20180CTF udp writeup

原创

Flying_Fatty 于 2018-04-30 08:12:04 发布 376 收藏
分类专栏： CTF之旅 reverse

一道linux下的父子进程调试题

首先分析main：

A：绑定了5999端口

```
29  addr.sa_family = 2;
30  *(_DWORD *)&addr.sa_data[2] = htonl(0x7F000001u);
31  *(_WORD *)addr.sa_data = htons(5999u);
```

B：fork了4000个子进程，并对齐进行阻塞

```
38  for ( main_for_i = 0; (unsigned int)main_for_i <= 3999; ++main_for_i )
39  {
40    printf("setup worker %d\n", (unsigned int)main_for_i);
```

```
56    if ( recvfrom(fd, &buf, 4uLL, 0, (struct sockaddr *)&s, &addr_len) != 4 )// waiting for sth
```

C：对于fork的4000个子进程的每一个，进行一次数据交互，先发送，再等待接收

```
83    v17 = htonl(0x7F000001u);
84    v16 = htons(i + 6000);
85    if ( sendto(fd, &v7, 4uLL, 0, (const struct sockaddr *)&s, 0x10u) != 4 )// sending sth to son
```

```
92    if ( recvfrom(fd, &buf, 4uLL, 0, (struct sockaddr *)&s, &addr_len) != 4 )// waiting sth
```

D：持续和子进程0进行交互，发送3过去，当接收回来是4，flag++，接收回来是5，输出flag

```
118   v17 = htonl(0x7F000001u);
119   v16 = htons(6000u);                       // first son
120   if ( sendto(fd, &v7, 4uLL, 0, (const struct sockaddr *)&s, 0x10u) != 4 )// sending 3 to first son
```

```
140   if ( buf != 4 && buf != 5 )
141   {
142     fwrite("[ERROR]\n", 1uLL, 8uLL, stderr);
143     fflush(stderr);
144     _exit(1);
145   }
146   if ( buf != 4 )
147     break;
148   printf("0x%lx\n", ++v13);                 // buf == 4 printf(sth)
149 }
150 printf("flag{%lx}\n", v13);                // buf == 5 printf(flag)
151 return 0LL;
```

然后就是另一个重要函数：0x400B45，son_udp

```
48    if ( !v12 )
49    {
50      close(fd);
51      son_udp();
52    }
```

在main中每一个子进程fork成功时，都会调用该函数

A：与main交互，发送数据

```
44  buf = 0;
45  v12 = 16;
46  v17.sa_family = 2;
47  *(_DWORD *)&v17.sa_data[2] = htonl(0x7F000001u);
48  *(_WORD *)v17.sa_data = htons(5999u);
49  if ( sendto(fd, &buf, 4uLL, 0, &v17, 0x10u) != 4 )
50  {
51    fwrite("[ERROR]\n", 1uLL, 8uLL, stderr);
52    fflush(stderr);
53    _exit(1);
54  }
```

B：在子进程中，开始对main中的数据值进行判断，并针对操作

当recvbuf为3时，说明是子进程0在操作

```
88    if ( recvbuf == 3 )
89    {
90      sendbuf = 5;                          // sendbuf == 5 -> main: printf(flag)
91      if ( main_for_i == 1 )
92      {
93        sendbuf = 4;                        // sendbuf == 4 -> main: printf(sth)
94      }
```

C：在之后的分析中，多次发现一个重要数据字段

```
140           if ( v5 == 4 )
141           {
142             --*(_QWORD *)(8LL * i + qword_8014110);
143             sendbuf = 4;
144           }
145           else if ( v5 != 5 )
146           {
147             exit(1);
148           }
149           if ( sendbuf == 4 )
150             break;
151         }
152       }
153     }
154     if ( sendbuf == 4 && v13 != -1 )
155       ++*(_QWORD *)(8LL * v13 + qword_8014110);
```

其实在该函数最前面有过初始化的

```
26  qword_8014110 = 32000LL * (unsigned int)main_for_i + 0x6020E0;
```

剩下的逻辑，看这篇writeup，很详细了

https://www.jianshu.com/p/d906619a01b7