# serial-150 攻防世界



版权

北风~ ● 于 2020-04-08 22:00:04 发布 ● 880 ☆ 收藏
 分类专栏: 逆向与保护
 版权声明:本文为博主原创文章,遵循 <u>CC 4.0 BY-SA</u>版权协议,转载请附上原文出处链接和本声明。
 本文链接: <u>https://blog.csdn.net/weixin\_45055269/article/details/105397707</u>

<u>逆向与保护 专栏收录该内容</u>

65 篇文章 4 订阅 订阅专栏

ida64 直接反编译,你会发现没有函数,而且代码段里是数据,这一点决定让我们使用动态调试,看程序跑起来之后,代码段里的数据会变成什么代码。

IDA远程动态调试配置:无坑版 🗆

这里分析如何找到输入字符后对字符检测的部分

#### 1.刚打开IDA时的样子

🖪 ID.	A View-RIP	- 8	×	X
	.text:000000000400890 ; Attributes: noreturn			4
	.text:00000000400890		Ξ	5
	.text:000000000400890    public _start			Ē
RIP	.text:000000000400890 _start proc near			H.
R12	.text:000000000400890 xor ebp, ebp			l (
the second s	.text:00000000400892 mov r9, rdx			
•	.text:00000000400895 pop rsi			4
•	.text:00000000400896 mov rdx, rsp			
•	.text:000000000400899 and rsp, 0FFFFFFFFF6h			
•	.text:0000000040089D push rax			
•	.text:0000000040089E			
•	.text:00000000040089F mov r8, offsetlibc_csu_fini			l.
•	.text:0000000004008A6 mov rcx, offsetlibc_csu_init			
	UNKNOWN 000000000400890: _start (Synchronized with RIP) https://blog.csdn.net/web/		520	þ

2.先点击红框,看程序停下来,再点击绿框内容

<u>F</u> ile	<u>E</u> dit Jump Searc <u>h V</u> iew Deb <u>ugg</u> e ■ Remote Linux debugger	er <u>O</u> ptions	<u>W</u> indow ⊋⊋⊉∎	/s Help II 💽 🛐 🗄 🗊 🕈 😭 🕯	2 of of 🎼 🗊   💷 🐓	<b>U</b>			
Library function 📕 Regular function 📕 Instruction 📕 Data 📕 Unexplored 📕 External symbol									
	Debug View	×	A	Structu	res	×	E	lnums	
📑 IDA	View-RIP							- <i>e</i>	× 👿
	libc_2.29.so:00007FBDA12BB87E libc_2.29.so:00007FBDA12BB87F libc_2.29.so:00007FBDA12BB880 libc_2.29.so:00007FBDA12BB8881	db 0C0h db 0Fh db 5 ;							
IP	<pre>libc_2.29.so:00007FBDA12BB881 libc_2.29.so:00007FBDA12BB887 libc_2.29.so:00007FBDA12BB889 libc_2.29.so:00007FBDA12BB889</pre>	cmp ja retn	rax, 0FF short lo	FFFFFFFFFF600h c_7FBDA12BB8E0					
•	libc_2.29.so:00007FBDA12BB88A libc_2.29.so:00007FBDA12BB88B libc_2.29.so:00007FBDA12BB88B libc_2.29.so:00007FBDA12BB88C libc_2.29.so:00007FBDA12BB88B libc_2.29.so:00007FBDA12BB88E	db 66h db 0Fh db 1Fh db 44h db 0	; f ; D						D
↓ ○ Hey	UNKNOWN 00007FBDA12BB881: libc_2.29.	so:read+11	(Synchroni	ized with RIP)					▼ 55269

程序停下来,发现是让你输入字符先随便输入一串a



#### 3.然后会发现进入7F开头的地址里,一直按F8,直到出现4开头的地址

D	A View-RIP		- <i>8</i>	>
•	libstdcso.6.0.28:00007FBDA15ABF6D add	rbp, 1		Π,
•	libstdcso.6.0.28:00007FBDA15ABF71 mov	[rsp+10h], rax		
1	libstdcso.6.0.28:00007FBDA15ABF76 mov	rax, [rbx+10h]		
1	libstdcso.6.0.28:00007FBDA15ABF7A cmp	rax, rdx		
ł	libstdcso.6.0.28: <mark>00007FBDA15ABF7D</mark>	loc_7FBDA15AC013		
1	libstdcso.6.0.28:00007FBDA15ABF83 add	rax, 1		
1	libstdcso.6.0.28:00007FBDA15ABF87 mov	[rbx+10h], rax		
	libstdcso.6.0.28:00007FBDA15ABF8B			
	libstdcso.6.0.28:00007FBDA15ABF8B loc_7F	BDA15ABF8B:	; CODE XREF: libstdcso.6.0.28:std::	: []
1	libstdcso.6.0.28:00007FBDA15ABF8B cmp	rax, rdx		-
-	libstdcso.6.0.28:00007FBDA15ABF8E jnb	loc_7FBDA15AC098		

INKNOWN 00007FRDA15ARF7D: libstde so 6.0.28.std: hasic istream(char std: char traits(char)) & std: operator))(char (Synchronized with RIP)).

### 4.发现关键比较,字符长度16

.text:000000000400A16 db 74h ; t .text:000000000400A17 db 0FAh
.text:000000000400A17 db 0FAh
taxt:00000000000000000000000000000000000
. LEXC. 00000000400A18 00 0181
.text:000000000400A19 ;
.text:000000000400A19 lea rax, [rbp-200h]
.text:000000000400A20 mov rdi, rax
.text:000000000400A23 <u>call sub_40085</u> 0
.text:000000000400A28 cmp rax, 10h
.text:0000000000400A2C jz short near ptr loc_400A3B+1
.text:000000000400A2E
.text:0000000000400A2E loc 400A2E: ; CODE XREF: .text:000000000400A341j

## 5.设置断点,不断满足条件写出flag

首位字符规定好,再规定末位字符加首位字符的和,这样两两一组逐渐往里缩,不断修改尝试

4	a 101	I VICW IVII				-
		.text:000000000400A16	db 74h	; t		
		.text:000000000400A17	db 0FAh			
	•	.text:000000000400A18	db 0E8h			
	_	.text:000000000400A19	;			
R	IP	.text:000000000400A19	lea	rax, [rbp-200h]		
	. •	.text:000000000400A20	mov	rdi, rax		
	•	.text:000000000400A23	call	sub_400850		
		.text:000000000400A28	cmp	rax, 10h		
	- <b>-</b>	.text:000000000400A2C	jz	short near ptr loc_400A3B+1		
		.text:000000000400A2E				
		.text:000000000400A2E	loc_4004	2E:	; CODE XREF: .text:0000000000400A34↓j	
		.text:000000000400A2E	mov	ax, 5EBh		
	•	.text:000000000400A32	xor	eax, eax		

EZ9dmq4c8g9G7bAV