciertas operaciones sobre un objeto.
 - Ejemplos: SELinux o AppArmor.
 - Otros modelos de CA.



Otros mecanismos de defensa

- ▷ Detección de intrusiones – *Firewall*, sistemas de detección basada en red (NIDS) o *hosts* (HIDS).
- ▷ Defensa frente al *malware* – antivirus (*antimalware* en general), buenas prácticas de navegación, ...
- ▷ Software seguro:
 - Actualizaciones de software (SO y aplicaciones).
 - Defensas en tiempo de compilación
 - Defensas en tiempo de ejecución. Ej. EMET en Windows.
 - Programación defensiva.

