关于泄漏的艺术

早就想接着swan博士6月底的文章《Cookie已死，有事烧纸》继续八卦一下，但无奈世事纷繁，不胜其扰，酝酿了很久却一拖再拖。眼看马上就2014年了，也正好为今年画上一个句号，总结也好，备忘也罢，还请各位神牛高抬贵手，写的不对的地方还请多多指教。所谓一个过全部，就是用一个漏洞绕过所有的anti-exploit机制，当然这里指的是DEP/ASLR/vt\_guard以及各种anti-heap-fengshui/spray技术，SE H和沙盒类的就不包括在内了。再多说一句，今天就不谈寻找固定地址和重庆高人的CPU-SE H了，就谈谈使用和研究越来越多，也越来越实用的“泄漏”的艺术，个人觉得这东西补起来和Alexander Sotirov的heap-fengshui一样慢，反正至少拿去参加pwn2own2014是够了（当然是你有bypass sandbox的东东了），赚上10刀补了也不会特别心疼和可惜。

首先说的是越界类，这类用计算机语言描述就是“相对地址读/写”。越界类的根本原因也不多，基本就是整数问题和type-confusion(小对象混淆成大对象)。这类的写居多，读居少（典型的是flash的cve-2012-0779）。至于又能读又能写各种bug以至于像极了后门的出了一个，Nicolas Joly大神的cve-2013-2551，这种神洞这里就不讨论了。只说大多数的情况，读就是直接读一个神马出来，打破随机性；写的话分连续写和不连续写，思路差不多，反正就要找个好东西来写，这个好东西还能放在被越界的堆后面（这里最典型的例子是cve-2013-0634和cve-2012-1876）。

围绕着这个，出现了一点点对抗，最开始IE8下最好用的BSTR，到了IE9及以上，jscript9的引入，就从这两方面进行了围堵。但好的exploiter总能找到弱点，重庆高人提出的强制加载低版本的javascript引擎，Keen的CL提出的vbarray，都是从找弱点的方向进行对抗，当然不利用未被保护的引擎也是可以的，比如Nicolas Joly的element attribute。这些东西都是好东西，既能和被越界堆所在一个堆块，又有好的东西可被改写，甚至能泄漏和执行代码一次性完成。element attribute这里多说一句，由于其内存分布的特殊性，可利用的地方比较广，所以很可能会成为今后的主力。总的来说heap-layout就这点东西，各位大神们也普遍认为这类“一个过全部”是很轻松愉快的。个人认为，浏览器里各种对象繁多，就算不利用低版本引擎，硬生生在浏览器里面找满足以上两个条件的对象，也总能找到，多说一句，不一定非要在jscript和mshtml里找啊：）这方面Peter Vreugdenhil的course里应该会讲的比较多，听过他们课程的牛们能不能爆点IE9及以上的non-heapspray的料啊。

下面讲讲第二大类，未初始化内存类，在栈里的叫局部变量未初始化，在堆里的叫UAF。这类问题属于时序问题，引起这类问题的原因非常多，插入卸载，事件回调，append/remove造成的引用计数问题等等，五花八门，不一而足。虽然找漏洞要跟着原因分类找，但利用起来其实都可归为一类。

在具体谈这类的一个过全部之前，先说说现在公开半公开的技术。今年10月份DM牛在FireEye的报告中公布了一些在0day中使用的技术。显现出了被捕到的大多使用的都是实用的泄漏技术。越界类的找到了array/vector（java和flash中）作为合适的布局对象，一点也不新奇，UAF类的全都（转化）成了绝对地址写，然后辅以spray一些合适的对象（flash-vector/ff-arraybuffer）用以覆盖。其实早在2012年的blackhat上，Fermin J. Serna就针对一个过全部进行了演讲《The info leak era on software exploitation》，但一谈到UAF的一个过全部就开始讲Plototype，变成了只讲思路，不谈技术的议题，用一句话总结，就是“找到non-virtual完成任意地址读/写/释放”，看似说了很多，实则关键的地方什么都没说。总之感觉这人有点小气，今天就不吐他的槽了。

今天要谈的IE上UAF的一个过全部，思路还是两个字，泄漏。其实UAF和任何东西一样，也是分369等的。上等的UAF是free的对象是脚本层面可以直接访问的对象，比如cve-2011-1996的Option对象，ms-13-055的anchor对象，cve-2012-1875的img和div对象。这种类型的UAF，重用的时机（何时重用）和流程（重用流程可控，包括控制v-table调用）是可控的，给UAF带来了极大的灵活性，更容易控制把UAF转换成任意绝对地址读/写，从而完成泄漏。中等的UAF是脚本层面不能直接访问的对象，比如贡献了很多cve的CTreeNode，cve-2013-1690的DocumentViewerImpl ,cve-2013-3163的CBlockContainerBlock ，这种类型的UAF重用的时机大多不可控（一般是事件回调里的固定流程或是在渲染页面时的固定流程），流程也基本固定（调用v-table的第几个虚函数是固定的），但还是能通过伪造对象/控制数据改变流程达到目的，比如说任意地址写。

这里稍微多谈谈call [exx+8]的情况，如果固定流程可走到+8的流程，那么可以通过填充相同大小的对象，漏洞触发后可错误的释放刚刚被填充的对象（+8的虚函数是com接口规范中的release方法），这样就相当于进行了一次type-confusion，结果是多了一个脚本层面可访问到的，实际内存却被释放的对象，变成了第一种上等的UAF，后面的变化也就多了。典型的示范见cve-2012-1889。

最后一种下等的UAF，一句话略过，就是释放和重用之间很短暂，短暂到插不进任何东西，占不了位，也就谈不上利用了。

具体操作时，free后的第一步还是占位。虽然占位早就成了老掉牙的动作了，但怎么占，占什么还是有新花样的。早期人们就是占字符串（准确的说就是前四字节）喜欢用img.src，后来规定动作变成了obj.id/className/title等等（当然，同理把占位的字符串拉长就变成spray了），但后来又发现如果需要占位的某些字段为0，str就不行了，这时候Peter Vreugdenhil又提出了area.coords的方法可以让任意字段为任意值。后来为了需要，也不时的出现占用相同大小对象的情况。如果被free对象的大小不合适，还牵扯到LFH堆的整理等比较复杂的情况。总体目标就是往上等的UAF上靠，靠不上去了，再考虑怎么伪造对象在现有的路径中找到可用的路径。

之后怎么伪造对象，怎么控制流程，怎么避免崩溃，怎么确保找到的这条路径没有virtual-call就是个体力活了。总结下来，任意绝对地址写会比读更容易找，另外如果用心找很多UAF都能一个过全部。

前几年喜闻乐见的的指令是找崩溃点的附近的call来控制eip，而现在人见人爱的是”mov/add/or [exx+0xxx]，0xxx”，真有点“不是我不明白，这时代变化快”的味道。在有了任意绝对地址写之后，能做什么，Peter Vreugdenhil和Ivan Fratric两位搞IE的大神也相继发表了自己的成果《A browser is only as strong as its weakest byte – Part 2》和《Exploiting Internet Explorer 11 64-bit on Windows 8.1 Preview》。虽然提到了一些本质问题，但终究没有通用/稳定的解决IE下覆盖什么的问题（当然或许是大神们不想说：））。

该到爆料和8挂时间了，今年9月份发现的那个问题存在于新版javascript引擎的堆管理机制中，它能泄漏任意对象的地址，用来通用/稳定的解决IE下覆盖什么的问题，并且在寻找路径时可以不用规避virtual-call，极大的提高成功率和效率。

写到最后觉得和原本想的有点不一样，把一篇8挂文写成了科普文。前段时间偶然看到xcon2005的议题和speaker，顿时无限感慨和唏嘘，老人们大多数奔去了大辽，留在大宋的骨灰级神牛还奋战在一线也寥寥数几，而大宋的中生代，暗日无光，不知何时天亮，更别说讨论和分享了，最后的最后，分享一段狄更斯的《双城记》，与小伙伴们共勉，大家新年快乐。

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way--in short, the period was so. far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

ga1ois

2013/12/31