Fix Time For Java Application Using JavaAgent

zcgonvh / 2017-12-06 15:36:00 / 浏览数 2375 安全工具 工具 顶(0) 踩(0)

Usage

```
java -agentpath:JavaTimeAgent.dll YOUR_JAVA_APPLICATION
```

JavaTimeAgent-master.zip (0.01 MB) 下载附件

上一篇：T00ls 2017第四期线下沙龙... 下一篇：2017年要结束了，我们是否来个年...

1. 1 条追加内容

追加 于 2017年12月6日 16:49

原理：

windows上的jre使用GetSystemTimeAsFileTime
API获取系统时间，对API下个断点，把poi(esp+4)改成0x01d263c1`feb0c000，就能看到所有的时间都变成了2017/1/1 X:0:0：

```
bp kernelbase!GetSystemTimeAsFileTime "r @$t0=poi(esp+4);gu;eq @$t0 0x01d263c1`feb0c000;g"
```

之后burp就能打开了：

中文操作系统默认时区是东八区，所以时间是2017/1/1 8:0:0，其他系统以此类推。

但现在出现了一个问题：由于所有时间都变成了相同的，所以history等按照时间排序的功能就失效了。
解决方式也很简单，只改日期不改时间即可。GetSystemTimeAsFileTime被我们和谐了，换用GetSystemTime来获取时间，返回的SYSTEMTIME也更方便处理。
（windows时间API基本都是从SharedUserData中直接取值，所以不用担心修改一个API会影响到其他的问题）

burp的进程是javaw，所以必须要做个hook。
注入+hook的代码网上一找一大把，不过java提供了agent功能进行动态修改，其中native agent本身就是一个dll，这样就不用单独写一个注入的exe了。

native
agent需要导出三个函数：Agent_OnLoad和Agent_OnAttach用于初始化，Agent_OnUnload用于卸载，考虑到并不需要访问jvm信息，用PVOID和long代替jvm类型

```
long WINAPI Agent_OnLoad(PVOID *vm, char *options, void *reserved);
long WINAPI Agent_OnAttach(PVOID *vm, char *options, void *reserved);
long WINAPI Agent_OnUnload(PVOID *vm);
```

hook实现方面，需要用到两个API：GetSystemTime获取当前时间，把年月日改成2017/1/1；SystemTimeToFileTime将修改后的时间转换为FileTime并返回传递给原

```
void WINAPI hook(LPFILETIME ft)
{
   SYSTEMTIME st = { 0 };
   GetSystemTime(&st);
   st.wYear = 2017;
```

```
    st.wMonth = 1;
    st.wDay = 1;
    SystemTimeToFileTime(&st, ft);
}
```

之后是hook，直接jmp，考虑到64位jre的问题要写两种：

```
#ifdef WIN64
    BYTE shellcode[] =
    {
        0x48, 0xb8,               //mov rax,
        0, 0, 0, 0, 0, 0, 0, 0,    //hook addr
        0xff, 0xe0                  //jmp rax
    };
#else
    BYTE shellcode[] =
    {
        0x90, 0xb8,               //mov eax,
        0, 0, 0, 0,                   //hook addr
        0xff, 0xe0                  //jmp rax
    };
#endif
```

编译加载运行，发现burp直接退出了，反编译BurpLoader，看到了一个可疑调用：

（吐槽一句jd-gui，反编译出的字节码结果还不如javap）

查询官方文档 https://docs.oracle.com/javase/7/docs/api/java/lang/management/RuntimeMXBean.html
得知`getInputArguments`方法返回jvm参数配置信息，反编译jre发现最终调用了一个native实现：`sun.management.VMManagementImpl.getVmArguments0`。

native实现必然封装于dll中，退出之前加载的最后一个java模块为management.dll：

查看导出函数，果然发现了名为`_Java_sun_management_VMManagementImpl_getVmArguments0@8`的导出。
那么和谐这个函数就好了。

由于java层的包装会自动处理null的问题，也就不用费力气写hook函数了，改成return NULL即可：

```
#ifdef WIN64
    BYTE shellcode2[] =
    {
        0x48, 0x33, 0xc0,         //xor rax,rax
        0xc3, 0x00, 0x00          //ret
    };

#else
    BYTE shellcode2[] =
    {
        0x33, 0xc0,               //xor eax,eax
        0xc2, 0x08, 0x00          //ret 8
    };
#endif
```

编译加载运行，这次没有问题了：

---

1. 12 条回复


nigh****ry911 2017-12-06 15:40:58

这个看起来好厉害

0 回复Ta


fiend 2017-12-06 16:16:38

向大佬低头

0 回复Ta

dkive 2017-12-06 20:27:33

■■■■

0 回复Ta

jean 2017-12-07 16:06:14

666

0 回复Ta

浮萍 2017-12-08 18:05:18

已经用上了，感谢ing...

0 回复Ta

爱豆人的马小跳47 2017-12-08 22:25:23

感谢大佬

0 回复Ta

gorg****@gmail.c 2017-12-10 20:00:22

感谢大佬

0 回复Ta

dav****a34b 2017-12-10 21:11:15

感谢大佬，厉害！

0 回复Ta


torah 2017-12-11 11:15:11

刚好非常需要，可以根据这个文档结合以前的整理搞点事情。

0 回复Ta


l3m0n 2017-12-11 16:28:09

学习一下，瞧一瞧

0 回复Ta


myse**** 2017-12-19 14:02:05

支持一下啊！

0 回复Ta

guiyingchongc*** 2017-12-25 11:55:15

感谢大佬，厉害。。。

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板