Dynamic Linked Library (.dll) e Shared Object (.so)

estudo

Lucas Skywalker

14 de agosto de 2015

1 Intro

As .dll ou .so são bibliotecas que contém rotinas já previamente compiladas. Para que o processo de reutilização dessas rotinas pelo seus programas seja mais ágil e menos custoso, pois enxergasse que serão reutilizadas muitas vezes pelos seus programas.

2 Estabelecendo ligação dinâmica pela primeira vez

1. De repente o sistema operacional se depara com uma rotina não local, ele gera um desvio para poder resolver o problema que apareceu para ele, de uma rotina contida em algum .dll ou .so, que será um `jal [LABEL1]` (o jal é porque ele deve retornar ao PC+4 depois dessa chamada para rotinas.)

2. A [LABEL1] é o local dentro do programa que irá ler a memória onde está guardada o endereço da sua rotina não local, isto é, ele irá executar lw $s1, [LABEL2], corregando o endereço da rotina não local em um registrador (neste caso $s1), e logo após irá executar um jr $s1 para desviar para o valor contido no endereço [LABEL2]. Como é a nossa primeira execução, o valor dentro de [LABEL2] não é a rotina que nós queremos ainda, ao invés disso ele irá ter o endereço do identificador da rotina, logo ele irá desviar para o identificador.

3. O código do identificador da rotina não local que o programa quer, será visto pelo montador como um `li $s0, I` (li = load immediate), logo após carregar **[LC]** ele desviará para o ligador/carregador (LC).

4. O LC irá caregar o código da rotina em memória e irá mapear o endereço do código para o endereço da memória [LABEL2], aí então ele desviará para a torina não local e a rotina será executada.

5. Executa a rotina não local e retorna para o programa.

Layout de Memória Antes e Depois da Ligação
dinâmica de um programa.o com a rotina printf