Fortify Security Report

2024-6-21

ASUS

Executive Summary

Issues Overview

On 2024-6-19, a source code review was performed over the AdAway code base. 1,769 files, 4,619 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 7 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
High	6
Critical	1

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: E:/workspace/repos/AdAway
Number of Files: 1769
Lines of Code: 4619
Build Label: <No Build Label>

Scan Information

Scan time: 01:38
SCA Engine version: 20.1.1.0007
Machine Name: DESKTOP-MK5UPFE
Username running scan: ASUS

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
Command Line Arguments:
null.null.null
null.AndroidLocaleChecker.main

File System:
null.file.__init__
java.io.FileInputStream.FileInputStream
java.io.FileInputStream.FileInputStream

Stream:
java.io.FileInputStream.read
java.io.InputStream.read

System Information:
null.null.null
java.lang.System.getProperty
java.lang.Throwable.getMessage
os.null.getcwd

Filter Set Summary

Current Enabled Filter Set:
Quick View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High
If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:
If impact is not in range [2.5, 5.0] Then hide issue
If likelihood is not in range (1.0, 5.0] Then hide issue

Audit Guide Summary

J2EE Bad Practices

Hide warnings about J2EE bad practices.
Depending on whether your application is a J2EE application, J2EE bad practice warnings may or may not apply. AuditGuide can hide J2EE bad practice warnings.
Enable if J2EE bad practice warnings do not apply to your application because it is not a J2EE application.

Filters:
If category contains j2ee Then hide issue
If category is race condition: static database connection Then hide issue

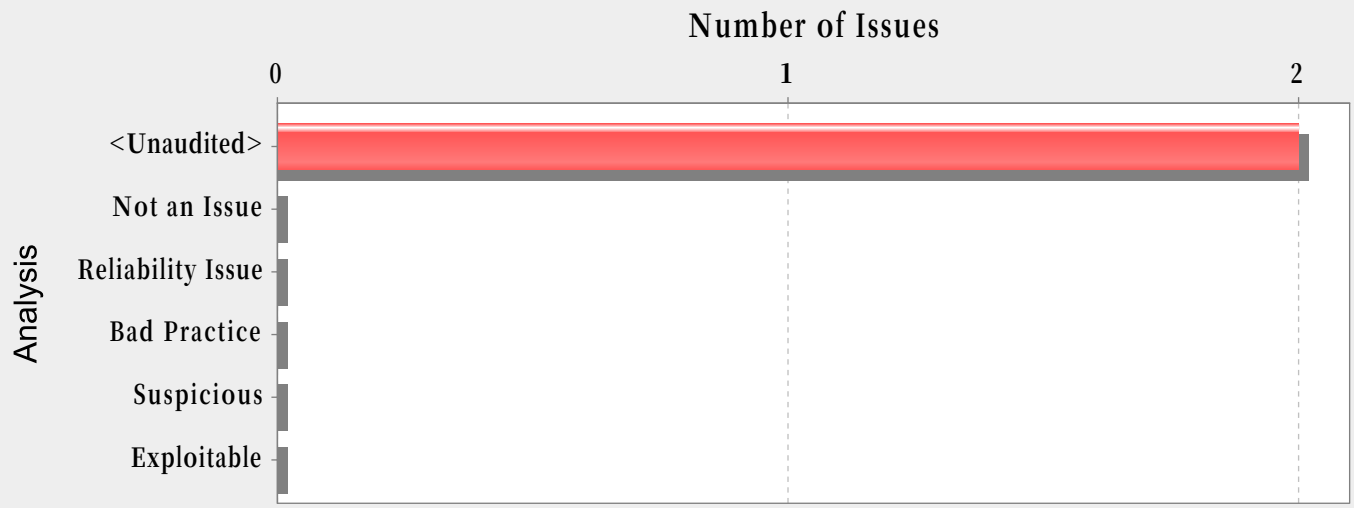
Results Outline

Overall number of results

The scan found 7 issues.

Vulnerability Examples by Category

Category: Android Bad Practices: Missing Component Permission (2 Issues)



Abstract:

在 AndroidManifest.xml 的第 147 行，程序未用导出标记或权限对此公共组件进行保护。

Explanation:

任何应用程序都能访问在显式定义中未明确分配访问权限的公共组件。

Android 应用程序中活动、接收者和服务组件的 exported 属性的默认值取决于是否存在 intent-filter。如果存在 intent-filter，则说明该组件供外部使用。因此应将 exported 属性设为 true。现在，Android 平台上的任何其他应用程序都可以访问该组件。

示例 1：以下是一个 Android 活动示例，该示例存在 intent-filter 且未设置明确访问权限。

```
<activity android:name=".AndroidActivity"/>
<intent-filter android:label="activityName"/>
<action android:name=".someFunAction"/>
</intent-filter>
...
</activity>
```

该活动可能被恶意应用程序利用。

Recommendations:

没有明确的访问权限的组件应该为异常。除非该组件确实需要被所有应用程序访问，否则开发人员应该通过在显式文件中明确定义访问权限来保护组件，以免其被恶意应用程序滥用。

示例 2：下面是Example 1 中使用明确分配的访问权限进行重写的活动组件声明。

```
<activity android:name=".AndroidActivity" android:permission="FriendsOnly"/>
<intent-filter android:label="activityName"/>
<action android:name=".someFunAction"/>
</intent-filter>
...
</activity>
```

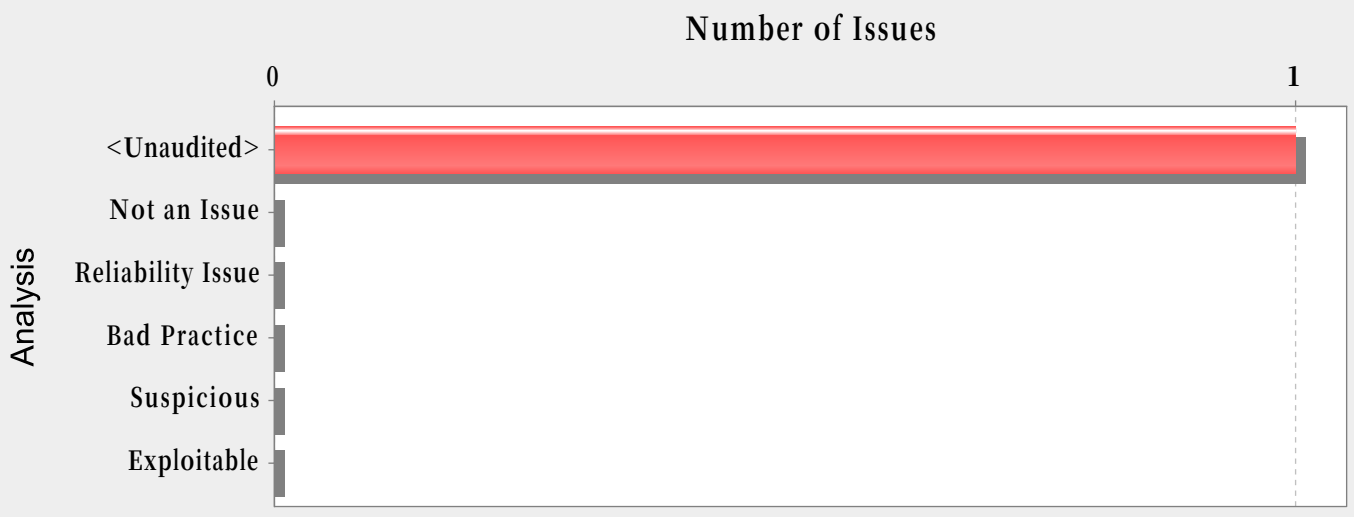
AndroidManifest.xml, line 147 (Android Bad Practices: Missing Component Permission)

Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	在 AndroidManifest.xml 的第 147 行，程序未用导出标记或权限对此公共组件进行保护。		

Sink: AndroidManifest.xml:147 null()

```
145     <receiver
146       android:name=".broadcast.BootReceiver"
147       android:exported="true">
148       <intent-filter>
149         <action android:name="android.intent.action.BOOT_COMPLETED" />
```

Category: Android Bad Practices: Missing Google Play Services Updated Security Provider (1 Issues)



Abstract:

应用程序不使用 Google Play 服务更新的安全提供程序，这可能使其未来易遭受 OpenSSL 库中漏洞的攻击。

Explanation:

Android 依赖于可提供安全网络通信的安全提供程序。 但是，有时漏洞存在于默认安全提供程序中。 为了防范这些漏洞，Google Play 服务可提供用于自动更新设备安全提供程序的方法，以防御已知盗取手段。 通过调用 Google Play 服务方法，您的应用程序可以确保其在具有最新更新的设备上运行，以防御已知盗取手段。

Recommendations:

修补安全提供程序最简单的方法是调用同步法 installIfNeeded()。 如果在等待操作完成的过程中用户体验不会受到线程阻止的影响，则此方法适用，否则它应该以异步方式完成。

示例：以下代码可实现用于更新安全提供程序的同步适配器。 由于同步适配器在后台运行，因此在等待安全提供程序更新的过程中若出现线程阻止也没有影响。 同步适配器调用 installIfNeeded() 以更新安全提供程序。 如果方法正常返回，则同步适配器了解安全提供程序为最新程序。 如果方法抛出异常，则同步适配器可采取相应的操作（如提示用户更新 Google Play 服务）。

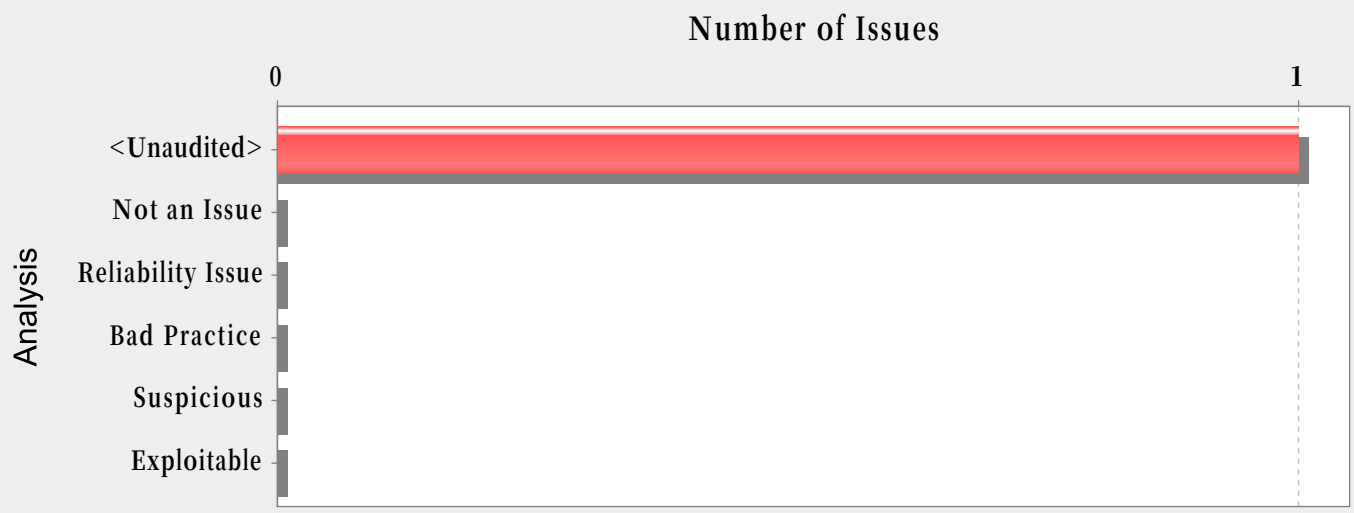
```
public class SyncAdapter extends AbstractThreadedSyncAdapter {

...

// This is called each time a sync is attempted; this is okay, since the
// overhead is negligible if the security provider is up-to-date.
@Override
public void onPerformSync(Account account, Bundle extras, String authority, ContentProviderClient provider, SyncResult
syncResult) {
try {
ProviderInstaller.installIfNeeded(getContext());
} catch (GooglePlayServicesRepairableException e) {

// Indicates that Google Play services is out of date, disabled, etc.

// Prompt the user to install/update/enable Google Play services.
GooglePlayServicesUtil.showErrorNotification(e.getConnectionStatusCode(), getContext());

// Notify the SyncManager that a soft error occurred.
syncResult.stats.numIOExceptions++;
return;

} catch (GooglePlayServicesNotAvailableException e) {
// Indicates a non-recoverable error; the ProviderInstaller is not able
// to install an up-to-date Provider.

// Notify the SyncManager that a hard error occurred.
syncResult.stats.numAuthExceptions++;
return;
}

// If this is reached, you know that the provider was already up-to-date,
```

```
// or was successfully updated.
}
}
```

AndroidManifest.xml, line 51 (Android Bad Practices: Missing Google Play Services Updated Security Provider)			
Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	应用程序不使用 Google Play 服务更新的安全提供程序，这可能使其未来易遭受 OpenSSL 库中漏洞的攻击。		
Sink:	AndroidManifest.xml:51 null()		

```
49      android:supportsRtl="true"
50      android:theme="@style/Theme.AdAway"
51      tools:ignore="GoogleAppIndexingWarning">
52      <activity
53        android:name=".ui.home.HomeActivity"
```

Category: Android Bad Practices: Unnecessary Component Exposure (1 Issues)



Abstract:
尽管没有必要，但此接收者组件可供第三方组件访问，从而增加了恶意信息注入的风险。

Explanation:
由于此组件只应接受系统而非其他组件的广播消息，因而它应是专用的（其他组件无法访问）。

Recommendations:
将 android:exported 标记设为 'false' 并将 android:enabled 标记设为 'true'。通过这样设置，可以使组件接收系统广播而非第三方广播（无论显式还是隐式消息状态）。

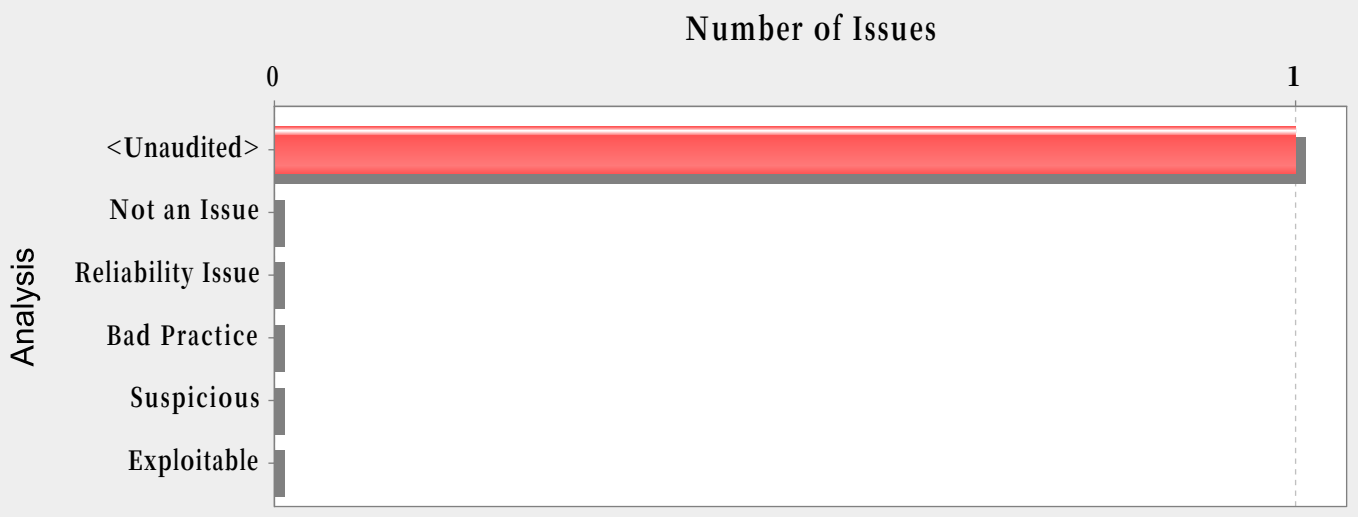
AndroidManifest.xml, line 147 (Android Bad Practices: Unnecessary Component Exposure)

Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	尽管没有必要，但此接收者组件可供第三方组件访问，从而增加了恶意信息注入的风险。		

Sink: AndroidManifest.xml:147 null()

```
145     <receiver
146       android:name=".broadcast.BootReceiver"
147       android:exported="true">
148       <intent-filter>
149         <action android:name="android.intent.action.BOOT_COMPLETED" />
```

Category: Insecure Transport (1 Issues)



Abstract:
该应用程序会使用未加密的协议（而非加密的协议）与服务器通信。

Explanation:
所有利用 HTTP、FTP 或 Gopher 的通信均未经过身份验证和加密。因此可能面临风险，特别是在移动环境中，设备要利用 WiFi 连接来频繁连接不安全的公共无线网络。

示例 1：以下 Spring Boot 配置文件禁用 TLS 协议，因此使用 HTTP 协议。

```
server.ssl.enabled=false
```

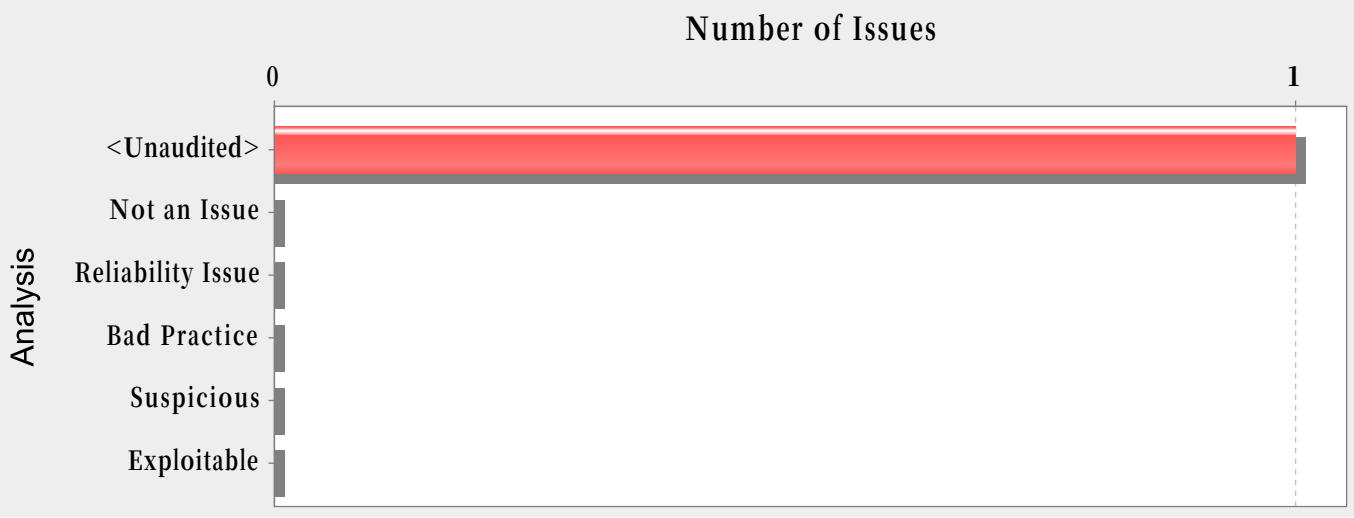
Recommendations:
应尽可能使用 HTTPS 等安全协议与服务器交换数据。

network_security_config.xml, line 3 (Insecure Transport)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Security Features		
Abstract:	该应用程序会使用未加密的协议（而非加密的协议）与服务器通信。		
Sink:	network_security_config.xml:3 null()		

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <network-security-config>
3    <domain-config>
4      <domain includeSubdomains="true">localhost</domain>
5      <trust-anchors>
```

Category: Privilege Management: Android Network (1 Issues)



Abstract:
程序在 AndroidManifest.xml 的第 31 行请求建立网络连接的权限。

Explanation:
授予此权限会使该软件能够打开网络套接字。这个权限会授予程序
对设备的控制权，从而对用户造成负面影响。因为此类型的权限会带来潜在风险，系统不会自动将此权限授予请求者
。

示例 1：以下 AndroidManifest.xml 中的 <uses-permission .../> 元素包含一个网络权限属性。

```
<uses-permission android:name="android.permission.INTERNET"/>
```

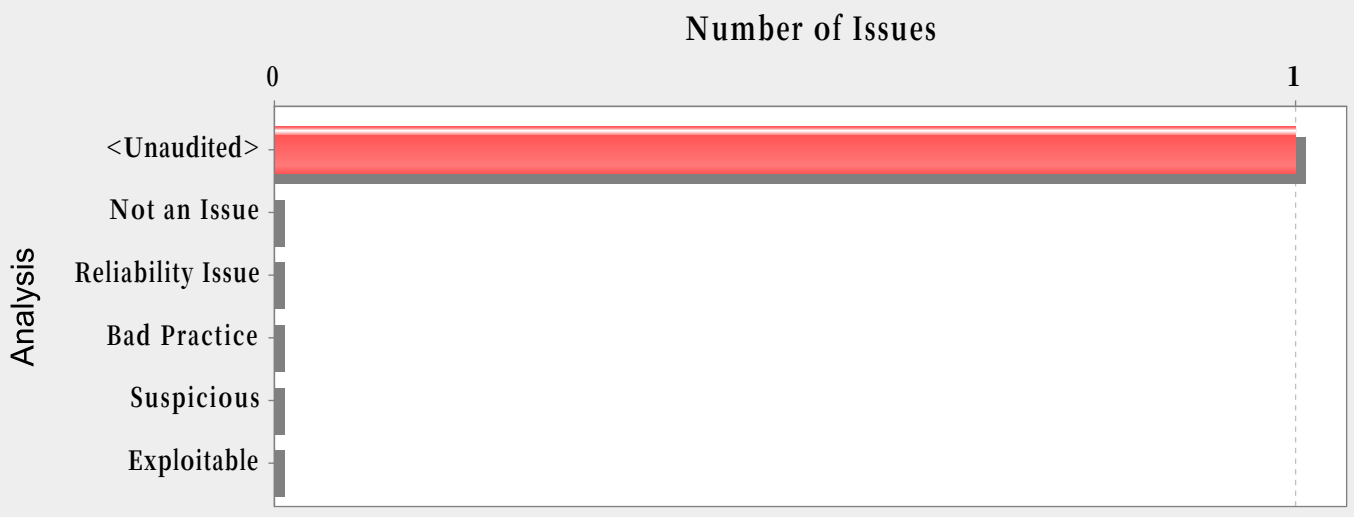
Recommendations:
请求此权限时请慎重考虑。如果程序不需要此权限，
用户可能会拒绝安装。

AndroidManifest.xml, line 31 (Privilege Management: Android Network)

Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	程序在 AndroidManifest.xml 的第 31 行请求建立网络连接的权限。		
Sink:	AndroidManifest.xml:31 null()		

```
29      android:xlargeScreens="true" />
30
31    <uses-permission android:name="android.permission.INTERNET" />
32    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
33    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
```

Category: Privilege Management: Unnecessary Permission (1 Issues)



Abstract:
应用程序若不能遵守最低权限原则，便会大大增加引发其他漏洞的风险。

Explanation:
应用程序应仅拥有正常执行所需的最小权限。权限过多会导致用户不愿意安装该应用程序。此权限对于该程序可能是不必要的。

Recommendations:
考虑应用程序是否需要请求的权限来保证正常运行。如果不需要，则应将相应的权限从 AndroidManifest.xml 文件中删除。除了请求应用程序真正需要的权限之外，切忌因请求更多权限而导致对应用程序过度授权。这会导致在设备上安装的其他恶意应用程序利用这种过度授权的应用程序对用户体验及存储的数据造成负面影响。另外，设置过多的权限可能会适得其反，导致客户不愿意安装您的应用程序。

AndroidManifest.xml, line 32 (Privilege Management: Unnecessary Permission)

Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	应用程序若不能遵守最低权限原则，便会大大增加引发其他漏洞的风险。		
Sink:	AndroidManifest.xml:32 null()		

```
30
31    <uses-permission android:name="android.permission.INTERNET" />
32    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
33    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
34    <uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
```

Issue Count by Category

Issues by Category	
Android Bad Practices: Missing Component Permission	2
Android Bad Practices: Missing Google Play Services Updated Security Provider	1
Android Bad Practices: Unnecessary Component Exposure	1
Insecure Transport	1
Privilege Management: Android Network	1
Privilege Management: Unnecessary Permission	1

Issue Breakdown by Analysis

Issues by Analysis