软件源代码安全实践

韩建

OWASP 中国

The Open Web Application Security Project

OWASP 中国
The Open Web Application Security Project

1、开源代码安全现状和实例分析

2、针对软件生产安全的思考

3、探索软件安全在高校中的方法

软件安全事件层出不穷

OWASP 中国
The Open Web Application Security Project

OpenSSL是一个开源的安全套接字层密码库，2014年曝光重大安全漏洞Heartbleed。攻击者通过构造异常的数据包进行攻击，获取用户敏感信息。

ElasticSearch是全文搜索引擎开源代码项目。2014年和2015年分别爆出远程任意命令执行漏洞。攻击者可利用远程任意命令执行漏洞获取主机最高权限。

Struts2是Apache软件基金会赞助的一个开源项目，近年来频繁爆发安全漏洞。影响国内电商、银行、运营商等诸多大型网站和为数众多的政府网站。

Tomcat是一个免费的开源的Serlvet容器，近年来爆出多个严重的安全漏洞，其中包括多个DoS漏洞和信息泄露漏洞。

开源软件安全的思考

OWASP 中国
The Open Web Application Security Project

基础原材料

2010年，Gartner采访了来自11个国家的547位公司负责人，在被调查的公司当中超过一半采用了开源软件作为其IT战略的组成部分。

安全性

2012年，Aspect Security和Sonatype公开的一份调查报告显示，最受欢迎的31个开源项目中，其不安全的版本被下载了超过4,600万次。

法律风险

开源不等于免费，开源软件许可协议背后的条条框框你清楚吗？

开源项目检测计划

OWASP 中国
The Open Web Application Security Project

开源项目检测计划（www.codesafe.cn）是由360代码卫士团队发起，针对开源项目进行的一项公益安全检测计划，旨在让广大开发者关注和了解开源代码安全问题，提高软件安全开发意识和技能。

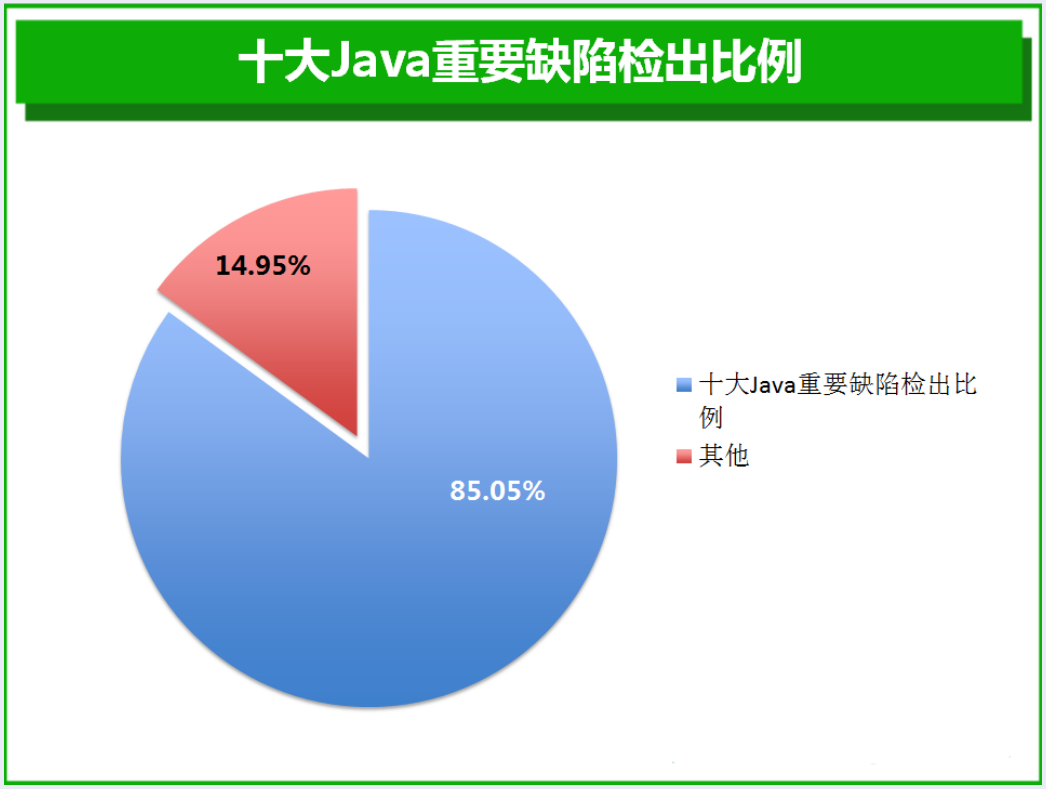
注：开源项目检测计划使用的检测工具是360自主研发的源代码检测引擎“代码卫士”。


OWASP 中国
The Open Web Application Security Project
代码卫士
codesafe
首页
开源项目检测计划
安全资讯
企业服务
免费检测
登录 | 注册
您的专属代码体检专家
代码卫士为每一位开发者提供免费的源代码缺陷检测服务，和您一起打造更安全、更有生命力的源代码
免费检测
目前我们已检测全球 2228 个开源项目 共检测到 2626352 个安全缺陷
开源项目检测计划
安全资讯
企业服务

OWASP 中国
The Open Web Application Security Project

开源项目检测计划—十大JAVA严重缺陷统计

十大 Java 重要缺陷	缺陷总数（个）
SQL 注入	2491
跨站脚本	5011
路径遍历	17852
密码管理	21273
HTTP 消息头注入	3106
命令注入	765
资源注入	12555
资源未释放	75450
系统信息泄露	113429
跨站请求伪造	10157
总计	262089



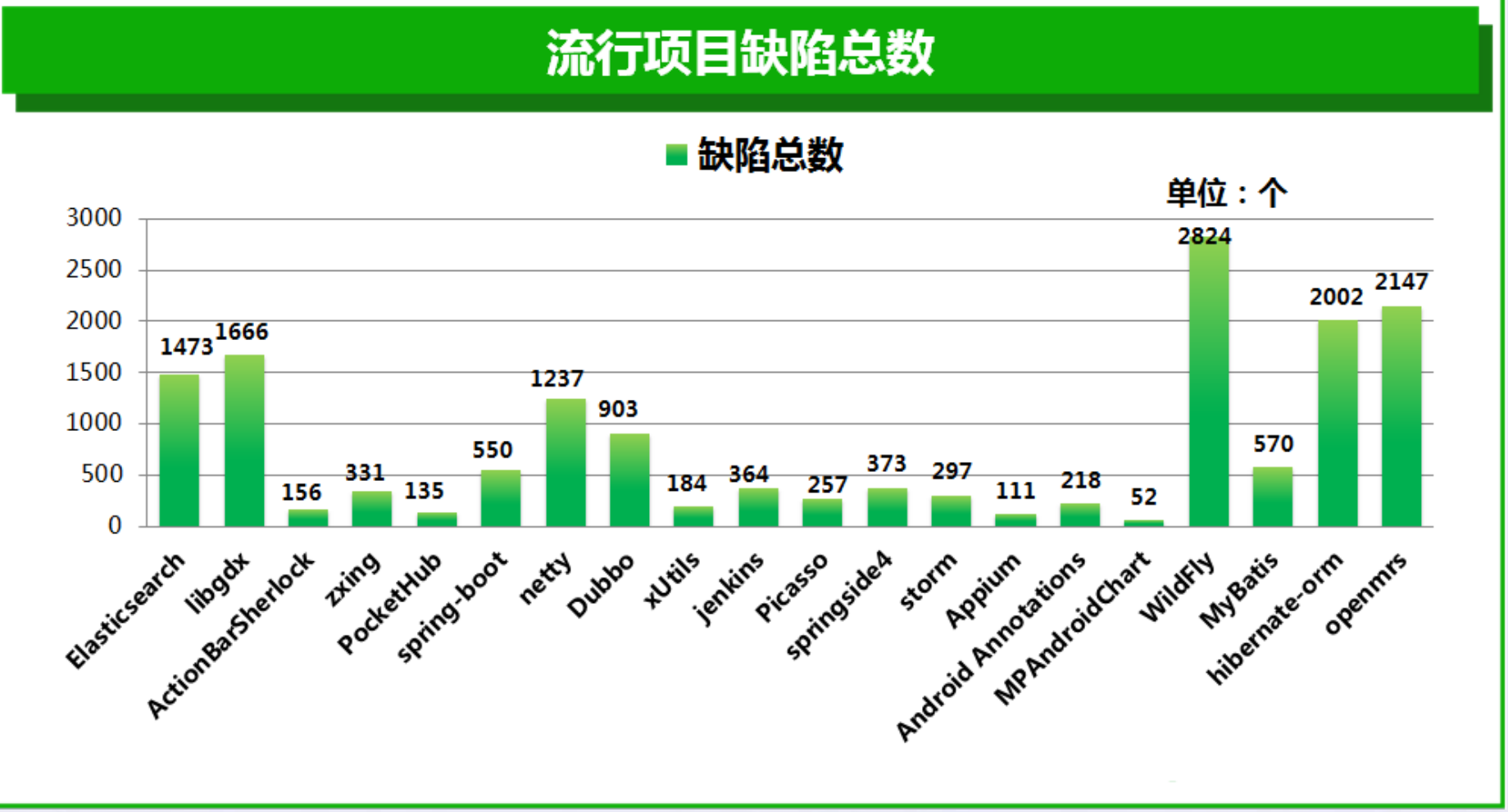
OWASP 中国
The Open Web Application Security Project

开源项目检测计划—20个流行项目

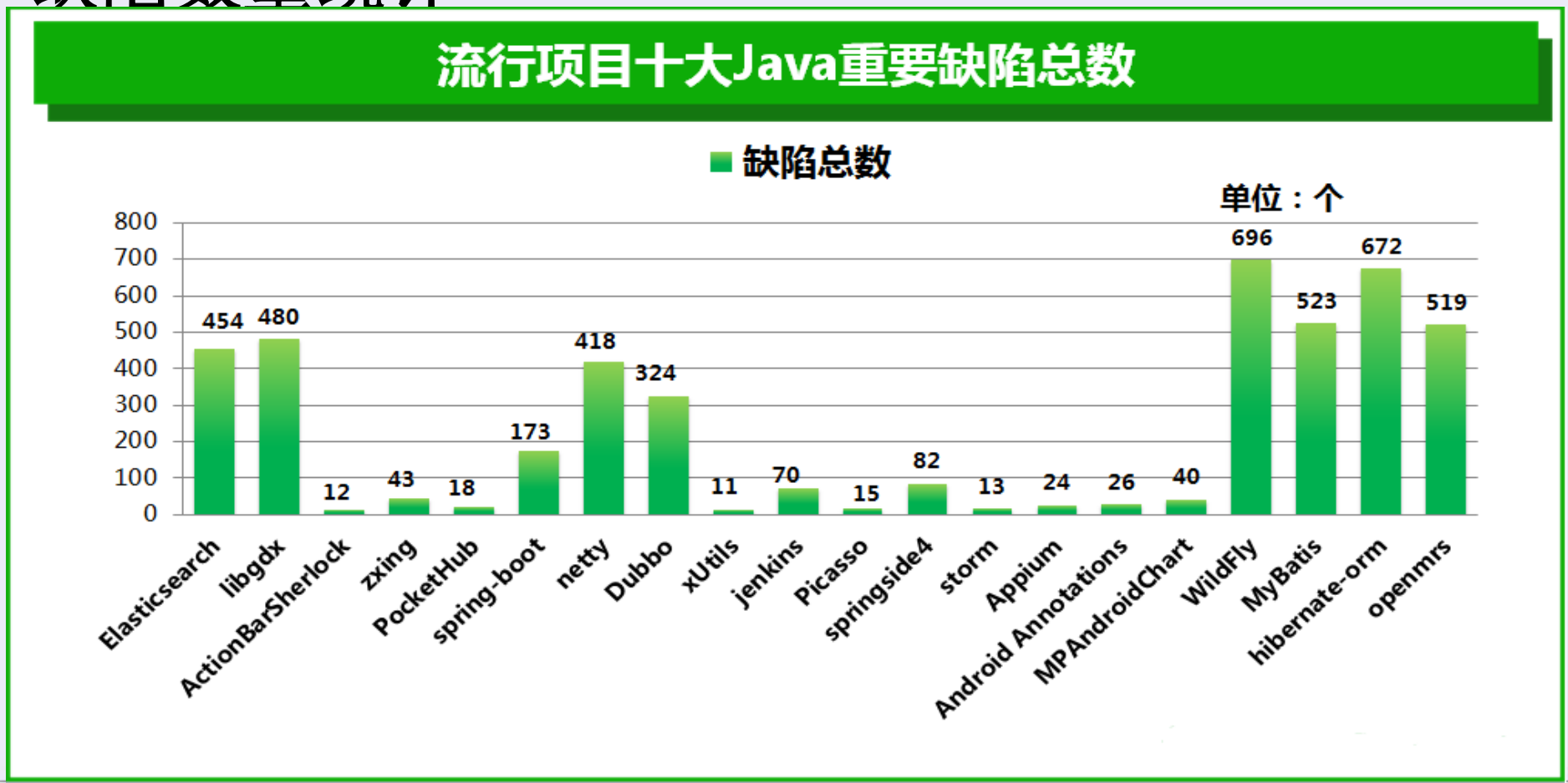
序号	项目名称	Fork	Star	Watch	版本号	缺陷总数
1	Elasticsearch	4132	12758	1303	1.5.1	1473
2	libgdx	4100	7008	936	1.5.0	1666
3	ActionBarSherlock	4096	7029	853	4.4.0	156
4	zxing	3854	6116	813	3.1.0	331
5	PocketHub	3441	6950	1054	1.9.0	135
6	spring-boot	3020	2903	501	1.3.0.M1	550
7	netty	2655	5223	813	4.0.24.Final	1237
8	Dubbo	2459	2321	758	2.5.3	903
9	xUtils	2288	3046	546	2.6.14	184
10	jenkins	2119	4532	577	1.616	364
11	Picasso	2042	7278	682	2.5.2	257
12	springside4	2038	2757	747	4.2.3.GA	373
13	storm	1748	8686	1174	0.9.0.1	297
14	Appium	1744	2600	464	1.4.10	111
15	Android Annotations	1739	6024	617	3.3.2	218
16	MPAndroidChart	1731	5194	395	2.0.8	52
17	WildFly	1432	1439	172	10.0.0.Beta1	2824
18	MyBatis	1401	1657	409	3.3.0	570
19	hibernate-orm	1172	1374	201	5.0.0.CR4	2002
20	openmrs	1161	234	82	1.9.1	2147

OWASP 中国
The Open Web Application Security Project

开源项目检测计划—20个流行项目缺陷总数统计

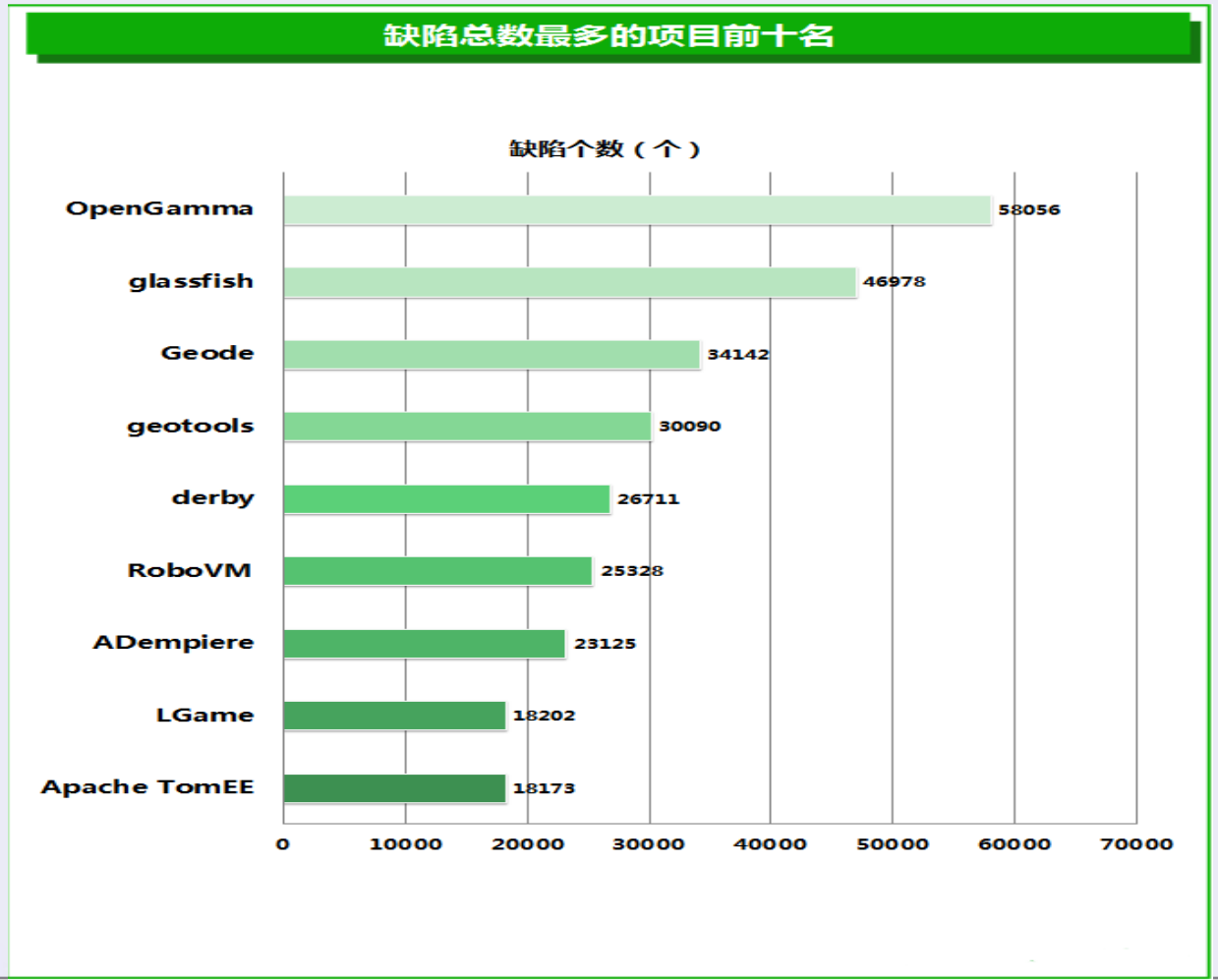


开源项目检测计划—20个流行项目十大JAVA重要缺陷数量统计



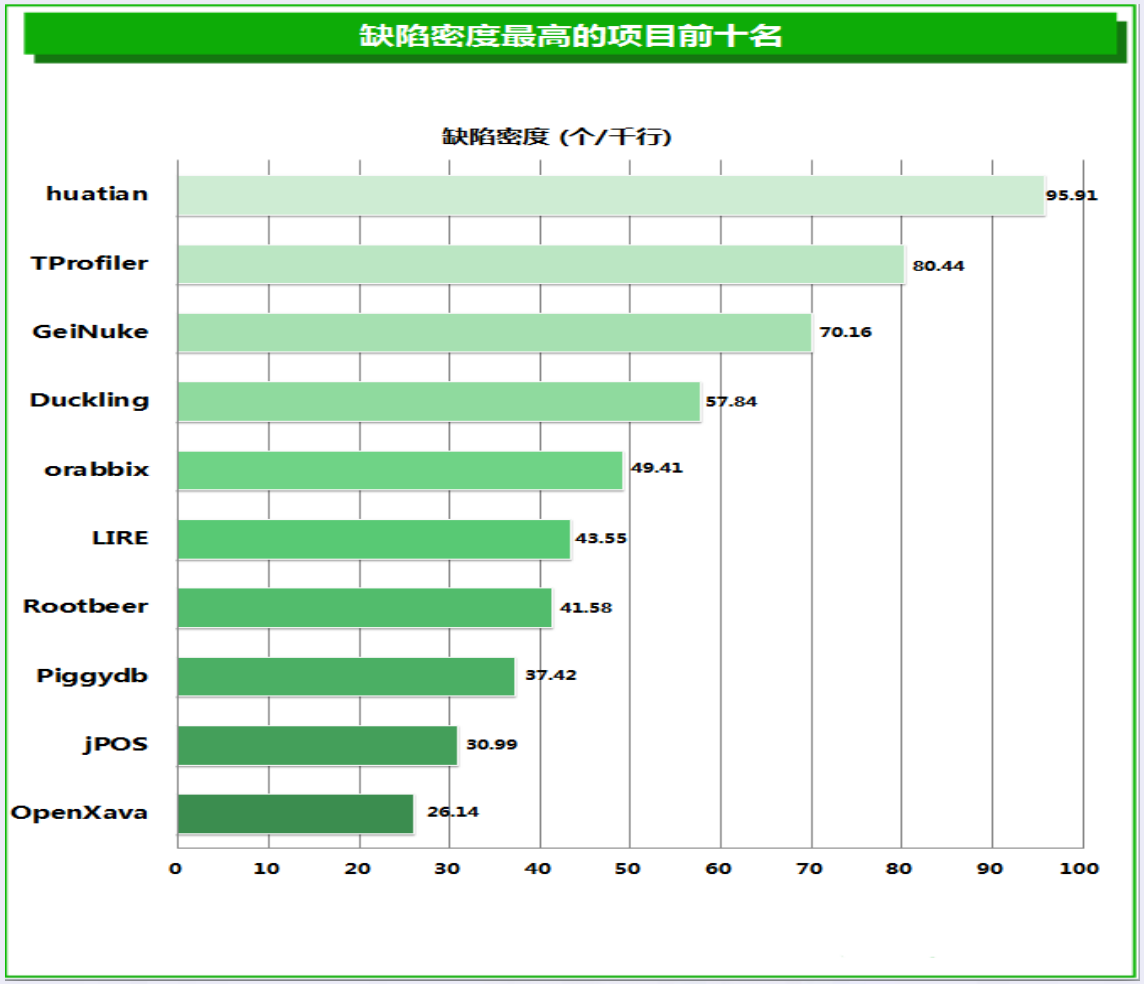
OWASP 中国
The Open Web Application Security Project

开源项目检测计划—缺陷数量TOP 10项目



OWASP 中国
The Open Web Application Security Project

开源项目检测计划—缺陷密度TOP 10项目



OWASP 中国
The Open Web Application Security Project

实例分析1—某开源论坛项目XSS漏洞

```
14 <form id="create_form" action="${baseUrl!}/topic/save" method="post">
15     <select name="sid" id="sid" class="form-control" style="width: 20%; margin-bottom: 5px;">
16         <#list sections as section>
17             <option value="${section.id}">${section.name}</option>
18         </#list>
19     </select>
20     <input type="text" placeholder="标题字数10字以上" id="title" name="title" class="form-control"
21     <input type="text" placeholder="原文地址（原创可不写）" id="original_url" name="original_url" cla
22     <div id="content" style="margin-bottom: 5px;"><textarea name="content"></textarea></div>
23     <input type="button" onclick="submitForm()" value="提交" class="btn btn-primary">
24 </form>
```

```
108         String content = getPara("content");
109         String original_url = getPara("original_url");
110         Topic topic = new Topic();
111         topic.set("id", StrUtil.getUUID())
112                 .set("in_time", new Date())
113                 .set("s_id", sid)
114                 .set("title", title)
115                 .set("content", content)
```

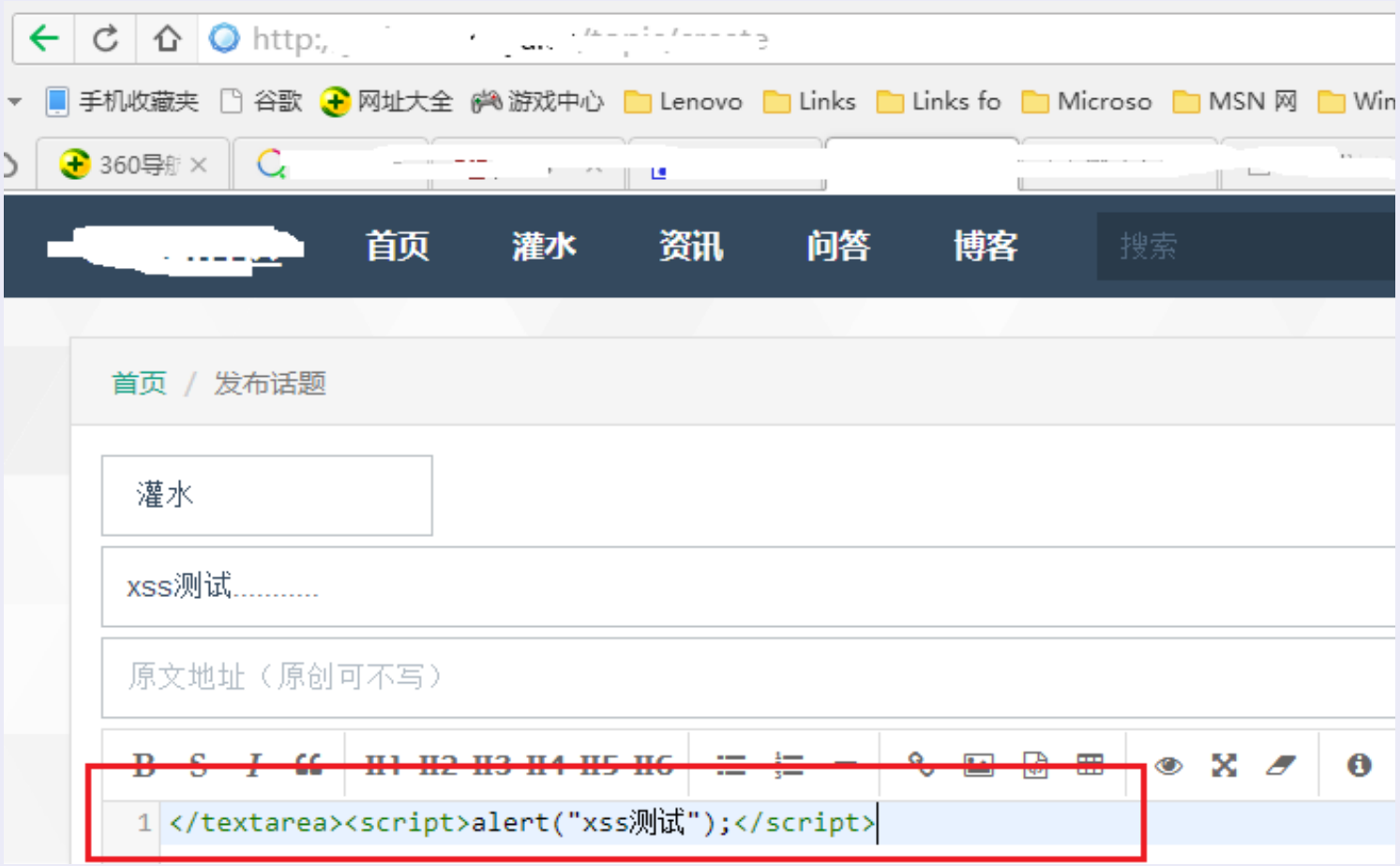
```
109     public String getPara(String name) {
110         return request.getParameter(name);
111     }
```

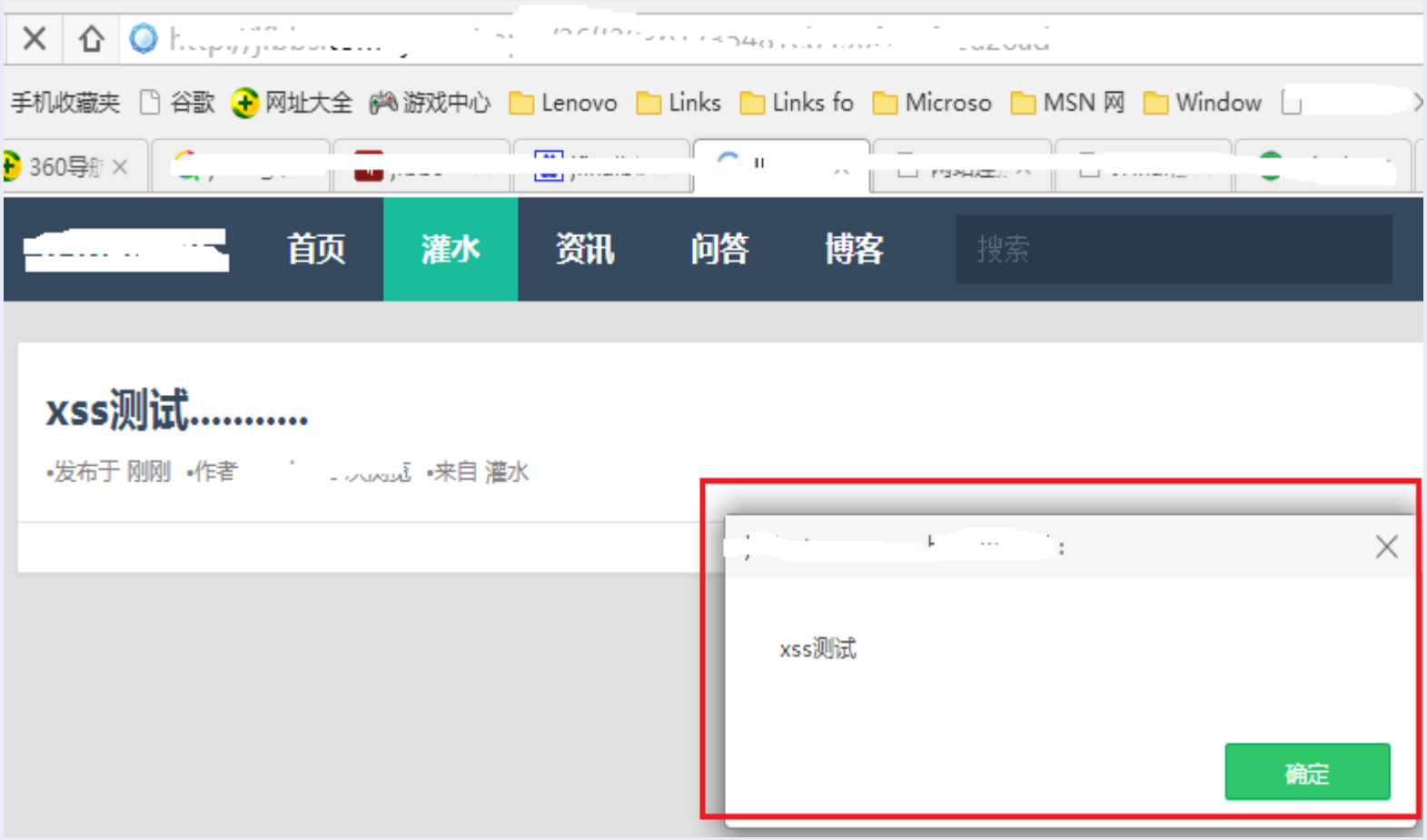
```
24          Topic topic = Topic.me.findByIdWithUser(id);
25          if (topic != null) {
26              List<Reply> replies = Reply.me.findByTid(id);
27              setAttr("topic", topic);
```

```
74      public Controller setAttr(String name, Object value) {
75          request.setAttribute(name, value);
76          return this;
77      }
```

```
62      <div class="panel-body" style="border-top: 1px #E5E5E5 solid; padding-top: 10px">
63          <div id="topic_content">
64              <textarea id="_topic_content" style="display: none;">${topic.content!}</textarea>
65          </div>
66          <#if topic.reposted?? && topic.reposted == 1>
```

XSS


首页
灌水
资讯
问答
博客
搜索
首页 / 发布话题
灌水
xss测试.....
原文地址（原创可不写）
</textarea><script>alert("xss测试");</script>


手机收藏夹
谷歌
网址大全
游戏中心
Lenovo
Links
Links fo
Microso
MSN 网
Window
首页
灌水
资讯
问答
博客
搜索
xss测试.....
•发布于 刚刚 •作者
•来自 灌水
xss测试
确定

实例分析2—某开源流媒体解析工具包类型混淆漏洞

- 该开源软件用于解析RTMP流媒体，支持RTMP协议及RTMPT/RTMPS/RTMPE等各类变种。
- 该开源软件及其库文件被广泛用于各种视频及音频流解决方案，如FFmpeg(被多款音视频播放器所使用)、 SMPlayer(ubuntu14.x默认自带)、Mplayer等。
- 该开源软件类型混淆漏洞的产生是由于对union的不当使用，导致double数据的高八位被当作指针进行处理，进而导致对非法内存地址的访问。

```
02909: HandleInvoke(RTMP *r, const char *body, unsigned int nBodySize)
02910: {
02911:   AMFObject obj;
02912:   AVal method;
02913:   double txn;
02914:   int ret = 0, nRes;
02915:   if (body[0] != 0x02)       /* make sure it is a string method name we start with */
02916:     {
02917:       RTMP_Log(RTMP_LOGWARNING, "%s, Sanity failed. no string method in invoke packet",
02918:       __FUNCTION__);
02919:       return 0;
02920:     }
02921:
02922:   nRes = AMF_Decode(&obj, body, nBodySize, FALSE);
```

```
01175:
01176:       nRes = AMFProp_Decode(&prop, pBuffer, nSize, bDecodeName);
01177:       if (nRes == -1)
01178:     bError = TRUE;
```

```
00653:   switch (prop->p_type)
00654:     {
00655:     case AMF_NUMBER:
00656:       if (nSize < 8)
00657:     return -1;
00658:       prop->p_vu.p_number = AMF_DecodeNumber(pBuffer);
00659:       nSize -= 8;
00660:       break;
00661:     case AMF_BOOLEAN:
00662:       if (nSize < 1)
00663:     return -1;
00664:       prop->p_vu.p_number = (double)AMF_DecodeBoolean(pBuffer);
00665:       nSize--;
00666:       break;
00667:     case AMF_STRING:
00668:       {
00669:     unsigned short nStringSize = AMF_DecodeInt16(pBuffer);
00670:
00671:     if (nSize < (long)nStringSize + 2)
00672:       return -1;
00673:     AMF_DecodeString(pBuffer, &prop->p_vu.p_aval);
00674:     nSize -= (2 + nStringSize);
00675:     break;
00676:       }
00677:     case AMF_OBJECT:
00678:       {
00679:     int nRes = AMF_Decode(&prop->p_vu.p_object, pBuffer, nSize, TRUE);
00680:     if (nRes == -1)
```

```
02909: HandleInvoke(RTMP *r, const char *body, unsigned int nBodySize)
02910: {
02911:   AMFObject obj;
02912:   AVal method;
02913:   double txn;
02914:   int ret = 0, nRes;
02915:   if (body[0] != 0x02)       /* make sure it is a string method name we start with */
02916:     {
02917:       RTMP_Log(RTMP_LOGWARNING, "%s, Sanity failed. no string method in invoke packet",
02918:       __FUNCTION__);
02919:       return 0;
02920:     }
02921:
02922:   nRes = AMF_Decode(&obj, body, nBodySize, FALSE);
```

```
03104:   else if (AVMATCH(&method, &av_onStatus))
03105:     {
03106:       AMFObject obj2;
03107:       AVal code, level;
03108:       AMFProp_GetObject(AMF_GetProp(&obj, NULL, 3), &obj2);
03109:       AMFProp_GetString(AMF_GetProp(&obj2, &av_code, -1), &code);
03110:       AMFProp_GetString(AMF_GetProp(&obj2, &av_level, -1), &level);
```

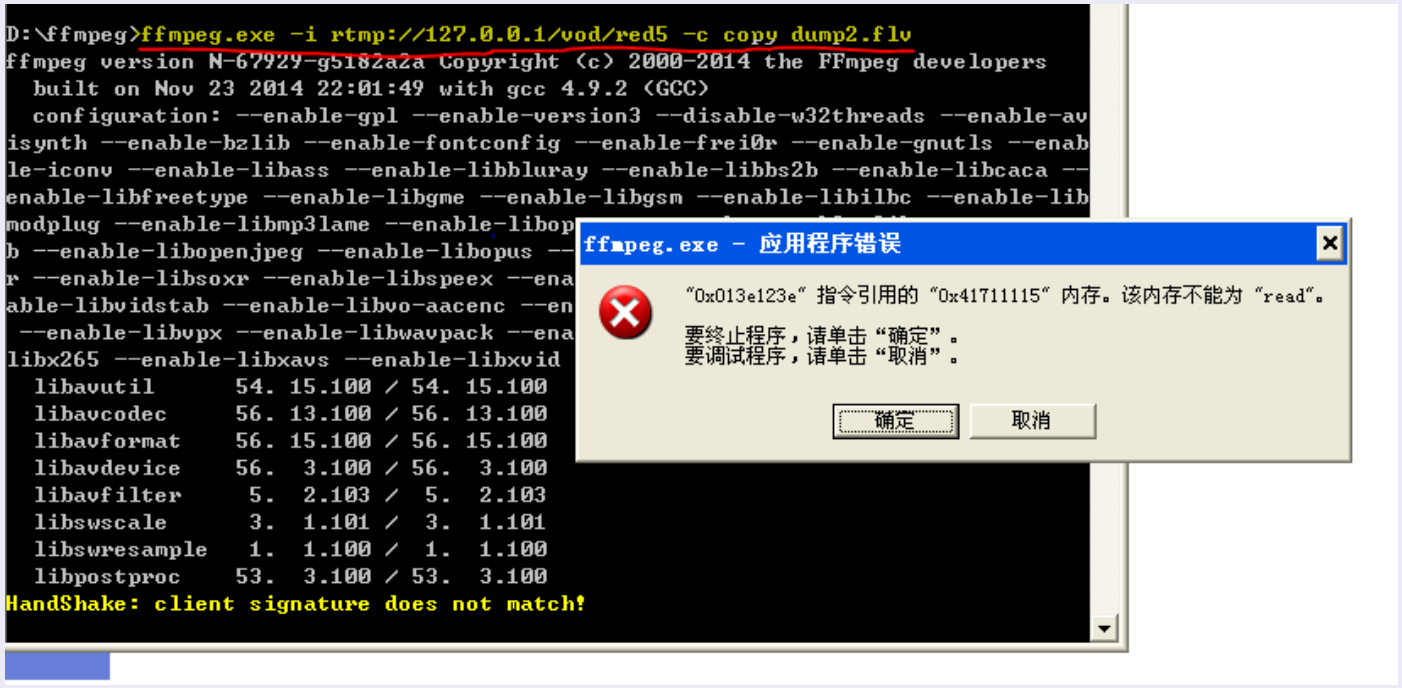
```
AMF_GetProp(AMFObject *obj, const AVal *name, int nIndex)
{
  if (nIndex >= 0)
    {
      if (nIndex < obj->o_num)
      return &obj->o_props[nIndex];
    }
  else
    {
      int n;
      for (n = 0; n < obj->o_num; n++)
    {
      if (AVMATCH(&obj->o_props[n].p
      return &obj->o_props[n];
    }
    }

  return (AMFObjectProperty *)&AMFProp_Invalid;
} ? end AMF_GetProp ?
```

Crash

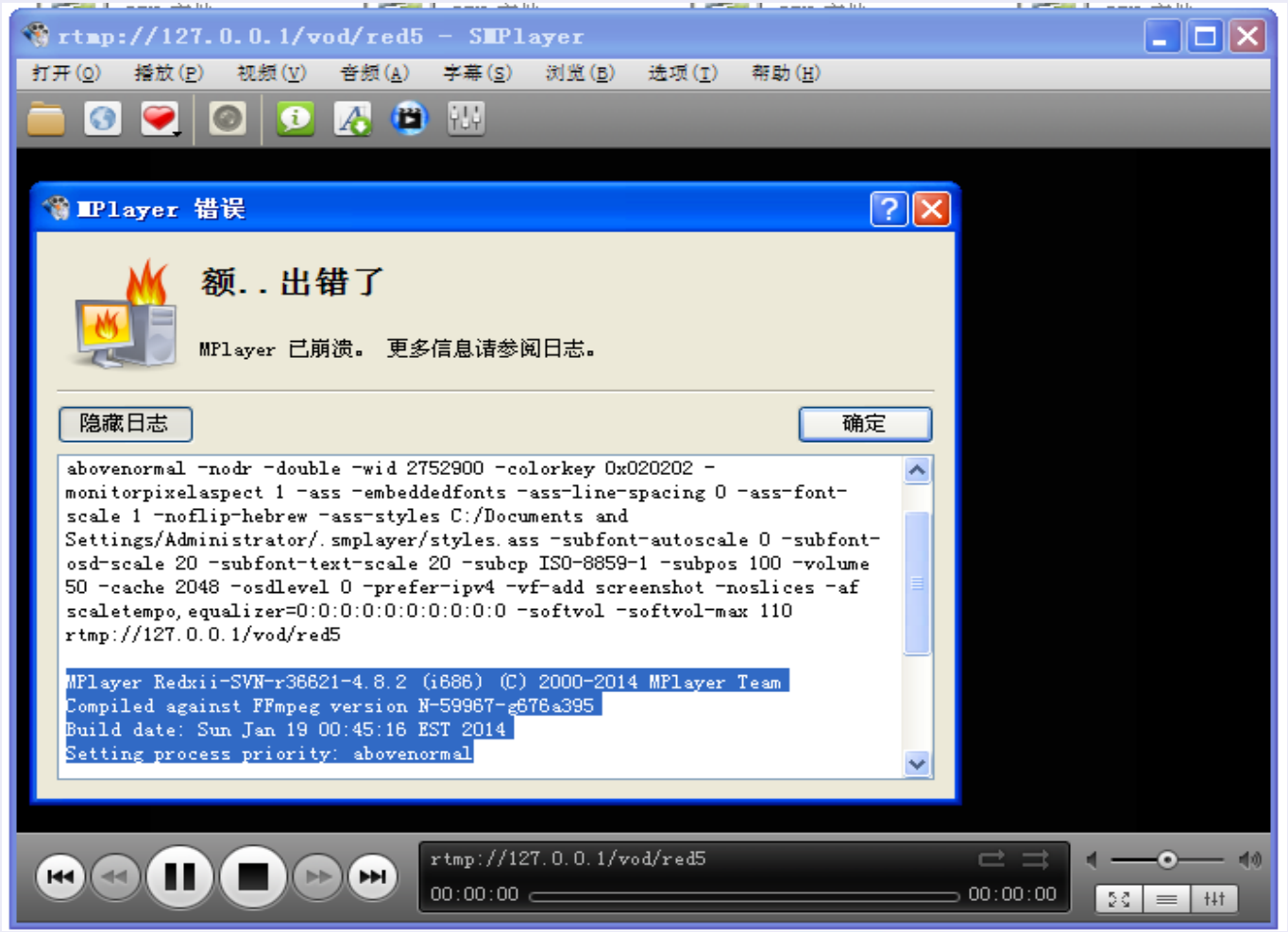
```
00347: AMFProp_GetObject(AMFObjectProperty *prop, AMFObject *obj)
00348: {
00349:   *obj = prop->p_vu.p_object;
00350: }
```

Windows平台下Ffmpeg验证结果：



可以看到FFmpeg程序崩溃，并指出了对地址0x41711115的非法访问

Windows平台下SMPlayer验证结果：



Ubuntu自带播放器SMPlayer验证结果：

OWASP 中国
The Open Web Application Security Project

1、开源代码安全现状和实例分析

2、针对软件生产安全的思考

3、探索软件安全在高校中的方法

OWASP 中国
The Open Web Application Security Project

软件安全生产中的问题

软件原材料（开源软件）是否安全

软件开发过程中是否遵守安全开发规范

软件成品中是否存在缺陷和漏洞

软件开发时有没有统一的安全管理策略

软件安全生产的思考

- 互联网+、中国制造2025时代背景下，人类现实社会与赛博空间（CyberSpace）逐步融合，形成真正的“信息社会”
  - 现实社会中，建筑、实物、人等是基本元素
  - 赛博空间中，软件是最重要的基本元素
  - 信息社会中，人类的生产、生活与互联网融合为一体，建筑、实物、人、软件同等重要
- “信息社会”时代，对软件的安全性提出了更高的要求。软件不再是虚拟的存在，它将与人的生产、生活直接关联，看得见摸得着，是支撑起信息社会正常运转的基本元素！

- 现实社会中，建筑、实物的生产按照工业标准严格对待，人的孕育要经过一系列周密详尽的孕期检查。
- 需要重新思考软件的生产过程：软件同样需要“质量安全（QS）”，软件同样需要“优生优育”。

- 软件生产安全的三要素：
  - 原材料安全（开源代码安全）
  - 安全生产规程（安全编码规范）
  - 软件成品检测（缺陷检测+漏洞检测）

软件安全生产的“3+1”解决方案

OWASP 中国
The Open Web Application Security Project

- 3=原材料安全检测+安全生产规程检查+软件成品检测；
  1=统一的代码质量监控中心，提供目标管理、持续监测、差距分析、修复跟踪等一系列软件代码的安全可视化管理

  - 原材料安全检测
    - 软件的原材料是否安全，是否有“三聚氰胺”
    - 开源代码漏洞检查
      - » 知识产权风险
      - » 已知的安全漏洞

- 安全生产规程检查
  - 软件是否按照安全规范“生产”
  - 编码的合规性检测
    - » 国际标准：CERT C/C++/Java/Android；MISRA C/C++；ISO/IEC
    - » 企业自有标准
- 软件成品检测
  - 生产的软件中是否存在安全瑕疵
  - 代码缺陷检测+漏洞检测
    - » 源代码缺陷分析
    - » 可执行码分析：静态分析、模糊测试
- 统一代码质量监控中心
  - 安全生产的过程管理
  - 安全开发生命周期管理
    - » 目标管理、策略管理、持续监测、流程整合

OWASP 中国
The Open Web Application Security Project

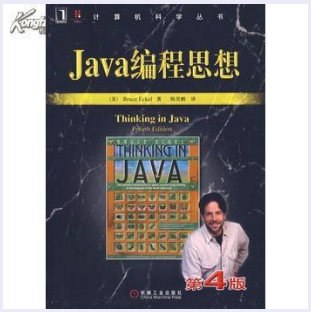
1、开源代码安全现状和实例分析

2、针对软件生产安全的思考

3、探索软件安全在高校中的方法

软件安全在高校中的方法

OWASP 中国
The Open Web Application Security Project



安全编码教材制定

联合教学

搭建实验平台

企业实习

OWASP 中国
The Open Web Application Security Project

Thanks!