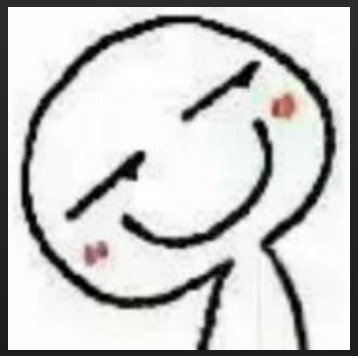
动态分析在安全检测中的应用

Fr1day @ 0keeTeam

ABOUT ME

- 微博： @吃瓜群众-Fr1day
- Web安全、安全开发
  - Python*
  - 前端安全



- 0keeTeam:
  - 360 信息安全部
  - Skywolf
  - 护心镜
  - 天相扫描系统
  - .....

为啥别人能扫到的洞我扫不到?

- AJAX 请求无法抓取到?
- DOM XSS无法自动检测?
- 任意URL跳转只能检查302跳转?
- 还在用复杂的正则匹配特性?
- 长得不够帅?

目录

- 动态分析是什么
- PhantomJS
- 应用
  - 爬虫
  - XSS
  - 任意URL跳转
- 总结

WEB 2.0 给安全检测带来的挑战

- 如何 **抓取更多** 的 HTTP 请求?
  - 扫描的基础是HTTP请求
  - 爬虫要抓取到尽可能全面的请求
- 如何保证扫描的 **有效率**?
  - http://xxx.com/index.php?id=1
  - http://xxx.com/index.php?id=2
  - http://xxx.com/index.php?id=3
  - …
- 如何保证扫描的 **准确率**?
  - 精准判断，是否存在漏洞（XSS、URL跳转等）

动态分析 - PhantomJs

- PhantomJS 是 **无界面的 Webkit 解析器**，提供了 JavaScript API

- 其他的解析器
  - 基于 Webkit 内核的 QtWebkit、Ghost.py
  - 基于的 Firefox Gecko 内核的 SlimerJS
  - 基于 PhantomJS、SlimerJS 封装的 CasperJS
  - Chrome Headless

WEB2.0 爬虫

静态分析获取页面所有<a>链接：

```
1  res = requests.get("http://named.cn/.mine")
2
3  pq = pyquery.PyQuery(res.content)
4  for i in pq("a"):
5      print "[%d]: %s" % (count, pq(i).attr("href"))
```

结果：

空

WEB2.0 爬虫

动态分析获取页面所有<a>链接：

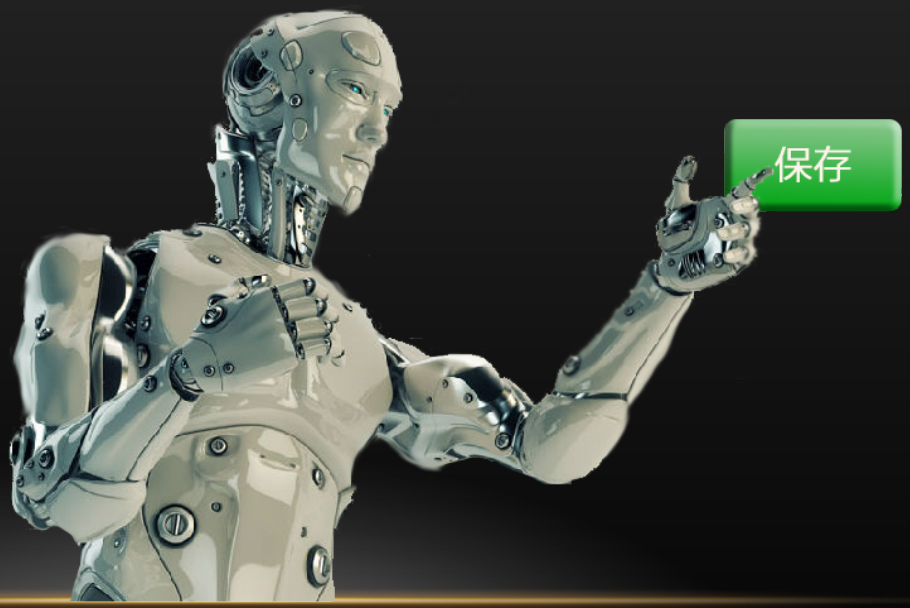
```
var page = require('webpage').create();

page.onCallback = function() {
    page.evaluate(function(){
        atags = document.getElementsByTagName("a");
        for(var i=0;i<atags.length;i++){
            if (atags[i].getAttribute("href")){
                alert(atags[i].getAttribute("href"));
            }
        }
    })

    phantom.exit()
};

page.open("http://named.cn/.mine", "get", "", function (status) {
    page.evaluateAsync(function(){
        if (typeof window.callPhantom === 'function') {
            window.callPhantom();
        }
    }, 10000)
});
```

结果：

.mine
/cmd2/controls/signin
/cmd2/controls/getcode
/download.html
/.blog
/.mine
/.at
/~发现推荐.findbbs
/help.html
/江南水乡.bbs/7313348
/摄情男女.bbs/7313242
/欢乐一家亲.bbs/7313356
/深夜食堂.bbs/7313168
/家有熊孩子.bbs/7313321
/乐淘亲子营.bbs/7313320
.../*省略*/...
/婚礼记.bbs/7277165
/不知道的事情.bbs/7277164
/不知道的事情.bbs/7277162
/婚礼记.bbs/7277160
/不知道的事情.bbs/7277016
http://www.miitbeian.gov.cn/
/cmd2/controls/mailpost/内容举报
download.html

WEB2.0 爬虫

- 场景1　滚动到页面尾部后自动加载下一页
- 场景2　点击了某个按钮后弹出相关数据
- 场景3　写文章自动保存草稿



WEB2.0 爬虫

事件	如何执行	执行过程	结果
滚动	触发 onscroll 事件	发送AJAX请求	动态更新页面
点击	触发 onclick 事件	发送AJAX请求	动态更新页面
定时保存	定时器执行 (setInterval)	获取表单内容 发送AJAX请求	动态更新页面

爬虫 – 如何执行

- 直接通过行内绑定
- 直接通过 addEventListener 绑定事件
- 用了封装的JS库，如 jQuery、Vue等

```
<div id="test" onclick="/**/"></div>
```

```
<div id="test"></div>
<script type="text/javascript">
test.addEventListener("click", function(e){
    /**/
})
</script>
```

```
<div id="test"></div>
<script src="http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js"></script>
<script type="text/javascript">
$("#test").click(function(){
    /**/
})
</script>
```

爬虫 – 如何执行

```
<div id="test" onclick="/**/"></div>
```

遍历dom节点，筛选出 onxxxx 的属性值，执行JS代码

```
for (var i = 0; i < document.all.length; i++) {
    var attrs = document.all[i].attributes;
    for (var j = 0; j < attrs.length; j++) {
        attr_name = attrs[j].nodeName;
        attr_value = attrs[j].nodeValue;
        if (attr_name.substr(0, 2) == "on") {
            // 匿名函数执行，防止由于执行return false;导致的函数意外退出
            eval("(function(){" + attr_value + "})()");
        }
        // 省略处理 href="javascript:alert(1)" 的情况
    }
}
```

爬虫 – 如何执行

```
<div id="test"></div>
<script type="text/javascript">
test.addEventListener("click", function(e){
    /**/
})
</script>
```

Hook addEventListener函数，获取function函数的内容，直接执行

```
_addEventListener = Element.prototype.addEventListener;
Element.prototype.addEventListener = function(a,b,c) {
    eval(b)
    _addEventListener.apply(this, arguments);
}
```

爬虫 – 如何执行

```
<div id="test"></div>
<script src="http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js"></script>
<script type="text/javascript">
$("#test").click(function(){
    /**/
})
</script>
```

Hook addEventListener函数 仍然能解决问题?

```
_addEventListener = Element.prototype.addEventListener;
Element.prototype.addEventListener = function(a,b,c) {
    eval(b)
    _addEventListener.apply(this, arguments);
}
```

```
function (b){return typeof n!==U&&n.event.triggered!==b.type?n.event.dispatch.apply(a,arguments):void 0}
▶ Uncaught SyntaxError: Unexpected token (
    at HTMLDivElement.Element.addEventListener (file:///C:/Users/zhaobeibei/Desktop/test.html:8:21)
    at Object.add (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:3:3645)
    at HTMLDivElement.<anonymous> (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:3:10983)
    at Function.each (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:2:2880)
    at n.fn.init.each (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:2:847)
    at n.fn.init.on (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:3:10959)
    at n.fn.init.n.fn.(anonymous function) [as click] (http://apps.bdimg.com/libs/jquery/2.1.1/jquery.min.js:4:48
    at file:///C:/Users/zhaobeibei/Desktop/test.html:12:12
```

爬虫 – 如何执行

JavaScript dispatchEvent，可以触发节点的绑定事件

```
var evt = document.createEvent('CustomEvent');
evt.initCustomEvent("click", true, true, null);
Test.dispatchEvent(evt);
```

如何获取 **节点** 和 **节点的绑定事件** 呢?

- 遍历节点
- Hook addEventListener

```
Element.prototype.addEventListener = function(a,b,c) {
    EVENT_LIST.push({"event": event, "element": this})
    _addEventListener.apply(this, arguments);
};
```

爬虫 – 捕获AJAX请求

- Hook XMLHttpRequest（示例代码在下一页）
  - 缺点：无法准确捕获Headers
  - 优点： 捕获到的所有请求都是有效请求
- Phantomjs onResourceRequested（示例代码在下一页）
  - 优点： 准确捕获Headers
  - 缺点： 会抓取到静态资源的请求(other frame)

爬虫 – 捕获AJAX请求

- Hook XMLHttpRequest

```
_open = XMLHttpRequest.prototype.open
XMLHttpRequest.prototype.open = function (method, url) {
    if (!this._url) {
        this._url = url;
        this._method = method;
    }
    _open.apply(this, arguments);
};
_send = XMLHttpRequest.prototype.send
XMLHttpRequest.prototype.send = function (data) {
    $Result$.add_ajax(this._url, this._method, data);
    _send.apply(this, arguments);
};
```

- Phantomjs onResourceRequested

```
page.onResourceRequested = function(requestData, networkRequest) {
    if (!(/[a-zA-Z0-9_\-\.]+?\.(js|css)/gi).test(requestData["url"])){
        navigate_urls.push(JSON.stringify({
            "url": requestData["url"], "method": requestData["method"],
            "data": requestData["postData"]? requestData["postData"]: "",
            "headers": get_headers(requestData["headers"]),
            "from": "phantom_requested"
        }))
    }
};
```

爬虫 – 分析结果

- Html5特性 – MutationObserver
- 事件 – DOMNodeInserted

```
document.addEventListener('DOMNodeInserted', function(e) {
        var node = e.target;
        if(node.src || node.href){
            LINKS_RESULT.push(node.src || node.href);
        }
}, true);
```

爬虫动态解析 - 流程


页面加载前
Hook addEventListener、XMLHttpRequest
Hook alert、prompt、confirm等
页面加载完成
获取当前所有<a>、<form>、<iframe>
监听 DOMNodeInserted
禁止页面跳转
触发事件阶段
自动填写表单
触发 inline事件 & 绑定事件
结束
去除静态资源
输出结果

爬虫 – 去重

/index.php?m=home&c=shop&a=show&id=2

逻辑参数： m、c、a

变量参数： id

/index.php?m=home&c=shop&a=show&id={{int}}

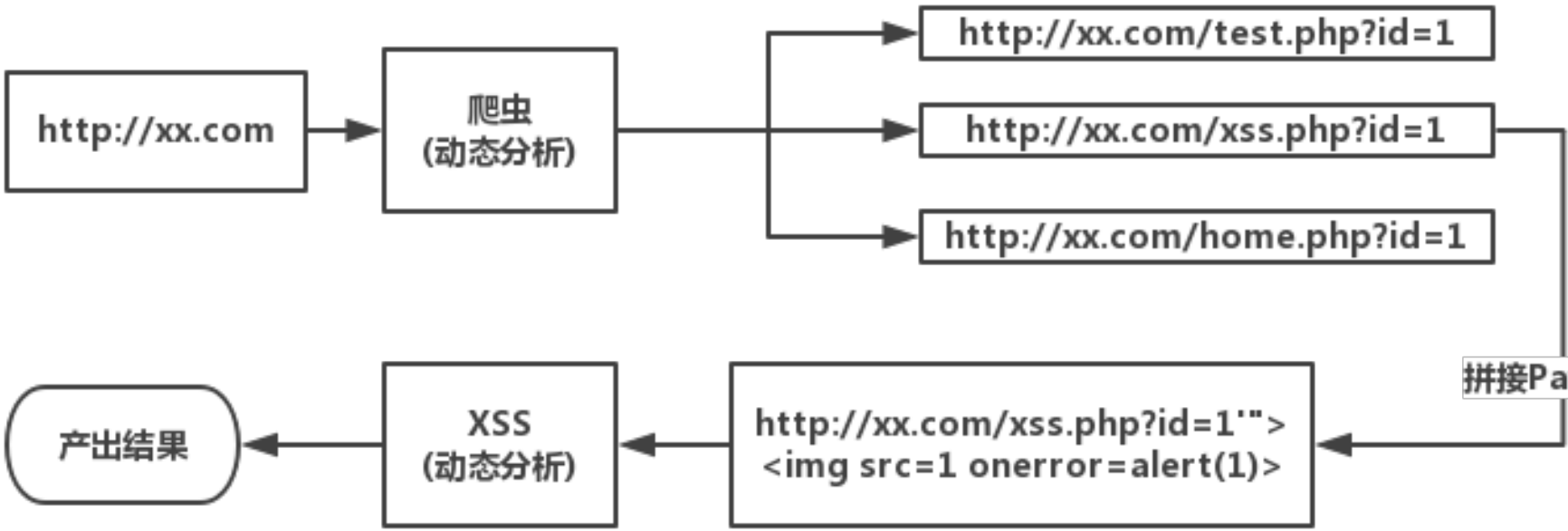
爬虫 – 去重

- 第一遍处理
  - 预处理
    - 去除非抓取目标的URL、静态资源等
    - 将中文、Hash、URL编码、纯数字等替换为占位符
  - 解析URL，分析 Param 和 Path
    - 参数名相同的URL，归到一类
    - Path类似的URL，归到一类
- 第二遍处理
  - 处理之前归类的 URL，数量超过一定值，强制去重
  - 对含占位符的 URL 进行去重

爬虫 – 对比

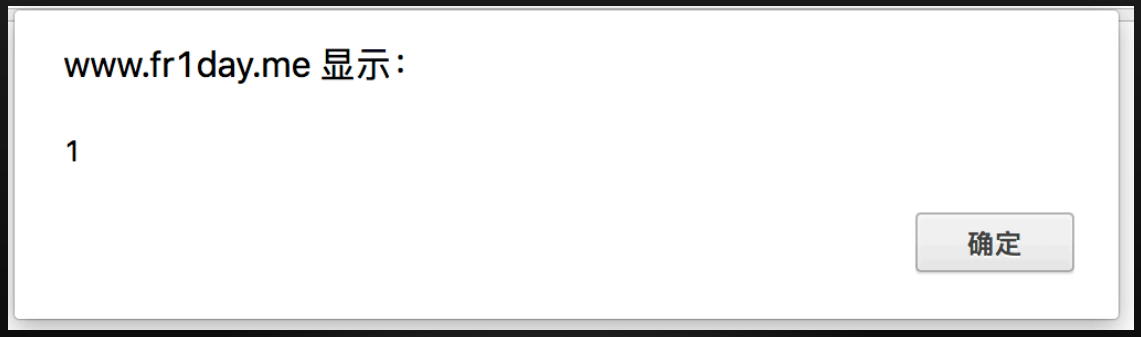
	基本抓取功能（AiSec）	抓取效率（tieba）
KSpider	Time: 33 Result: 18 Request Count: 23 有效抓取率： 100%	Time: 339 Result: 535 Request Count: 410 有效抓取率: 100%(相对)
WVSSpider	Time: 23 Result: 11 Request Count: 31 有效抓取率： 100%	Time: 220 Result: 101 Request Count: 201 有效抓取率：38%

爬虫 -> XSS


http://xx.com
爬虫
(动态分析)
http://xx.com/test.php?id=1
http://xx.com/xss.php?id=1
http://xx.com/home.php?id=1
拼接Payloa
产出结果
XSS
(动态分析)
http://xx.com/xss.php?id=1'">
<img src=1 onerror=alert(1)>

动态分析 – XSS

- 判断是否存在XSS
  - [静态] 是否存在特定关键字: <img src=1 onerror=alert(1)>
  - [动态] 是否执行了指定的代码
  - [动态] 是否有特定的DOM节点添加



动态分析 – XSS

- 是否执行了指定的代码： alert(1)

```
/?u=<img src=1 onerror=alert(1)>
/?u='"onfocus=alert(1) autocus
/?u='"></script><svg onload=alert(1)>
```

```
Window.alert = function(msg){
    if(msg == "1"){
        phantom_exit(1)
    }
    return true
}
```

动态分析 – XSS

- 是否有新的DOM节点添加： <xsstest>

```
/?u='"></script><xsstest></xsstest>
```

```
if (document.getElementsByTagName("xsstest").length > 0){
    phantom_exit(true)
}
```

动态分析 – XSS流程


页面加载前
Hook addEventListener
Hook alert、prompt、confirm等
页面加载完成
禁止页面跳转
触发事件阶段
触发 inline事件 & 绑定事件
结束
是否有特定JS代码执行
是否有特定DOM节点新增

动态分析 – URL跳转

- 服务器端 302跳转
  - Header(“Location: xxx”)
- 浏览器端 跳转
  - Window.location.href = xxx
  - <meta http-equiv="refresh" content=“0;url=xxx">

- PhantomJS
  - Payload：http://xx.com/?url={{target_url}}

```
page.onNavigationRequested = function(url) {
    if (url == target_url){
        phantom_exit(1)
    }
};
```

优劣

- 动态分析
  - 爬虫单次解析耗时： 3s – 20s
  - CPU和内存占用率较高
  - 爬虫、漏洞扫描： 覆盖率高 & 漏报率低
- 静态分析
  - 爬虫(PyQuery)单次解析耗时： 0.5s – 10s
  - 开发成本低，资源消耗少
  - 网络IO是主要瓶颈
  - 覆盖率低 & 漏报率高

- Chrome Headless
  - 优点
    - 官方正版
    - 支持各种新功能
  - 缺点
    - 实验性接口太多，有可能去掉或者改动参数，不太稳定
    - 部分实验性接口用不起来：拦截请求、控制页面大小
    - 无法锁定页面
    - 没有办法拦截掉资源请求
    - ….

总结

- 未来
  - 爬虫智能化
  - 存储型XSS
- 更多应用
  - 自动登录
  - 复杂加密的场景
  - 浏览器漏洞Fuzz
  - XSS bot

Q&A