链滴

如何在二进制文件中查找全局变量的值

作者：dafsic

原文链接：https://ld246.com/article/1547651846124

来源网站：链滴

<h2 id="背景">背景</h2>
从二进制文件中找出一个初始化了的全局变量的值。
<h2 id="原理">原理</h2>
初始化的全局变量的值，一定是在可执行文件中的，就看怎么定位到这个地址了。
<h2 id="环境说明">环境说明</h2>
可执行二进制文件为 linux 下 elf 文件格式。二进制文件的符号表没被删除，并且知道这个全局量名。
<h2 id="方法">方法</h2>

先在文件中找到这个变量的地址。

<blockquote>
readelf -s prob |grep accessID
</blockquote>
<pre><code class="highlight-chroma">319: 0000000000cf2610   16 OBJECT GLOBAL DEFALT   9 main.accessID
4414: 00000000009ad660   17 OBJECT GLOBAL DEFAULT   2 main.accessID.str
</code></pre>
prob 为二进制文件名，accessID 就是我想知道的全局变量名，readelf -s 查看 prob 的符号表找到 accessID 这个 变量。
从符号名上应该能看出第二条记录是 我们需要的：(实际应该从第一条记录去找，因为我知道这变量是个字符串，所以这个变量的值 应该是个地址，而这个地址里的内容才是真正的 字符串值。 经证，0xcf2610 这个地址的内容就是 0x9ad660)

名称 -> main.accessID.str
内存地址 -> 0x9ad660
size -> 17 字节

但是 0x9ad660 这个地址是 应用加载进内存之后，在内存中的地址，并不是在二进制文件中的址。
<ol start="2">
找到变量在文件 中相对与文件头的偏移

<blockquote>
readelf -S prob
</blockquote>
<pre><code class="highlight-chroma">There are 24 section headers, starting at offset 0x1c8:

Section Headers:
[Nr] Name   Type   Address   Offset
Size   EntSize   Flags Link Info Align
[0]   NULL   0000000000000000 00000000
0000000000000000 0000000000000000   0   0   0
[1] .text   PROGBITS   0000000000401000 00001000
0000000000424f78 0000000000000000 AX   0   0   16
[2] .rodata   PROGBITS   0000000000826000 00426000
000000000019e7c4 0000000000000000   A   0   0   32
[3] .shstrtab   STRTAB   0000000000000000 005c47e0
0000000000000193 0000000000000000   0   0   1
</code></pre>

```
[ 4] .typelink         PROGBITS         00000000009c4980  005c4980
0000000000004078  0000000000000000   A       0     0     32
[ 5] .itablink         PROGBITS         00000000009c89f8  005c89f8
0000000000000f88  0000000000000000   A       0     0     8
[ 6] .gosymtab         PROGBITS         00000000009c9980  005c9980
0000000000000000  0000000000000000   A       0     0     1
[ 7] .gopclntab        PROGBITS         00000000009c9980  005c9980
00000000002fa65b  0000000000000000   A       0     0     32
[ 8] .noptrdata        PROGBITS         0000000000cc4000  008c4000
000000000002da61  0000000000000000  WA       0     0     32
[ 9] .data             PROGBITS         0000000000cf1a80  008f1a80
000000000000cb50  0000000000000000  WA       0     0     32
[10] .bss              NOBITS           0000000000cfe5e0  008fe5e0
000000000001fe50  0000000000000000  WA       0     0     32
</code></pre>

<p>readelf -S 查看文件的 section 表， 可以看出我们要找的变量 0x9ad660 这个地址在.rodata 段只读数据段， 字符串在这个段应该没错了)， 这个段的信息：</p>
<ul>
<li>名称 -> .rodata</li>
<li>内存起始地址 -> 0x826000</li>
<li>相对于文件头的 偏移 -> 0x426000</li>
<li>size -> 0x19e7c4 字节</li>
</ul>
<p>根据 0x826000 < 0x9ad660 < (0x826000+0x19e7c4)，确定这个变量在这个段的。</p>
<p>设 该变量在文件中的偏移为 X，则，</p>
<p>0x9ad660 - X = 0x826000 - 0x426000， X = 0x9ad660 - 0x400000 = 0x5ad660</p>
<ol start="3">
<li>输出这个变量的内容</li>
</ol>
<blockquote>
<p>hexdump prob -C -s 0x5ad660 -n 17</p>
</blockquote>
<pre><code class="highlight-chroma">005ad660  4c 54 41 49 62 42 67 6a  62 39 59 59 72 4
 4b 68  |LTAIbBgjb9YYrMKh|
005ad670  00                                       |.|
</code></pre>
<p>既然找到了偏移， 查看的方式有很多，其中**-C 是为了输出 ascii 值**，<strong>-s 起始地</strong>，<strong>-n 17 是之前 确认的 size 大小</strong></p>
<h2 id="结束">结束</h2>
<p>好多说不清楚，没办法再说下 elf 文件格式， 程序加载到内存变成进程的过程，以及 section 和 egment 的区别。这些都可以 在 《程序员的自我修养 》这本书上学到。</p>
```