Linking dinamico

Enlace Dinámico

El **enlace dinámico** permite generar un ejecutable sin incluir todo el código binario necesario para su ejecución. En su lugar, ciertas referencias externas pueden resolverse posteriormente, ya sea en el **tiempo de carga** o en el **tiempo de ejecución**. Esto significa que, al principio, el ejecutable contiene referencias sin resolver que serán gestionadas por el sistema en el momento adecuado.

Enlace Dinámico en Tiempo de Carga (Load-Time Dynamic Linking)

En este enfoque, el **loader** combina el **load module** con las bibliotecas dinámicas requeridas en el momento en que se carga el ejecutable en memoria. Cuando el programa hace referencia a un módulo externo, el loader lo busca, lo carga en memoria y actualiza la referencia dentro del ejecutable para que apunte correctamente a la dirección donde se ubicó el módulo.

Proceso del Enlace Dinámico en Tiempo de Carga:

1. Se carga en memoria el **load module**.
2. Se identifican referencias a módulos externos.
3. El **loader** busca los módulos externos requeridos.
4. Los módulos externos son cargados en memoria y se actualizan las referencias en el programa.

Ventajas:

- Permite actualizar módulos externos sin necesidad de recompilar el ejecutable principal.
- El sistema operativo puede cargar y compartir una única versión del módulo externo entre varios procesos.
- Facilita el desarrollo de bibliotecas de enlace dinámico (ejemplo: archivos **.so** en Unix/Linux).

Enlace Dinámico en Tiempo de Ejecución (Run-Time Dynamic Linking)

En este caso, el ejecutable contiene referencias a módulos externos que **no se resuelven en el momento de la carga**, sino que se gestionan en **tiempo de ejecución**. El sistema operativo reconoce estas referencias cuando el programa intenta utilizarlas y, en ese momento, busca la biblioteca dinámica correspondiente, la carga en memoria y ajusta los enlaces.

Proceso del Enlace Dinámico en Tiempo de Ejecución:

1. Se ejecuta el programa con referencias a módulos externos sin resolver.
2. Cuando se invoca una función o recurso externo, el sistema operativo detecta la referencia.
3. Se busca el módulo requerido en una biblioteca dinámica.

4. El módulo se carga en memoria y se actualizan las referencias en el programa.

Ventajas:

- Reduce el uso de memoria, ya que los módulos externos solo se cargan cuando realmente se necesitan (ejemplo: **DLLs en Windows**).
- Permite la carga y descarga dinámica de funcionalidades, optimizando el rendimiento del sistema.