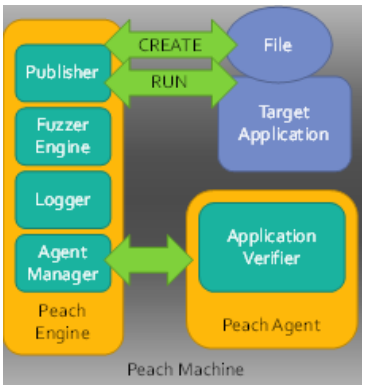
(http://blog.nsfocus.net/)

浅析Peach Fuzz

4 years ago (http://blog.nsfocus.net/peach-fuzz/) 绿盟科技 (http://blog.nsfocus.net/author/nsfocus/)
(http://blog.nsfocus.net/peach-fuzz/)



阅读： 27,394

浅析 Peach Fuzz 安全研究部 刘永军

摘要：本文简要介绍了Fuzz 工具Peach的使用，并通过文件格式 Fuzz举例阐述了 Peach Pit 文件的编写。

文章目录

1、引言

Fuzz（模糊测试）是一种通过提供非预期的输入并监视异常结果来发现软件安全漏洞的方法。模糊测试在很大程度上是一种强制性的技术，简单并且有效，但测试存在盲目性。

典型地模糊测试过程是通过自动的或半自动的方法，反复驱动目标软件运行并为其提供构造的输入数据，同时监控软件运行的异常结果。

Fuzz被认为是一种简单有效的黑盒测试，随着Smart Fuzz的发展，RCE（逆向代码工程）需求的增加，其特征更符合一种灰盒测试。

Peach是一个优秀的开源Fuzz框架。

2、Peach简介

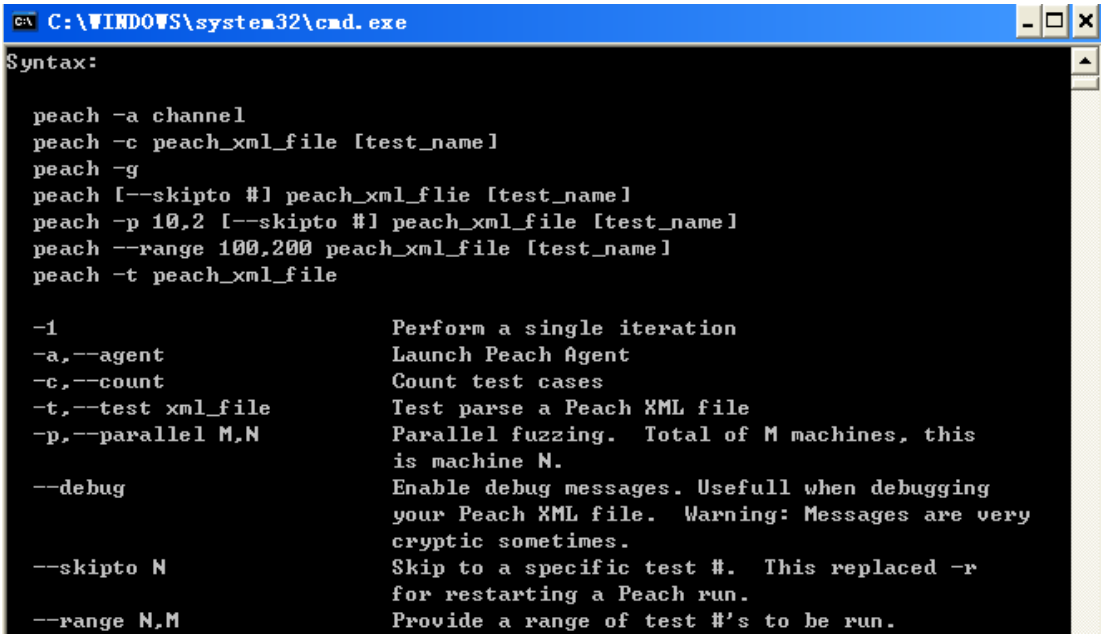
2.1 概述

Michael Eddington等人开发的Peach是一个遵守MIT开源许可证的模糊测试框架，最初采用Python语言编写，发布于2004年，第二版于2007年发布，最新的第三版使用C#重写了整个框架。

Peach支持对文件格式、ActiveX、网络协议、API等进行Fuzz测试；Peach Fuzz的关键是编写Peach Pit配置文件。

Windows下使用Peach3需要预先安装.net 4和windbg；Linux、OS X下需要安装Mono .net开发框架。

2.2 命令行参数



(http://blog.nsfocus.net/wp-content/uploads/2015/07/image001.png)

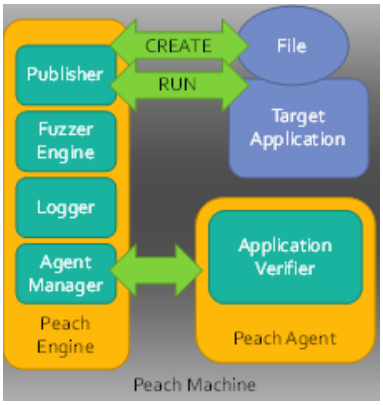
- -1：执行第1次测试。
- -a：启动Peach代理。不指定”channel”默认为本地代理（默认支持，无需显式启动）；

“channel”可以指定为”tcp”远程代理。

- -c：统计测试用例数。
- -t：验证Peach Pit xml文件正确性。
- -p：并行Fuzz。运行Peach的机器总数为M，这是第N个。
- –debug：调试信息开关。
- –skipto：指定Fuzz跳过的测试用例数。
- –range：指定Fuzz的测试用例范围。

3、Peach文件Fuzz

3.1 Peach文件Fuzz流程图



(http://blog.nsfocus.net/wp-content/uploads/2015/07/image002.png)

3.2 Peach Pit

在使用Peach进行Fuzz之前需要编写被称为"Peach Pit"的xml配置文件，其中包含着如何进行Fuzz的关键信息，如下图：

```
<?xml …版本，编码之类…>
<Peach …版本，作者介绍之类…>
<Include ...包含的外部文件… />
<DataModel >原始数据结构定义、可嵌套</DataModel>
<StateModel >测试逻辑，状态转换定义</StateModel>
<Agent >监测被测目标的反应，如crash等</Agent>
<Test >指定使用哪个StateModel 、Agent 、Publisher、
Strategy、 Logger等</Test>
</Peach>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image003.png)

其主要元素包括：

- **DataModel**

一个Pit文件至少会包括一个或多个DataModel，描述数据类型信息，关系（大小、数量、偏移量），和其它允许智能Fuzz的信息。如下图：

```
<DataModel name="Template">
  <String name="Key" />
  <String value=": " token="true" />
  <String name="Value" />
  <String value="\r\n" token="true" />
</DataModel>

<DataModel name="Customized" ref="Template">
  <String name="Key" value="Content-Length" />
  <String name="Value">
    <Relation type="size" of="HttpBody" />
  </String>
  <Blob name="HttpBody" />
</DataModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image004.png)

其属性包括：

> 1) name：数据模型的名字[必须]。
> 2) ref：引用模版数据模型[可选]。DataModel有ref属性时，与被引用DataModel类似

于子类与基类的关系，基类数据会被子类继承，子类子元素会覆盖基类同名子元素，

> 3) mutable：数据元素可变异性[可选，默认true]。

其主要子元素：Blob、Block、Choice、Flags、String、Number、Relation等。

1) Blob：常用于表示没有类型定义和格式的数据，如下图：

```
<Blob name="Unknown1" valueType="hex" value="01 06 22 03"/>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image005.png)

其主要属性包括：

- value：Blob默认值。
- length： blob的字节长度， blob长度判断会根据后续有token元素的位置计算。
- token：这个元素解析是否作为"标记"，默认false。

2) Block：用来组合一个或者多个的其他元素。Block和DataModel是很类似的，一个重要区别在于它们的位置，DataModel是顶级元素， Block是其子元素。

其不同于DataModel的属性包括：

- minOccurs：这个Block所必须出现的最低次数[可选] 。
- maxOccurs ： 这个Block可能会出现的最高次数[可选]。

3) Choice：每次选择其中一个元素，类似switch语句。如下图：

```
<DataModel name="ChoiceExample1">
 <Choice name="Choice1" minOccurs="3" maxOccurs="6">

  <Block name="Type1">
    <Number name="Str1" size="8" value="1" token="true" />
    <Number size="32"/>
  </Block>

  <Block name="Type2">
    <Number name="Str2" size="8" value="2" token="true" />
    <Blob length="255" />
  </Block>

  <Block name="Type3">
    <Number name="Str3" size="8" value="3" token="true" />
    <String length="8" />
  </Block>
 </Choice>
</DataModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image006.png)

minOccurs为最小生成Choice数；maxOccurs为最大生成Choice数，-1为无上限；occurs为必须产生的次数，如果不能达到这个次数，异常退出。具体匹配实现按照Choice中Block顺序，crack（解析）数据时根据token匹配一个Block后，数据位置后移匹配Block大小，继续按照Choice中Block顺序从头匹配。

4) Flags： Flag元素定义包含在Flags容器中的位字段，如下图：

```
<Flags name="options" size="16">
  <Flag name="compression" position="0" size="1" />
  <Flag name="compressionType" position="1" size="3" />
  <Flag name="opcode" position="10" size="2" value="5" />
</Flags>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image007.png)

其主要属性包括：

- size：大小，以位数为单位[必须] 。
- position：flag的起始位置（以0为基准）[必须]。

5) String：定义一个或者双字节的字符串，如下图：

```
<String value="Null terminated string" nullTerminated="true" />
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image008.png)

其主要属性包括：

- nullTerminated：字符串是以null结尾[可选] 。
- type：字符编码类型，默认”ascii”，可用选项有ascii, utf7, utf8, utf16, utf16be,

utf32 [可选]。

- padCharacter：填充字符串，来填充达到length的长度，默认是0x00[可选]。

6) Number：定义了长度为8，16，24，32 或者64位的二进制数字，如下图：

```
<DataModel name="NumberExample7">
 <Number name="Hi5" value="AB CD" valueType="hex" size="16" signed="false" endian="little" />
</DataModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image009.png)

其主要属性包括：

- size：Number的大小，以位为单位。有效的选择是1-64 [可选]。
- endian：数字的字节顺序，默认是小端字节［可选］。
- signed：是否是有符号，默认是true[可选]。

7) Relation：用于连接两个大小、数据、偏移量相关元素，如下图：

```
<Number size="32" signed="false">
  <Relation type="size" of="Value"
  expressionGet="size/2" expressionSet="size*2" />
</Number>
<String name="Value" />
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image010.png)

type类型为size时，of表示Number 是Value字符串的字节数。expressionGet用于crack过程，表示读”Value”多少字节。expressionSet用于publishing过程，为Publisher 生成Number值。

- StateModel

用于定义测试的逻辑，实际上相当于一个状态机。如下图：

```
<!-- This is our simple wave state model -->
<StateModel name="TheState" initialState="Initial">
      <State name="Initial">
            <!-- Write out our wave file -->
            <Action type="output">
                  <DataModel ref="Wav"/>
                  <!-- This is our sample file to read in -->
                  <Data fileName="sample.wav"/>
            </Action>
            <Action type="close"/>
            <!-- Launch the target process -->
            <Action type="call" method="StartMPlayer" publisher="Peach.Agent" />
      </State>
</StateModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image011.png)

下级标签包括State，每个State中又可以包含若干个Action标签。

1) State：表示一个状态，不同的State之间可以根据一些判断条件进行跳转，通常和Action的when属性联合使用。如下图：

```
<DataModel name="InputModel">
        <Number name="Type" size="32" />
</DataModel>
<DataModel name="OutputModelA">
        <Number name="Type" size="32" value="11 22 33 44" valueType="hex"
</DataModel>
<StateModel name="StateModel" initialState="InitialState">
        <State name="InitialState">
                <Action type="input">
                        <DataModel ref="InputModel" />
                </Action>
                <Action type="changeState" ref="State2" when="int
                (StateModel.states['InitialState']
                .actions[0].dataModel['Type'].InternalValue) == 2"/>
        </State>
        <State name="State2">
                <Action type="output">
                        <DataModel ref="OutputModelA" />
                </Action>
        </State>
</TheStateModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image012.png)

2) Action：用于完成StateModel中的各种操作，是给Publisher发送命令的主要方式。Action能发送输出、接收输入、打开连接，也能改变State等。主要属性：

- type：操作类型[必须]。主要类型：

start：启动Publisher，隐含动作，一般不需要。

stop：停止Publisher，隐含动作，一般不需要。

input：接收或者读取来自Publisher的输入，需要指定DataModel，用于crack和包含输入数据。

ouput：通过Publisher发送或者写输出，需要一个DataModel ，包含可选data，如下图：

```
<!-- Write out our wave file -->
<Action type="output">
        <DataModel ref="Wav"/>
        <!-- This is our sample file to read in -->
        <Data fileName="sample.wav"/>
</Action>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image013.png)

- when：如果提供的表达式为true，完成操作；否则，跳过。
- ref：状态变更后的引用[type=changeState] 。
- method：call的方法 [必须, type=call]，调用Publisher可选参数定义的方法，不

是所有Publisher都支持。

- Agent

是能够运行在本地或者远程的特殊的peach进程，这些进程能够启动监视器监控被测目标，如附加调试器、检测crash等。如下图：

```
<Agent name="LocalAgent">
 <Monitor class="WindowsDebugger">
      <!-- The command line to run.  Notice the filename provided matched up
           to what is provided below in the Publisher configuration -->
      <Param name="CommandLine" value="C:\mplayer\mplayer.exe fuzzed.wav" />
      <!-- This parameter will cause the debugger to wait for an action-call in
           the state model with a method="StartMPlayer" before running
           program.
      -->
      <Param name="StartOnCall" value="StartMPlayer" />
  </Monitor>
  <!-- Enable heap debugging on our process as well. -->
  <!--Monitor class="PageHeap">
      <Param name="Executable" value="hmplayer.exe"/>
  </Monitor-->
</Agent>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image014.png)

远程Agent需要首先在远程目标机通过peach –a tcp启动远程代理，无需pit文件。本地peach pit文件添加如下图location，其中ip为目标机ip。

```
<Agent name="RemoteAgent" location="tcp://192.168.1.1:9001">
  <!-- Monitors -->
</Agent>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image015.png)

可用Monitor如下图：

Windows Monitors

- Windows Debugger Monitor
- Cleanup Registry Monitor
- Page Heap Monitor
- Popup Watcher Monitor
- Windows Service Monitor

OS X Monitors

- OS X Crash Wrangler Monitor
- OS X Crash Reporter Monitor

Linux Monitors

- Linux Crash Monitor

Cross Platform Monitors

- CanaKit Relay Monitor
- Cleanup Folder Monitor
- IpPower9258 Monitor
- Memory Monitor
- Pcap Network Monitor
- Ping Monitor
- Process Launcher Monitor
- Process Killer Monitor
- Save File Monitor
- Socket Listener Monitor
- SSH Monitor
- SSH Downloader Monitor
- Vmware Control Monitor

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image016.png)

Windows Debugger Monitor通过windbg控制一个windows调试实例，主要参数：

- CommandLine：运行的命令行，如下图：

```
<Param name="CommandLine" value="c:\\mplayer\\mplayer.exe fuzzed.wav" />
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image017.png)

文件fuzz时上述文件名fuzzed.wav需要与Publisher参数一致。如下图：

```
<Publisher class="File">
    <Param name="FileName" value="fuzzed.wav"/>
</Publisher>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image018.png)

- SymbolsPath：windbg符号路径。
- StartOnCall：StateModel有匹配调用时附加调试器。
- NoCpuKill：默认false，表示当被测目标进程cpu占用为0时将其结束。

Peach3对非内核目标使用的混合调试模式，首先通过CreateProcess DEBUG_PROCESS参数创建调试进程，当检测到被测目标有感兴趣faults产生时会使用windbg的dbgeng.dll进行重现调试，最后利用windbg插件msec.dll的!exploitable命令对漏洞的可利用性进行初步判断，记录结果。

- Test

指定使用哪个Agent、StateModel，Publisher用什么方法发送数据，使用什么方法变异数据，日志文件路径等。可以有多个Test，使用时通过peach命令行指定要运行的Test名称，未指定默认运行名称为”Default”的Test。如下图：

```
<Test name="Default" waitTime="5">
  <Strategy class="RandomDeterministic"/>
  <Agent ref="LocalAgent"/>
  <StateModel ref="TheState"/>
  <Publisher class="File">
      <Param name="FileName" value="fuzzed.wav"/>
  </Publisher>
  <Logger class="Filesystem">
      <Param name="Path" value="logs" />
  </Logger>
</Test>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image019.png)

Strategy（变异策略）包括：

- Random：默认会随机选择最大6个元素（可以通过参数MaxFieldsToMutate设置）利用随机mutator（变异器）进行变异。
- Sequential：Peach会顺序对每个元素使用其所有可用的Mutators进行变异。
- RandomDeterministic：Peach默认规则。这个规则对pit xml文件中元素根据Mutators

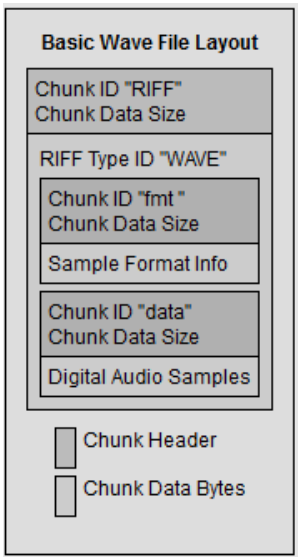
生成的Iterations链表做相对随机（由链表中元素数目决定）的顺序混淆，所以每个xml文件每次运行生成的测试用例多少、顺序固定，这样才能保证skipto的准确性。Peach3包括元素增、删、改、交换，经验值，逐位、双字等Mutators，见下图：

- ArrayNumericalEdgeCasesMutator
- ArrayRandomizeOrderMutator
- ArrayReverseOrderMutator
- ArrayVarianceMutator
- BlobBitFlipperMutator
- BlobDWORDSliderMutator
- BlobMutator
- DataElementDuplicateMutator
- DataElementRemoveMutator
- DataElementSwapNearNodesMutator
- FiniteRandomNumbersMutator
- NumericalEdgeCaseMutator
- NumericalVarianceMutator
- SizedDataNumericalEdgeCasesMutator
- SizedDataVarianceMutator
- SizedNumericalEdgeCasesMutator
- SizedVarianceMutator
- StringCaseMutator
- StringMutator
- UnicodeBadUtf8Mutator
- UnicodeBomMutator
- UnicodeStringsMutator
- UnicodeUtf8ThreeCharMutator
- ValidValuesMutator
- WordListMutator
- XmlW3CMutator

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image020.png)

3.3 Fuzz Wav文件

- Wav文件格式



(http://blog.nsfocus.net/wp-content/uploads/2015/07/image021.png)

- Pit文件

```
<?xml version="1.0" encoding="utf-8"?>
<Peach xmlns="http://phed.org/2012/Peach" xmlns:xsi="http://www.w3.org
  xsi:schemaLocation="http://phed.org/2012/Peach ../peach.xsd">

  <!-- Defines the common wave chunk -->
  <DataModel name="Chunk">
        <String name="ID" length="4" padCharacter=" " />
        <Number name="Size" size="32" >
                <Relation type="size" of="Data"/>
        </Number>
        <Blob name="Data" />
  </DataModel>

  <DataModel name="ChunkFmt" ref="Chunk">
        <String name="ID" value="fmt " token="true"/>
        <Block name="Data">
                <Number name="CompressionCode" size="16" />
                <Number name="NumberOfChannels" size="16" />
                <Number name="SampleRate" size="32" />
                <Number name="AverageBytesPerSecond" size="32" />
                <Number name="BlockAlign" size="16" />
                <Number name="SignificantBitsPerSample" size="16" />
                <Number name="ExtraFormatBytes" size="16" />
                <Blob name="ExtraData" />
        </Block>
  </DataModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image022.png)

```
  省略ChunkData等其它DataModel

  <!-- Defines the format of a WAV file -->
  <DataModel name="Wav">
        <!-- wave header -->
        <String value="RIFF" token="true" />
        <Number size="32" />
        <String value="WAVE" token="true" />

        <Choice maxOccurs="30000">
                <Block ref="ChunkFmt"/>
                <Block ref="ChunkData"/>
                <Block ref="ChunkFact"/>
                <Block ref="ChunkSint"/>
                <Block ref="ChunkWavl"/>
                <Block ref="ChunkCue"/>
                <Block ref="ChunkPlst"/>
                <Block ref="ChunkLtxt"/>
                <Block ref="ChunkSmpl"/>
                <Block ref="ChunkInst"/>
                <Block ref="Chunk"/>
        </Choice>
  </DataModel>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image023.png)

```
<!-- This is our simple wave state model -->
<StateModel name="TheState" initialState="Initial">
      <State name="Initial">
            <!-- Write out our wave file -->
            <Action type="output">
                  <DataModel ref="Wav"/>
                  <!-- This is our sample file to read in -->
                  <Data fileName="sample.wav"/>
            </Action>
            <Action type="close"/>
            <!-- Launch the target process -->
            <Action type="call" method="StartMPlayer" publisher="Peach.Agent" />
      </State>
</StateModel>

<Agent name="LocalAgent">
<!--Agent name="LocalAgent"  location="tcp://192.168.126.175:9001"-->
 <Monitor class="WindowsDebugger">
      <!-- The command line to run.  Notice the filename provided matched up
           to what is provided below in the Publisher configuration -->
      <Param name="CommandLine" value="C:\Program Files\Haihaisoft
        Universal Player\hmplayer.exe fuzzed.wav" />
      <!-- This parameter will cause the debugger to wait for an action-call in
           the state model with a method="StartMPlayer" before running
           program.
      -->
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image024.png)

```
            <Param name="StartOnCall" value="StartMPlayer" />
        </Monitor>
    </Agent>

    <Test name="Default" waitTime="3">
        <Agent ref="LocalAgent"/>
        <StateModel ref="TheState"/>
        <Publisher class="File">
            <Param name="FileName" value="fuzzed.wav"/>
        </Publisher>
        <Logger class="Filesystem">
            <Param name="Path" value="logs" />
        </Logger>
    </Test>

</Peach>
```

(http://blog.nsfocus.net/wp-content/uploads/2015/07/image025.png)

参考文献

www.peachFuzzer.com

《浅析Peach-Fuzz》文章下载

浅析Peach Fuzz (http://blog.nsfocus.net/wp-content/uploads/2015/07/%E6%B5%85%E6%9E%90Peach-Fuzz.pdf)

更多文章请访问：

http://www.nsfocus.com.cn/research/qyjs.html (http://www.nsfocus.com.cn/research/qyjs.html)

文章分类： 安全分享 (http://blog.nsfocus.net/category/safeshare/)
文章关键词： fuzz (http://blog.nsfocus.net/tag/fuzz/), fuzz测试 peach (http://blog.nsfocus.net/tag/fuzz%e6%b5%8b%e8%af%95-peach/), peach (http://blog.nsfocus.net/tag/peach/), peach fuzz使用 (http://blog.nsfocus.net/tag/peach-fuzz%e4%bd%bf%e7%94%a8/)

← 绿盟科技下一代网络安全预警决策体系 (http://blog.nsfocus.net/nsfocus-network-security-warning-decision-making-system/)

恶意文件分析系统中的数字签名验证 → (http://blog.nsfocus.net/digital-signature-with-malware-analysis/)

发表评论

要发表评论，您必须先登录 (http://blog.nsfocus.net/wp-login.php?redirect_to=http%3A%2F%2Fblog.nsfocus.net%2Fpeach-fuzz%2F)。

绿盟科技博客精华文章

学习手册：浅析DDoS的攻击及防御 (http://blog.nsfocus.net/analysis-ddos-attack-defense/)

学习手册：盘点DDoS带来的误会 (http://blog.nsfocus.net/inventory-ddos-errors/)

移动APP安全测试要点 (http://blog.nsfocus.net/mobile-app-security-security-test/)

Dedecms远程写文件漏洞分析 (http://blog.nsfocus.net/dedecms-write-file-vuln/)

【干货分享】XSS攻击进阶篇——那些年我们看不懂的XSS (http://blog.nsfocus.net/xss-advance/)

两步邮件订阅，方便获取文章

欢迎订阅！现在已有6 646个朋友订阅了。
在后续邮件的尾部，您可以退订及修改订阅内容。
选择订阅组：

- [ ] **最新文章**
- [ ] **技术分享**
- [ ] **漏洞分析**
- [ ] **运维安全**
- [ ] **Web安全**
- [ ] **安全报告**

邮件 *

马上订阅！

绿盟科技博客功能

注册 (http://blog.nsfocus.net/wp-login.php?action=register)

登录 (http://blog.nsfocus.net/wp-login.php)

文章RSS (Really Simple Syndication) (http://blog.nsfocus.net/feed/)

评论RSS (Really Simple Syndication) (http://blog.nsfocus.net/comments/feed/)

WordPress.org (https://cn.wordpress.org/)

绿盟科技博客文章标签

绿盟科技 (http://blog.nsfocus.net/tag/%e7%bb%bf%e7%9b%9f%e7%a7%91%e6%8a%80/) 漏洞 (http://blog.nsfocus.net/tag/%e6%bc%8f%e6%b4%9e/) 远程代码执行漏洞 (http://blog.nsfocus.net/tag/%e8%bf%9c%e7%a8%8b%e4%bb%a3%e7%a0%81%e6%89%a7%e8%a1%8c%e6%bc%8f%e6%b4%9e/) 绿盟科技漏洞库 (http://blog.nsfocus.net/tag/%e7%bb%bf%e7%9b%9f%e7%a7%91%e6%8a%80%e6%bc%8f%e6%b4%9e%e5%ba%93/) NSFOCUS (http://blog.nsfocus.net/tag/nsfocus/) EnglishVersion (http://blog.nsfocus.net/tag/englishversion/) DDoS (http://blog.nsfocus.net/tag/ddos/) 远程代码执行 (http://blog.nsfocus.net/tag/%e8%bf%9c%e7%a8%8b%e4%bb%a3%e7%a0%81%e6%89%a7%e8%a1%8c/) 技术博客

(http://blog.nsfocus.net/tag/%e6%8a%80%e6%9c%af%e5%8d%9a%e5%ae%a2/) 云安全
(http://blog.nsfocus.net/tag/%e4%ba%91%e5%ae%89%e5%85%a8/) 威胁情报
(http://blog.nsfocus.net/tag/%e5%a8%81%e8%83%81%e6%83%85%e6%8a%a5/) 网络安全
(http://blog.nsfocus.net/tag/%e7%bd%91%e7%bb%9c%e5%ae%89%e5%85%a8/) 物联网安全
(http://blog.nsfocus.net/tag/%e7%89%a9%e8%81%94%e7%bd%91%e5%ae%89%e5%85%a8/) 威胁
(http://blog.nsfocus.net/tag/%e5%a8%81%e8%83%81/) 高危漏洞
(http://blog.nsfocus.net/tag/%e9%ab%98%e5%8d%b1%e6%bc%8f%e6%b4%9e/) 绿盟科技博客
(http://blog.nsfocus.net/tag/%e7%bb%bf%e7%9b%9f%e7%a7%91%e6%8a%80%e5%8d%9a%e5%ae%a2/) 威胁通告
(http://blog.nsfocus.net/tag/%e5%a8%81%e8%83%81%e9%80%9a%e5%91%8a/) rsa2018 (http://blog.nsfocus.net/tag/rsa2018/) 安全会议
(http://blog.nsfocus.net/tag/%e5%ae%89%e5%85%a8%e4%bc%9a%e8%ae%ae/) 安全周报
(http://blog.nsfocus.net/tag/%e5%ae%89%e5%85%a8%e5%91%a8%e6%8a%a5/) ddos攻击事件
(http://blog.nsfocus.net/tag/ddos%e6%94%bb%e5%87%bb%e4%ba%8b%e4%bb%b6/) DDoS攻击
(http://blog.nsfocus.net/tag/ddos%e6%94%bb%e5%87%bb/) 云计算 (http://blog.nsfocus.net/tag/%e4%ba%91%e8%ae%a1%e7%ae%97/) 大数据 (http://blog.nsfocus.net/tag/%e5%a4%a7%e6%95%b0%e6%8d%ae/) 勒索软件
(http://blog.nsfocus.net/tag/%e5%8b%92%e7%b4%a2%e8%bd%af%e4%bb%b6/) 安全意识
(http://blog.nsfocus.net/tag/%e5%ae%89%e5%85%a8%e6%84%8f%e8%af%86/) 恶意软件
(http://blog.nsfocus.net/tag/%e6%81%b6%e6%84%8f%e8%bd%af%e4%bb%b6/) 物联网
(http://blog.nsfocus.net/tag/%e7%89%a9%e8%81%94%e7%bd%91/) 漏洞分析
(http://blog.nsfocus.net/tag/%e6%bc%8f%e6%b4%9e%e5%88%86%e6%9e%90/) 态势感知
(http://blog.nsfocus.net/tag/%e6%80%81%e5%8a%bf%e6%84%9f%e7%9f%a5/)

最新文章

【威胁通告】开源压缩库Libarchive代码执行漏洞（CVE-2019-18408） (http://blog.nsfocus.net/cve-2019-18408/)

绿盟科技互联网安全威胁周报NSFOCUS-2019-44 (http://blog.nsfocus.net/nsfocus-2019-44/)

网络安全威胁月报NSFOCUS-2019-10 (http://blog.nsfocus.net/nsfocus-2019-10/)

Jenkins 路由解析及沙箱绕过漏洞分析报告(上) (http://blog.nsfocus.net/jenkins-routing-resolution-and-sandbox-bypass-vulnerability-analysis-report/)

【威胁通告】Apache Solr Velocity远程代码执行漏洞处置手册 (http://blog.nsfocus.net/ns-2019-0046/)

【威胁通告】 Apache Solr远程命令执行漏洞 (http://blog.nsfocus.net/apache-solr2019-10-31/)

APT技术观察——APT与隐写术 (http://blog.nsfocus.net/apt-technology-observation-apt-and-steganography/)

绿盟科技互联网安全威胁周报NSFOCUS-2019-43 (http://blog.nsfocus.net/nsfocus-2019-43/)

Splinter新APT攻击工具透析 (http://blog.nsfocus.net/splinters-new-apt-attack-tool-dialysis/)

Web安全之防止浏览器自动代填和回显已保存账号 (http://blog.nsfocus.net/web-security-prevents-browsers-from-automatically-filling-in-and-echoing-saved-accounts/)

友情链接

绿盟科技官网 (http://www.nsfocus.com.cn/)

绿盟威胁情报中心NTI (https://nti.nsfocus.com/)

绿盟科技博客读者群

群名称:绿盟安全爱好者
群　号:956543462