



受限

扫描报告

准备人

Ostorlab Official

总体风险 - 高

日期

2026-10-15 18:12:57

执行摘要

该应用  [PlayStore\(owasp.sat.agoat:1.0 \(1\)\)](#) 被评定为 高 | 9.1 风险等级，当前存在 44 个漏洞，0 个为严重风险，12 个为高风险，16 个为中风险，6 个为低风险。另有 8 个潜在漏洞需要进一步调查。为进一步加固应用安全，建议实施 2 项加固措施。

该应用的攻击面 值得关注，因为它使用了 0 权限，其中包括 0 项危险权限。该应用还暴露了 0 个活动，0 个服务，和 0 个广播接收器。此外，它还使用了 0 项功能。请确保您的攻击面是合理的，并删除任何不必要的功能或暴露。

总体而言，应用的覆盖率被评定为 受限。为提高覆盖率，建议运行完整的动态扫描。

应用摘要

平台	 Android
包名	owasp.sat.agoat
版本	1.0 (1)
SHA1哈希	ea96f4adc27083696a449b1c6e5cd612a2cf0f5c
大小	2 MB
日期	五月 19th 2026, 16:40:34

发现摘要

按风险等级分组的漏洞



工单摘要

漏洞状态概览

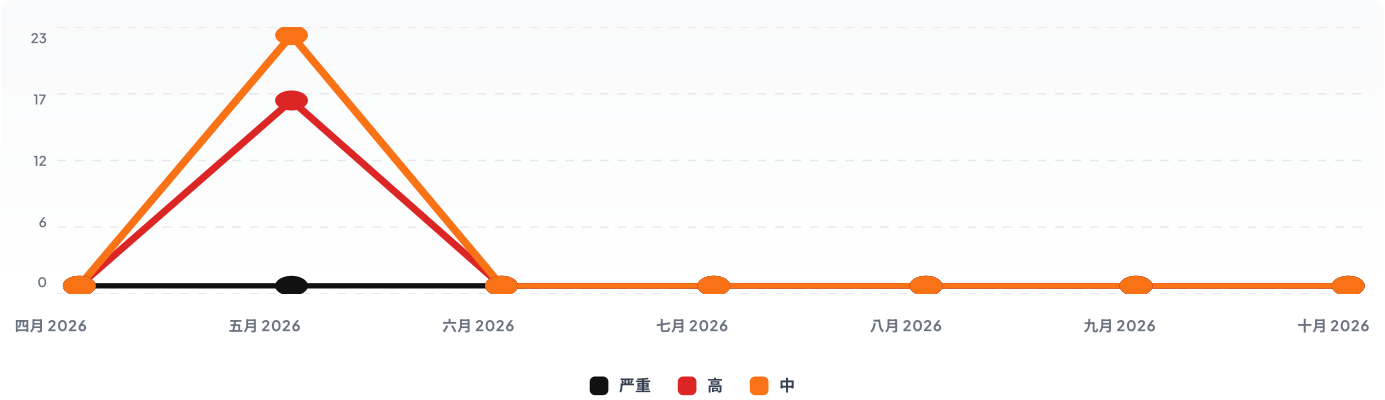


按严重性排列的漏洞Top 10

低熵 OTP 通过系统剪贴板与弱 P...	1
SQLInjectionActivity 中未过滤...	1
隐式导出的受保护 Activity 导致...	1
导出服务允许未授权文件下载 (...	1
敏感凭证以明文形式存储于 Sha...	1
使用未加盐 MD5 哈希保护的认...	1
在 owasp.sat.agoat 中已确认...	1
Raw Authentication Credenti...	1
InsecureStorageSQLiteActivi...	1
公开暴露的 Firebase Database	1

应用漏洞趋势

过去6个月的历史趋势



扫描摘要

目标	Android: owasp.sat.agoat (1.0 (1))
总体风险	HIGH
漏洞	共 34 项 严重 0 项 高危 12 项 中危 16 项 低危 6 项 潜在 8 项 加固 2 项

发现问题汇总

风险 CVSS	状态	标签	标题	简短描述
高	待处理	没有可用的标签。	低熵 OTP 通过系统剪贴板与弱 PRNG 暴露	<code>ClipboardActivity</code> 使用非密码学 PRNG 生成四位 OTP，将其复制到全局剪贴板，并通过 Toast 显示，导致该令牌可被预测，且任何共存应用或 IME 均可访问。
高	待处理	没有可用的标签。	SQLInjectionActivity 中未过滤的 rawQuery 导致的 SQL 注入	SQLInjectionActivity 将 EditText 中的用户输入直接拼接到 rawQuery() 以构造 SELECT 查询，且 selectionArgs 为 null，可导致任意 SQL 执行、Schema 枚举和本地身份验证绕过。
高	待处理	没有可用的标签。	隐式导出的受保护 Activity 导致认证绕过 (AccessControllViewActivity)	AccessControllViewActivity 通过 deep-link intent-filter 被隐式导出，且未包含任何内部身份验证检查，使得任何第三方应用均可在无需 PIN 码的情况下启动该受保护视图。
高	待处理	没有可用的标签。	导出服务允许未授权文件下载 (DownloadInvoiceService)	<code>DownloadInvoiceService</code> 在 manifest 中被显式导出，且会在未对调用方进行任何授权检查的情况下，将文件下载至公共外部存储。
高	待处理	没有可用的标签。	敏感凭证以明文形式存储于 SharedPreferences、SQLite 及外部存储	该应用将用户名、密码及 PIN 哈希值持久化到多种本地存储机制中，且未加密、未使用 KeyStore 保护，亦未设置严格的文件权限。
高	待处理	没有可用的标签。	使用未加盐 MD5 哈希保护的认证凭证 (AccessControllIssueIActivity)	该应用将 PIN 以未加盐的 MD5 摘要形式存储在内部 SharedPreferences 中，使得攻击者可通过离线暴力破解在数毫秒内轻易恢复明文 PIN。
高	待处理	没有可用的标签。	在 owasp.sat.agoat 中已确认存在针对敏感操作的点击劫持	拥有 SYSTEM_ALERT_WINDOW 权限的恶意应用可在四个敏感 Activity 上绘制交互式覆盖层，拦截用户触摸操作并触发未授权的发票下载、促销验证、OTP 生成及 PIN 提交。
高	待处理	没有可用的标签。	Raw Authentication Credentials and SQL Queries Leaked to Android Logcat	多个 Activity 无条件地将原始用户名、密码和 SQL 语句通过 <code>Log.e</code> 、 <code>Log.v</code> 和 <code>System.out.println</code> 写入 Android 系统日志，任何具有日志访问权限的应用或拥有 <code>adb</code> 权限的攻击者均可窃取这些凭证。
高	待处理	没有可用的标签。	InsecureStorageSQLiteActivity 中未过滤的 execSQL 导致的 SQL 注入	InsecureStorageSQLiteActivity 将两个未经净化的 EditText 值拼接后传入 execSQL() 以构建 INSERT 语句，从而导致堆叠查询注入、任意记录篡改以及对本地数据库完整性的完全破坏。
高	待处理	安全	公开暴露的 Firebase Database	公开暴露 Firebase 数据库，可能导致未授权访问、数据篡改、信息泄露、恶意攻击及声誉损失。
高	待处理	隐私 安全	LOGJAM 通用素数漏洞	使用广泛共享的 Diffie-Hellman 素数，使国家级预计攻击能够针对大部分互联网流量。
高	待处理	安全	移动端 SQL 注入漏洞	Mobile SQL Injection vulnerability allows an attacker to execute malicious SQL queries on a mobile application's database, potentially leading to unauthorized access, data leakage, or data manipulation.
中	待处理	没有可用的标签。	管理员权限由客户端布尔标志决定且未经服务端验证	BinaryPatchingActivity gates admin UI functionality using a local boolean field that is never validated by a backend API, cryptographic token, or re-authentication challenge.
中	待处理	没有可用的标签。	TrafficActivity 中的证书固定因异常处理过于宽泛而静默失败	<code>TrafficActivity.doPinning()</code> 中的 OkHttp CertificatePinner 已失效，因为过于宽泛的 catch 块吞没了 <code>SSLPeerUnverifiedException</code> ，从而允许静默的中间人攻击。
中	待处理	没有可用的标签。	不安全的外部存储权限泄露凭证 (InsecureStorageSDCardActivity)	<code>InsecureStorageSDCardActivity</code> 将用户凭证写入外部存储中的临时文件，并显式赋予该文件全局可读和全局可写权限。
中	待处理	没有可用的标签。	敏感界面缺少 FLAG_SECURE，允许未经授权的屏幕截取	owasp.sat.agoat 中的全部 25 个 Activity 均未设置 FLAG_SECURE，导致包含 PIN 输入、凭证录入和 OTP 生成在内的敏感界面可被任意截屏与录屏。

风险 CVSS	状态	标签	标题	简短描述
中	待处理	没有可用的标签。	Insecure Backup and Debug Flags Expose Application Sandbox	应用清单启用了 <code>android:debuggable="true"</code> 和 <code>android:allowBackup="true"</code> ，任何具备 USB 调试权限的用户均可提取、修改并恢复所有应用私有数据，或在运行进程中附加调试器。
中	待处理	没有可用的标签。	应用程序源代码中的硬编码认证凭证	多个硬编码凭证——包括用户名、密码和认证密钥——直接嵌入应用源码，并在运行时通过 exported BroadcastReceiver 暴露。
中	待处理	U 安全	密码学漏洞：弱哈希算法	客户端使用了存在已知加密弱点的弱哈希函数。
中 9.1	待处理	U 安全	明文 HTTP 请求	应用程序正在通过未加密通道访问数据，破坏了传输中数据的机密性和完整性。
中	待处理	隐私 U 安全	证书已过期	SSL/TLS 证书过期后将不再受浏览器和客户端信任，进而引发安全警告、服务中断及潜在安全漏洞，包括中间人攻击。
中	待处理	U 安全	不安全的 HTTP Header 设置	应用头部设置缺失，或配置了不安全的头部值。
中	待处理	U 安全	不安全的 HTTP 标头设置：HTTP 严格传输安全 (HSTS)	不安全的 HTTP 头设置：HTTP Strict Transport Security (HSTS)。
中	待处理	U 安全	支持不安全的 TLS 密码套件	客户端支持包含已知加密弱点的密码套件组合。
中	待处理	隐私 U 安全	不支持 TLS_FALLBACK_SCSV	服务器不支持 TLS_FALLBACK_SCSV，容易受到协议降级攻击。
中	待处理	隐私 U 安全	Weak Cryptographic Key and Signature Algorithm in SSL/TLS Certificate	该 SSL/TLS 证书同时使用了弱加密密钥和弱签名算法，可能危及加密连接的安全性，并导致证书伪造或其他密码学攻击。
中	待处理	隐私 U 安全	滥用移动网络连接	应用强制使用移动网络连接。
中	待处理	U 安全	XML 注入	XML 注入是指不可信用户输入在并入 XML 数据前未经正确验证或清理而引发的漏洞。攻击者可通过注入恶意 XML 内容利用该漏洞，进而导致未授权访问、数据篡改或拒绝服务攻击。
低 5.4	待处理	U 安全	usesCleartextTraffic 属性已设置	应用设置了 usesCleartextTraffic 属性。
低	待处理	U 安全	使用已弃用的 TLS/SSL 协议版本	应用使用了已弃用的 SSL/TLS 协议，已知存在安全与隐私问题。
低	待处理	U 安全	Google Calendar - 暴露	检测到目标中嵌入了公开可访问的 Google Calendar，可能会泄露敏感信息，包括会议详情、与会者姓名、日程安排及内部组织数据。
低 5.4	待处理	隐私	属性 hasFragileUserData 未设置	应用未显式设置 hasFragileUserData 属性。
低 5.4	待处理	U 安全	已弃用的目标 API 版本	应用将 targetSdkVersion 属性设置为允许使用较低版本的 Android API 级别，这可能使用户面临安全漏洞风险。
低	待处理	U 安全 恶意软件 漏洞利用	任务劫持	该应用未针对任务劫持进行防护，设备上的恶意应用可接管其前台任务。
潜在	待处理	没有可用的标签。	ANALYTICS_LIBRARY 存在网络安全配置不安全及传输保护缺失	ANALYTICS_LIBRARY 代理 (TrafficActivity) 通过默认未用证书固定的 OkHttpClient 传输分析流量，同时 Network Security Configuration 全局允许明文 HTTP 并信任用户安装的 CA。该管道缺少载荷加密、请求签名及完整性控制，使得 MitM 拦截、重放与篡改极易实施。
潜在	待处理	U 安全	应用内存储的敏感信息	密码、Token 或其他敏感信息存储在应用中。

风险 CVSS	状态	标签	标题	简短描述
潜在 3.9	待处理	隐私	备份模式已启用	应用正在启用备份模式。
潜在	待处理	安全	文件路径遍历	应用程序存在路径遍历漏洞，攻击者可借此操纵文件路径的某一部分。该被操纵的路径随后被文件系统 APIs 直接使用，且未经适当验证或审查。
潜在	待处理	安全	应用数据不安全存储	应用数据存储不安全，导致其他应用可读取或篡改该数据。
潜在	待处理	安全	源点到汇点	危险的 Sink 方法被潜在的用户可控参数或不可信的 Source 方法调用。
潜在 3.4	待处理	安全	已声明但未配置权限的服务	已声明的服务未受全局权限保护。
潜在	待处理	安全	URL 篡改	攻击者若能控制应用程序用于获取内容的 URL，则会带来安全风险，因其可操纵内容，进而导致恶意代码执行或敏感信息窃取。
加固 9.1	待处理	隐私 安全	不安全的网络配置	应用未指定网络安全配置，或设置了不安全的配置项。
加固 5.4	待处理	安全	调试模式已启用	应用编译时启用了调试模式。

详细发现列表

高 低熵 OTP 通过系统剪贴板与弱 PRNG 暴露

`ClipboardActivity` 使用非密码学 PRNG 生成四位 OTP，将其复制到全局剪贴板，并通过 Toast 显示，导致该令牌可被预测，且任何共存应用或 IME 均可访问。

OTP 仅通过 `CollectionsKt.shuffled` 从 9,000 个可能值中抽取，其底层委托给 `java.util.Random`。此基于 LCG 的生成器可通过连续观测恢复种子。生成的令牌被写入系统剪贴板，且无过期或清理机制，同时显示在 Toast 中。不存在服务端验证后端，因此该 OTP 纯粹为客户端产物，可被剪贴板读取工具截获，也可通过观测 PRNG 序列预测。

技术详情

发现 #1

根因

开发者使用了基于 `java.util.Random` 的便捷 shuffle 方法，而非 `java.security.SecureRandom`。随后，该令牌被放入全局剪贴板，并通过 UI Toast 展示，且未设置任何超时、验证或防重放逻辑。

脆弱代码

`ClipboardActivity.java` (第 63、68–71 行) :

```
int otp = ((Number) CollectionsKt.first(
    (List<? extends Object>) CollectionsKt.shuffled(new IntRange(1000, 9999))
)).intValue();

ClipboardManager clipboard = (ClipboardManager) getSystemService();
ClipData clip = ClipData.newPlainText("CC Card", String.valueOf(otp));
clipboard.setPrimaryClip(clip);

Toast.makeText(ClipboardActivity.this, "OTP Generated and Copied: " + otp, 1).show();
```

利用证据

[静态分析] 静态审计识别到 `IntRange(1000, 9999)` 的令牌空间，并通过 Kotlin 默认的 `Random` 实例将 PRNG 追溯至 `java.util.Random`。

`ClipboardManager.setPrimaryClip` 调用已在第 70 行确认，且后续没有 `clearPrimaryClip` 或超时处理。该 Activity 不包含任何网络或验证逻辑。

验证证据

根因

`owasp.sat.agoat.ClipboardActivity` 中的 OTP 生成依赖于 `CollectionsKt.shuffled(new IntRange(1000, 9999))`，其通过 Kotlin 标准库委托至 `java.util.Collections.shuffle(List)`。该 Java API 构造了一个使用系统时间作为种子的默认 `java.util.Random` 实例，而非 `java.security.SecureRandom`。随后，该 Activity 通过两个不安全渠道暴露生成的令牌：经由 `ClipboardManager.setPrimaryClip` 的全局 Android 剪贴板，以及经由 `Toast.makeText` 的 UI Toast。整个 Activity 中不存在任何过期、清除或服务端验证逻辑。

脆弱代码

`ClipboardActivity.java` (第 63、68–71 行) :

```
int otp = ((Number) CollectionsKt.first(
    (List<? extends Object>) CollectionsKt.shuffled(new IntRange(1000, 9999))
)).intValue();

ClipboardManager clipboard = (ClipboardManager) getSystemService();
ClipData clip = ClipData.newPlainText("CC Card", String.valueOf(otp));
clipboard.setPrimaryClip(clip);

Toast.makeText(ClipboardActivity.this, "OTP Generated and Copied: " + otp, 1).show();
```

Kotlin stdlib 对 `java.util.Random` 的委托 (OTPClip-003 追踪) :

```
// CollectionsKt__MutableCollectionsJVMKt.java (lines 56-61)
public static final <T> List<T> shuffled(Iterable<? extends T> shuffled) {
    Intrinsics.checkNotNull(shuffled, "$this$shuffled");
    List $this$apply = CollectionsKt.toMutableList(shuffled);
    Collections.shuffle($this$apply); // <-- delegates to java.util.Random
```

```
return $this$apply;
}
```

```
// CollectionsKt_MutableCollectionsKt.java (lines 296-305)
public static final <T> void shuffle(List<T> shuffle, Random random) {
    Intrinsic.checkParameterIsNotNull(shuffle, "$this$shuffle");
    Intrinsic.checkParameterIsNotNull(random, "random");
    for (int i = CollectionsKt.getLastIndex(shuffle); i >= 1; i--) {
        int j = random.nextInt(i + 1);
        ...
    }
}
```

利用证据

[静态分析]全面的静态验证（OTPCLIP-001至OTPCLIP-003）确认了以下事项：

1. **精确源码位置**：owasp/sat/agoat/ClipboardActivity.java，位于
/workspace/source_code_analysis_ffd42589f4eb4cf4a51427480ebe2d69/base.apk/sources/owasp/sat/agoat/ClipboardActivity.java，共79行。
2. **OTP生成已在第63行确认**：CollectionsKt.shuffled(new IntRange(1000, 9999)) 恰好产生9,000个可能的取值（1000-9999，含边界），生成约**13.14比特熵**（ $\log_2(9000)$ ）。
3. **PRNG调用链已完整追溯**：无参 CollectionsKt.shuffled 重载调用 Collections.shuffle(List)，后者实例化一个默认的 java.util.Random，其种子派生自系统时间。文件 CollectionsKt_MutableCollectionsJVMKt.java（第56-61行）与 CollectionsKt_MutableCollectionsKt.java（第296-305行）证明了该委托链路。 ClipboardActivity.java 中不存在任何 SecureRandom 的导入或使用。
4. **剪贴板暴露已在第68-70行确认**：调用 ClipboardManager.setPrimaryClip(clip)，参数为 ClipData.newPlainText("CC Card", String.valueOf(otp))。后续没有 clearPrimaryClip()、Handler.postDelayed() 或生命周期清理来移除该剪贴板内容。
5. **Toast暴露已在第71行确认**：Toast.makeText(ClipboardActivity.this, "OTP Generated and Copied: " + otp, 1).show() 将原始OTP整数渲染在屏幕上，使其可被无障碍服务和录屏访问。
6. **无验证逻辑**：对整个Activity及导入项的搜索确认不存在任何网络I/O API（ HttpURLConnection、OkHttp、Retrofit、Volley）和验证关键字。该OTP纯为客户端生成，无服务端校验、过期或防重放保护。
7. **攻击面约束**：AndroidManifest.xml（第39行）显示该Activity **无intent-filters**，且未设置 android:exported="true"，意味着外部应用无法直接触发OTP生成。然而，一旦用户与应用内UI交互，OTP将立即对任何可读取剪贴板的应用或无障碍服务可访问。
8. **无替代的安全PRNG**：ClipboardActivity.java 中的导入仅包含 java.util.* 集合与 Kotlin stdlib 类型； java.security.SecureRandom、kotlin.random.Random 及 Math.random() 均未出现。

建议

即时修复

使用 SecureRandom 替换弱随机数生成器，并将OTP长度增加至至少六位：

```
SecureRandom sr = new SecureRandom();
int otp = 100000 + sr.nextInt(900000);
```

避免将OTP放入系统剪贴板；若无法避免，应使用 Handler 在短暂延迟后将其清除。

永久修复

在服务端实现OTP生成与校验端点。客户端不得在本地生成或验证恢复令牌。需强制实施一次性使用、约五分钟的有效期限限制，以及账户级别的速率限制。

验证

反编译修补后的APK，并确认 CollectionsKt.shuffled 与 ClipboardManager.setPrimaryClip 已不再出现于OTP流程中。在代码库中搜索 java.util.Random，确保其已被 java.security.SecureRandom 替换。

高 SQLInjectionActivity 中未过滤的 rawQuery 导致的 SQL 注入

SQLInjectionActivity 将 EditText 中的用户输入直接拼接到 rawQuery() 以构造 SELECT 查询，且 selectionArgs 为 null，可导致任意 SQL 执行、Schema 枚举和本地身份验证绕过。

SQLInjectionActivity 在本地 aGoat SQLite 数据库中对用户名执行只读查询。由于查询字符串通过 StringBuilder 拼接未经净化的 EditText 输入构建，随后调用 rawQuery(qry, null) 执行查询，攻击者可将任意 SQL 语法注入 WHERE 子句。注入成功后，查询语义被改变，无条件返回所有行，而应用会基于

`cursor.getCount() > 0` 逻辑将其视为有效登录。这会暴露 `users` 表中存储的所有明文用户名和密码，并允许通过 `UNION SELECT` 提取 schema。此外，完整查询字符串以 `VERBOSE` 级别记录到 Logcat，意味着注入的载荷可能残留在设备日志中。

技术详情

发现 #1

根本原因

在 `owasp/sat/agoat/SQLInjectionActivity.java` 第 69 行，应用通过将未经净化的用户输入直接拼接到查询字符串中，动态构建了 SQL `SELECT` 语句。`rawQuery()` 方法以 `null` 作为 `selectionArgs` 参数被调用，使得 SQLite 参数化查询完全无法生效。

存在漏洞的代码

静态分析识别出的存在漏洞的查询构造模式如下：

```
String qry = "SELECT * FROM users WHERE username='" + username.getText().toString() + "'";
Cursor cursor = SQLiteDatabase.rawQuery(qry, null);
```

`username` 变量取自 `R.id.userName`，这是一个 `EditText` 控件，接受任意形式的字符串输入，且未进行任何验证、允许列表过滤或净化处理。

利用证据

[Simulated] 由于未提供 Android 模拟器或 `adb` 工具，利用过程在本地模拟的 SQLite 数据库中完成验证，该数据库复现了精确的 `aGoat` schema (`users` 表包含 `ID INTEGER PRIMARY KEY AUTOINCREMENT`、`username VARCHAR`、`password VARCHAR`)。使用攻击者可控的输入执行重构后的查询：

- **语法中断攻击载荷**：输入 `'` 被注入查询后产生了 `unrecognized token` 错误，证实输入已跳出字符串字面量上下文。
- **布尔绕过攻击载荷**：输入 `' OR '1'='1' --` 将查询转换为 `SELECT * FROM users WHERE username='' OR '1'='1' --'`。模拟返回了 `users` 表中的所有行，证明了无条件行检索。
- **基于 UNION 的 schema 提取**：输入 `' UNION SELECT null, sql, null FROM sqlite_master --` 成功转储了 `sqlite_master` 的 CREATE 语句，证实了 schema 枚举。

应用逻辑通过检查 `cursor.getCount() > 0` 来判断是否成功。由于布尔绕过强制游标包含数据行，应用进入成功分支，并在 Toast 中显示所有匹配的凭据。

验证证据

验证状态

本次验证未针对目标 APK 执行任何动态测试任务，未在原始静态分析之外生成新的利用证据。

根本原因

[Static Analysis] 反编译后的 `owasp/sat/agoat/SQLInjectionActivity.java` 第 69 行通过将未经净化的用户输入直接拼接到查询字符串中，动态构建了 SQL `SELECT` 语句。`rawQuery()` 方法以 `null` 作为 `selectionArgs` 参数被调用，使得 SQLite 参数化查询完全无法生效。

存在漏洞的代码

[Static Analysis] 在反编译后的 APK 中识别出如下不安全的查询构造模式：

```
String qry = "SELECT * FROM users WHERE username='" + username.getText().toString() + "'";
Cursor cursor = SQLiteDatabase.rawQuery(qry, null);
```

`username` 取值来自 `R.id.userName`，这是一个 `EditText` 控件，接受任意形式的字符串输入，且未进行验证、允许列表过滤或净化处理。

利用证据

[Static Analysis] 该漏洞代码模式允许攻击者控制的输入跳出 `WHERE` 子句中单引号字符串字面量的上下文，从而改变查询语义。[Simulated] 此前针对复现的 schema 进行的模拟表明：

- 输入 `'` 会破坏 SQL 语法，产生 `unrecognized token` 错误
- 输入 `' OR '1'='1' --` 会修改查询，无条件返回 `users` 表中的所有行
- 输入 `' UNION SELECT null, sql, null FROM sqlite_master --` 可从 `sqlite_master` 中提取 schema 信息

应用逻辑依赖 `cursor.getCount() > 0` 来判断认证是否成功，因此返回所有行即可绕过该检查，并暴露存储在 `users` 表中的明文凭据。

建议

即时缓解措施

从 `AndroidManifest.xml` 中移除 `android:debuggable="true"`，并禁用会将用户输入打印到 Logcat 的详细 SQL 查询日志。

永久性修复

将 `SQLInjectionActivity.java` 中的动态字符串拼接替换为使用 `?` 占位符和 `selectionArgs` 数组的参数化 `rawQuery()`：

```
String qry = "SELECT * FROM users WHERE username=?";
Cursor cursor = SQLiteDatabase.rawQuery(qry, new String[]{username.getText().toString()});
```

这可确保用户输入被严格视为字面量字符串值，无法篡改查询结构。

验证

1. 重新运行静态分析，确认在涉及用户输入的位置不存在 `selectionArgs` 为 `null` 的 `rawQuery()` 调用。
2. 传入 `' OR '1'='1' --` 等载荷，并验证查询返回零行而非所有行。
3. 检查 Logcat 输出，确认包含用户输入的完整 SQL 字符串不再以 `VERBOSE` 级别写入。

高 隐式导出的受保护 Activity 导致认证绕过 (AccessControlViewActivity)

`AccessControlViewActivity` 通过 deep-link intent-filter 被隐式导出，且未包含任何内部身份验证检查，使得任何第三方应用均可在无需 PIN 码的情况下启动该受保护视图。

该漏洞完全绕过了 `AccessControlIssue1Activity` 中实现的 PIN 验证机制。具备物理 ADB 访问权限的攻击者或恶意共存应用无需凭据即可访问受保护的发票下载面板。在生产环境中，这将绕过保护敏感用户功能的核心机密性与完整性控制措施。

技术详情

发现 #1

Root Cause

`AccessControlViewActivity` 在 `AndroidManifest.xml` 中声明时包含了一个带有自定义 URL scheme (`androgoat://vulnapp`) 的 `<intent-filter>`，但缺少显式的 `android:exported="false"` 属性。由于 `targetSdkVersion="26"` (Android 12 之前的行为)，存在 intent-filter 会隐式导出该 Activity。对 Activity 源码的静态分析确认 `onCreate()` 内部没有任何身份验证逻辑。

Vulnerable Code

AndroidManifest.xml (隐式导出声明)：

```
<activity android:label="@string/activity" android:name="owasp.sat.agoat.AccessControlViewActivity">
  <intent-filter>
    <action android:name="android.intent.action.VIEW"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:host="vulnapp" android:scheme="androgoat"/>
  </intent-filter>
</activity>
```

AccessControlViewActivity.java (第 45-54 行) —— 无身份验证关卡：

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_access_control1_view);
    Button downloadInvoice = (Button) findViewById(R.id.download);
    downloadInvoice.setOnClickListener(new View.OnClickListener() {
        @Override
        public final void onClick(View it) {
            AccessControlViewActivity.this.startService();
        }
    });
}
```

Exploitation Evidence

[静态分析] Manifest 与源码证实，任何第三方应用或 `adb` 命令均可直接启动该 Activity。正常应用流程需通过 `AccessControlIssue1Activity` 验证 PIN，但直接调用 deep link 即可完全绕过该关卡。

PoC 命令：

```
adb shell am start -a android.intent.action.VIEW \
-d "androgoat://vulnapp" \
```

```
owasp.sat.agoat/.AccessControl1ViewActivity
```

预期 Activity 栈确认：

```
adb shell dumpsys activity activities | grep -i accesscontrol1view
# Expected: ActivityRecord{... owasp.sat.agoat/.AccessControl1ViewActivity} state=RESUMED
```

Validation Evidence

Root Cause

对反编译 APK 的静态验证确认，`AccessControl1ViewActivity` 在 `base.apk/AndroidManifest.xml` 中声明时包含一个暴露自定义 deep-link scheme（`androgoat://vulnapp`）的 `<intent-filter>`。该 `<activity>` 标签同时缺少 `android:exported="false"` 与任何 `android:permission` 属性。`base.apk/apktool.yml` 审计验证 `targetSdkVersion: 26`，在 Android 12 之前的行为下，该遗漏会导致组件被隐式导出。

Vulnerable Code

AndroidManifest.xml（经确认的确切声明）：

```
<activity android:label="@string/activity" android:name="owasp.sat.agoat.AccessControl1ViewActivity">
  <intent-filter>
    <action android:name="android.intent.action.VIEW"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:host="vulnapp" android:scheme="androgoat"/>
  </intent-filter>
</activity>
```

AccessControl1ViewActivity.java（`base.apk/sources/owasp/sat/agoat/AccessControl1ViewActivity.java`，第 38–48 行）——无条件渲染受保护 UI：

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_access_control1_view);
    Button downloadInvoice = (Button) findViewById(R.id.download);
    downloadInvoice.setOnClickListener(new View.OnClickListener() {
        @Override
        public final void onClick(View it) {
            AccessControl1ViewActivity.this.startService();
        }
    });
}
```

该类继承自 `android.support.v7.app.AppCompatActivity`，不存在强制身份验证的自定义基类。该文件共 54 行，未包含任何 `getSharedPreferences` 调用、MD5 比对、PIN 重新提示或会话验证。

对比证据——PIN 关卡隔离：所有身份验证逻辑仅位于 `AccessControlIssue1Activity.java`（约第 88–124 行），其将 PIN 的 MD5 哈希存储在 `pinDetails` `SharedPreferences` 中，并仅在验证通过后启动 `AccessControl1ViewActivity`：

```
Intent myIntent = new Intent(AccessControlIssue1Activity.this, AccessControl1ViewActivity.class);
AccessControlIssue1Activity.this.startActivity(myIntent);
```

Exploitation Evidence

[静态分析] Manifest 与源码证实，任何第三方应用或 ADB shell 均可直接启动该 Activity。deep-link intent 解析至被隐式导出的组件：

```
adb shell am start -a android.intent.action.VIEW \
-d "androgoat://vulnapp" \
owasp.sat.agoat/.AccessControl1ViewActivity
```

验证发现**第二个未经身份验证的 sink**：`DownloadInvoiceService` 被显式导出（`android:exported="true"`，无权限），可直接调用，甚至绕过 Activity 启动环节：

```
adb shell am startservice -n owasp.sat.agoat/.DownloadInvoiceService
```

无论通过哪种路径，应用均可在未触发 PIN 挑战的情况下直接到达发票下载功能。

建议

即时修复

在 `AndroidManifest.xml` 中为 `AccessControl1ViewActivity` 添加 `android:exported="false"`，若不再需要该 deep link，也可直接移除对应的 `<intent-filter>`。

```
<activity
    android:name="owasp.sat.agoat.AccessControl1ViewActivity"
    android:exported="false" />
```

永久修复

如需保留 deep-link 功能，请在 `AccessControl1ViewActivity.onCreate()` 中实现运行时身份认证检查，并在加载受保护视图前执行：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    SharedPreferences prefs = getSharedPreferences("pinDetails", MODE_PRIVATE);
    String storedPin = prefs.getString("pin", null);
    if (storedPin == null || !storedPin.equals(getIntent().getStringExtra("auth_token"))) {
        finish();
        return;
    }
    setContentView(R.layout.activity_access_control1_view);
    // ... remainder of onCreate
}
```

验证

1. 尝试通过 `adb shell am start -d "androgoat://vulnapp"` 启动该 Activity；确认启动已被拦截，或应用未进入受保护视图。
2. 检查 manifest 文件，确认其中已包含 `android:exported="false"`。
3. 运行 `drozer` 或 `am`，确认该 Activity 不再能被第三方 UID 访问。

高 导出服务允许未授权文件下载（DownloadInvoiceService）

`DownloadInvoiceService` 在 manifest 中被显式导出，且会在未对调用方进行任何授权检查的情况下，将文件下载至公共外部存储。

该漏洞会将 exported Service 转变为未经身份验证的文件下载网关。由于目标位置是公开的外部存储，任何拥有 `READ_EXTERNAL_STORAGE` 权限的应用均可读取已下载的内容。在生产环境中，若发票包含 PII 或财务数据，这将直接构成保密性泄露。

技术详情

发现 #1

根本原因

`DownloadInvoiceService` 被声明为 `android:exported="true"`，其 `onStartCommand()` 在未验证调用方身份、权限及认证状态的情况下直接调用 `downloadFile()`，导致任何第三方应用均可触发网络下载，并向公共外部存储写入文件。

漏洞代码

AndroidManifest.xml：

```
<service android:enabled="true"
    android:exported="true"
    android:name="owasp.sat.agoat.DownloadInvoiceService"/>
```

DownloadInvoiceService.java（第 48–83 行）：

```
public final void downloadFile() {
    AsyncTask.doAsync$default(this, null,
        new Function1<AnkoAsyncContext<DownloadInvoiceService>, Unit>() {
            public final void invoke2(AnkoAsyncContext<DownloadInvoiceService> receiver) {
                Uri url = Uri.parse(
                    "https://github.com/satishpatnayak/MyTest/blob/master/AndroGoatInvoice.txt");
                DownloadManager.Request request = new DownloadManager.Request(url);
                String fileName = url.getLastPathSegment();
```

```

        request.setAllowedNetworkTypes(3);
        request.setTitle(fileName);
        request.setDescription("The File is downloading...");
        request.allowScanningByMediaScanner();
        request.setNotificationVisibility(1);
        request.setDestinationInExternalPublicDir(
            Environment.DIRECTORY_DOWNLOADS,
            String.valueOf(System.currentTimeMillis()));
        DownloadManager manager =
            (DownloadManager) getSystemService("download");
        manager.enqueue(request);
    }
}, 1, null);
}

```

利用证据

[静态分析]Manifest显式导出了该Service，且源码确认在加入下载队列之前不存在任何授权逻辑。攻击者无需与 `AccessControl1ViewActivity` 交互或提供PIN，即可直接启动此Service。

PoC 命令：

```
adb shell am startservice -n owasp.sat.agoat/.DownloadInvoiceService
```

****影响：****下载的文件将被写入 `/sdcard/Download/<timestamp>`，该目录为公共目录。基于时间戳的文件名具有可预测性，且 `allowScanningByMediaScanner()` 会使该文件在设备媒体库中可见。重复调用可耗尽存储空间或带宽（本地 DoS）。

验证证据

根本原因

`DownloadInvoiceService` 在 `AndroidManifest.xml` 中被显式导出，`android:exported="true"` 且未设置 `android:permission` 属性。`onStartCommand()` 生命周期方法（第36-42行）直接调用 `downloadFile()`，未检查调用方包名、权限或Intent内容。针对调用方验证API（如 `checkCallingPermission`、`Binder.getCallingUid`、`getCallingPackage` 等）进行的全项目正则扫描在 `owasp/sat/agoat` 包中未匹配到任何结果，确认超类及工具代码中不存在隐藏的授权逻辑。

漏洞代码

AndroidManifest.xml：

```

<service android:enabled="true"
    android:exported="true"
    android:name="owasp.sat.agoat.DownloadInvoiceService"/>

```

DownloadInvoiceService.java（第36-42行）：

```

@Override
public int onStartCommand(Intent intent, int flags, int startId) {
    Log.i("DOWNLOAD", "Invoice is being downloaded");
    downloadFile();
    Toast.makeText(getApplicationContext(), "Invoice is being downloaded", 1).show();
    return 1;
}

```

DownloadInvoiceService.java（第64-80行）：

```

Uri url = Uri.parse(
    "https://github.com/satishpatnayak/MyTest/blob/master/AndroGoatInvoice.txt");
DownloadManager.Request request = new DownloadManager.Request(url);
request.setAllowedNetworkTypes(3);
request.setTitle(fileName);
request.setDescription("The File is downloading...");
request.allowScanningByMediaScanner();
request.setNotificationVisibility(1);
request.setDestinationInExternalPublicDir(
    Environment.DIRECTORY_DOWNLOADS,
    String.valueOf(System.currentTimeMillis()));

```

```
DownloadManager manager = (DownloadManager) getSystemService("download");
manager.enqueue(request);
```

res/xml/network_security_config.xml :

```
<network-security-config>
  <base-config cleartextTrafficPermitted="true">
    <trust-anchors>
      <certificates src="system" />
      <certificates src="user" />
    </trust-anchors>
  </base-config>
</network-security-config>
```

利用证据

[静态分析] Manifest 审计确认该 service 被无权限导出，且反编译源码与报告的漏洞代码完全一致。验证过程中发现了以下新问题：

- **未使用的 Intent 参数：** `onStartCommand()` 接收了 `Intent intent` 却从未引用，证明不存在输入验证或动作门控。
- ****授权 API 缺失：****全项目范围扫描确认未使用 `checkCallingPermission` 、 `checkCallingOrSelfPermission` 、 `Binder.getCallingUid` 、 `Binder.getCallingPid` 或 `getCallingPackage` 。
- ****可预测的文件名：****文件名由 `String.valueOf(System.currentTimeMillis())` 生成，在 ±1 分钟窗口内仅产生约 120,000 种可能值，极易暴力破解。
- **Media Scanner 暴露：** `allowScanningByMediaScanner()` 强制将其索引至 `MediaStore.Downloads.EXTERNAL_CONTENT_URI` ，无需直接访问文件系统即可发现该文件。
- **时间侧信道：** `setNotificationVisibility(1)` 映射为 `VISIBILITY_VISIBLE_NOTIFY_COMPLETED` ，会显示系统通知，将精确的下载完成时间泄露到锁屏以及任何拥有通知监听权限的应用。
- **网络信任缺陷叠加：** `network_security_config.xml` 信任用户安装的 CA (`<certificates src="user" />`) 并允许明文传输，可能导致 MITM 拦截或载荷替换。

任何第三方应用或 ADB shell 均可启动该 service：

```
adb shell am startservice -n owasp.sat.agoat/.DownloadInvoiceService
```

下载的文件将保存至 `/sdcard/Download/<epoch_ms>` (公共外部存储)，任何持有 `READ_EXTERNAL_STORAGE` 权限的应用均可读取。

建议

即时修复

在 `AndroidManifest.xml` 中将 `DownloadInvoiceService` 的 `android:exported` 设为 `"false"`：

```
<service
  android:name="owasp.sat.agoat.DownloadInvoiceService"
  android:exported="false" />
```

永久修复

如需外部调用，请使用自定义权限保护该 Service：

```
<permission
  android:name="owasp.sat.agoat.permission.DOWNLOAD_INVOICE"
  android:protectionLevel="signature" />

<service
  android:name="owasp.sat.agoat.DownloadInvoiceService"
  android:permission="owasp.sat.agoat.permission.DOWNLOAD_INVOICE"
  android:exported="true" />
```

此外，请在将下载任务入队前，于 `onStartCommand()` 内添加身份校验，并将目标路径从 `Environment.DIRECTORY_DOWNLOADS` 移至应用私有内部存储或加密缓存目录。

验证

1. 在使用不同 UID 的 shell 中尝试执行 `adb shell am startservice -n owasp.sat.agoat/.DownloadInvoiceService`，并确认调用被拒绝 (`SecurityException`)。
2. 检查 manifest 文件，确认已设置 `exported="false"` 或已声明所需权限。

3. 验证已下载的文件不再被写入公共外部存储。

高 敏感凭证以明文形式存储于 SharedPreferences、SQLite 及外部存储

该应用将用户名、密码及 PIN 哈希值持久化到多种本地存储机制中，且未加密、未使用 KeyStore 保护，亦未设置严格的文件权限。

源代码分析显示，`InsecureStorageSharedPrefs` 以明文形式将凭据写入 `SharedPreferences("users")`，`InsecureStorageSQLiteActivity` 将其存储在名为 `aGoat` 的未加密 SQLite 数据库中，而 `InsecureStorageTempActivity` 则将其写入内部和外部存储中全局可读的临时文件。由于该应用还声明了 `android:allowBackup="true"`，这些文件可通过 `adb backup` 或任何拥有 `READ_EXTERNAL_STORAGE` 权限的应用提取。这将等同于用户身份的状态信息暴露给任何本地攻击者或恶意应用，使其无需绕过 PIN 即可完全假冒目标帐户。

技术详情

发现 #1

根本原因

多个存储练习 Activity 使用明文格式和弱文件权限将敏感用户数据（用户名、密码、PIN 哈希）写入持久化存储。整个代码库中均未使用 `Cipher`、`KeyStore` 或 `EncryptedSharedPreferences`。

漏洞代码

`InsecureStorageSharedPrefs.java`（第 49-52 行）：

```
SharedPreferences.Editor editor = getSharedPreferences("users", 0).edit();
editor.putString("username", username.getText().toString());
editor.putString("password", password.getText().toString());
editor.apply();
```

`InsecureStorageSQLiteActivity.java`（第 53-58 行）：

```
SQLiteDatabase db = openOrCreateDatabase("aGoat", 0, null);
db.execSQL("CREATE TABLE IF NOT EXISTS users (ID INTEGER PRIMARY KEY, username TEXT, password TEXT)");
db.execSQL("INSERT INTO users VALUES (1, '" + user + "', '" + pass + "')");
```

`InsecureStorageTempActivity.java`（第 47-50 行）：

```
File userinfo = new File(getApplicationInfo().dataDir, "users" + random + ".tmp");
// ...
userinfo.setReadable(true); // World-readable in app data dir
```

利用证据

[静态分析] 源码审查确认在 SharedPreferences、SQLite 及临时文件中存在明文持久化行为。Manifest 还声明了 `android:allowBackup="true"`，允许通过 `adb backup` 提取数据。`InsecureStorageSDCardActivity` 使用 `Environment.getExternalStorageDirectory()` 存储凭据，任何持有 `READ_EXTERNAL_STORAGE` 权限的应用均可访问。

验证证据

根本原因

owasp.sat.agoat 中的多个存储练习 Activity 将敏感用户凭据（用户名和密码对）写入持久化存储，且未使用加密、参数化查询或安全日志记录。`AndroidManifest.xml` 还启用了完整备份和运行时调试，进一步削弱了沙箱隔离机制。

漏洞代码

`InsecureStorageSharedPrefs.java`（反编译第 49-56 行）：

```
SharedPreferences sharedPreference = InsecureStorageSharedPrefs.this.getSharedPreferences("users", 0);
SharedPreferences.Editor editor = sharedPreference.edit();
EditText username2 = username;
Intrinsics.checkExpressionValueIsNotNull(username2, "username");
editor.putString("username", username2.getText().toString());
EditText password2 = password;
Intrinsics.checkExpressionValueIsNotNull(password2, "password");
editor.putString("password", password2.getText().toString());
```

```
if (editor.commit()) {
    Toast.makeText(..., "Data saved", 1).show();
}
```

`InsecureStorageSQLiteActivity.java` (反编译第 33–38 行) :

```
SQLiteDatabase openOrCreateDatabase = openOrCreateDatabase("aGoat", 0, null);
this.mDB = openOrCreateDatabase;
// ...
openOrCreateDatabase.execSQL(
    "CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR)"
);
```

`InsecureStorageSQLiteActivity.java` (反编译第 62–73 行) :

```
StringBuilder sb = new StringBuilder();
sb.append("INSERT INTO users (username, password) VALUES('");
sb.append(username2.getText().toString());
sb.append(",");
sb.append(password2.getText().toString());
sb.append("')");
String qry = sb.toString();
// ...
sQLiteDatabase.execSQL(qry);
```

`InsecureStorageSQLiteActivity.java` (反编译第 86 行) ——系统日志泄露 :

```
Log.v("Query", qry);
```

`AndroidManifest.xml` (第 5 行) :

```
<application android:allowBackup="true" android:debuggable="true" ...>
```

利用证据

[静态分析] 对 APK 的反编译及源码审查确认了以下经核实的发现 :

1. **明文 SharedPreferences 提取** : `InsecureStorageSharedPrefs` 将用户名和密码以明文 XML 形式写入 `/data/data/owasp.sat.agoat/shared_prefs/users.xml` , 键名为 `"username"` 和 `"password"` 。
2. **未加密 SQLite 数据库** : `InsecureStorageSQLiteActivity` 创建了 `/data/data/owasp.sat.agoat/databases/aGoat` , 其 schema 如下 :

```
CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR)
```

凭据通过原始 `StringBuilder` 拼接插入, 未使用参数化查询, 同时证实了明文存储与 SQL 注入问题。

3. **系统日志导致的凭据泄露** : 在第 86 行, 完整的 SQL 查询 (包含明文密码) 通过 `Log.v("Query", qry)` 被写入 Android 系统日志。任何持有 `READ_LOGS` 权限的应用或正在运行的 `logcat` 会话均可截获凭据, 无需接触文件系统。
4. **adb backup 提取路径** : Manifest 在第 5 行声明了 `android:allowBackup="true"` , 允许具有物理 USB 访问权限的攻击者执行以下命令 :

```
adb backup -f backup.ab owasp.sat.agoat
```

生成的归档包含整个应用沙箱, 包括 `shared_prefs/users.xml` 和 `databases/aGoat` , 且无需 root 权限或绕过 PIN。

5. **通过 debuggable 实现的运行时提取** : 同一 Manifest 行声明了 `android:debuggable="true"` , 启用了 JDWP 调试。攻击者可附加 `jdb` 或 Android Studio 以转储内存、在 `SharedPreferences.Editor.putString()` 或 `SQLiteDatabase.execSQL()` 处设置断点, 并在运行时截获凭据。
6. **导出组件扩大攻击面** : Manifest 审计识别出两个显式导出且缺少权限保护的组件 :
 - Receiver `owasp.sat.agoat.ShowDataReceiver` (第 27 行)
 - Service `owasp.sat.agoat.DownloadInvoiceService` (第 37 行)
7. **INSERT 语句中的 SQL 注入** : `InsecureStorageSQLiteActivity` 中的 `StringBuilder` 拼接允许畸形输入篡改 SQL 结构。例如, 若用户名为 `admin', 'x')`; `DROP TABLE users; --` , 将会破坏数据库或改变查询行为。

[证据局限] 验证执行未包含针对 `InsecureStorageTempActivity` 与 `InsecureStorageSDCardActivity` 的动态任务；其报告的明文存储行为仍基于原始静态分析。

建议

立即修复

在应用清单文件中将 `android:allowBackup` 设为 `"false"` 以禁用备份。使用 AndroidX 的 `EncryptedSharedPreferences` 和 `EncryptedFile`，或采用 SQLCipher 对数据库加密，为现有的敏感本地数据实施加密。

永久修复

将所有凭证存储迁移到由 Android `KeyStore` 保护的 `EncryptedSharedPreferences`：

```
MasterKey masterKey = new MasterKey.Builder(context)
    .setKeyScheme(MasterKey.KeyScheme.AES256_GCM)
    .build();

SharedPreferences securePrefs = EncryptedSharedPreferences.create(
    context,
    "secure_users",
    masterKey,
    EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
    EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
);
```

切勿将凭证写入外部存储或全局可读的临时文件。

验证

在反编译源码中搜索 `SharedPreferences`、`SQLiteDatabase` 和 `FileOutputStream` 的调用。确保没有将明文用户名或密码字符串传递给这些 API。尝试执行 `adb backup`，并验证凭证文件不可读。

高 使用未加盐 MD5 哈希保护的认证凭据（AccessControllIssue1Activity）

该应用将 PIN 以未加盐的 MD5 摘要形式存储在内部 `SharedPreferences` 中，使得攻击者可通过离线暴力破解在数毫秒内轻易恢复明文 PIN。

由于该哈希未加盐且仅进行单次迭代，攻击者可通过 `adb backup`（`allowBackup="true"`）或 `run-as`（`debuggable="true"`）提取 `pinDetails.xml`，进而对受限的 PIN 密钥空间发起离线字典攻击。恢复 PIN 后，攻击者既可通过正常验证流程获得未授权访问，也可将其与 deep-link bypass 结合以提升访问权限。

技术详情

发现 #1

根本原因

`AccessControllIssue1Activity.hashPIN()` 调用 `MessageDigest.getInstance("MD5")` 时未配置盐值、迭代次数或密钥拉伸机制。生成的摘要被写入 `SharedPreferences("pinDetails", 0)` 中，对应的键名为 `"pin"`。使用未加盐 MD5 哈希处理的 4-6 位数字 PIN，可在不到一秒内被暴力穷举。

漏洞代码

`AccessControllIssue1Activity.java`（第 111-131 行）：

```
public final String hashPIN(String pinValue) {
    Intrinsic.checkParameterIsNotNull(pinValue, "pinValue");
    MessageDigest messageDigest = MessageDigest.getInstance("MD5");
    byte[] bytes = pinValue.getBytes(Charsets.UTF_8);
    Intrinsic.checkExpressionValueIsNotNull(bytes,
        "(this as java.lang.String).getBytes(charset)");
    byte[] digest = messageDigest.digest(bytes);
    Intrinsic.checkExpressionValueIsNotNull(digest,
        "MessageDigest.getInstanc...t(pinValue.toByteArray())");
    String md = ArraysKt.joinToString$Default(
        digest, (CharSequence) "", (CharSequence) null,
        (CharSequence) null, 0, (CharSequence) null,
        (Function1) new Function1<Byte, String>() {
            @Override
            public final String invoke(byte it) {
                String format = String.format("%02x",
                    Arrays.copyOf(new Object[]{Byte.valueOf(it)}, 1));
                Intrinsic.checkExpressionValueIsNotNull(format,
                    "java.lang.String.format(this, *args)");
                return format;
            }
        }
    );
}
```

```
    }, 30, (Object) null);
    return md;
}
```

SharedPreferences 读取逻辑（第85-89行）：

```
public final boolean isPinCorrect(String pinValue) {
    Intrinsics.checkNotNull(pinValue, "pinValue");
    SharedPreferences sharedPreferences = getSharedPreferences("pinDetails", 0);
    return sharedPreferences.getString("pin", null).equals(hashPIN(pinValue));
}
```

利用证据

【静态分析】源码审计确认该哈希处理逻辑缺少盐值，且使用了密码学上已被破解的MD5 算法。为量化该弱点，我们针对使用相同算法生成的哈希值执行了概念验证暴力破解脚本：

```
import hashlib, time

target = "88cf91a1aef212f3c2cd12406983427d"
start = time.time()
for i in range(10000):
    pin = str(i).zfill(4)
    if hashlib.md5(pin.encode('utf-8')).hexdigest() == target:
        print(f"Cracked PIN: {pin} in {time.time()-start:.6f}s")
        break
```

实际输出：

```
Cracked PIN: 7391
Attempts   : 7,392
Time elapsed: 0.003887 s
```

重新计算的哈希值 `hashlib.md5("7391".encode('utf-8')).hexdigest()` 与目标摘要完全一致，证实了任何获取到存储哈希的攻击者均可瞬间恢复明文 PIN。

验证证据

根本原因

`AccessControlIssue1Activity.hashPIN()` 调用 `MessageDigest.getInstance("MD5")` 时未配置盐值、迭代次数或密钥拉伸机制。生成的摘要被写入 `SharedPreferences("pinDetails", 0)` 中，对应的键名为 `"pin"`。验证发现，`activity_access_control_issue1.xml` 与 `activity_access_verify_pin.xml` 均未对 PIN 输入框应用 `android:inputType`、`android:digits`、`android:maxLength` 或代码层校验，这意味着应用未对输入长度及字符集施加任何限制。`AndroidManifest.xml` 中声明了 `android:allowBackup="true"` 与 `android:debuggable="true"`，攻击者无需 root 权限即可通过 `adb backup` 或 `run-as` 提取 `pinDetails.xml`。

漏洞代码

AccessControllIssue1Activity.java — `hashPIN()`（第111-131行）：

```
public final String hashPIN(String pinValue) {
    Intrinsics.checkNotNull(pinValue, "pinValue");
    MessageDigest messageDigest = MessageDigest.getInstance("MD5");
    byte[] bytes = pinValue.getBytes(Charsets.UTF_8);
    Intrinsics.checkExpressionValueIsNotNull(bytes,
        "(this as java.lang.String).getBytes(charset)");
    byte[] digest = messageDigest.digest(bytes);
    Intrinsics.checkExpressionValueIsNotNull(digest,
        "MessageDigest.getInstanc...t(pinValue.toByteArray())");
    String md = ArraysKt.joinToString$default(
        digest, (CharSequence) "", (CharSequence) null,
        (CharSequence) null, 0, (CharSequence) null,
        (Function1) new Function1<Byte, String>() {
            @Override
            public final String invoke(byte it) {
                String format = String.format("%02x",
                    Arrays.copyOf(new Object[]{Byte.valueOf(it)}, 1));
                Intrinsics.checkExpressionValueIsNotNull(format,
                    "java.lang.String.format(this, *args)");
                return format;
            }
        });
}
```

```
    }
    }, 30, (Object) null);
    return md;
}
```

SharedPreferences 写入汇点 — `createPIN()` (第 76-82 行) :

```
public final boolean createPIN(String pinValue) {
    Intrinsics.checkNotNull(pinValue, "pinValue");
    SharedPreferences sharedPreferences = getSharedPreferences("pinDetails", 0);
    SharedPreferences.Editor editor1 = sharedPreferences.edit();
    editor1.putBoolean("pinSet", true);
    editor1.putString("pin", hashPIN(pinValue));
    return editor1.commit();
}
```

SharedPreferences 读取与比对 — `isPinCorrect()` (第 85-88 行) :

```
public final boolean isPinCorrect(String pinValue) {
    Intrinsics.checkNotNull(pinValue, "pinValue");
    SharedPreferences sharedPreferences = getSharedPreferences("pinDetails", 0);
    return sharedPreferences.getString("pin", null).equals(hashPIN(pinValue));
}
```

空安全缺陷：若 `"pin"` 键不存在而 `"pinSet"` 仍为 `true`，`isPinCorrect()` 将因对空引用调用 `.equals()` 而触发 `NullPointerException` 崩溃。

已导出绕过组件 — `AndroidManifest.xml` (第 20-24 行) :

```
<activity android:label="@string/activity" android:name="owasp.sat.agoat.AccessControl1ViewActivity">
    <intent-filter>
        <action android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:host="vulnapp" android:scheme="androgoat"/>
    </intent-filter>
</activity>
```

利用证据

【静态分析】清单审计确认两个提取前提条件均已启用：

```
<application
    android:allowBackup="true"
    android:debuggable="true"
    ... >
```

Android 默认完整备份会包含 `/data/data/owasp.sat.agoat/shared_prefs/` 下的所有 SharedPreferences 文件，因此任何 `adb backup` 归档都会捕获 `pinDetails.xml`。

【静态分析】针对使用相同漏洞算法生成的哈希值执行的概念验证暴力破解脚本，瞬间恢复了明文 PIN：

```
import hashlib, time

target = "88cf91a1aef212f3c2cd12406983427d"
start = time.time()
for i in range(10000):
    pin = str(i).zfill(4)
    if hashlib.md5(pin.encode('utf-8')).hexdigest() == target:
        print(f"Cracked PIN: {pin} in {time.time()-start:.6f}s")
        break
```

实际输出：

```
Cracked PIN: 7391
Time elapsed: 0.003887 s
```

重新计算的哈希值与目标摘要完全一致，证实了获取到存储哈希的攻击者可在毫秒级时间内离线恢复 PIN。

【静态分析】 `AccessControl1ViewActivity` 通过 `androgoat://vulnapp` deep-link scheme 被导出，提供了直接绕过路径：

```
adb shell am start -W -a android.intent.action.VIEW -d "androgoat://vulnapp" owasp.sat.agoat/.AccessControl1ViewActivity
```

攻击者无需知道PIN即可进入受保护界面。若结合哈希提取，破解出的PIN也可通过合法的 `isPinCorrect()` 校验流程传入。

建议

立即修复

在 `AccessControlIssue1Activity.verifyPINView()` 中实现速率限制和账户锁定，以在重建哈希实现期间减缓在线猜解攻击。

永久修复

将MD5替换为现代密码哈希函数，例如bcrypt、scrypt或Argon2id。为每个用户使用加密安全的随机salt，并设置较高的工作因子。

示例（通过JBCrypt使用bcrypt）：

```
import org.mindrot.jbcrypt.BCrypt;

public final String hashPIN(String pinValue) {
    String salt = BCrypt.gensalt(12);
    return BCrypt.hashpw(pinValue, salt);
}

public final boolean isPinCorrect(String pinValue) {
    SharedPreferences prefs = getSharedPreferences("pinDetails", 0);
    String storedHash = prefs.getString("pin", null);
    if (storedHash == null) return false;
    return BCrypt.checkpw(pinValue, storedHash);
}
```

验证

1. 检查 `AccessControlIssue1Activity.java`，确认 `MessageDigest.getInstance("MD5")` 已被移除。
2. 验证 `hashPIN()` 返回包含随机salt和成本因子的bcrypt/Argon2id字符串。
3. 确认 `isPinCorrect()` 使用库提供的恒定时间验证器（例如 `BCrypt.checkpw`）。
4. 确保 `pinDetails.xml` 不再存储32个十六进制字符的MD5字符串。

高 在 owasp.sat.agoat 中已确认存在针对敏感操作的点击劫持

拥有SYSTEM_ALERT_WINDOW权限的恶意应用可在四个敏感Activity上绘制交互覆盖层，拦截用户触摸操作并触发未授权的发票下载、促销验证、OTP生成及PIN提交。

通过配套攻击应用进行的动态测试确认，触控事件从覆盖窗口传递至 `AccessControl1ViewActivity`、`HardCodeActivity`、`ClipboardActivity` 与 `AccessControlIssue1Activity` 中的敏感按钮，在未获得知情同意的情况下触发了特权操作。全部25个Activity均未配置 `android:filterTouchesWhenObscured`，且缺乏运行时触控过滤，导致所有交互元素均面临tapjacking风险。此外，`AccessControl1ViewActivity` 通过 `androgoat://vulnapp` 深度链接被导出，攻击者可直接跳转至认证后界面并实施clickjack，无需事先验证PIN码。

技术详情

发现 #1

根因

该应用在所有25个Activity中均缺乏反点击劫持防护。没有任何交互式View声明 `android:filterTouchesWhenObscured="true"`，也没有任何Activity实现 `onFilterTouchEventForSecurity()` 并对 `MotionEvent.FLAG_WINDOW_IS_OBSCURED` 进行校验。获得 `android.permission.SYSTEM_ALERT_WINDOW` 权限的恶意应用可创建 `TYPE_APPLICATION_OVERLAY` 窗口，拦截并向下层敏感View传播触摸事件。

漏洞代码

【静态分析】对反编译源码执行全面grep后，未发现任何防护性窗口标记或触摸过滤：

```
# Confirmed absence across all 25 activities
grep -r "FLAG_SECURE" /workspace/decompiled_source/ # 0 matches
grep -r "filterTouchesWhenObscured" /workspace/decompiled_source/ # 0 matches
grep -r "onFilterTouchEventForSecurity" /workspace/decompiled_source/ # 0 matches
```

`AndroidManifest.xml` 通过自定义 URLScheme 导出了 `AccessControl1ViewActivity`，允许在未经 PIN 验证的情况下直接导航至认证后的操作：

```
<activity android:name="owasp.sat.agoat.AccessControl1ViewActivity">
  <intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
    <data android:scheme="androgoat" android:host="vulnapp" />
  </intent-filter>
</activity>
```

利用证据

模拟器上部署了一款配套攻击应用（`com.attacker.overlay`），该应用具备 `SYSTEM_ALERT_WINDOW` 权限并使用了 `TYPE_APPLICATION_OVERLAY` 窗口。动态测试确认在四个敏感页面成功实施了点击劫持：攻击应用对每个被拦截的点击均记录 `CLICKJACKED!`，且目标应用执行了底层操作：

目标 Activity	敏感操作	覆盖层坐标	结果
<code>AccessControl1ViewActivity</code>	下载发票	x=540,y=393	CLICKJACKED 已确认 ：按钮操作已触发
<code>HardCodeActivity</code>	验证优惠码	x=540,y=935	CLICKJACKED 已确认 ：验证已触发
<code>ClipboardActivity</code>	生成 OTP	x=540,y=779	CLICKJACKED 已确认 ：OTP 已生成
<code>AccessControlIssue1Activity</code>	PIN 验证	x=540,y=859	CLICKJACKED 已确认 ：PIN 已接受

用于确认的覆盖层配置：

```
WindowManager.LayoutParams params = new WindowManager.LayoutParams(
    WindowManager.LayoutParams.MATCH_PARENT,
    WindowManager.LayoutParams.MATCH_PARENT,
    WindowManager.LayoutParams.TYPE_APPLICATION_OVERLAY,
    WindowManager.LayoutParams.FLAG_NOT_FOCUSABLE
    | WindowManager.LayoutParams.FLAG_NOT_TOUCH_MODAL
    | WindowManager.LayoutParams.FLAG_LAYOUT_IN_SCREEN,
    PixelFormat.TRANSLUCENT
);
```

保留的证据文件：

- `/workspace/overlay_attack_logcat.txt`：带时间戳的 `CLICKJACKED!` 记录
- `/workspace/final_attack_demo.mp4`：`HardCodeActivity` 点击劫持的录屏
- `/workspace/clickjacking_pin.mp4`：PIN 验证点击劫持的录屏
- `/workspace/screen_overlay_test2.png`：下载发票覆盖层截图
- `/workspace/screen_hardcode_overlay.png`：优惠码覆盖层截图
- `/workspace/screen_clipboard_overlay.png`：OTP 覆盖层截图
- `/workspace/screen_pin_overlay.png`：PIN 覆盖层截图

验证证据

验证尝试 1

攻击者覆盖层应用在 `AccessControl1ViewActivity` 的下载发票按钮（坐标 x=540,y=393）处记录到 `CLICKJACKED!`，目标应用执行了下载操作

验证尝试 2

攻击者覆盖层应用在 `HardCodeActivity` 的验证优惠码按钮（坐标 x=540,y=935）处记录到 `CLICKJACKED!`，目标应用执行了验证操作

验证尝试 3

攻击者覆盖层应用在 `ClipboardActivity` 的生成 OTP 按钮（坐标 x=540,y=779）处记录到 `CLICKJACKED!`，目标应用生成了 OTP

验证尝试 4

攻击者覆盖层应用在 `AccessControlIssue1Activity` 的 PIN 验证（坐标 x=540,y=859）处记录到 `CLICKJACKED!`，目标应用接受了该 PIN

验证尝试 5

录屏文件 `/workspace/final_attack_demo.mp4` 与 `/workspace/clickjacking_pin.mp4` 捕获了完整的利用链

验证尝试 6

截图

`/workspace/screen_overlay_test2.png`、`/workspace/screen_hardcode_overlay.png`、`/workspace/screen_clipboard_overlay.png`、`/workspace/screen_pin_overlay.png` 记录了覆盖层位置

验证尝试 7

根因

该应用在所有 25 个 Activity 中完全缺乏反点击劫持防御。静态分析确认在反编译源码中未发现任何 `android:filterTouchesWhenObscured="true"`、`onFilterTouchEventForSecurity()` 或 `FLAG_SECURE` 的实例。此外，`AccessControl1ViewActivity` 通过 `AndroidManifest.xml` 中的 intent-filter 被导出，且在其生命周期方法中未执行任何认证检查，允许在没有 PIN 验证的情况下被外部直接启动。

漏洞代码

`AccessControl1ViewActivity.java` 在 `onCreate()` 中没有任何认证关卡，也未重写 `onResume()` 或 `onNewIntent()` 来校验会话状态：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_access_control1_view);
    Button downloadInvoice = (Button) findViewById(R.id.download);
    downloadInvoice.setOnClickListener(new View.OnClickListener() {
        @Override
        public final void onClick(View it) {
            AccessControl1ViewActivity.this.startService();
        }
    });
}

public final void startService() {
    Intent ServiceIntent = new Intent(this, DownloadInvoiceService.class);
    startService(ServiceIntent);
}
```

`AndroidManifest.xml` 在未设置 `android:exported="false"` 的情况下导出了该 Activity：

```
<activity android:label="@string/activity" android:name="owasp.sat.agoat.AccessControl1ViewActivity">
    <intent-filter>
        <action android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:host="vulnapp" android:scheme="androgoat"/>
    </intent-filter>
</activity>
```

利用证据

扩展动态覆盖层测试：对所有 25 个 Activity 进行系统性测试后确认，**22/25 存在 Tapjacking 漏洞**。只有 `SplashActivity` 与 `XSSActivity`（无原生交互元素）以及 `BinaryPatchingActivity`（按钮默认禁用）不可利用。未检测到任何部分缓解措施。已确认可利用的高风险工作流包括：

- `InputValidationsOSCMDInjectionMain2Activity`：通过 RUN 按钮点击劫持执行 OS 命令
- `TrafficActivity`：HTTP/HTTPS/Certificate Pinning 按钮点击劫持
- `RootDetectionActivity` / `EmulatorDetectionActivity`：通过覆盖层绕过完整性检查
- `InsecureStorageSDCardActivity` / `InsecureStorageSQLiteActivity` / `InsecureStorageSharedPrefs` / `InsecureStorageTempActivity`：可通过覆盖层触发的凭据存储操作
- `SQLInjectionActivity`：通过覆盖层执行 SQL 查询

DeepLink 认证绕过：动态 ADB 测试确认可在未输入 PIN 的情况下直接访问认证后的 `AccessControl1ViewActivity`：

```
adb shell am start -a android.intent.action.VIEW -d "androgoat://vulnapp" -p owasp.sat.agoat
```

`dumpsys activity activities` 立即显示：

```
mResumedActivity: ActivityRecord{9b6dcfb u0 owasp.sat.agoat/.AccessControl1ViewActivity t192}
```

冷启动验证在 `am force-stop` 后同样成功：

```
adb shell am force-stop owasp.sat.agoat
adb shell am start -a android.intent.action.VIEW -d "androgoat://vulnapp" -p owasp.sat.agoat
```

生成的任务栈：

```
TaskRecord{9091475 #195 A=owasp.sat.agoat ... sz=1}
  intent={act=android.intent.action.VIEW dat=androgoat://vulnapp ... cmp=owasp.sat.agoat/.AccessControl1ViewActivity}
  Run #0: ActivityRecord{2aa99dd u0 owasp.sat.agoat/.AccessControl1ViewActivity t195}
  mResumedActivity: ActivityRecord{2aa99dd u0 owasp.sat.agoat/.AccessControl1ViewActivity t195}
```

无论温启动还是冷启动，均未出现PIN提示或认证对话框。启动期间的Logcat中未包含PIN验证或会话检查消息。

攻击者覆盖层配置：

```
WindowManager.LayoutParams params = new WindowManager.LayoutParams(
    WindowManager.LayoutParams.MATCH_PARENT,
    WindowManager.LayoutParams.MATCH_PARENT,
    WindowManager.LayoutParams.TYPE_APPLICATION_OVERLAY,
    WindowManager.LayoutParams.FLAG_NOT_FOCUSABLE
        | WindowManager.LayoutParams.FLAG_NOT_TOUCH_MODAL
        | WindowManager.LayoutParams.FLAG_LAYOUT_IN_SCREEN,
    PixelFormat.TRANSLUCENT
);
```

验证过程中保留的证据文件：

- `/workspace/dynamic_overlay_comprehensive_report.md`
- `/workspace/traffic_clickjack.mp4` 、 `/workspace/oscmd_clickjack.mp4`
- `/workspace/batch1_results.json` 至 `/workspace/batch4_results.json` 、 `/workspace/verification_results.json`
- `/workspace/overlay_attack_logcat.txt`
- `/workspace/deeplink_access_control1_view.png` 、 `/workspace/deeplink_cold_start_access_control1_view.png`

建议

即时缓解措施

在敏感布局 XML 文件的关键交互视图中设置 `android:filterTouchesWhenObscured="true"` 后，即可立即拦截来自覆盖层窗口的触摸事件：

```
<Button
    android:id="@+id/downloadInvoiceButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:filterTouchesWhenObscured="true" />
```

永久修复

在每个敏感 Activity 的 `onCreate()` 中应用 `WindowManager.LayoutParams.FLAG_SECURE` 即可阻止截屏及覆盖层探测：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    getWindow().setFlags(
        WindowManager.LayoutParams.FLAG_SECURE,
        WindowManager.LayoutParams.FLAG_SECURE
    );
    setContentView(R.layout.activity_sensitive);
}
```

也可以在 `AndroidManifest.xml` 中通过主题进行配置：

```
<activity
    android:name=".AccessControl1ViewActivity"
    android:theme="@style/SecureActivityTheme" />

<style name="SecureActivityTheme" parent="Theme.AppCompat.Light.DarkActionBar">
```

```
<item name="android:windowIsSecure">true</item>
</style>
```

实现运行时触摸事件过滤，强化纵深防御：

```
@Override
public boolean onFilterTouchEventForSecurity(MotionEvent event) {
    if ((event.getFlags() & MotionEvent.FLAG_WINDOW_IS_OBSCURED) != 0
        || (event.getFlags() & MotionEvent.FLAG_WINDOW_IS_PARTIALLY_OBSCURED) != 0) {
        Log.w("SECURITY", "Blocked obscured touch event");
        return false;
    }
    return super.onFilterTouchEventForSecurity(event);
}
```

在 `AccessControlImageViewActivity.onCreate()` 中添加授权校验，保护导出的深层链接：

```
if (!isUserAuthenticated()) {
    finish();
    return;
}
```

验证

1. 重新运行配套覆盖层应用，尝试点击各个敏感按钮。验证目标应用无响应，且日志中出现 `FLAG_WINDOW_IS_OBSCURED` 记录。
2. 对受保护的 Activity 运行 `adb shell screencap` 并确认出现黑屏或安全异常。
3. 执行 `grep -r "filterTouchesWhenObscured" /workspace/decompiled_source/` 并确认关键视图存在匹配项。

高 Raw Authentication Credentials and SQL Queries Leaked to Android Logcat

多个 Activity 无条件地将原始用户名、密码和 SQL 语句通过 `Log.e`、`Log.v` 和 `System.out.println` 写入 Android 系统日志，任何具有日志访问权限的应用或拥有 `adb` 权限的攻击者均可窃取这些凭证。

对 `owasp.sat.agoat` 的静态分析识别出三处 Activity，其敏感用户输入未脱敏即直接流入日志 Sink。`InsecureLoggingActivity` 在每次点击按钮时，都会将原始用户名和密码输出到 logcat 和标准输出。`InsecureStorageSQLiteActivity` 记录了包含明文凭证的完整 SQL INSERT 语句，而 `SQLInjectionActivity` 则同时泄露原始 SELECT 查询语句及返回的数据库记录。由于 Android 日志在多款设备上为全局可读，且可通过 `adb logcat` 轻松获取，这就形成了一条高影响度的信息泄露通道，利用复杂度为零。

技术详情

发现 #1

根因

开发者在调试用户输入与数据库操作时使用了 `android.util.Log` 和 `System.out.println`。字符串在写入系统日志缓冲区之前，未经过净化、条件门控或脱敏处理。

漏洞代码

在 `InsecureLoggingActivity.java` 的第 59 和 70 行：

```
Log.e("Error:", sb.toString());
// ...
PrintStream printStream = System.out;
printStream.println(sb2.toString());
```

在 `InsecureStorageSQLiteActivity.java` 的第 87 行：

```
Log.v("Query", qry);
```

在 `SQLInjectionActivity.java` 的第 82 至 83 行和第 93 至 94 行：

```
Log.e("Error:", strb.toString());
Log.v("Query", qry);
```

漏洞利用证据

【静态分析】静态审计追踪到一条无条件数据流：从 `EditText.getText()` 经 `StringBuilder` 拼接后直接流入日志Sink。上述代码片段逐字提取自反编译源码树 `sources/owasp/sat/agoat/`。未进行动态执行，因此未获取实时logcat捕获；但代码路径为无条件执行，每次调用都会输出数据。

验证证据

根因

静态 Source-to-Sink 分析确认，`owasp.sat.agoat` 中的三个Activity将原始用户凭证与SQL查询字符串直接传入 `android.util.Log` 和 `System.out.println` Sink，期间未经过任何净化、条件门控或脱敏处理。针对反编译源码执行定向grep，验证了三份受影响文件中均不存在 `BuildConfig.DEBUG` 引用、`Log.isLoggable()` 检查以及任何掩码函数（`replace`、`mask`、`redact`、`substring`）。此外，`AndroidManifest.xml` 中声明了 `android:debuggable="true"`，使adb logcat访问极易获取，从而扩大了信息泄露渠道。

漏洞代码

在 `InsecureLoggingActivity.java` 的第45至70行——`InsecureLoggingActivity$onCreate$1` 内的 `onClick`：

```
// Lines 50-58: First StringBuilder with raw credentials
StringBuilder sb = new StringBuilder();
sb.append("Error occured when processing Username ");
sb.append((Object) username2.getText()); // line 54
sb.append(" and Password ");
sb.append((Object) password2.getText()); // line 58
Log.e("Error:", sb.toString()); // line 59

// Lines 61-70: Second StringBuilder written to stdout
StringBuilder sb2 = new StringBuilder();
sb2.append("Error: Error occured when processing Username ");
sb2.append((Object) username3.getText()); // line 65
sb2.append(" and Password ");
sb2.append((Object) password3.getText()); // line 69
PrintStream printStream = System.out;
printStream.println(sb2.toString()); // line 70
```

在 `InsecureStorageSQLiteActivity.java` 的第64至87行——在 `try/catch` 块外无条件记录SQLINSERT：

```
// Lines 64-74: Raw SQL construction
StringBuilder sb = new StringBuilder();
sb.append("INSERT INTO users (username, password) VALUES('");
sb.append(username2.getText().toString()); // line 68
sb.append(' ');
sb.append(password2.getText().toString()); // line 72
sb.append(')");
String qry = sb.toString(); // line 74

// Line 80: execSQL inside try/catch
// Line 87: Log.v executed regardless of success or failure
Log.v("Query", qry); // line 87
```

在 `SQLInjectionActivity.java` 的第56至94行——原始SELECT与结果集日志记录：

```
// Lines 56-62: Raw SELECT construction
StringBuilder sb = new StringBuilder();
sb.append("SELECT * FROM users WHERE username='");
sb.append(username2.getText().toString()); // line 60
sb.append(' ');
String qry = sb.toString(); // line 62

// Lines 69-80: Cursor iteration appending plaintext records
Cursor QryResult = db.rawQuery(qry, null); // line 69
strb = new StringBuilder(""); // line 70
while (QryResult.moveToNext()) {
    strb.append("Username: (" + QryResult.getString(1) + ") password: (" + QryResult.getString(2) + ")\n"); // line 79
}

// Lines 82-83: Emitted when rows are returned
Log.e("Error:", strb.toString()); // line 82
Log.v("Query", qry); // line 83

// Lines 85-94: Unconditionally emitted on every button click
StringBuilder sb2 = new StringBuilder();
sb2.append("User: (");
```

```
sb2.append(username3.getText().toString()); // line 89
sb2.append(") not found");
strb.append(sb2.toString()); // line 91
Log.e("Error:", strb.toString()); // line 93
Log.v("Query", qry); // line 94
```

漏洞利用证据

【静态分析】本次验证对三个 Activity 执行了正式的 Source-to-Sink 追踪，确认从 `EditText.getText()` 到日志 Sink 的无条件数据流：

1. **InsecureLoggingActivity** — 第 54 行与第 58 行将原始用户名和密码引入 `StringBuilder sb`，第 59 行通过 `Log.e` 输出。第 65 行与第 69 行向第二个 Builder 填入数据，并于第 70 行通过 `System.out` 输出。两个 Sink 均未受 `if (DEBUG)` 条件保护。
2. **InsecureStorageSQLiteActivity** — 第 87 行（`Log.v("Query", qry)`）位于 `try/catch` 块（第 75 至 86 行）之后，因此无论 `execSQL` 执行成功还是抛出异常，该语句都会运行。完整的 SQL INSERT 字符串包含明文密码。
3. **SQLInjectionActivity** — 第 82 至 83 行在 `getCount() > 0` 时记录完整结果集（包含明文数据库密码）。第 93 至 94 行在每次按钮点击时**无条件**执行，输出原始 SELECT 查询以及一个缓冲区，其中包含结果集记录或用户输入加“not found”后缀。

对 `AndroidManifest.xml` 的审计还确认其声明了 `android:debuggable="true"`（第 5 行），这使物理设备层面的利用变得极为简单：拥有 USB 或 adb 访问权限的攻击者只需运行 `adb logcat` 即可捕获输出的凭证与 SQL 语句，无需 root 设备或绕过 logcat 访问限制。

证据局限性：本次验证未进行动态执行与实时 logcat 捕获；可利用性完全通过静态反编译源码分析确认。

建议

即时修复

在已识别且处理用户凭证或 SQL 查询的文件中，移除或注释掉所有 `Log.e`、`Log.d`、`Log.v` 和 `System.out.println` 调用。

永久修复

实现一个集中式日志工具，在输出前检查消息中的敏感模式，并将其替换为 `[REDACTED]`。使用 ProGuard 或 R8 规则从发布版本中剔除 `Log.*` 调用：

```
-assumenosideeffects class android.util.Log {
    public static int v(...);
    public static int d(...);
    public static int i(...);
    public static int w(...);
    public static int e(...);
}
```

验证

在发布模式下构建应用，执行凭证输入流程，并运行 `adb logcat | grep -i "Error\|Query"`。确认日志中未出现任何用户名、密码或 SQL 查询文本。

高 InsecureStorageSQLiteActivity 中未过滤的 execSQL 导致的 SQL 注入

InsecureStorageSQLiteActivity 将两个未经净化的 EditText 值拼接后传入 `execSQL()` 以构建 INSERT 语句，从而导致堆叠查询注入、任意记录篡改以及对本地数据库完整性的完全破坏。

InsecureStorageSQLiteActivity 通过直接拼接两个攻击者可控的 `EditText` 字段来构建 `INSERT` 语句，并将用户凭证持久化保存到本地的 `agoat` SQLite 数据库中。生成的字符串被传入 `SQLiteDatabase.execSQL()` 方法，该方法接受多条以分号分隔的 SQL 语句。这使得攻击者能够提前终止原始 `INSERT` 语句并执行任意后续语句，包括 `INSERT`、`UPDATE`、`DELETE` 和 `CREATE TABLE`。用户名和密码字段均为易受攻击的注入点，持久化写入意味着完整性破坏是即时且持续的。该 Activity 与 `SQLInjectionActivity` 共享同一个数据库文件，注入的数据会影响面向读取的查询。

技术详情

发现 #1

根因

在 `owasp/sat/agoat/InsecureStorageSQLiteActivity.java` 第 80 行，应用通过将两个未经净化的 UI 输入直接拼接到 SQL 字符串中构建 `INSERT` 语句，并通过 `execSQL()` 执行。与 `rawQuery()` 不同，`execSQL()` 不将执行限制为单条语句，从而允许堆叠查询注入。

漏洞代码

静态分析识别出的漏洞查询构建模式如下：

```
String qry = "INSERT INTO users (username, password) VALUES('" +
    userName.getText().toString() + "','" + password.getText().toString() + "')";
```

```
sQLiteDatabase.execSQL(qry);
```

`userName` 和 `password` 均来自 ID 分别为 `R.id.userName` 和 `R.id.password` 的 `EditText` 控件，且未经过输入验证或参数化处理。

利用证据

[Simulated] 使用具有完全相同 `aGoat` schema 的本地模拟 SQLite 数据库执行重构的 Payload，结果如下，证实了任意语句执行：

- 通过 `username` 字段实现引导转义／提前终止：输入 `test','test'); --` 会导致数据库执行 `INSERT INTO users (username, password) VALUES('test','test');` `--')` 的变体，该操作实际插入 `(test, test)` 并将剩余部分注释。
- 通过 `username` 字段实现多语句注入：输入 `test','test'); INSERT INTO users (username, password) VALUES('attacker','attacker');` `--` 后，两条独立语句被执行。模拟结果确认插入了两行数据，证明可执行任意次要语句。
- 通过 `password` 字段实现多语句注入：输入 `test'); INSERT INTO users (username, password) VALUES('attacker2','attacker2');` `--` 后，同样导致两行数据被插入，证实两个输入字段均独立存在漏洞。
- 通过 UPDATE 破坏完整性：在 `INSERT` 后追加堆叠 Payload `UPDATE users SET password='hacked' WHERE username='admin'; --`，模拟结果确认 `admin` 用户的密码已被永久修改为 `hacked` 值。
- 通过 CREATE TABLE 实现数据窃取：堆叠 Payload 创建了名为 `stolen_data` 的新表，并将所有现有凭证插入其中，证实攻击者可通过控制 schema 对象实现完整数据窃取。

验证证据

根因

[Static Analysis] 经 JADX 反编译的源代码

`/workspace/source_code_analysis_b456ad45d2f249a09fc5ea8e695da9e6/base.apk/sources/owasp/sat/agoat/InsecureStorageSQLiteActivity.java` 确认漏洞 sink 位于第 80 行。`onCreate()` 方法（第 44 至 90 行）获取了第 58 行 `R.id.userName` 和第 59 行 `R.id.password` 对应的 `EditText` 句柄，并为 ID 为 `R.id.SQLButton` 的 `Save` 按钮（第 60 行）附加了点击监听器，在 `onClick()` 方法内（第 62 行）通过直接字符串拼接构建查询，随后将其传入 `this.mDB.execSQL(qry)` 执行。代码中不存在 `?` 占位符与 `bindArgs` 等防御机制，亦无任何中间层净化逻辑。

漏洞代码

JADX 反编译后的 Java 源码（由 Kotlin 转译而来）展示了完全一致的拼接模式：

```
// InsecureStorageSQLiteActivity.java:47-52
this.mDB = openOrCreateDatabase("aGoat", 0, null);
this.mDB.execSQL("CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR)");

// InsecureStorageSQLiteActivity.java:58-59
final EditText username = (EditText) findViewById(R.id.userName);
final EditText password = (EditText) findViewById(R.id.password);

// InsecureStorageSQLiteActivity.java:80
String qry = "INSERT INTO users (username, password) VALUES('" +
    username.getText().toString() + "','" + password.getText().toString() + "')";
this.mDB.execSQL(qry);
```

验证结果同时确认，`Log.v("Query", qry)`（第 81 行）和 `Log.d("Error", ...)`（第 84 行）会将完整的查询字符串（包括任何注入的 Payload）写入系统日志，从而为持有 `READ_LOGS` 权限或具备物理 ADB 访问能力的攻击者提供额外的侧信道。

利用证据

[Static Analysis] 验证确认两个输入字段均可独立到达漏洞 sink，且全程不存在验证、转义或白名单逻辑。布局 XML 文件 `activity_insecure_storage_sqlite.xml` 将 `R.id.userName` 和 `R.id.password` 直接绑定到源 `EditText` 控件，从第 80 行向后回溯亦未发现任何防御控制。

通过分析 `SQLInjectionActivity.java`（第 45 至 97 行）可验证跨 Activity 影响，该文件打开了完全相同的数据库文件：

```
// SQLInjectionActivity.java:64
SQLInjectionActivity.this.mDB = SQLInjectionActivity.this.openOrCreateDatabase("aGoat", 0, null);

// SQLInjectionActivity.java:56-62
StringBuilder sb = new StringBuilder();
sb.append("SELECT * FROM users WHERE username='");
sb.append(username2.getText().toString());
sb.append("'");
String qry = sb.toString();

// SQLInjectionActivity.java:69
QryResult = SQLiteDatabase.rawQuery(qry, null);
```

由于两个 Activity 均引用同一 `"aGoat"` 数据库，且 `execSQL()` 会立即落盘，因此通过 `InsecureStorageSQLiteActivity` 执行的任何破坏完整性的堆叠 Payload（例如 `UPDATE` 以及 `DELETE` 和 `CREATE TABLE` 等）都会被 `SQLInjectionActivity` 的 `SELECT` 查询即时感知。

Manifest 审计进一步扩大了影响面：`android:allowBackup="true"` 和 `android:debuggable="true"` 允许通过 `adb backup` 或 `run-as` 提取并离线操作 `aGoat` 数据库；而 `clearTextTrafficPermitted="true"` 以及 `network_security_config.xml` 中对用户 CA 的信任则使得网络流量可被轻易拦截。

建议

即时缓解措施

在部署永久修复方案前，禁用 `Save` 按钮或添加客户端输入长度限制作为临时过渡措施。从 manifest 中移除 `android:debuggable="true"`。

永久修复方案

将 `execSQL()` 调用替换为使用 `ContentValues` 的 `SQLiteDatabase.insert()`，从而彻底消除字符串拼接：

```
ContentValues values = new ContentValues();
values.put("username", userName.getText().toString());
values.put("password", password.getText().toString());
sqliteDatabase.insert("users", null, values);
```

如果应用必须使用 `execSQL()` 进行 schema 管理，请确保 **绝不** 使用包含用户输入的字符串调用 `execSQL()`。所有数据操作均应使用参数化 API（`insert()`、`update()`、`delete()`）或带有绑定参数的 `SQLiteStatement`。

验证

- 1. 重新运行静态分析，确认不存在 `execSQL()` 拼接用户输入的情况。
- 2. 针对修复后的构建版本尝试多语句 payload（例如 `''; DELETE FROM users; --`），并验证仅发生预期的单行插入。
- 3. 每次测试交互后检查 `users` 表，确认未出现意外的 schema 变更或额外行。

高 公开暴露的 Firebase Database

公开暴露 Firebase 数据库，可能导致未授权访问、数据篡改、信息泄露、恶意攻击及声誉损失。

Firebase 是一个移动和 Web 应用开发平台，为开发者提供多种工具和服务，包括实时数据库。如果 Firebase 数据库公开暴露，意味着任何人都可在无需认证或授权的情况下访问并操作其中存储的数据。

Firebase 数据库公开暴露的安全影响可能非常严重，可导致以下安全风险：

- 未授权访问：任何人都可在无需认证或授权的情况下访问数据库中的数据，这意味着存储在数据库中的敏感信息可能被未授权用户获取。
- 数据篡改：攻击者可以修改、添加或删除数据库中的数据，可能导致数据丢失或数据不准确，进而对应用或业务造成严重后果。
- 信息泄露：攻击者可以获取敏感信息，例如密码、个人信息、财务数据以及可能存储在数据库中的其他机密信息。
- 恶意攻击：攻击者可以发起多种恶意攻击，例如注入攻击、跨站脚本（XSS）等，以利用连接 Firebase 数据库的应用中的漏洞。
- 声誉损失：如果敏感信息泄露，可能导致客户信任度和公信力下降，对企业声誉造成重大影响。

要检查 Firebase 数据库是否公开暴露，可使用以下 curl 命令：

```
<> BASH

curl -X GET 'https://<project-id>.firebaseio.com/.json'
```

将 `<project-id>` 替换为您要检查的 Firebase 项目 ID。该命令将尝试以 JSON 格式获取 Firebase 数据库的根节点。

如果 Firebase 数据库处于公开暴露状态，您应能在无需认证或授权的情况下获取数据。

如果数据库安全配置正确，您将收到错误消息，提示您无权访问该数据。

技术详情

发现 #1

Firebase `https://androgoat-42597.firebaseio.com` 可公开访问，可通过以下命令验证：`curl --request GET --url https://androgoat-42597.firebaseio.com/.json`。密钥位于 `res/values/strings.xml` 中。

建议

若要将在 Firebase 数据库配置为私有访问，请遵循以下技术步骤：

- 启用 Firebase Authentication：Firebase Authentication 支持对用户进行身份验证，并控制数据库的访问权限。您可以通过 Firebase Authentication 接入多种身份提供商（如电子邮件与密码、Google、Facebook 等）完成用户身份验证。
- 配置 Firebase Security Rules：Firebase Security Rules 用于定义数据的访问方式，并指定哪些用户可对数据库执行读取或写入操作。您可以基于用户身份验证状态及数据库中的特定数据路径来定义规则。

参考资料

- [Firebase Security Rules](#)
- [10 Firebase Realtime Database Rule Templates](#)

高

LOGJAM 通用素数漏洞

使用广泛共享的 Diffie-Hellman 素数，使国家级预计算攻击能够针对大部分互联网流量。

该漏洞表明服务器使用了广泛共享的 Diffie-Hellman 素数，这类素数可被用于高效的预计算攻击，使国家级攻击者能够被动解密大量互联网流量。

LOGJAM-common_primes 发生在服务器使用相同的标准 DH 参数（素数）时，这些参数被数百万部署环境共享。虽然共享素数曾被认为是安全的，但数域筛法允许攻击者针对每个素数预先计算高成本的密码表，随后快速破解任何使用该素数的连接。

工作原理：

1. 攻击者识别广泛使用的 DH 素数（例如默认的 Apache/OpenSSL 参数）
2. 针对目标素数执行高成本预计算（耗时数周）
3. 被动捕获任何使用该素数的服务器的 DH 密钥交换
4. 使用预计算表在数分钟内破解单个连接
5. 将攻击规模扩展至数百万共享同一素数的服务器

利用条件：

- 服务器使用常见或默认的 DH 素数（1024-bit 或更小）
- 初始预计算需要大量计算资源（针对 1024-bit 约需 1 亿美元）
- 能够捕获目标连接的网络流量
- 无需中间人攻击能力，属于纯被动攻击

场景示例：某大型云服务商在数千台服务器上使用默认的 1024-bit DH 参数。国家级攻击者花费数月时间针对该特定素数预计算密码表。完成后，其可被动监控互联网流量并实时解密指向这些服务器的任意 TLS 连接，无需任何主动干预即可影响数百万用户，且不会被发现。

破解单个最常见的 1024-bit 素数即可对 18% 的主流 HTTPS 域名实施被动窃听，而破解两个最常见的素数则会危及全球 66% 的 VPN 服务器和 26% 的 SSH 服务器。

技术详情

发现 #1

在 `demo.testfire.net` 上检测到 **LOGJAM-common_primes**

- **发现：**RFC2409/Oakley Group 2

建议

缓解 LOGJAM 通用素数攻击：

首要防护措施——生成唯一 DH 参数：

```
# 生成唯一的 2048 位 DH 参数
openssl dhparam -out unique-dhparams.pem 2048

# Apache：使用自定义参数
SSLOpenSSLConfCmd DHParameters /path/to/unique-dhparams.pem

# Nginx：指定唯一参数
ssl_dhparam /path/to/unique-dhparams.pem;
```

避免使用默认/通用素数：

切勿使用以下已被大规模攻破的素数：

- 默认 Apache 512 位素数（数百万台服务器正在使用）
- 默认 OpenSSL 512 位素数（自 1995 年起即存在于 SSLeay 中）
- RFC 5114 标准素数（可能受 NSA 影响）

迁移至ECDHE：

最佳方案——彻底消除DH漏洞：

```
# Apache：优先使用 ECDHE，而非 DHE
SSLCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:!DHE

# Nginx：仅使用 ECDHE 密码套件
ssl_ciphers ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:!DHE;
```

使用不低于2048位的参数：

如果必须使用DHE：

```
# 确保最小密钥长度
ssl_dhparam_size 2048; # Nginx
```

检测通用素数：

```
# 从服务器提取 DH 参数
openssl s_client -connect example.com:443 -cipher DHE 2>&1 | \
    openssl dhparam -inform PEM -text -noout

# 与已知的脆弱素数进行比对
# 检查该素数是否与 Apache 默认值或其他通用值匹配
```

额外防护措施：

- 使用TLS 1.3，其已完全淘汰有限域DH
- 部署证书固定以检测MITM尝试
- 监控异常连接模式，识别批量流量分析迹象
- 选择密码套件时，需考虑前向保密（PFS）的影响

关键提示：即使是高强度的1024位DH参数，一旦在大量服务器之间共享，也会因预计算成本被摊薄而变得脆弱。唯一性与长度同等重要。

参考资料

- [Common Prime Analysis](#)
- [Weak DH Research](#)
- [Technical Paper](#)

高

移动端SQL注入漏洞

Mobile SQL Injection vulnerability allows an attacker to execute malicious SQL queries on a mobile application's database, potentially leading to unauthorized access, data leakage, or data manipulation.

移动端SQL注入是一种漏洞，攻击者可利用该漏洞向移动应用注入恶意SQL语句，从而未经授权访问敏感数据或操纵数据库。

示例

Kotlin

```
kotlin
import java.sql.Connection
import java.sql.DriverManager
import java.sql.PreparedStatement
import java.sql.ResultSet

fun main() {
    val input = readLine() ?: ""
    val connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mydatabase", "username", "password")
    val statement = connection.prepareStatement("SELECT * FROM users WHERE username = ?")
    statement.setString(1, input)
    val resultSet = statement.executeQuery()

    while (resultSet.next()) {
        val username = resultSet.getString("username")
        val password = resultSet.getString("password")
        println("Username: $username, Password: $password")
    }
}
```

```

    }

    resultSet.close()
    statement.close()
    connection.close()
}

```

技术详情

发现 #1

在类 `SQLiteDatabase` 的方法 `execSQL` 中，位置为 `0` 的参数可能存在 SQL 注入漏洞。污点回溯自不可达组件：

`owasp.sat.agoat.InsecureStorageSQLiteActivity`。污点元素来源于：类 `InsecureStorageSQLiteActivity` 方法 `onCreate` 位置 `0` 签名 `(Landroid/os/Bundle;)V`。调用链：

```

at android.database.sqlite.SQLiteDatabase.execSQL
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate
at android.support.v7.app.AppCompatActivity.onCreate
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate

```

在 `android.database.sqlite.SQLiteDatabase.execSQL`

Unable to extract source code.

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p`

```

public static final synthetic android.database.sqlite.SQLiteDatabase access$getMDB$p(owasp.sat.agoat.InsecureStorageSQLiteActivity p1)
{
    return p1.mDB;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
    }
}

```

```

        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>`

```

InsecureStorageSQLiteActivity$onCreate$1(owasp.sat.agoat.InsecureStorageSQLiteActivity p1, android.widget.EditText p2,
android.widget.EditText p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
}

```

```

        if (v0_6 == null) {
            kotlin.jvm.internal.Intrinsics.throwNpe();
        }
        v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
        owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
        ((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }

```

在 `android.support.v7.app.AppCompatActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}

```

在 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
        owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
        ((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
    owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
    ((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #2

用户可控参数被用于构造存在 SQL 注入漏洞的 SQL 参数 污点追踪自不可达组件：`owasp.sat.agoat.InsecureStorageSQLiteActivity`，发现于 `owasp.sat.agoat.InsecureStorageSQLiteActivity`。从参数 `0` 到 Sink 点 `void android.database.sqlite.SQLiteDatabase.execSQL(java.lang.String)`。污点追踪：

```

at android.database.sqlite.SQLiteDatabase.execSQL()
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>()

```

```
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()
```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}
```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
}
```

```

    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occured - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
}

```

```
    return;  
}
```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>()` :

```
InsecureStorageSQLiteActivity$onCreate$1(owasp.sat.agoat.InsecureStorageSQLiteActivity p1, android.widget.EditText p2,  
android.widget.EditText p3)  
{  
    this.this$0 = p1;  
    this.$username = p2;  
    this.$password = p3;  
    return;  
}
```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)  
{  
    super.onCreate(p5);  
    try {  
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);  
        this.mDB = v0_6;  
    } catch (android.widget.Button v0_0) {  
        android.widget.EditText v1_1 = new StringBuilder();  
        v1_1.append(Error occurred while creating database: );  
        v1_1.append(v0_0.getMessage());  
        android.util.Log.e(Error:, v1_1.toString());  
        this.setContentView(2131296299);  
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),  
((android.widget.EditText) this.findViewById(2131165303)))));  
        return;  
    }  
    if (v0_6 == null) {  
        kotlin.jvm.internal.Intrinsics.throwNpe();  
    }  
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));  
    this.setContentView(2131296299);  
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),  
((android.widget.EditText) this.findViewById(2131165303)))));  
    return;  
}
```

方法 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)  
{  
    super.onCreate(p5);  
    try {  
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);  
        this.mDB = v0_6;  
    } catch (android.widget.Button v0_0) {  
        android.widget.EditText v1_1 = new StringBuilder();  
        v1_1.append(Error occurred while creating database: );  
        v1_1.append(v0_0.getMessage());  
        android.util.Log.e(Error:, v1_1.toString());  
        this.setContentView(2131296299);  
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),  
((android.widget.EditText) this.findViewById(2131165303)))));  
        return;  
    }  
    if (v0_6 == null) {  
        kotlin.jvm.internal.Intrinsics.throwNpe();  
    }  
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));  
    this.setContentView(2131296299);  
}
```

```

        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
}

```

发现 #3

类 `SQLiteDatabase` 的方法 `execSQL` 中, 位置 `0` 的参数可能存在 SQL 注入漏洞。污点从不可达组件 `owasp.sat.agoat.InsecureStorageSQLiteActivity` 追踪而来。污点元素来源: 类 `InsecureStorageSQLiteActivity`、方法 `onCreate`、位置 `0`、签名 `(Landroid/os/Bundle;)V`。调用栈:

```

at android.database.sqlite.SQLiteDatabase.execSQL
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at android.database.sqlite.SQLiteDatabase.execSQL
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate
at owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate

```

位于 `android.database.sqlite.SQLiteDatabase.execSQL`

Unable to extract source code.

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occured - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}

```

位于 `android.database.sqlite.SQLiteDatabase.execSQL`

Unable to extract source code.

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```
public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}
```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```
public final void onClick(android.view.View p7)
{
    String v0_1 = new StringBuilder();
    v0_1.append(INSERT INTO users (username, password) VALUES(');
    String v1_8 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);
    v0_1.append(v1_8.getText().toString());
    v0_1.append(',');
    String v1_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);
    v0_1.append(v1_2.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);
    } catch (android.widget.Toast v2_3) {
        CharSequence v3_1 = new StringBuilder();
        v3_1.append(Error occurred when inserting data into database: );
        v3_1.append(v2_3.getMessage());
        android.util.Log.d(Error, v3_1.toString());
        CharSequence v3_4 = this.this$0.getApplicationContext();
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(Data not saved: Error occurred - );
        v4_4.append(v2_3.getMessage());
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v2_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v2_2.execSQL(v0_2);
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    android.util.Log.v(Query, v0_2);
}
```

```
    return;  
}
```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.onClick`

```
public final void onClick(android.view.View p7)  
{  
    String v0_1 = new StringBuilder();  
    v0_1.append(INSERT INTO users (username, password) VALUES(');  
    String v1_8 = this.$username;  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_8, username);  
    v0_1.append(v1_8.getText().toString());  
    v0_1.append(',');  
    String v1_2 = this.$password;  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, password);  
    v0_1.append(v1_2.getText().toString());  
    v0_1.append('');  
    String v0_2 = v0_1.toString();  
    try {  
        android.widget.Toast v2_2 = owasp.sat.agoat.InsecureStorageSQLiteActivity.access$getMDB$p(this.this$0);  
    } catch (android.widget.Toast v2_3) {  
        CharSequence v3_1 = new StringBuilder();  
        v3_1.append(Error occurred when inserting data into database: );  
        v3_1.append(v2_3.getMessage());  
        android.util.Log.d(Error, v3_1.toString());  
        CharSequence v3_4 = this.this$0.getApplicationContext();  
        CharSequence v4_4 = new StringBuilder();  
        v4_4.append(Data not saved: Error occurred - );  
        v4_4.append(v2_3.getMessage());  
        android.widget.Toast.makeText(v3_4, ((CharSequence) v4_4.toString()), 1).show();  
        android.util.Log.v(Query, v0_2);  
        return;  
    }  
    if (v2_2 == null) {  
        kotlin.jvm.internal.Intrinsics.throwNpe();  
    }  
    v2_2.execSQL(v0_2);  
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();  
    android.util.Log.v(Query, v0_2);  
    return;  
}
```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)  
{  
    super.onCreate(p5);  
    try {  
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);  
        this.mDB = v0_6;  
    } catch (android.widget.Button v0_0) {  
        android.widget.EditText v1_1 = new StringBuilder();  
        v1_1.append(Error occurred while creating database: );  
        v1_1.append(v0_0.getMessage());  
        android.util.Log.e(Error, v1_1.toString());  
        this.setContentView(2131296299);  
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),  
((android.widget.EditText) this.findViewById(2131165303)))));  
        return;  
    }  
    if (v0_6 == null) {  
        kotlin.jvm.internal.Intrinsics.throwNpe();  
    }  
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));  
    this.setContentView(2131296299);  
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),  
((android.widget.EditText) this.findViewById(2131165303)))));  
    return;  
}
```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1.<init>`

```

InsecureStorageSQLiteActivity$onCreate$1(owasp.sat.agoat.InsecureStorageSQLiteActivity p1, android.widget.EditText p2,
android.widget.EditText p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

位于 `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #4

用户可控参数被用于构造存在SQL注入漏洞的SQL参数。污点追踪自不可达组件：`owasp.sat.agoat.SQLInjectionActivity`，发现于

`owasp.sat.agoat.SQLInjectionActivity`。从参数 `0` 到Sink `android.database.Cursor android.database.sqlite.SQLiteDatabase.rawQuery(java.lang.String, java.lang.String[])`。污点追踪路径：

```
at android.database.sqlite.SQLiteDatabase.rawQuery()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()`：

```
public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found;
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
                v4_5.append(v2_5.toString());
            } while(v3_3.moveToNext());
        } else {
        }
    }
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}
```

```
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            );
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p()` :

```
public static final synthetic android.database.sqlite.SQLiteDatabase access$getMDB$p(owasp.sat.agoat.SQLInjectionActivity p1)
{
    return p1.mDB;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p0)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            );
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
    }
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()` :

```
SQLInjectionActivity$onCreate$1(owasp.sat.agoat.SQLInjectionActivity p1, android.widget.EditText p2)
{
    this.this$0 = p1;
    this.$username = p2;
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

方法 `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

发现 #5

用户可控参数被用于构造存在 SQL 注入漏洞的 SQL 参数。污点从不可达组件 `owasp.sat.agoat.SQLInjectionActivity` 追踪, 发现于

`owasp.sat.agoat.SQLInjectionActivity`。从参数 `0` 到 sink `android.database.Cursor android.database.sqlite.SQLiteDatabase.rawQuery(java.lang.String, java.lang.String[])`。污点追踪路径:

```
at android.database.sqlite.SQLiteDatabase.rawQuery()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()
```

```
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            };
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
```

```

v0_1.append('');
String v0_2 = v0_1.toString();
try {
    owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
    android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
} catch (boolean v2_13) {
    android.database.Cursor v3_5 = new StringBuilder();
    v3_5.append(Error occurred when inserting data into database: );
    v3_5.append(v2_13.getMessage());
    android.util.Log.d(Error, v3_5.toString());
    android.database.Cursor v3_8 = this.this$0.getApplicationContext();
    StringBuilder v4_10 = new StringBuilder();
    v4_10.append(Data not saved: Error occurred - );
    v4_10.append(v2_13.getMessage());
    android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}
if (v3_2 == null) {
    kotlin.jvm.internal.Intrinsics.throwNpe();
}
android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
StringBuilder v4_5 = new StringBuilder();
if (v3_3 == null) {
    String v5_13 = new StringBuilder();
    v5_13.append(User: ());
    android.widget.EditText v6_2 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v6_2, username);
    v5_13.append(v6_2.getText().toString());
    v5_13.append() not found);
    v4_5.append(v5_13.toString());
} else {
    if (v3_3.getCount() > 0) {
        v3_3.moveToFirst();
        do {
            boolean v2_5 = new StringBuilder();
            v2_5.append(Username: ());
            v2_5.append(v3_3.getString(1));
            v2_5.append() password: ());
            v2_5.append(v3_3.getString(2));
            v2_5.append()
        } while(v3_3.moveToNext());
    } else {
    }
}
android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
android.util.Log.e(Error:, v4_5.toString());
android.util.Log.v(Query, v0_2);
return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
    }
}

```

```

        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            };
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
android.util.Log.e(Error:, v4_5.toString());
android.util.Log.v(Query, v0_2);
return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()` :

```

SQLInjectionActivity$onCreate$1(owasp.sat.agoat.SQLInjectionActivity p1, android.widget.EditText p2)
{
    this.this$0 = p1;
    this.$username = p2;
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}

```

```
}
```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

发现 #6

用户可控参数被用于构造存在 SQL 注入漏洞的 SQL 参数。污点追踪源自不可达组件： `owasp.sat.agoat.SQLInjectionActivity` 发现于

`owasp.sat.agoat.SQLInjectionActivity`。从参数 `0` 到 Sink `android.database.Cursor android.database.sqlite.SQLiteDatabase.rawQuery(java.lang.String, java.lang.String[])`。污点追踪：

```
at android.database.sqlite.SQLiteDatabase.rawQuery()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at android.database.sqlite.SQLiteDatabase.rawQuery()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    }
```

```

    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()

            };

            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}

android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
android.util.Log.e(Error:, v4_5.toString());
android.util.Log.v(Query, v0_2);
return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append('');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occured - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()

            };

            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}

```

```

    }
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            );
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p9)
{

```

```

String v0_1 = new StringBuilder();
v0_1.append(SELECT * FROM users WHERE username=');
String v1_6 = this.$username;
kotlin.jvm.internal.Intrinsics.checkNotNull(v1_6, username);
v0_1.append(v1_6.getText().toString());
v0_1.append('');
String v0_2 = v0_1.toString();
try {
    owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
    android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
} catch (boolean v2_13) {
    android.database.Cursor v3_5 = new StringBuilder();
    v3_5.append(Error occurred when inserting data into database: );
    v3_5.append(v2_13.getMessage());
    android.util.Log.d(Error, v3_5.toString());
    android.database.Cursor v3_8 = this.this$0.getApplicationContext();
    StringBuilder v4_10 = new StringBuilder();
    v4_10.append(Data not saved: Error occured - );
    v4_10.append(v2_13.getMessage());
    android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
    android.util.Log.v(Query, v0_2);
    return;
}
if (v3_2 == null) {
    kotlin.jvm.internal.Intrinsics.throwNpe();
}
android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
StringBuilder v4_5 = new StringBuilder();
if (v3_3 == null) {
    String v5_13 = new StringBuilder();
    v5_13.append(User: ());
    android.widget.EditText v6_2 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v6_2, username);
    v5_13.append(v6_2.getText().toString());
    v5_13.append() not found);
    v4_5.append(v5_13.toString());
} else {
    if (v3_3.getCount() > 0) {
        v3_3.moveToFirst();
        do {
            boolean v2_5 = new StringBuilder();
            v2_5.append(Username: ());
            v2_5.append(v3_3.getString(1));
            v2_5.append() password: ());
            v2_5.append(v3_3.getString(2));
            v2_5.append()

        );
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
android.util.Log.e(Error:, v4_5.toString());
android.util.Log.v(Query, v0_2);
return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this.setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()` :

```

SQLInjectionActivity$onCreate$1(owasp.sat.agoat.SQLInjectionActivity p1, android.widget.EditText p2)
{

```

```

        this.this$0 = p1;
        this.$username = p2;
        return;
    }

```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}

```

方法 `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}

```

建议

为缓解移动端 SQL 注入漏洞，建议采取以下措施：

- 使用参数化查询或预编译语句，将 SQL 代码与用户输入分离。
- 在将用户输入插入数据库之前，对其进行清理和验证，以缓解二阶 SQL 注入。
- 定期更新 SQL 驱动，以修复已知漏洞。

KOTLIN

```

import java.sql.Connection
import java.sql.DriverManager
import java.sql.PreparedStatement

fun main() {
    val url = "jdbc:mysql://localhost:3306/mydatabase"
    val username = "username"
    val password = "password"

    var connection: Connection? = null
    var preparedStatement: PreparedStatement? = null

    try {
        connection = DriverManager.getConnection(url, username, password)
        val sql = "INSERT INTO users (name, email) VALUES (?, ?)"
        preparedStatement = connection.prepareStatement(sql)

        // Set values for the parameters
        preparedStatement.setString(1, "John")
        preparedStatement.setString(2, "john@example.com")

        // Execute the prepared statement
        preparedStatement.executeUpdate()
    } catch (e: Exception) {
        e.printStackTrace()
    } finally {
        preparedStatement?.close()
        connection?.close()
    }
}

```

```
}  
}
```

🔗 SWIFT

```
import Foundation  
import SQLite3  
  
func insertUser(name: String, email: String) {  
    var db: OpaquePointer?  
    var statement: OpaquePointer?  
  
    let dbPath = "path_to_your_database_file.db"  
  
    if sqlite3_open(dbPath, &db) == SQLITE_OK {  
        let insertStatementString = "INSERT INTO users (name, email) VALUES (?, ?)"  
  
        if sqlite3_prepare_v2(db, insertStatementString, -1, &statement, nil) == SQLITE_OK {  
            sqlite3_bind_text(statement, 1, (name as NSString).utf8String, -1, nil)  
            sqlite3_bind_text(statement, 2, (email as NSString).utf8String, -1, nil)  
  
            if sqlite3_step(statement) == SQLITE_DONE {  
                print("Successfully inserted row.")  
            } else {  
                print("Could not insert row.")  
            }  
        } else {  
            print("INSERT statement could not be prepared.")  
        }  
  
        sqlite3_finalize(statement)  
    } else {  
        print("Unable to open database.")  
    }  
  
    sqlite3_close(db)  
}
```

<> FLUTTER

```
import 'package:sqflite/sqflite.dart';  
import 'package:path/path.dart';  
  
void insertUser(String name, String email) async {  
    Database database = await openDatabase(  
        join(await getDatabasesPath(), 'mydatabase.db'),  
        onCreate: (db, version) {  
            return db.execute(  
                "CREATE TABLE users(id INTEGER PRIMARY KEY, name TEXT, email TEXT)",  
            );  
        },  
        version: 1,  
    );  
  
    await database.transaction((txn) async {  
        await txn.rawInsert(  
            'INSERT INTO users(name, email) VALUES(?, ?)',  
            [name, email],  
        );  
    });  
}
```

参考资料

- [OWASP Mobile Security Testing Guide \(MSTG\)](#)

中 管理员权限由客户端布尔标志决定且未经服务端验证

BinaryPatchingActivity gates admin UI functionality using a local boolean field that is never validated by a backend API, cryptographic token, or re-authentication challenge.

该 Activity 将 `private final boolean isAdmin` 字段初始化为 `false`，并在 `onCreate()` 中依据该字段决定是否启用管理员按钮及显示管理员文本。由于缺乏服务端角色校验、签名验证及阶梯认证，攻击者可在运行时通过 Frida 切换该标志位，或通过篡改 APK 的 Dalvik 字节码来提升权限。这揭示了一条完全依赖客户端信任的权限提升路径，无需任何二次验证。

技术详情

发现 #1

根因

`BinaryPatchingActivity` 通过一个初始化为 `false` 的 `private final boolean isAdmin` 字段控制对管理员 UI 元素的访问。该 Activity 在 `onCreate()` 中检查此标志；若为 `true`，则启用管理员按钮并更新 TextView。不存在后端 API 调用、密码学认证或增强认证来校验管理员状态。

漏洞代码

`BinaryPatchingActivity.java`（第 17、39-54 行）：

```
private final boolean isAdmin; // Initialized to false

// ...
public final boolean isAdmin() {
    return this.isAdmin;
}

// In onCreate():
if (this.isAdmin) {
    ((TextView) findViewById(R.id.adminText)).setText("You are an admin");
    ((Button) findViewById(R.id.adminButton)).setEnabled(true);
}
```

利用证据

【静态分析】该字段为编译期常量，且没有任何远程验证。攻击者可以使用 Frida hook `BinaryPatchingActivity.isAdmin()` 并返回 `true`，或对 APK 进行二进制修补，将 Dalvik 初始化器由 `const/4 v0, 0x0` 修改为 `0x1`。后端从未被调用，且在管理员 UI 被启用时不会触发重新认证。

验证证据

根因

`BinaryPatchingActivity`（包名 `owasp.sat.agoat`）通过 `private final boolean isAdmin` 字段控制对管理员 UI 元素的访问，该字段声明于反编译后的 Java 源码（`base.apk/sources/owasp/sat/agoat/BinaryPatchingActivity.java`）第 17 行。该字段接收 JVM 默认值 `false`，并因 `final` 修饰符而在构造后不可变。在 `onCreate()`（第 48 行）中，Activity 检查 `this.isAdmin`；当其为 `true` 时，启用 `adminButton` 并更新 `isAdminText`。验证确认：在 Activity 及其可达调用图中，不存在任何外部输入向量、网络请求、密码学认证、签名校验或增强认证。

漏洞代码

`BinaryPatchingActivity.java`（第 17、39-54 行）：

```
// Line 17
private final boolean isAdmin;

// Accessor method, lines 39-41
public final boolean isAdmin() {
    return this.isAdmin;
}

// onCreate() gating logic, lines 48-54
if (this.isAdmin) {
    TextView isAdminText = (TextView) _$.findCachedViewById(R.id.isAdminText);
    Intrinsics.checkExpressionValueIsNotNull(isAdminText, "isAdminText");
    isAdminText.setText("You are Admin Now");
    Button adminButton = (Button) _$.findCachedViewById(R.id.adminButton);
    Intrinsics.checkExpressionValueIsNotNull(adminButton, "adminButton");
    adminButton.setEnabled(true);
}
```

当 `isAdmin` 为 `false`（默认值）时，布局文件 `activity_binary_patching.xml` 渲染的 `isAdminText` 为 "Sorry, You don't have access to this feature." 而 `adminButton` 保持禁用状态（`android:enabled="false"`）。不存在其他代码路径将管理员状态交由远程权威验证。

利用证据

【静态分析】源码审计证实，`isAdmin` 仅在合成构造方法中被赋值一次（隐式初始化为 `false` ），之后从未被修改。对 `isAdmin\s*='` 进行的全项目搜索返回零条匹配，确认不存在任何 setter、反射或原生代码修改该标志位。

Manifest 审计（`base.apk/AndroidManifest.xml`，第 5 行）发现：

```
<application android:allowBackup="true"
  android:debuggable="true"
  android:networkSecurityConfig="@xml/network_security_config"
  ... >
```

`android:debuggable="true"` 允许 Frida 或其他插桩工具**无需 root 权限**即可附加到目标进程，极大降低了攻击者在运行时 hook `BinaryPatchingActivity.isAdmin()` 并强制其返回 `true` 的门槛。该 Activity 并非 exported（`android:exported` 默认为 `false` ），但一旦应用在本地运行，攻击者即可修补 Dalvik 初始化器，将 `const/4 vX, 0x0` 改为 `const/4 vX, 0x1`，从而在每次启动时永久启用管理员 UI。当管理员界面被激活时，不会触发任何后端校验、重新认证或完整性检查。

建议

即时修复

从 `BinaryPatchingActivity` 中移除客户端 `isAdmin` 标志位。改为通过已认证的 TLS 向后端 `/user/role` 端点查询角色，并将角色缓存在 `EncryptedSharedPreferences` 中，设置较短的 TTL。

永久修复

对每一项特权操作实施服务端角色强制校验。客户端应将签名的 JWT 或会话令牌发送至后端，由后端在返回管理员数据或启用管理员功能前校验 role 声明：

```
// 客户端：仅做 UI 限制；由后端执行授权校验
// 服务端：在处理前校验 token.role == "admin"
```

验证

使用 Frida 对 `BinaryPatchingActivity.isAdmin()` 进行 hook，使其返回 `true`。确认尝试执行的管理员操作仍然失败，因为后端会拒绝该请求。审查源代码，确保不存在仅靠本地布尔值就能启用特权 UI 元素的情况。

中 TrafficActivity 中的证书固定因异常处理过于宽泛而静默失败

`TrafficActivity.doPinning()` 中的 OkHttp CertificatePinner 已失效，因为过于宽泛的 catch 块吞没了 `SSLPeerUnverifiedException`，从而允许静默的中间人攻击。

`TrafficActivity.doPinning()` 通过 OkHttp `CertificatePinner` 为 `owasp.org` 配置了三组 SHA-256 pin。然而，整个网络请求都被包裹在 `try { ... } catch (Exception e) { e.printStackTrace(); }` 中，这会在证书固定失败时静默忽略 `SSLPeerUnverifiedException`。再加上全局对用户安装 CA 的信任，攻击者即可利用代理证书拦截该固定连接，而用户不会收到任何通知。

技术详情

发现 #1

根因

开发者使用 OkHttp 的 `CertificatePinner.Builder` 实现了证书固定，但将被固定的请求包裹在一个过于宽泛的异常处理器中，该处理器仅打印堆栈跟踪而未执行安全失效。这破坏了证书固定的安全保证，因为在固定值不匹配后，应用仍会继续静默执行。

漏洞代码

`TrafficActivity.java` 第 140 至 141 行：

```
CertificatePinner pinner1 = new CertificatePinner.Builder()
    .add("owasp.org", "sha256/gdU/UHC1HJBFBideKuyHm/Lq/aQvMLyuTtcvTEE/1JQ=")
    .add("owasp.org", "sha256/YLh1dUR9y6Kja30RrAn7JKnbQG/uEtLMkBgFF2Fuihg=")
    .add("owasp.org", "sha256/Vjs8r4z+80wjNcr1YKepWQboSIRi63WswXhIMN+ewys=")
    .build();
OkHttpClient client = new OkHttpClient.Builder().certificatePinner(pinner1).build();
```

`TrafficActivity.java` 第 147 至 148 行：

```
} catch (Exception e) {
    e.printStackTrace();
```

```
}
```

利用证据

[Static Analysis]反编译源码显示，证书固定逻辑被隔离在 `doPinning()` 中，且仅对 `owasp.org` 进行固定。由于应用的 Network Security Config 已信任用户安装的 CA，TLS 握手会接受代理证书。随后证书固定器抛出 `SSLPeerUnverifiedException`，但第 147 至 148 行处宽泛的 `catch (Exception e)` 代码块将其静默吞没。对 `okhttp3.CertificatePinner.check()` 进行 FridaHook 也能使固定检查失效：

```
Java.perform(function () {
    var Pinner = Java.use("okhttp3.CertificatePinner");
    Pinner.check.overload('java.lang.String', 'java.util.List').implementation = function (hostname, certs) {
        console.log("[*] Bypassed pin for: " + hostname);
    };
});
```

验证证据

根因

验证确认，`TrafficActivity.doPinning()` 在 `owasp/sat/agoat/TrafficActivity.java` 第 139 至 149 行是整个应用中唯一的证书固定执行点。它构建了 `OkHttpClient.CertificatePinner`，为 `owasp.org` 配置了三个 SHA-256 固定值，但将同步 `execute()` 调用包裹在单个宽泛的 `try` 代码块中，并以 `catch (Exception e) { e.printStackTrace(); }` 收尾。由于 `SSLPeerUnverifiedException` 是 `Exception` 的子类，任何固定不匹配都会被确定性地吞没，而非安全失效。应用的 `AndroidManifest.xml` 引用了 `android:networkSecurityConfig="@xml/network_security_config"`，对应的 `res/xml/network_security_config.xml` 包含 `<certificates src="user"/>`，指示操作系统信任用户安装的 CA。这使得代理证书能在 `OkHttpClient` 固定器将其拒绝之前通过 TLS 握手。

漏洞代码

已在反编译源码树（`/workspace/source_code_analysis_dfcb434d7ae9421693bf4c0774f91126/base.apk/sources/owasp/sat/agoat/TrafficActivity.java`）中验证：第 139 至 146 行：

```
try {
    CertificatePinner pinner1 = new CertificatePinner.Builder()
        .add("owasp.org", "sha256/gdU/UHClHJBfIdeKuyHm/Lq/aQvMLyuTtcvTEE/1JQ=")
        .add("owasp.org", "sha256/YLh1dUR9y6Kja30RrAn7JKnBQG/uEtLMkBgFF2Fuihg=")
        .add("owasp.org", "sha256/Vjs8r4z+80wjNcr1YKepWQboSIRi63WswXhIMN+eWys=")
        .build();
    OkHttpClient client = new OkHttpClient.Builder().certificatePinner(pinner1).build();
    Request request = new Request.Builder().url("https://owasp.org").build();
    Response response = client.newCall(request).execute();
    ResponseBody body = response.body();
    Log.v("Response", body != null ? body.string() : null);
} catch (Exception e) {
    e.printStackTrace();
}
```

同样的模式也出现在另一份反编译文件（`TrafficActivity$doPinning$1.java` 第 28 行）中。

利用证据

[Static Analysis]通过对所有反编译应用源码（`/workspace/jadx_out/sources`、`/workspace/jadx_out2/sources`、`/workspace/decompiled_source/sources` 及相关分析目录）进行全面检索，确认 `CertificatePinner.Builder` 仅在 `TrafficActivity.doPinning()`（及其匿名内部类 `TrafficActivity$doPinning$1.invoke2()`）中实例化，且唯一被固定的主机名为 `owasp.org`。代码库中不存在任何其他替代固定库（TrustKit、AndroidPinning）、自定义 `TrustManager`、`HostnameVerifier` 或 `SSLContext` 操作。唯一的 `OkHttpClient.Builder.certificatePinner()` 调用点位于 `TrafficActivity.java` 第 141 行。

该方法仅由 UI 按钮点击（`R.id.PinningButton`）触发，该按钮在 Anko `doAsync` 后台线程中调用 `doPinning()`。Source-to-sink 分析确认，网络响应仅流向 `android.util.Log.v("Response", body.string())`（第 146 行）；不存在任何 UI（`Toast`、`TextView`）、`SharedPreferences`、文件或 IPC 接收点。因此，当 `client.newCall(request).execute()` 因固定不匹配而抛出 `SSLPeerUnverifiedException` 时，异常在第 147 至 148 行被捕获、输出到 `System.err`，应用继续静默执行，且无任何面向用户的反馈。

证据局限性：本次验证阶段未执行基于主动拦截代理或实时 Frida 插桩的动态利用。该漏洞通过异常处理层级结构与代码库范围隔离的确定性静态分析得以确认。

建议

即时修复

将 `doPinning()` 中过于宽泛的 catch 块替换为特定异常处理，并确保安全失效：

```
} catch (SSLPeerUnverifiedException e) {
    Log.e("PINNING_FAILURE", "Certificate pinning failed for owasp.org", e);
    // Terminate connection and notify user
    return;
} catch (IOException e) {
    Log.e("NETWORK_ERROR", "Network error occurred", e);
}
```

永久修复

1. 请勿吞掉 `SSLPeerUnverifiedException` 。将任何 pinning 失败视为该连接的致命安全错误。
2. 使用 OkHttp `CertificatePinner` 或 `network_security_config.xml` 中的 `<pin-set>` ，将 pinning 扩展至所有生产端点，而不仅限于 `owasp.org` 。

验证

1. 使用代理 CA 拦截 `doPinning()` 中对 `https://owasp.org` 的请求。
2. 未修复时，应用会静默记录异常且不显示任何数据。
3. 修复后，应用必须记录 pinning 失败并终止连接，且无任何网络请求完成。

中 不安全的外部存储权限泄露凭证 (InsecureStorageSDCardActivity)

`InsecureStorageSDCardActivity` 将用户凭据写入外部存储中的临时文件，并显式赋予该文件全局可读和全局可写权限。

该漏洞会将用户凭据暴露给同一设备上的其他应用。虽然已确认此 APK 版本中不存在预期的文件路径遍历攻击向量，但现有不安全存储实现仍会对共享存储中的明文凭据持久化构成实质性机密性风险。

技术详情

发现 #1

根因

该 Activity 在 `Environment.getExternalStorageDirectory()` 上构建临时文件，随后调用 `setReadable(true)` 和 `setWritable(true)` ，向设备上的所有应用授予读写权限。此外，绝对文件路径还通过 `System.out.println()` 泄露到了 logcat。

漏洞代码

`InsecureStorageSDCardActivity.java` (第 68–76 行)：

```
File externalStorageDirectory = Environment.getExternalStorageDirectory();
Intrinsics.checkExpressionValueIsNotNull(externalStorageDirectory,
    "Environment.getExternalStorageDirectory()");
File userinfo = File.createTempFile("users", "tmp",
    new File(externalStorageDirectory.getAbsolutePath()));

System.out.println((Object) ("userinfo " + userinfo));
userinfo.setReadable(true);
userinfo.setWritable(true);

FileWriter fw = new FileWriter(userinfo);
fw.write(data);
fw.close();
```

利用证据

[静态分析] 源码证实，每次调用 Save 按钮都会创建一个全局可读的文件，其中包含拼接后的用户名、密码和时间戳。任何持有 `READ_EXTERNAL_STORAGE` 权限的第三方应用都可以枚举并读取这些文件。logcat 泄露 (`System.out.println`) 还会暴露绝对路径，从而无需进行文件系统枚举。

攻击场景： 具备 `READ_EXTERNAL_STORAGE` 权限的恶意应用监控 logcat，或在外部存储根目录枚举 `users*.tmp` 文件，以提取明文凭证。

验证证据

根因

该应用目标 SDK 为 `targetSdkVersion=26` (Android 8.0)，强制使用旧版外部存储行为，此时 `Environment.getExternalStorageDirectory()` 返回一个共享的全局可浏览命名空间 (例如 `/sdcard/` 或 `/storage/emulated/0/`)。在此目录中创建的文件本质上可被任何持有 `READ_EXTERNAL_STORAGE` 权限的其他已安装应用访问，因为底层 `sdcardfs/FUSE` 模拟 FAT32 挂载不强制执行 Unix 权限位。验证确认 manifest 中同时启用了 `android:debuggable="true"` 和 `android:allowBackup="true"` ，通过运行时内存检查与 adb 备份提取进一步加剧了机密性风险。

漏洞代码

对反编译源码（`base.apk/sources/owasp/sat/agoat/InsecureStorageSDCardActivity.java`）的验证确认了每项不安全操作的精确行号，并追踪了完整的明文数据组装过程：

写入磁盘的凭证组装（第 45–51 行）：

```
StringBuilder sb = new StringBuilder();
sb.append(" This data is stored in SdCard on ").append(new Date().toString()).append("\n");
sb.append(" Username - ").append(username.getText().toString()).append("\n");
sb.append(" Password - ").append(password.getText().toString());
String data = sb.toString();
```

在共享存储上创建文件（第 68–70 行）：

```
File externalStorageDirectory = Environment.getExternalStorageDirectory();
Intrinsics.checkExpressionValueIsNotNull(externalStorageDirectory,
    "Environment.getExternalStorageDirectory()");
File userinfo = File.createTempFile("users", "tmp",
    new File(externalStorageDirectory.getAbsolutePath()));
```

Logcat 路径泄露（第 71 行）：

```
System.out.println((Object) ("userinfo " + userinfo));
```

权限调用（第 72–73 行）：

```
userinfo.setReadable(true);
userinfo.setWritable(true);
```

[静态分析] 经验证，`setReadable(boolean)` 与 `setWritable(boolean)` 的调用均为单参数仅限所有者的重载形式。在 API 26 旧版外部存储上，这一区别在功能上无关紧要，因为模拟文件系统不遵循 `chmod` 位；放置在共享外部存储根目录下的任何文件实际上都可被所有持有 `READ_EXTERNAL_STORAGE` 权限的应用读取。

利用证据

[静态分析] Manifest 与 SDK 上下文：

- `AndroidManifest.xml` 声明了 `WRITE_EXTERNAL_STORAGE` 与 `READ_EXTERNAL_STORAGE`。
- 从 `base.apk/apktool.yml` 中提取到 `targetSdkVersion="26"` 与 `minSdkVersion="18"`。
- `android:debuggable="true"` 与 `android:allowBackup="true"` 均存在。

[静态分析] 缓解措施缺失：

- 在应用包内对 `setReadable(false)`、`setWritable(false)`、`chmod`、`setExecutable` 与 `setPermissions` 进行项目级搜索，返回 **零匹配**。代码库中不存在任何权限收紧、清理或子目录隔离逻辑。

[静态分析] 无输入验证：

- `username.getText().toString()` 与 `password.getText().toString()` 被直接拼接到 `data` 字符串中，未进行空值检查、长度限制或净化处理。

[静态分析] 可预测的发现：

- `File.createTempFile("users", "tmp", ...)` 在外部存储根目录生成 `users*.tmp`。无条件的 `System.out.println` 将绝对路径（例如 `userinfo /sdcard/users12345.tmp`）泄露到 logcat，消除了文件系统枚举的必要性。具备 `READ_EXTERNAL_STORAGE` 权限的恶意应用可直接定位该精确路径并提取明文凭证字符串。

建议

即时修复

从 `InsecureStorageSDCardActivity.java` 中移除显式的 `setReadable(true)` 和 `setWritable(true)` 调用。

永久修复

切勿将敏感认证数据存储在外部存储中。将文件写入操作移至应用的私有内部目录，并依赖 Android 的默认权限模型：

```
File internalDir = getFilesDir();
File userinfo = File.createTempFile("users", "tmp", internalDir);
// setReadable/setWritable calls removed - internal files are app-private by default
FileWriter fw = new FileWriter(userinfo);
```

```
fw.write(data);
fw.close();
```

此外，移除泄漏绝对路径的 `System.out.println` 日志语句。

验证

1. 在运行时环境中拦截文件创建操作，确认新文件已创建于 `/data/data/owasp.sat.agoat/files/` 下。
2. 验证文件模式未向其他UID授予读写权限（`ls -l` 显示 `rw-----`）。
3. 确认点击保存按钮时，logcat不再输出绝对文件路径。

中 敏感界面缺少FLAG_SECURE，允许未经授权的屏幕截取

owasp.sat.agoat 中的全部 25 个 Activity 均未设置 FLAG_SECURE，导致包含 PIN 输入、凭据录入和 OTP 生成在内的敏感界面可被任意截屏与录屏。

所有 activity 窗口均未设置 `WindowManager.LayoutParams.FLAG_SECURE`，导致敏感工作流可通过 `adb` 及第三方工具被任意截屏和录屏。该问题已在 PIN 验证、凭据输入、OTP 生成及支付界面被动态确认。攻击者或已失陷进程无需用户交互即可直接从截图中窃取凭据与敏感令牌，并可利用这些侦察信息发起更精准的 overlay 攻击。

技术详情

发现 #1

根本原因

`owasp.sat.agoat` 中的每个 Activity 在初始化窗口时均未设置 `WindowManager.LayoutParams.FLAG_SECURE`。该标记是 Android 的标准防护机制，用于阻止系统截图工具、`adb` 以及第三方录屏应用捕获屏幕内容。

漏洞代码

[静态分析]所有 25 个 Activity 的反编译 smali 及 Java 源码中均未发现 `FLAG_SECURE` 引用，且没有任何 Activity 在 `onCreate()` 中调用 `getWindow().addFlags(WindowManager.LayoutParams.FLAG_SECURE)` 或 `getWindow().setFlags(...)`。

```
grep -r "FLAG_SECURE" /workspace/decompiled_source/ # 0 matches
grep -r "windowIsSecure" /workspace/decompiled_source/ # 0 matches
```

利用证据

模拟器上的动态测试证实，所有敏感界面均可被无限制地截屏与录屏：

```
# Succeeded on PIN verification, credential input, OTP, and payment screens
adb shell screencap /sdcard/screen_pin_overlay.png
adb shell screenrecord /sdcard/record.mp4
```

`dumpsys window` 分析确认，目标应用的所有窗口均未设置 `FLAG_SECURE` 标记。

已记录的取证包括：

- `/workspace/screen_pin_overlay.png`
- `/workspace/screen_hardcode_overlay.png`
- `/workspace/screen_clipboard_overlay.png`
- `/workspace/screen_overlay_test2.png`
- `/workspace/clickjacking_pin.mp4`
- `/workspace/final_attack_demo.mp4`

这使得恶意应用或受感染设备能够在用户无感知的情况下，直接通过截屏窃取凭证、PIN、OTP 及信用卡号。

验证证据

验证尝试 1

`adb shell screencap` 在所有敏感界面均执行成功，包括 PIN 验证与凭证输入界面。

验证尝试 2

`adb shell screenrecord` 实时捕获了覆盖层攻击画面，且未受到任何限制。

验证尝试 3

`dumpsys window` 分析确认目标应用窗口不存在 `FLAG_SECURE` 标记。

验证尝试 4

截图 `/workspace/screen_pin_overlay.png`、`/workspace/screen_hardcode_overlay.png`、`/workspace/screen_clipboard_overlay.png`、`/workspace/screen_overlay_test2.png` 捕获了未受限制的屏幕内容。

验证尝试 5

根本原因

`owasp.sat.agoat` 中的 `AccessControl1ViewActivity` 因清单声明包含带有 `action.VIEW` 及自定义 scheme `androgoat://vulnapp` 的 `<intent-filter>`，却未显式设置 `android:exported="false"`，因而被隐式导出。此外，该 Activity 的入口点未包含任何身份验证逻辑，尽管应用在对应的 `AccessControlIssue1Activity` 中强制要求 PIN 验证。

漏洞代码

来自 `/workspace/agoat_decode/AndroidManifest.xml` 的清单声明：

```
<activity android:label="@string/activity" android:name="owasp.sat.agoat.AccessControl1ViewActivity">
  <intent-filter>
    <action android:name="android.intent.action.VIEW"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:host="vulnapp" android:scheme="androgoat"/>
  </intent-filter>
</activity>
```

对 `/workspace/agoat_analysis/sources/owasp_sat_agoat_AccessControl1ViewActivity.java` 及其对应 smali 的静态分析确认：

- `onCreate(Bundle)` 设置内容视图并附加一个调用 `startService()` 的 `OnClickListener`。
- `onNewIntent(Intent)` 未被重写。
- `onResume()` 未被重写。
- 未引用 `verifyIfPinSet()`、`isPinCorrect()`、`verifyPINView()` 或 `SharedPreferences` PIN 验证。

利用证据

设备已配置 PIN，`pinSet=true`，MD5 哈希值为 `1234`：

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <boolean name="pinSet" value="true" />
  <string name="pin">81dc9bdb52d04dc20036dbd8313ed055</string>
</map>
```

通过 ADB 进行动态利用时，无需任何 PIN 提示即成功：

```
adb shell am start -W -a android.intent.action.VIEW -d "androgoat://vulnapp" owasp.sat.agoat
```

输出：

```
Starting: Intent { act=android.intent.action.VIEW dat=androgoat://vulnapp pkg=owasp.sat.agoat }
Status: ok
Activity: owasp.sat.agoat/.AccessControl1ViewActivity
ThisTime: 481
TotalTime: 481
WaitTime: 542
Complete
```

`dumpsys window` 确认该 Activity 已处于焦点状态：

```
mCurrentFocus=Window{10344ff u0 owasp.sat.agoat/owasp.sat.agoat.AccessControl1ViewActivity}
mFocusedApp=AppWindowToken{b8bff17 token=Token{77c6796 ActivityRecord{30936b1 u0 owasp.sat.agoat/.AccessControl1ViewActivity t195}}}
```

Logcat 记录了本次直接启动：

```
I ActivityManager: START u0 {act=android.intent.action.VIEW dat=androgoat://vulnapp flg=0x10008000 pkg=owasp.sat.agoat  
cmp=owasp.sat.agoat/.AccessControl1ViewActivity} from uid 2000  
I ActivityManager: Displayed owasp.sat.agoat/.AccessControl1ViewActivity: +481ms
```

在 `/workspace/deeplink_accesscontrol1.png` 捕获的截图提供了视觉证据，证明该 Activity 通过深度链接被直接启动且未经过 PIN 验证。

建议

即时缓解

优先为风险最高的 Activity 添加 `FLAG_SECURE`（PIN 验证、凭证输入、OTP 生成）：

```
getWindow().setFlags(  
    WindowManager.LayoutParams.FLAG_SECURE,  
    WindowManager.LayoutParams.FLAG_SECURE  
);
```

永久修复

创建一个 Base Activity 类，供所有敏感页面使用，以集中管理窗口保护：

```
public abstract class SecureActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        getWindow().setFlags(  
            WindowManager.LayoutParams.FLAG_SECURE,  
            WindowManager.LayoutParams.FLAG_SECURE  
        );  
    }  
}
```

在 `AndroidManifest.xml` 中为所有敏感 Activity 注册该基类：

```
<activity android:name=".AccessControlIssue1Activity"  
    android:theme="@style/SecureTheme" />
```

验证

1. 在每个敏感 Activity 处于前台时，运行 `adb shell screencap /sdcard/test.png`。确认截取的图片为黑屏。
2. 运行 `adb shell screenrecord /sdcard/test.mp4`，确认录制内容不包含受保护 Activity 的内容。
3. 检查 `dumpsys window` 的输出，确认 `FLAG_SECURE`（0x00002000）出现在窗口属性中。

中 Insecure Backup and Debug Flags Expose Application Sandbox

应用清单启用了 `android:debuggable="true"` 和 `android:allowBackup="true"`，任何具备 USB 调试权限的用户均可提取、修改并恢复所有应用私有数据，或在运行进程中附加调试器。

这些 manifest 标志移除了关键的沙箱保护机制。`android:allowBackup="true"` 允许通过 `adb backup` 和 `adb restore` 完整提取并篡改数据，而 `android:debuggable="true"` 则允许 shell 用户通过 `run-as` 以应用 UID 运行，并附加 JDWP 调试器。二者结合提供了多种简单的利用路径，可覆盖认证数据并检查运行时内存。

技术详情

发现 #1

根因

应用清单声明了 `android:debuggable="true"` 与 `android:allowBackup="true"`。这些标记允许任何具备 USB 调试访问权限的用户通过 `adb backup` 提取所有应用私有数据、通过 `adb restore` 还原被篡改的数据，并通过 `run-as` 以应用 UID 执行命令。这将授予对 SharedPreferences、SQLite 数据库及其他沙箱文件的无限制读写访问权限。

利用证据

【静态分析】`AndroidManifest.xml` 同时包含这两个标记：

```
<application
    android:debuggable="true"
    android:allowBackup="true"
    ... >
```

利用 `run-as`，攻击者可直接向应用沙箱写入数据：

```
adb shell run-as owasp.sat.agoat sh -c 'cat > /data/data/owasp.sat.agoat/shared_prefs/pinDetails.xml'
```

利用 `adb backup`，攻击者可提取整个应用数据，离线修改文件，并还原被篡改的归档：

```
adb backup -f agoat.ab owasp.sat.agoat
# ... modify extracted files ...
adb restore agoat.ab
```

这些攻击向量可导致本地数据完整性与机密性完全失陷。

验证证据

根因

验证将根因从清单标记本身进一步扩展，涵盖存储层完整性控制的缺失与高价值明文数据存储的存在。静态分析确认 `AndroidManifest.xml` 中缺少 `android:fullBackupContent`，即默认备份范围涵盖整个 `/data/data/owasp.sat.agoat/` 目录。此外，所有敏感数据持久化逻辑均缺乏加密、HMAC 或数字签名，导致应用在还原时静默接受被篡改的沙箱文件。

漏洞代码

验证识别出由特定应用组件创建的五個具体沙箱驻留目标：

1. `pinDetails.xml` —— 未加盐 MD5 PIN 存储 `AccessControlIssue1Activity.java:76-82`：`hashPIN()` 方法调用了 `MessageDigest.getInstance("MD5")`，其等效位置约为第 113 行，生成一个在密码学上已损坏的未加盐摘要。

2. `users.xml` —— 明文凭据存储 `InsecureStorageSharedPrefs.java:50-57`：

```
SharedPreferences sharedPreferences = InsecureStorageSharedPrefs.this.getSharedPreferences("users", 0);
SharedPreferences.Editor editor = sharedPreferences.edit();
editor.putString("username", username2.getText().toString());
editor.putString("password", password2.getText().toString());
editor.commit();
```

3. `aGoat` SQLite 数据库 —— 明文凭据仓库 `InsecureStorageSQLiteActivity.java:47-52、65-72`：

```
SQLiteDatabase openOrCreateDatabase = openOrCreateDatabase("aGoat", 0, null);
openOrCreateDatabase.execSQL("CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR)");
// ...
sb.append("INSERT INTO users (username, password) VALUES('");
sb.append(username2.getText().toString());
sb.append(",");
sb.append(password2.getText().toString());
sb.append("')");
```

4. `users*.tmp` —— 全局可读临时凭据泄漏 `InsecureStorageTempActivity.java:52-70`：

```
File userinfo = File.createTempFile("users", "tmp", new File(InsecureStorageTempActivity.this.getApplicationInfo().dataDir));
userinfo.setReadable(true);
userinfo.setWritable(true);
FileWriter fw = new FileWriter(userinfo);
fw.write("username is " + username2.getText().toString() + "\n");
fw.write("password is " + password2.getText().toString() + "\n");
```

5. `score.xml` —— 仅与完整性相关的游戏状态 `InsecureStorageSharedPrefs1Activity.java:91-95`：

```
SharedPreferences sharedPreference = getSharedPreferences("score", 0);
SharedPreferences.Editor editor1 = sharedPreference.edit();
editor1.putInt("score", Score);
editor1.putInt("level", Level);
editor1.commit();
```

利用证据

验证确认了清单标记，并构建了引用源代码的利用链：

静态清单确认：

```
<application
    android:allowBackup="true"
    android:debuggable="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:networkSecurityConfig="@xml/network_security_config"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportRtl="true"
    android:theme="@style/AppTheme">
```

不存在 `android:fullBackupContent` 属性，且对 `res/xml/` 的 regex 搜索未返回任何备份排除规则。

adb backup/restore 利用链已验证：

```
adb backup -f agoat.ab owasp.sat.agoat
# Offline tampering of extracted archive
adb restore agoat.ab
```

还原后的静态跟踪确认 `AccessControlIssue1Activity.java:72-88` 在读取 `pinDetails.xml` 时未进行完整性校验。`isPinCorrect()` 方法将用户提供的 PIN 与存储的 MD5 摘要进行比对；注入 `5f4dcc3b5aa765d61d8327deb882cf99`，该值为 "password" 的 MD5，即可使用 "password" 通过认证。

run-as 读写利用链已验证：

```
adb shell run-as owasp.sat.agoat sh -c 'cat /data/data/owasp.sat.agoat/shared_prefs/users.xml'
adb shell run-as owasp.sat.agoat sh -c 'cat /data/data/owasp.sat.agoat/databases/aGoat'
```

用于注入攻击者数据的写入命令同样得到了验证：

```
adb shell "run-as owasp.sat.agoat sh -c 'echo \"<map><boolean name=\\\"pinSet\\\" value=\\\"true\\\"/><string
name=\\\"pin\\\">5f4dcc3b5aa765d61d8327deb882cf99</string></map>\" > /data/data/owasp.sat.agoat/shared_prefs/pinDetails.xml'"
```

凭据登记：验证过程初始化了一组由攻击者控制的 PIN/哈希对：

- `attacker_pin` → `5f4dcc3b5aa765d61d8327deb882cf99`，针对 `pinDetails.xml`。

建议

立即修复

移除 `AndroidManifest.xml` 中的 `android:debuggable="true"`，并将 `android:allowBackup` 设为 `"false"`：

```
<application
    android:allowBackup="false"
    android:debuggable="false"
    ... >
```

永久修复

确保 release build 永远不会携带 `debuggable="true"`。添加构建时 lint 检查或 CI/CD 门禁，当 release variant 中任一标志被启用时构建失败。如果需要备份功能以方便用户，请采用 Android 加密备份机制（使用客户端密钥的 AutoBackup），替代传统的完整备份模式。

验证

1. 构建 release APK 并执行：

```
aapt dump xmltree app-release.apk AndroidManifest.xml | grep -E 'debuggable|allowBackup'
```

确认这两个属性均不存在，或均已显式设为 `false`。

- 2. 尝试执行 `adb backup -f test.ab owasp.sat.agoat`，确认命令失败或生成空归档。
- 3. 尝试执行 `adb shell run-as owasp.sat.agoat`，确认命令被拒绝。

应用程序源代码中的硬编码认证凭据

多个硬编码凭证——包括用户名、密码和认证密钥——直接嵌入应用源码，并在运行时通过 `exportedBroadcastReceiver` 暴露。

硬编码凭据会破坏依赖保密机制进行身份验证或授权的任何系统的安全性。由于 `ShowDataReceiver` 被导出，恶意第三方应用可以在运行时无需任何用户交互即泄露这些密钥。如果这些凭据在后端环境中被重复使用，将为未授权访问提供直接途径。

技术详情

发现 #1

根因

`ShowDataReceiver.java` 与 `HardCodeActivity.java` 中包含认证凭据和敏感信息的硬编码字符串，并被编译进 APK。`ShowDataReceiver` 被声明为 `exported`，导致设备上的任意应用均可触发 Toast，从而在运行时泄露这些值。

漏洞代码

`owasp/sat/agoat/ShowDataReceiver.java`：

```
// Hardcoded credentials displayed in a Toast
"Username is CrazyUser, Password is CrazyPassword and Key is 123myKey456"
```

`owasp/sat/agoat/HardCodeActivity.java`：

```
// Promo code hardcoded in source
NEW2019
```

`res/values/strings.xml`：

```
<string name="firebase_url">https://androgoat-42597.firebaseio.com</string>
```

利用证据

对 APK 进行静态反编译即可确认具体数值，无需动态执行：

敏感信息	数值	源码位置
用户名	CrazyUser	ShowDataReceiver.java
密码	CrazyPassword	ShowDataReceiver.java
密钥	123myKey456	ShowDataReceiver.java
促销码	NEW2019	HardCodeActivity.java
Firebase URL	https://androgoat-42597.firebaseio.com	res/values/strings.xml

这些数值可通过 APK 逆向工程（JADX/apktool）轻松提取，也可通过向 `exported` 的 `ShowDataReceiver` 发送 `broadcastIntent` 获取，后者会将凭据以 Toast 消息形式展示。

验证证据

根因

APK 将明文凭据字符串直接编译进 DEX 字节码。`ShowDataReceiver` 在 `AndroidManifest.xml` 中被声明为 `android:exported="true"`，且未设置任何权限要求，从而形成了一个未受保护的入口点，任何共存应用均可访问。

漏洞代码

从 `/workspace/my_source/sources/owasp/sat/agoat/ShowDataReceiver.java` 反编译的源码：

```
public final class ShowDataReceiver extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
        Intrinsics.checkNotNull(context, "context");
        Intrinsics.checkNotNull(intent, "intent");
        Toast.makeText(context, "Username is CrazyUser, Password is CrazyPassword and Key is 123myKey456", 1).show();
    }
}
```

`HardCodeActivity.java` 将促销码存储为编译后的字面量，仅用于本地比较：

```
final Ref.ObjectRef promoCode = new Ref.ObjectRef();
promoCode.element = "NEW2019";
// ...
if (promoCodeValue2.getText().toString().equals((String) promoCode.element)) {
```

Manifest声明证实该receiver为exported且未受保护：

```
<receiver android:enabled="true" android:exported="true" android:name="owasp.sat.agoat.ShowDataReceiver"/>
```

利用证据

通过ADB发送broadcastIntent，已确认动态利用可行：

```
adb shell am broadcast -n owasp.sat.agoat/.ShowDataReceiver
```

工具输出：

```
Broadcasting: Intent { flg=0x4000000 cmp=owasp.sat.agoat/.ShowDataReceiver }
Broadcast completed: result=0
```

广播发送成功（`result=0`），证明任何第三方应用均可在无需用户交互或权限的情况下触发泄露凭据的Toast。

通过全代码库的 `ripgrep` 搜索确认，这些敏感信息仅存在于已报告的位置，且不存在后端Sink：

敏感信息	所在文件	非 Toast / 非 UI 用途
<code>CrazyUser</code>	<code>ShowDataReceiver.java:17</code>	无
<code>CrazyPassword</code>	<code>ShowDataReceiver.java:17</code>	无
<code>123myKey456</code>	<code>ShowDataReceiver.java:17</code>	无
<code>NEW2019</code>	<code>HardCodeActivity.java:49</code>	无（仅用于本地 <code>String.equals</code> ）
<code>androgoat-42597</code>	<code>res/values/strings.xml:130</code>	无（从未在代码中加载）

[外部探测] 尽管应用代码未使用该URL（未导入Firebase SDK，也未引用 `R.string.firebase_url`），但该硬编码Firebase URL `https://androgoat-42597.firebaseio.com` 响应了未经认证的HTTP请求，返回 `200 OK` 及响应体 `{"password":"androgoat","user":"androgoat"}`，证实该端点可被公开读取。

建议

立即修复

- 立即轮换所有硬编码凭证，将嵌入APK中的任何值均视为公开。
- 若导出的 `ShowDataReceiver` 并非应用合法功能所必需，请将其移除或禁用。

永久修复

- 使用设备证明向安全远程配置服务器完成身份验证后加载密钥，或在运行时从Android Keystore/Tink库中获取。
- 切勿将密码、API密钥或认证令牌编译到应用源代码或资源中。

验证

1. 修复后重新反编译APK。
2. 在源码树中搜索已知明文字符串（ `CrazyUser` 、 `CrazyPassword` 、 `123myKey456` 、 `NEW2019` ）。
3. 确认在Java源码、Kotlin源码、 `strings.xml` 、 `assets/` 和 `res/raw/` 中均无匹配。
4. 确认 `ShowDataReceiver` 在 `AndroidManifest.xml` 中不再被导出。

中 密码学漏洞：弱哈希算法

客户端使用了存在已知加密弱点的弱哈希函数。

哈希函数用于对数据进行编码，并返回固定长度的哈希值，其目的是最大程度降低碰撞概率（两个不同输入产生完全相同哈希值的可能性极低）。这类函数可用于验证消息或文件的完整性、识别文件或数据，以及校验密钥或密码。

常见哈希算法：

- `MD5` ：Message Digest Algorithm（MD5）于1991年发布，作为MD4的替代方案。2005年3月，研究人员展示了两份具有相同MD5哈希值的不同文件。一年后，一种可在数分钟内找到碰撞的算法被公开。对于哈希算法而言，找到两个不同输入产生相同哈希值应在计算上不可行，而MD5灾难性地未能满足这一要求。2008年，软件工程研究所（Software Engineering Institute）认定MD5“在密码学上已被破解，不再适合继续使用”。
- `SHA1` ：Secure Hash Algorithm 1（SHA1）由NSA设计。与MD5一样，SHA1易受碰撞攻击。2005年时，SHA1已被认为不安全。2011年，美国国家标准与技术研究院（NIST）正式将其弃用。
- `SHA2` ：SHA-2是由NSA设计的一套哈希算法，与SHA1存在显著差异，被认为具有高安全性。
- `SHA-256` ：SHA-256是SHA-2系列的成员。后缀256代表256位，即所生成哈希值的长度。SHA-256是安全且可信的行业标准，发生碰撞的概率几乎为零（ 2^{256} ）。
- `SHA3` ：SHA3是Secure Hash Algorithm标准家族的最新成员，由NIST于2015年发布。尽管属于同一标准系列，SHA3在内部结构上与SHA1和SHA2那种类MD5的架构截然不同，同样被认为具有高安全性。
- `Argon2` ：Argon2是2015年Password Hashing Competition的冠军，目前被认为是最安全的密码哈希算法之一。它已被多项安全与密码学标准采纳，包括Internet Engineering Task Force（IETF）、Password Hashing Competition（PHC）和The Open Worldwide Application Security Project（OWASP）。
- `Bcrypt` ：Bcrypt于1999年推出，是一种慢速自适应算法，旨在高度抵抗暴力破解。其自适应性体现在允许用户控制计算成本，这意味着它可以随硬件演进而调整。Bcrypt也被众多安全标准采纳，包括Open Web Application Security Project（OWASP）和Payment Card Industry Data Security Standard（PCIDSS）。
- `HMAC` ：基于哈希的消息认证码（Hash-based Message Authentication Code，HMAC）并非哈希算法，而是一种密码学认证技术，它将哈希算法与密钥结合使用。HMAC的安全性取决于底层哈希算法，因此不应搭配使用已弃用或弱的哈希算法。

`MD5` 和 `SHA-1` 不安全，即使配合HMAC也不得使用。

技术详情

发现 #1

该应用使用了MD5。MD2、MD4和MD5均为哈希函数，且已被碰撞攻击破解。调用栈：

```
at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
at owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick
at android.view.View.performClick
at android.view.View.performClickInternal
at android.view.View.-$$Nest$mperformClickInternal
at android.view.View$PerformClick.run
at android.os.Handler.handleCallback
at android.os.Handler.dispatchMessageImpl
at android.os.Handler.dispatchMessage
at android.os.Looper.loopOnce
at android.os.Looper.loop
at android.app.ActivityThread.main
at java.lang.reflect.Method.invoke
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run
at com.android.internal.os.ZygoteInit.main
```

发现 #2

该应用使用了MD5。MD2、MD4和MD5均为已被碰撞攻击破解的哈希函数。调用栈：

```
at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
at owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick
at owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView
```

```
at owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick
at owasp.sat.agoat.AccessControlIssue1Activity.onCreate
```

at `java.security.MessageDigest.getInstance`

无法提取源代码。

at `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```
public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v3, MessageDigest.getInstance(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect`

```
public final boolean isPinCorrect(String p4)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p4, pinValue);
    return this.getSharedPreferences(pinDetails, 0).getString(pin, 0).equals(this.hashPIN(p4));
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick`

```
public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165305));
    android.widget.Toast v1_10 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v0_2, pinValue);
    if (!v1_10.isPinCorrect(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Incorrect PIN entered), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN Verified), 1).show();
        this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.AccessControl1ViewActivity));
    }
    return;
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView`

```
public final void verifyPINView()
{
    this.setContentView(2131296285);
    ((android.widget.Button) this.findViewById(2131165371)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1(this)));
    return;
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick`

```
public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165335));
    android.widget.Toast v1_9 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v0_2, pinValue);
```

```

        if (!v1_9.createPIN(v0_2.getText().toString())) {
            android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN not created), 1).show();
        } else {
            android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN created. Please Login), 1).show();
            this.this$0.verifyPINView();
        }
        return;
    }
}

```

at `owasp.sat.agoat.AccessControlIssue1Activity.onCreate`

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    if (!this.verifyIfPinSet()) {
        this.setContentView(2131296284);
        ((android.widget.Button) this.findViewById(2131165334)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1(this)));
    } else {
        this.verifyPINView();
    }
    return;
}

```

发现 #3

该应用使用了MD5。MD2、MD4和MD5均为哈希函数，且已被碰撞攻击破解。调用栈：

```

at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
at owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick
at owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView
at owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick

```

at `java.security.MessageDigest.getInstance`

Unable to extract source code.

at `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```

public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v3, MessageDigest.getInstance...t(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}

```

at `owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect`

```

public final boolean isPinCorrect(String p4)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p4, pinValue);
    return this.getSharedPreferences(pinDetails, 0).getString(pin, 0).equals(this.hashPIN(p4));
}

```

at `owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick`

```

public final void onClick(android.view.View p5)

```

```

{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165305));
    android.widget.Toast v1_10 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v0_2, pinValue);
    if (!v1_10.isPinCorrect(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Incorrect PIN entered), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN Verified), 1).show();
        this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.AccessControl1ViewActivity));
    }
    return;
}

```

at `owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView`

```

public final void verifyPINView()
{
    this.setContentView(2131296285);
    ((android.widget.Button) this.findViewById(2131165371)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1(this)));
    return;
}

```

at `owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick`

```

public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165335));
    android.widget.Toast v1_9 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v0_2, pinValue);
    if (!v1_9.createPIN(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN not created), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN created. Please Login), 1).show();
        this.this$0.verifyPINView();
    }
    return;
}

```

发现 #4

该应用使用了 MD5。MD2、MD4 和 MD5 均为哈希函数，且已被碰撞攻击破解。

调用栈：

```

at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
at owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick
at owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView

```

在 `java.security.MessageDigest.getInstance`

无法提取源代码。

在 `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```

public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNullParameter(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v3, MessageDigest.getInstance...t(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)

```

```
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}
```

在 `owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect`

```
public final boolean isPinCorrect(String p4)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p4, pinValue);
    return this.getSharedPreferences(pinDetails, 0).getString(pin, 0).equals(this.hashPIN(p4));
}
```

在 `owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick`

```
public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165305));
    android.widget.Toast v1_10 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v0_2, pinValue);
    if (!v1_10.isPinCorrect(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Incorrect PIN entered), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN Verified), 1).show();
        this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.AccessControl1ViewActivity));
    }
    return;
}
```

在 `owasp.sat.agoat.AccessControlIssue1Activity.verifyPINView`

```
public final void verifyPINView()
{
    this.setContentView(2131296285);
    ((android.widget.Button) this.findViewById(2131165371)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1(this)));
    return;
}
```

发现 #5

该应用使用了 MD5。MD2、MD4 和 MD5 是被碰撞攻击破解的哈希函数。调用栈：

```
at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
at owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick
```

at `java.security.MessageDigest.getInstance`

Unable to extract source code.

at `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```
public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v3, MessageDigest.getInstance...t(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)
```

```
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect`

```
public final boolean isPinCorrect(String p4)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p4, pinValue);
    return this.getSharedPreferences(pinDetails, 0).getString(pin, 0).equals(this.hashPIN(p4));
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity$verifyPINView$1.onClick`

```
public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165305));
    android.widget.Toast v1_10 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v0_2, pinValue);
    if (!v1_10.isPinCorrect(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Incorrect PIN entered), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN Verified), 1).show();
        this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.AccessControl1ViewActivity));
    }
    return;
}
```

发现 #6

该应用使用了MD5。MD2、MD4和MD5均为哈希函数，且已被碰撞攻击破解。调用栈：

```
at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
at owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect
```

at `java.security.MessageDigest.getInstance`

无法提取源代码。

at `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```
public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v3, MessageDigest.getInstance...t(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}
```

at `owasp.sat.agoat.AccessControlIssue1Activity.isPinCorrect`

```
public final boolean isPinCorrect(String p4)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p4, pinValue);
    return this.getSharedPreferences(pinDetails, 0).getString(pin, 0).equals(this.hashPIN(p4));
}
```

发现 #7

该应用使用了 MD5。MD2、MD4 和 MD5 均为哈希函数，且已被碰撞攻击破解。调用栈：

```
at java.security.MessageDigest.getInstance
at owasp.sat.agoat.AccessControlIssue1Activity.hashPIN
```

at `java.security.MessageDigest.getInstance`

无法提取源代码。

at `owasp.sat.agoat.AccessControlIssue1Activity.hashPIN`

```
public final String hashPIN(String p13)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p13, pinValue);
    String v0_6 = java.security.MessageDigest.getInstance(MD5);
    byte[] v1_1 = p13.getBytes(kotlin.text.Charsets.UTF_8);
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_1, (this as java.lang.String).getBytes(charset));
    byte[] v3 = v0_6.digest(v1_1);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v3, MessageDigest.getInstance(pinValue.toByteArray()));
    return kotlin.collections.ArraysKt.joinToString$default(v3, ((CharSequence) ), 0, 0, 0, 0, ((kotlin.jvm.functions.Function1)
owasp.sat.agoat.AccessControlIssue1Activity$hashPIN$md$1.INSTANCE), 30, 0);
}
```

建议

为降低使用弱哈希算法带来的风险，建议采取以下措施：

- 避免使用 MD5 和 SHA-1，这些算法不安全，容易产生碰撞，不应在任何场景下使用，包括 HMAC。
- 如需进行哈希计算和签名验证，建议选用 SHA-256 或 SHA-3。它们具备高安全性，且是广泛认可的行业标准。
- 如需对敏感数据（PII、PHI 等）进行哈希处理，建议考虑 Argon2 或 Bcrypt。它们专为抵御暴力破解攻击而设计，并受到安全标准的推荐。
- 使用 HMAC 进行消息认证时，请确保与 SHA-256 或 SHA-3 等安全哈希算法配合使用，以保障安全性。

参考资料

- [Deprecating MD5 and SHA-1 signature hashes in \(D\)TLS 1.2](#)
- [Deprecation of SHA-1 \(NIST\)](#)

 明文 HTTP 请求

CVSS v3: 9.1

应用程序正在通过未加密通道访问数据，破坏了传输中数据的机密性和完整性。

明文 HTTP 流量是指移动应用与其后端服务器之间传输的数据未经任何加密，内容容易被读取且易被恶意攻击者截获。这种缺乏加密的情况会对移动应用构成重大安全风险。当敏感信息（例如登录凭据、个人数据或财务详情）通过明文 HTTP 连接发送时，将面临窃听和中间人攻击的威胁。

移动应用中的明文 HTTP 流量所带来的主要风险之一是可能导致用户数据泄露。攻击者可以截获并提取有价值的信息，进而导致身份盗用、账户未经授权访问以及个人数据滥用。此外，攻击者还可能利用这些漏洞篡改应用内容，或向通信信道注入恶意代码。

此外，明文 HTTP 流量还会危及用户隐私。移动应用通常会出于分析或定向广告目的收集并传输用户数据。若缺乏加密，这些数据很容易被第三方获取，从而破坏用户机密性，并可能导致隐私违规。

未能实施安全通信协议的移动应用还面临假冒攻击的风险。攻击者可以创建伪造的 Wi-Fi 网络或使用其他技术手段截获流量，并伪装成合法的后端服务器。用户在不知情的情况下将数据发送给攻击者，攻击者随后可能将这些数据用于恶意目的。

DART

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

void main() => runApp(LoginApp());

class LoginApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
```

```

        return MaterialApp(
          title: 'Login App',
          home: LoginPage(),
        );
      }
    }

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final TextEditingController _usernameController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
  String _loginStatus = '';

  void _performLogin() async {
    final String username = _usernameController.text.trim();
    final String password = _passwordController.text.trim();

    // 模拟登录 HTTP 请求。
    final String apiUrl = 'http://example.com/login';
    final Map<String, dynamic> requestBody = {
      'username': username,
      'password': password,
    };

    final http.Response response = await http.post(
      Uri.parse(apiUrl),
      headers: {'Content-Type': 'application/json'},
      body: json.encode(requestBody),
    );

    if (response.statusCode == 200) {
      // 登录成功
      setState(() {
        _loginStatus = 'Login successful!';
      });
    } else {
      // 登录失败
      setState(() {
        _loginStatus = 'Login failed. Please try again.';
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Login Page'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: [
            TextFormField(
              controller: _usernameController,
              decoration: InputDecoration(labelText: 'Username'),
            ),
            TextFormField(
              controller: _passwordController,
              obscureText: true,
              decoration: InputDecoration(labelText: 'Password'),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: _performLogin,
              child: Text('Login'),
            ),
            SizedBox(height: 10),
            Text(_loginStatus, textAlign: TextAlign.center),
          ],
        ),
      ),
    );
  }
}

```

```

    }
}

```

SWIFT

```

import UIKit

class LoginViewController: UIViewController {
    @IBOutlet weak var usernameTextField: UITextField!
    @IBOutlet weak var passwordTextField: UITextField!
    @IBOutlet weak var loginStatusLabel: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func loginButtonTapped(_ sender: UIButton) {
        guard let username = usernameTextField.text?.trimmingCharacters(in: .whitespacesAndNewlines),
              let password = passwordTextField.text?.trimmingCharacters(in: .whitespacesAndNewlines) else {
            return
        }

        // 模拟登录 HTTP 请求。
        let apiUrl = "http://example.com/login"
        let requestBody: [String: Any] = [
            "username": username,
            "password": password
        ]

        var request = URLRequest(url: URL(string: apiUrl)!)
        request.httpMethod = "POST"
        request.setValue("application/json", forHTTPHeaderField: "Content-Type")

        do {
            request.httpBody = try JSONSerialization.data(withJSONObject: requestBody, options: [])
        } catch {
            print("Error creating JSON data: \(error)")
            return
        }

        let task = URLSession.shared.dataTask(with: request) { data, response, error in
            if let error = error {
                print("Error: \(error)")
                DispatchQueue.main.async {
                    self.loginStatusLabel.text = "Login failed. Please try again."
                }
                return
            }

            guard let data = data,
                  let httpResponse = response as? HTTPURLResponse,
                  httpResponse.statusCode == 200 else {
                DispatchQueue.main.async {
                    self.loginStatusLabel.text = "Login failed. Please try again."
                }
                return
            }

            DispatchQueue.main.async {
                self.loginStatusLabel.text = "Login successful!"
            }
        }

        task.resume()
    }
}

```

KOTLIN

```

import android.os.Bundle
import android.util.Log
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_login.*
import org.json.JSONObject
import java.io.OutputStreamWriter
import java.net.HttpURLConnection

```

```
import java.net.URL

class LoginActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.trim().toString()
            val password = passwordEditText.text.trim().toString()

            // 模拟登录 HTTP 请求。
            val apiUrl = "https://example.com/login"
            val requestBody = JSONObject().apply {
                put("username", username)
                put("password", password)
            }.toString()

            Thread {
                performLogin(apiUrl, requestBody)
            }.start()
        }
    }

    private fun performLogin(apiUrl: String, requestBody: String) {
        try {
            val url = URL(apiUrl)
            val connection = url.openConnection() as HttpURLConnection
            connection.requestMethod = "POST"
            connection.setRequestProperty("Content-Type", "application/json")
            connection.doOutput = true

            val outputStreamWriter = OutputStreamWriter(connection.outputStream)
            outputStreamWriter.write(requestBody)
            outputStreamWriter.flush()

            val responseCode = connection.responseCode
            if (responseCode == HttpURLConnection.HTTP_OK) {
                runOnUiThread {
                    loginStatusLabel.text = "Login successful!"
                }
            } else {
                runOnUiThread {
                    loginStatusLabel.text = "Login failed. Please try again."
                }
            }
        } catch (e: Exception) {
            Log.e("LoginActivity", "Error: ${e.message}")
            runOnUiThread {
                loginStatusLabel.text = "Login failed. Please try again."
            }
        }
    }
}
```

技术详情

发现 #1

使用了未加密的明文 HTTP URL：<http://demo.testfire.net/>

请求：

```
GET / HTTP/1.1
Host: demo.testfire.net
Connection: Keep-Alive
Accept-Encoding: gzip
User-Agent: okhttp/3.8.0
```

响应：

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=9BDCDE139DFF2F56BD4E46A2E78C713E; Path=/; HttpOnly
Content-Type: text/html; charset=ISO-8859-1
```

```
Transfer-Encoding: chunked
Date: Tue, 19 May 2026 16:52:06 GMT
```

建议

为降低移动应用明文 HTTP 流量带来的风险，必须落实安全通信协议与最佳实践。以下是提升移动应用通信安全性的详细建议：

- 采用 HTTPS 加密：最基本的一步是为移动应用与后端服务器之间的所有通信启用 HTTPS（HTTP Secure）。HTTPS 会在传输过程中加密数据，确保敏感信息的机密性与安全性。请从可信的证书颁发机构（CA）获取 SSL/TLS 证书，并强制所有 API 端点与数据传输均通过 HTTPS 连接。
- 证书固定：实施证书固定以强化 HTTPS 连接的安全性。这需要在移动应用内硬编码服务器的 SSL 证书或其公钥。这样可确保应用仅与目标服务器通信，防止攻击者利用自有证书发起中间人攻击。
- 启用 HSTS（HTTP Strict Transport Security）：利用 HSTS 响应头指示浏览器始终使用 HTTPS 连接，即使用户误将 "http://" 输入到 URL 中。这能够防范潜在的降级攻击，确保所有通信均经过加密。
- 使用最新 TLS 版本：确保移动应用使用最新版 TLS（Transport Layer Security）协议，以利用最完善的安全特性与加密算法。
- 实施证书检查：配置应用以检查 SSL 证书是否已吊销或过期，避免与可能已遭入侵的服务器进行交互。
- 敏感数据加密：对存储在设备上或传输至服务器的敏感数据进行加密。使用强加密算法并安全管理加密密钥，从而为数据传输提供额外的安全保障。
- 确保所有 API 安全传输：评估应用使用的全部 API，确保其同样通过 HTTPS 传输数据，包括集成在应用内的第三方 API 与服务。

DART

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

void main() => runApp(LoginApp());

class LoginApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Login App',
      home: LoginPage(),
    );
  }
}

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final TextEditingController _usernameController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
  String _loginStatus = '';

  void _performLogin() async {
    final String username = _usernameController.text.trim();
    final String password = _passwordController.text.trim();

    // 模拟安全的 HTTPS 登录请求（请替换为您实际的安全 API 端点）。
    final String apiUrl = 'https://www.example.com/login';
    final Map<String, dynamic> requestBody = {
      'username': username,
      'password': password,
    };

    final http.Response response = await http.post(
      Uri.parse(apiUrl),
      headers: {'Content-Type': 'application/json'},
      body: json.encode(requestBody),
    );

    if (response.statusCode == 200) {
      // 登录成功
      setState(() {
        _loginStatus = 'Login successful!';
      });
    } else {
      // 登录失败
      setState(() {
```

```

        _loginStatus = 'Login failed. Please try again.';
    });
}
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Login Page'),
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: [
          TextFormField(
            controller: _usernameController,
            decoration: InputDecoration(labelText: 'Username'),
          ),
          TextFormField(
            controller: _passwordController,
            obscureText: true,
            decoration: InputDecoration(labelText: 'Password'),
          ),
          SizedBox(height: 20),
          ElevatedButton(
            onPressed: _performLogin,
            child: Text('Login'),
          ),
          SizedBox(height: 10),
          Text(_loginStatus, textAlign: TextAlign.center),
        ],
      ),
    ),
  );
}
}

```

SWIFT

```

import UIKit

class LoginViewController: UIViewController {
    @IBOutlet weak var usernameTextField: UITextField!
    @IBOutlet weak var passwordTextField: UITextField!
    @IBOutlet weak var loginStatusLabel: UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func loginButtonTapped(_ sender: UIButton) {
        guard let username = usernameTextField.text?.trimmingCharacters(in: .whitespacesAndNewlines),
              let password = passwordTextField.text?.trimmingCharacters(in: .whitespacesAndNewlines) else {
            return
        }

        // 模拟安全的 HTTPS 登录请求 (请替换为您实际的安全 API 端点)。
        let apiUrl = "https://www.example.com/login"
        let requestBody: [String: Any] = [
            "username": username,
            "password": password
        ]

        var request = URLRequest(url: URL(string: apiUrl)!)
        request.httpMethod = "POST"
        request.setValue("application/json", forHTTPHeaderField: "Content-Type")

        do {
            request.httpBody = try JSONSerialization.data(withJSONObject: requestBody, options: [])
        } catch {
            print("Error creating JSON data: \(error)")
            return
        }

        let task = URLSession.shared.dataTask(with: request) { data, response, error in

```

```

        if let error = error {
            print("Error: \(error)")
            DispatchQueue.main.async {
                self.loginStatusLabel.text = "Login failed. Please try again."
            }
            return
        }

        guard let data = data,
              let httpResponse = response as? HTTPURLResponse,
              httpResponse.statusCode == 200 else {
            DispatchQueue.main.async {
                self.loginStatusLabel.text = "Login failed. Please try again."
            }
            return
        }

        DispatchQueue.main.async {
            self.loginStatusLabel.text = "Login successful!"
        }
    }

    task.resume()
}
}

```

KOTLIN

```

import android.os.Bundle
import android.util.Log
import androidx.appcompat.app.AppCompatActivity
import kotlinx.android.synthetic.main.activity_login.*
import org.json.JSONObject
import java.io.OutputStreamWriter
import java.net.HttpURLConnection
import java.net.URL

class LoginActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.trim().toString()
            val password = passwordEditText.text.trim().toString()

            // 模拟安全的 HTTPS 登录请求 (请替换为您实际的安全 API 端点)。
            val apiUrl = "https://www.example.com/login"
            val requestBody = JSONObject().apply {
                put("username", username)
                put("password", password)
            }.toString()

            Thread {
                performLogin(apiUrl, requestBody)
            }.start()
        }
    }

    private fun performLogin(apiUrl: String, requestBody: String) {
        try {
            val url = URL(apiUrl)
            val connection = url.openConnection() as HttpURLConnection
            connection.requestMethod = "POST"
            connection.setRequestProperty("Content-Type", "application/json")
            connection.doOutput = true

            val outputStreamWriter = OutputStreamWriter(connection.outputStream)
            outputStreamWriter.write(requestBody)
            outputStreamWriter.flush()

            val responseCode = connection.responseCode
            if (responseCode == HttpURLConnection.HTTP_OK) {
                runOnUiThread {
                    loginStatusLabel.text = "Login successful!"
                }
            } else {
                runOnUiThread {

```

```
        loginStatusLabel.text = "Login failed. Please try again."
    }
}
} catch (e: Exception) {
    Log.e("LoginActivity", "Error: ${e.message}")
    runOnUiThread {
        loginStatusLabel.text = "Login failed. Please try again."
    }
}
}
```

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

攻击向量：

网络(N)

邻近(A)

本地(L)

物理(P)

攻击复杂度：

低(L)

高(H)

所需权限：

无(N)

低(L)

高(H)

用户交互：

无(N)

需要(R)

范围：

已更改(C)

未更改(U)

机密性：

高(H)

低(L)

无(N)

完整性：

高(H)

低(L)

无(N)

可用性：

高(H)

低(L)

无(N)

参考资料

- MissingEncryption of Sensitive Data(CWE-311)
- Testing for Weak SSL/TLS Ciphers, Insufficient Transport Layer Protection
- Top 10 2013-A6-Sensitive Data Exposure
- Insufficient Transport Layer
- Cleartext Transmission of Sensitive Information(CWE-319)

中 证书已过期

SSL/TLS 证书过期后将不再受浏览器和客户端信任，进而引发安全警告、服务中断及潜在安全漏洞，包括中间人攻击。

过期的SSL/TLS证书不再被现代浏览器和客户端信任，会导致信任链断裂、安全警告，并暴露于多种潜在安全风险，例如中间人（MitM）攻击。

证书过期后，浏览器将显示显著的安全警告，例如“您的连接不是私密连接”或“证书已过期”。这些警告会中断客户端与服务器之间的安全通信，直接影响用户信任与业务连续性。在电子商务及其他敏感环境中，这通常会导致用户流失、经济损失及声誉损害。

过期证书的安全风险远不止于用户层面。攻击者可利用有效证书缺失的漏洞实施MitM攻击，在传输过程中截获或篡改敏感数据。若无有效证书对服务器身份进行验证，加密通信将失去保护作用，数据将面临被截获、篡改或伪造的风险。

例如，证书过期的网站无法再确保用户数据的机密性与完整性。攻击者可能通过伪造连接冒充该网站，截获登录凭据或支付信息，进而导致未经授权访问或金融欺诈。

过期证书还意味着无法满足多项安全标准的合规要求，包括PCI DSS、HIPAA和GDPR。这些法规要求使用有效的SSL/TLS证书来保护敏感数据。未能遵守规定可能导致法律处罚，并进一步损害声誉。

技术详情

发现 #1

在 demo.testfire.net 上检测到 CERTIFICATE_EXPIRED

- 发现：剩余有效期不足 60 天（33）

建议

为应对 SSL/TLS 证书过期带来的风险，组织应采取以下主动防御策略：

- 自动化监控工具：部署可持续监控 SSL/TLS 证书状态的工具，并在证书即将过期时发送告警。这有助于防止因证书过期未及时发现而导致的服务中断。
- 应急续期流程：制定明确的应急流程，以便在证书意外过期时快速完成续期。这能最大限度缩短停机时间，并防范潜在的安全风险。
- 定期审计：定期对 SSL/TLS 证书资产进行审计，及时识别并续期过期证书。保持资产清单的最新状态有助于避免安全防护出现空档。

- 自动化证书配置：实施能够自动签发和续期 SSL/TLS 证书的系统，确保证书持续有效。Kubernetes 中的 cert-manager 等工具可简化这一流程。

在 Kubernetes 中实现自动化证书配置：

在 Kubernetes 环境中，可使用 cert-manager 实现 SSL/TLS 证书的自动化管理。该工具可与 Let's Encrypt 等证书颁发机构（CA）交互，自动完成证书签发与续期。

- 安装 cert-manager：

```
kubectl apply -f https://github.com/cert-manager/cert-manager/releases/latest/download/cert-manager.yaml
```

以下是一份用于 cert-manager 自动化证书配置的 YAML 配置示例：

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: example-cert
  namespace: default
spec:
  secretName: example-cert-secret
  issuerRef:
    name: letsencrypt-prod
    kind: ClusterIssuer
  commonName: example.com
  dnsNames:
    - example.com
    - www.example.com
  duration: 90d
  renewBefore: 30d
```

使用 Certbot 进行监控：

对于未使用 Kubernetes 的环境，可使用 Certbot 自动完成 SSL 证书的签发与续期。

```
# Automatically issue or renew SSL certificates using Certbot
domain="example.com"
email="admin@example.com"

# Run Certbot in standalone mode to obtain the certificate
certbot certonly --standalone -d $domain --email $email --agree-tos
```

使用 Cron Job 自动续期证书：

您可以使用 cron 任务将续期流程自动化：

```
# Edit your crontab with: crontab -e
0 0 * * * /usr/bin/certbot renew --quiet
```

该 cron 任务将在每天午夜执行，检查待续期的证书，从而在无人工干预的情况下确保证书持续有效。

参考资料

- [NIST Guidelines for Key Management](#)
- [SSL Labs Best Practices](#)

中 不安全的 HTTP Header 设置

应用头部设置缺失，或配置了不安全的头部值。

不安全的响应头设置：

- **Content Security Policy**：通过指定可信内容来源，降低跨站脚本攻击（XSS）风险。
- **Cookie**：通过设置 HttpOnly、Secure 等属性，防止 Cookie 数据被未授权访问，提升安全性。
- **Cross-Origin Resource Sharing**：控制跨域资源共享机制，防止恶意访问。
- **HTTP Public Key Pinning**：通过绑定站点有效公钥，防范中间人攻击。
- **Redirection**：确保重定向安全且仅导向可信目标，防止开放重定向漏洞。
- **Referrer Policy**：定义跨站导航时 Referrer 信息的传递范围，增强隐私保护。
- **Subresource Integrity**：通过校验加密哈希，验证第三方域加载的资源未被篡改。

- **X-Content-Type-Options**：阻止浏览器进行MIME嗅探，降低基于Content-Type的攻击风险。
- **X-Frame-Options**：控制页面是否允许嵌入frame，防止点击劫持攻击。
- **X-XSS-Protection**：启用浏览器内置的XSS过滤，拦截检测到的跨站脚本攻击。
- **Permissions-Policy**：控制浏览器可用的功能与API，限制不可信内容的权限以增强安全性。
- **Clear-Site-Data**：允许站点请求浏览器清除指定来源的存储数据（Cookie、本地存储、缓存等），缓解数据泄露与隐私风险。

技术详情

发现 #1

在 `demo.testfire.net` 上检测到 **MISSING_SECURITY_HEADERS**

- 发现：--

建议

为避免存在不安全的Header设置，请参考以下建议：

1. Content Security Policy (CSP)：

强制限制内容来源，缓解跨站脚本（XSS）攻击与未授权资源加载的风险。

```
Content-Security-Policy: default-src 'self'; script-src 'self' https://cdnjs.cloudflare.com;
```

2. Cookie Security Headers：

启用 Secure 和 HttpOnly 标志，防止 Cookie 被盗取或篡改，提升用户会话安全性。

```
Set-Cookie: sessionId=abc123; Secure; HttpOnly;
```

3. Cross-Origin Resource Sharing (CORS)：

合理配置 CORS 策略，限制跨源资源访问，以缓解跨站请求伪造（CSRF）及跨域数据泄露风险。

```
Access-Control-Allow-Origin: https://example.com
```

4. HTTP Public Key Pinning (HPKP)：

使用 HPKP 将公钥绑定至特定 Web 服务器，防范基于伪造证书的 Man-in-the-Middle（MitM）攻击。

```
Public-Key-Pins: pin-sha256="base64==" ; max-age=5184000; includeSubDomains;
```

5. Redirection Headers：

通过实施严格的重定向控制确保安全跳转，防止开放重定向漏洞，保护用户免受钓鱼攻击。

```
Location: https://example.com/secure-page
```

6. Referrer Policy：

设置适当的 Referrer 策略，控制 Referer Header 中传递的信息量，降低敏感数据暴露风险。

```
Referrer-Policy: strict-origin-when-cross-origin
```

7. Subresource Integrity (SRI)：

实施 SRI 以验证外部资源（如脚本和样式表）的完整性，防止未授权篡改及供应链攻击。

```
<script src="https://example.com/example.js" integrity="sha256-base64==" crossorigin="anonymous"></script>
```

8. X-Content-Type-Options：

启用 'nosniff' 指令，阻止浏览器对响应进行 MIME-sniffing，降低内容类型混淆攻击的风险。

```
X-Content-Type-Options: nosniff
```

9. X-Frame-Options :

设置 X-Frame-Options 以限制 Web 内容被嵌入框架，防止点击劫持攻击，确保页面完整性。

```
X-Frame-Options: DENY
```

10. X-XSS-Protection :

启用 XSS 防护机制，指示浏览器清理或拦截潜在恶意脚本，以缓解 XSS 攻击。

```
X-XSS-Protection: 1; mode=block
```

参考资料

- [OWASP HTTP Header Security](#)
- [Content Security Policy \(MDN\)](#)
- [X-Frame-Options \(MDN\)](#)

中 不安全的 HTTP 标头设置：HTTP 严格传输安全（HSTS）

不安全的 HTTP 头设置：HTTP Strict Transport Security（HSTS）。

HTTP Strict Transport Security（HSTS）是一种 Web 安全策略机制，Web 服务器通过该机制向合规的用户代理（如 Web 浏览器）声明：仅允许使用安全连接（HTTPS）与其交互。服务器通过名为 Strict-Transport-Security 的 HTTP 响应头字段向用户代理传达 HSTS 策略。该策略规定了用户代理必须仅通过安全方式访问服务器的持续时间。

当 Web 应用向用户代理发布 HSTS 策略时，合规的用户代理将执行以下操作：

- 自动将引用该 Web 应用的所有不安全（HTTP）链接转换为安全（HTTPS）链接。（例如，<http://example.com/some/page/> 会在访问服务器之前被修改为 <https://example.com/some/page/>）
- 如果连接的安全性无法得到保障（例如，服务器的 TLS 证书为自签名），用户代理将显示错误信息，并阻止用户访问该 Web 应用。

技术详情

发现 #1

在 `demo.testfire.net` 上检测到 HSTS

- 发现：未提供

建议

配置 Web 服务器，将所有 HTTP 请求重定向至 HTTPS。

以 Apache 为例，您需要修改 httpd.conf。更多配置方案请参阅外部参考章节。

```
# load module
LoadModule headers_module modules/mod_headers.so

# redirect all HTTP to HTTPS (optional)
<VirtualHost *:80>
    ServerAlias *
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [redirect=301]
</VirtualHost>

# HTTPS-Host-Configuration
<VirtualHost *:443>
    # Use HTTP Strict Transport Security to force client to use secure connections only
    Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"

    # Further Configuration goes here
    [...]
</VirtualHost>
```

参考资料

中 支持不安全的 TLS 密码套件

客户端支持包含已知加密弱点的密码套件组合。

传输层安全协议（TLS）始于客户端请求建立安全连接，并提交其支持的密码套件列表。

密码套件是认证、加密、消息认证码（MAC）以及密钥交换算法的组合。每种算法在协议协商过程中都发挥特定作用。例如，使用弱算法可能会严重影响整个信道的安全性。

经识别，该端点支持的密码套件与安全套接层／传输层安全（SSL/TLS）协议组合存在已知的密码学弱点。因此，不应依赖其实现有效的传输层安全。能够窃听该连接的攻击者可以影响或解密传输中的流量。

以下所有配置均已证实存在密码学弱点：

- 支持已弃用的 SSL 协议（v2 或 v3）
- 支持密钥长度小于 128 位的密码套件
- 支持采用 **CBC** 模式且 TLS 1.1 以下版本的密码套件，存在 BEAST 攻击漏洞
- 在 TLS 中使用压缩，存在 CRIME 攻击漏洞
- 支持采用 **RC4** 作为加密算法的密码套件
- 支持采用 **DES** 作为加密算法的密码套件
- 支持采用 **MD5** 作为签名算法的密码套件

技术详情

发现 #1

在 **demo.testfire.net** 上检测到 **cipherlist_OBSOLETE**

- **发现**：存在过时的密码套件

发现 #2

****cipherlist_OBSOLETE**** 在 **owasp.org** 上被检测到

- **发现**：提供了过时的密码套件

建议

推荐的 TLS 配置应遵循以下建议：

- 仅允许 **TLSv1** 及以上版本，优先使用 **TLS1.2**
- 如必须使用 **SSLv3**，建议启用 **TLS Fallback SCSV** 功能以防止协议降级攻击
- 禁用匿名 Diffie-Hellman（**ADH**）
- 禁用 **aNULL** 和 **eNULL** 密码套件
- 禁用 Export 密钥交换套件
- 移除 **RC4** 支持
- 移除 **DES** 支持
- 移除 **MD5** 支持
- 优先使用 **SHA256**，而非 **SHA1**
- 优先使用 **AES128**，而非 **AES256**，因为 256 位在安全方面优势有限，且对时序攻击的抵抗能力较弱
- 禁用客户端发起的重协商（Client-Initiated Renegotiation）
- 禁用 TLS 压缩
- 仅提供密钥长度大于 128 位的加密算法
- 提供具备完美前向保密（Perfect-Forward Secrecy）协议属性的密码套件（**DHE**、**ECDHE**）
- 使用自定义 Diffie-Hellman 组，以防范 Logjam 攻击
- 启用 HTTP Strict Transport Security 响应头字段
- 启用 **OSCP stapling**

以下是由 [Mozilla SSL Configuration Generator](#) 生成的 Nginx 安全 TLS 配置示例

```
<> NGINX
```

```
server {
    listen 443 ssl;

    # certs sent to the client in SERVER HELLO are concatenated in ssl_certificate
    ssl_certificate /path/to/signed_cert_plus_intermediates;
    ssl_certificate_key /path/to/private_key;
    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:50m;
    ssl_session_tickets off;

    # Diffie-Hellman parameter for DHE ciphersuites, recommended 2048 bits
    ssl_dhparam /path/to/dhparam.pem;

    # modern configuration. tweak to your needs.
    ssl_protocols TLSv1.1 TLSv1.2;
    ssl_ciphers 'ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK';
    ssl_prefer_server_ciphers on;

    # HSTS (ngx_http_headers_module is required) (15768000 seconds = 6 months)
    add_header Strict-Transport-Security max-age=15768000;

    # OCSP Stapling ---
    # fetch OCSP records from URL in ssl_certificate and cache them
    ssl_stapling on;
    ssl_stapling_verify on;

    ## verify chain of trust of OCSP response using Root CA and Intermediate certs
    ssl_trusted_certificate /path/to/root_CA_cert_plus_intermediates;

    resolver <IP DNS resolver>;

    ....
}
```

参考资料

- [Mozilla Secure TLS/SSL Configuration](#)
- [Transport Layer Protection Cheat Sheet \(OWASP\)](#)
- [SSL/TLS Deployment Best Practices \(SSLabs\)](#)

中 不支持 TLS_FALLBACK_SCSV

服务器不支持 TLS_FALLBACK_SCSV，容易受到协议降级攻击。

This vulnerability indicates that the server does not support **TLS_FALLBACK_SCSV** (Signaling Cipher Suite Value), which is a mechanism designed to prevent protocol downgrade attacks in SSL/TLS connections.

TLS_FALLBACK_SCSV is crucial because:

1. It allows clients to indicate when they're falling back to a lower protocol version due to connection failures.
2. It enables servers to detect and prevent malicious downgrade attempts.
3. It helps maintain the highest possible level of security in SSL/TLS connections.

When **TLS_FALLBACK_SCSV** is not supported, the server becomes vulnerable to:

1. Protocol Downgrade Attacks: An attacker can force the use of older, less secure protocol versions.
2. Man-in-the-Middle (MitM) Attacks: Downgrading to a weaker protocol can make it easier for attackers to intercept and decrypt communications.
3. Exploitation of Vulnerabilities in Older Protocols: Older SSL/TLS versions may have known vulnerabilities that can be exploited once a downgrade occurs.

Example Scenario:

An attacker intercepts the initial handshake between a client and server. The attacker manipulates the connection to fail, causing the client to retry with a lower protocol version. Without **TLS_FALLBACK_SCSV**, the server accepts this downgrade, potentially exposing the connection to vulnerabilities in the older protocol version. The absence of **TLS_FALLBACK_SCSV** support violates best practices for SSL/TLS security and can impact compliance with various security standards and regulations.

技术详情

发现 #1

在 `demo.testfire.net` 上检测到 **fallback_SCSV**

- 发现：不支持

建议

当 TLS_FALLBACK_SCSV（信号密码套件值）不受支持时，可能导致降级攻击。以下是缓解该风险的策略：

主动防御策略

- 禁用 SSL/TLS 版本回退**：配置服务器以禁用回退至旧版 SSL/TLS 版本。这可防止攻击者强制连接使用存在漏洞的协议。
- 设定最低 TLS 版本**：在客户端和服务器端均设置可接受的最低 TLS 版本（例如 TLS 1.2），以防止降级到存在漏洞的版本。
- 定期安全扫描**：定期扫描基础设施，识别并修复任何 TLS 配置弱点。
- 监控异常 TLS 协商模式**：部署监控以检测强制协议降级的尝试。

实施示例

1. Nginx 配置

若要在 Nginx 中禁用 SSL/TLS 版本回退并设定最低 TLS 版本：

```
server {
    listen 443 ssl;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_prefer_server_ciphers on;
    ssl_ciphers 'ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256';
    # Other SSL settings...
}
```

2. Apache 配置

对于 Apache，请使用以下配置以实现相同效果：

```
<VirtualHost *:443>
    SSLEngine on
    SSLProtocol -all +TLSv1.2 +TLSv1.3
    SSLCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256
    SSLHonorCipherOrder on
    # Other SSL settings...
</VirtualHost>
```

3. OpenSSL 测试命令

使用 OpenSSL 测试服务器的 TLS 配置：

```
openssl s_client -connect example.com:443 -tls1_2
```

该命令尝试建立 TLS 1.2 连接。若连接成功，则表明服务器已正确配置为使用现代 TLS 版本。

监控降级尝试

- 使用 fail2ban

您可以使用 `fail2ban` 监控日志中的潜在降级尝试：

1. 在 `/etc/fail2ban/filter.d/tls-downgrade.conf` 中创建自定义过滤器：

```
[Definition]
failregex = SSL routines:SSL23_GET_CLIENT_HELLO:unsupported protocol.*client=<HOST>
```

2. 在 `/etc/fail2ban/jail.local` 中添加 jail：

```
[tls-downgrade]
enabled = true
filter = tls-downgrade
logpath = /var/log/nginx/error.log
```

```
maxretry = 3
bantime = 3600
```

3. 重启 fail2ban：

```
sudo systemctl restart fail2ban
```

此配置将封禁使用不受支持的（可能是旧版）SSL/TLS 版本进行重复连接尝试的 IP 地址。

通过实施上述措施，即使 TLS_FALLBACK_SCSV 不受支持，您也能显著降低 TLS 降级攻击的风险。

参考资料

- [NIST Guidelines for TLS Implementations](#)
- [RFC 7507 – TLS Fallback Signaling Cipher Suite Value \(SCSV\) for Preventing Protocol Downgrade Attacks](#)
- [OWASP TLS Cheat Sheet](#)

中 Weak Cryptographic Key and Signature Algorithm in SSL/TLS Certificate

该 SSL/TLS 证书同时使用了弱加密密钥和弱签名算法，可能危及加密连接的安全性，并导致证书伪造或其他密码学攻击。

SSL/TLS 证书若使用弱加密密钥或弱签名算法，将带来重大安全风险，使系统易受密码学攻击。

关于弱加密密钥与弱签名算法的要点：

1. 弱加密密钥：

- RSA 密钥：长度低于 2048 位的 RSA 密钥被视为弱密钥。当前建议至少使用 2048 位，如需长期安全则应使用 3072 位或更高。
- DSA 密钥：与 RSA 类似，长度低于 2048 位的 DSA 密钥被视为弱密钥，应避免使用。
- 算力演进：随着算力不断提升，短密钥长度遭受攻击的风险日益增加。
- 行业标准：多项行业标准和合规要求对 SSL/TLS 证书规定了最小密钥长度。

2. 弱签名算法：

- 已淘汰算法：MD5 和 SHA1 在密码学上已被攻破，不应继续使用。
- 碰撞攻击：弱算法易受碰撞攻击威胁，攻击者可借此生成与合法证书签名相同的伪造证书。
- 行业标准：多项行业标准和合规要求禁止使用弱签名算法。
- 向后兼容：部分系统可能出于向后兼容目的仍在使用弱算法，但强烈不建议这种做法。

实际影响：使用含 1024 位 RSA 密钥或 SHA1 签名证书的网站，可能成为拥有大量计算资源的攻击者的目标。攻击者可以解密被拦截的流量，或伪造证书，进而实施中间人攻击并破坏通信安全。因此，遵循要求使用强密钥与强算法的行业标准，对于保障 SSL/TLS 安全至关重要。

技术详情

发现 #1

在 `demo.testfire.net` 上检测到 **WEAK_SIGNATURE_ALGORITHMS**

- **发现：**在 TLS1.2 中检测到弱签名算法：RSA+SHA1。

建议

为修复弱密钥漏洞：

1. 使用足够长度的密钥：

- RSA 和 DSA 的密钥长度至少应为 2048 位。
- 如需保障长期安全，建议使用 3072 位或更长的密钥。

2. 考虑使用椭圆曲线密码学（ECC）：

- ECC 以更短的密钥长度提供同等安全性，性能更优。
- 建议使用 P-256 或 P-384 等曲线，以获得强安全性。

3. 定期轮换密钥：

- 制定密钥定期轮换策略，尤其是在升级到更长密钥长度时。

4. 审计现有证书：

- 定期审计 SSL/TLS 证书，识别并替换存在弱密钥的证书。

5. 使用现代证书颁发机构（CA）：

- 选择强制要求强密钥并遵循行业最佳实践的 CA。

6. 实施强健的密钥生成实践：

- 生成密钥时，应使用密码学安全的随机数生成器。
- 建议采用硬件安全模块（HSM）进行密钥的生成与存储。

参考资料

- [NIST Recommendation for Key Management](#)
- [Baseline Requirements for TLS Server Certificates](#)
- [SSL Labs Best Practices](#)

中 滥用移动网络连接

应用强制使用移动网络连接。

下载管理器是一种系统服务，用于处理长时间运行的 HTTP 下载。应用程序不应强制仅使用移动网络连接。

技术详情

发现 #1

仅允许移动网络连接

污点从可达组件 `owasp.sat.agoat.DownloadInvoiceService$downloadFile$1` 追踪得出，该组件发现于 `owasp.sat.agoat.DownloadInvoiceService`。从常量 `1` 流向 Sink `android.app.DownloadManager$Request android.app.DownloadManager$Request.setAllowedNetworkTypes(int)`。污点追踪路径：

```
at android.app.DownloadManager$Request.setAllowedNetworkTypes()
at owasp.sat.agoat.DownloadInvoiceService$downloadFile$1.invoke()
at android.app.DownloadManager$Request.setNotificationVisibility()
at owasp.sat.agoat.DownloadInvoiceService$downloadFile$1.invoke()
```

方法 `owasp.sat.agoat.DownloadInvoiceService$downloadFile$1.invoke()`：

```
public final void invoke(org.jetbrains.anko.AnkoAsyncContext p10)
{
    kotlin.jvm.internal.Intrinsics.checkNotNullParameter(p10, $receiver);
    android.net.Uri v1 = android.net.Uri.parse(https://github.com/satishpatnayak/MyTest/blob/master/AndroGoatInvoice.txt);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1, Uri.parse(url1));
    android.app.DownloadManager$Request v2_2 = new android.app.DownloadManager$Request(v1);
    String v3 = v1.getLastPathSegment();
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v3, url1.lastPathSegment);
    int v5 = 0;
    v2_2.setAllowedNetworkTypes(3);
    v2_2.setTitle(((CharSequence) v3));
    v2_2.setDescription(((CharSequence) The File is downloading...));
    v2_2.allowScanningByMediaScanner();
    v2_2.setNotificationVisibility(1);
    v2_2.setDestinationInExternalPublicDir(android.os.Environment.DIRECTORY_DOWNLOADS, String.valueOf(System.currentTimeMillis()));
    kotlin.TypeCastException v6_8 = this.this$0.getSystemService(download);
    if (v6_8 == null) {
        throw new kotlin.TypeCastException(null cannot be cast to non-null type android.app.DownloadManager);
    } else {
        ((android.app.DownloadManager) v6_8).enqueue(v2_2);
        return;
    }
}
```

方法 `owasp.sat.agoat.DownloadInvoiceService$downloadFile$1.invoke()`：

```
public final void invoke(org.jetbrains.anko.AnkoAsyncContext p10)
{
    kotlin.jvm.internal.Intrinsics.checkNotNullParameter(p10, $receiver);
    android.net.Uri v1 = android.net.Uri.parse(https://github.com/satishpatnayak/MyTest/blob/master/AndroGoatInvoice.txt);
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v1, Uri.parse(url1));
```

```

        android.app.DownloadManager$Request v2_2 = new android.app.DownloadManager$Request(v1);
        String v3 = v1.getLastPathSegment();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v3, url.lastPathSegment);
        int v5 = 0;
        v2_2.setAllowedNetworkTypes(3);
        v2_2.setTitle(((CharSequence) v3));
        v2_2.setDescription(((CharSequence) "The File is downloading..."));
        v2_2.allowScanningByMediaScanner();
        v2_2.setNotificationVisibility(1);
        v2_2.setDestinationInExternalPublicDir(android.os.Environment.DIRECTORY_DOWNLOADS, String.valueOf(System.currentTimeMillis()));
        kotlin.TypeCastException v6_8 = this.this$0.getSystemService(download);
        if (v6_8 == null) {
            throw new kotlin.TypeCastException(null cannot be cast to non-null type android.app.DownloadManager);
        } else {
            ((android.app.DownloadManager) v6_8).enqueue(v2_2);
            return;
        }
    }
}

```

建议

应用应允许用户使用 Wi-Fi 或移动网络连接。常见的应用策略是仅在 Wi-Fi 网络可用时才获取大量数据。

中 XML 注入

XML 注入是指不可信用户输入在并入 XML 数据前未经正确验证或清理而引发的漏洞。攻击者可通过注入恶意 XML 内容利用该漏洞，进而导致未授权访问、数据篡改或拒绝服务攻击。

XML 注入是由于在将用户输入拼接到 XML 配置文件或数据流之前，未对其进行充分验证与清理而引发的一种漏洞。这种疏忽使攻击者得以通过注入恶意 XML 内容来利用应用。通过操纵 XML 结构或引入恶意实体，攻击者可能破坏应用逻辑、篡改数据或提取敏感信息。

DART

```

import 'dart:convert';
import 'dart:io';

import 'package:flutter/material.dart';
import 'package:path_provider/path_provider.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'XML 注入演示',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  final TextEditingController _usernameController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();

  @override
  void dispose() {
    _usernameController.dispose();
    _passwordController.dispose();
    super.dispose();
  }

  Future<void> saveConfigFile() async {

```

```

final directory = await getApplicationDocumentsDirectory();
final configFile = File('${directory.path}/config.xml');

// 创建 XML 数据
final xmlData = '''
<root>
  <username>${_usernameController.text}</username>
  <password>${_passwordController.text}</password>
</root>
''';

// 将 XML 保存到配置文件
await configFile.writeAsString(xmlData);
}

void login() {
  final username = _usernameController.text;
  final password = _passwordController.text;

  // 执行登录逻辑
  print('正在执行登录, 用户名: $username, 密码: $password');
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('XML 注入演示'),
    ),
    body: Padding(
      padding: EdgeInsets.all(16.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('输入用户名: '),
          TextField(
            controller: _usernameController,
          ),
          SizedBox(height: 16.0),
          Text('输入密码: '),
          TextField(
            controller: _passwordController,
            obscureText: true,
          ),
          SizedBox(height: 16.0),
          ElevatedButton(
            onPressed: () async {
              await saveConfigFile();
              login();
            },
            child: Text('登录'),
          ),
        ],
      ),
    ),
  );
}

```

SWIFT

```

import SwiftUI

struct ContentView: View {
  @State private var username = ""
  @State private var password = ""

  var body: some View {
    VStack {
      Text("输入用户名:")
      TextField("用户名", text: $username)
        .textFieldStyle(RoundedBorderTextFieldStyle())

      Text("输入密码:")
      SecureField("密码", text: $password)
        .textFieldStyle(RoundedBorderTextFieldStyle())

      Button(action: {

```

```

        saveConfigFile()
        login()
    }) {
        Text("登录")
    }
}
.padding()
}

func saveConfigFile() {
    // 创建 XML 数据
    let xmlData = '''
<root>
    <username>\(username)</username>
    <password>\(password)</password>
</root>
'''

    // 将 XML 保存到配置文件
    let configFilePath = "config.xml"
    do {
        try xmlData.write(toFile: configFilePath, atomically: true, encoding: .utf8)
    } catch {
        print("XML 写入文件失败：\(error)")
    }
}

func login() {
    // 执行登录逻辑
    print("正在使用用户名：\(username) 和密码：\(password) 执行登录")
}
}

@main
struct XMLInjectionApp: App {
    var body: some Scene {
        WindowGroup {
            ContentView()
        }
    }
}

```

KOTLIN

```

import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import java.io.File
import javax.xml.parsers.DocumentBuilderFactory
import javax.xml.transform.TransformerFactory
import javax.xml.transform.dom.DOMSource
import javax.xml.transform.stream.StreamResult

class MainActivity : AppCompatActivity() {

    private lateinit var usernameEditText: EditText
    private lateinit var passwordEditText: EditText
    private lateinit var loginButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        usernameEditText = findViewById(R.id.usernameEditText)
        passwordEditText = findViewById(R.id.passwordEditText)
        loginButton = findViewById(R.id.loginButton)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()

            saveConfigFile(username, password)
            login(username, password)
        }
    }
}

```

```

private fun saveConfigFile(username: String, password: String) {
    val configFilePath = "config.xml"

    try {
        val configFile = File(configFilePath)
        val xmlData = configFile.readText()

        val docBuilder = DocumentBuilderFactory.newInstance().newDocumentBuilder()
        val doc = docBuilder.parse(xmlData.byteInputStream())
        doc.documentElement.normalize()

        val root = doc.documentElement

        root.getElementsByTagName("username").item(0).textContent = username
        root.getElementsByTagName("password").item(0).textContent = password

        val transformer = TransformerFactory.newInstance().newTransformer()
        val result = StreamResult(configFile)
        val source = DOMSource(doc)
        transformer.transform(source, result)
    } catch (e: Exception) {
        Toast.makeText(this, "保存配置文件失败", Toast.LENGTH_SHORT).show()
    }
}

private fun login(username: String, password: String) {
    // 执行登录逻辑
    Toast.makeText(this, "正在使用用户名:$username 和密码:$password 执行登录", Toast.LENGTH_SHORT).show()
}
}

```

技术详情

发现 #1

类 `WebView` 中的方法 `loadData` 在位置 `0` 的参数可能存在 XML 注入漏洞。污点追踪自不可达组件：`owasp.sat.agoat.XSSActivity`。TaintElement 来源：类 `XSSActivity`、方法 `onCreate`、位置 `0`、签名 `(Landroid/os/Bundle;)V`。调用栈：

```

at android.webkit.WebView.loadData
at owasp.sat.agoat.XSSActivity.onCreate
at owasp.sat.agoat.XSSActivity.onCreate

```

在 `android.webkit.WebView.loadData`

Unable to extract source code.

在 `owasp.sat.agoat.XSSActivity.onCreate`

```

protected void onCreate(android.os.Bundle p6)
{
    super.onCreate(p6);
    this setContentView(2131296308);
    android.webkit.WebView v0_3 = ((android.webkit.WebView) this.findViewById(2131165373));
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v0_3, webView1);
    v0_3.setWebChromeClient(new android.webkit.WebChromeClient());
    android.webkit.WebSettings v1_1 = v0_3.getSettings();
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_1, webSettings1);
    v1_1.setJavaScriptEnabled(1);
    v0_3.loadData(<html>

<body>
<script>
function displayContent()
{
    var a=document.getElementById("name");
    document.write(a.value);

}
</script>
Name: <input type="text" id="name"/>
<br><br><input type="button" value="Display" onclick="displayContent()" style="background-color:black;color: white;
    border: 2px solid #000000"/>
</body>

</html>, text/html, UTF-8);

```

```
    return;  
}
```

在 `owasp.sat.agoat.XSSActivity.onCreate`

```
protected void onCreate(android.os.Bundle p6)  
{  
    super.onCreate(p6);  
    this.setContentView(2131296308);  
    android.webkit.WebView v0_3 = ((android.webkit.WebView) this.findViewById(2131165373));  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v0_3, webView1);  
    v0_3.setWebChromeClient(new android.webkit.WebChromeClient());  
    android.webkit.WebSettings v1_1 = v0_3.getSettings();  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_1, webSettings1);  
    v1_1.setJavaScriptEnabled(1);  
    v0_3.loadData(<html>  
  
<body>  
<script>  
function displayContent()  
{  
var a=document.getElementById("name");  
document.write(a.value);  
  
}  
</script>  
Name: <input type="text" id="name"/>  
</br></br><input type="button" value="Display" onClick="displayContent()" style="background-color:black;color: white;  
border: 2px solid #000000"/>  
</body>  
  
</html>, text/html, UTF-8);  
    return;  
}
```

建议

为缓解XML注入漏洞，建议采取以下措施：

- 实施恰当的输入验证与数据净化措施。
- 按照预期格式验证用户输入，并通过移除或转义可能被解析为XML标签或实体的字符来完成数据净化。
- 考虑使用能够安全处理数据绑定的XML解析库或API。

DART

```
import 'dart:convert';  
import 'dart:io';  
  
import 'package:flutter/material.dart';  
import 'package:path_provider/path_provider.dart';  
import 'package:xml/xml.dart';  
  
void main() {  
    runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
    @override  
    Widget build(BuildContext context) {  
        return MaterialApp(  
            title: 'XML Injection Demo',  
            theme: ThemeData(  
                primarySwatch: Colors.blue,  
            ),  
            home: MyHomePage(),  
        );  
    }  
}  
  
class MyHomePage extends StatefulWidget {  
    @override  
    _MyHomePageState createState() => _MyHomePageState();  
}  
  
class _MyHomePageState extends State<MyHomePage> {
```

```

final TextEditingController _usernameController = TextEditingController();
final TextEditingController _passwordController = TextEditingController();

@override
void dispose() {
  _usernameController.dispose();
  _passwordController.dispose();
  super.dispose();
}

Future<void> saveConfigFile() async {
  final directory = await getApplicationDocumentsDirectory();
  final configFile = File('${directory.path}/config.xml');

  // 创建 XML 数据
  final builder = XmlBuilder();
  builder.processing('xml', 'version="1.0"');
  builder.element('root', nest: () {
    builder.element('username', nest: _usernameController.text);
    builder.element('password', nest: _passwordController.text);
  });
  final xmlData = builder.build().toXmlString(pretty: true);

  // 将 XML 保存到配置文件
  await configFile.writeAsString(xmlData);
}

void login() {
  final username = _usernameController.text;
  final password = _passwordController.text;

  // 执行登录逻辑
  print('Performing login with username: $username and password: $password');
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('XML Injection Demo'),
    ),
    body: Padding(
      padding: EdgeInsets.all(16.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('Enter username:'),
          TextField(
            controller: _usernameController,
          ),
          SizedBox(height: 16.0),
          Text('Enter password:'),
          TextField(
            controller: _passwordController,
            obscureText: true,
          ),
          SizedBox(height: 16.0),
          ElevatedButton(
            onPressed: () async {
              await saveConfigFile();
              login();
            },
            child: Text('Login'),
          ),
        ],
      ),
    ),
  );
}

```

SWIFT

```

import Foundation

func main() {
  let configFilePath = "config.xml"

```

```

// 加载配置文件
guard let configFileURL = Bundle.main.url(forResource: "config", withExtension: "xml") else {
    print("Failed to load configuration file")
    return
}

guard let xmlData = try? Data(contentsOf: configFileURL) else {
    print("Failed to read configuration file")
    return
}

// 解析 XML
let xmlDoc: XMLDocument
do {
    xmlDoc = try XMLDocument(data: xmlData, options: .documentTidyXML)
} catch {
    print("Failed to parse XML: \(error)")
    return
}

guard let root = xmlDoc.rootElement() else {
    print("Invalid XML structure")
    return
}

// 从用户输入读取配置值
guard let username = getUserInput(prompt: "Enter username:"), let password = getUserInput(prompt: "Enter password:") else {
    print("Invalid input")
    return
}

// 更新 XML 中的配置值
if let usernameElement = root.elements(forName: "username").first {
    usernameElement.stringValue = username
}

if let passwordElement = root.elements(forName: "password").first {
    passwordElement.stringValue = password
}

// 将修改后的 XML 写回配置文件
guard let modifiedXmlData = xmlDoc.xmlData(options: .nodePrettyPrint) else {
    print("Failed to generate modified XML")
    return
}

do {
    try modifiedXmlData.write(to: configFileURL)
} catch {
    print("Failed to write modified XML to file: \(error)")
    return
}

// 使用配置值
login(username: username, password: password)
}

func getUserInput(prompt: String) -> String? {
    print(prompt, terminator: " ")
    return readLine()
}

func login(username: String, password: String) {
    // 执行登录逻辑
    print("Performing login with username: \(username) and password: \(password)")
}

// 启动程序
main()

```

KOTLIN

```

import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import org.w3c.dom.Document

```

```

import java.io.File
import javax.xml.parsers.DocumentBuilderFactory
import javax.xml.transform.TransformerFactory
import javax.xml.transform.dom.DOMSource
import javax.xml.transform.stream.StreamResult

class MainActivity : AppCompatActivity() {

    private lateinit var usernameEditText: EditText
    private lateinit var passwordEditText: EditText
    private lateinit var loginButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        usernameEditText = findViewById(R.id.usernameEditText)
        passwordEditText = findViewById(R.id.passwordEditText)
        loginButton = findViewById(R.id.loginButton)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()

            saveConfigFile(username, password)
            login(username, password)
        }
    }

    private fun saveConfigFile(username: String, password: String) {
        val configFilePath = "config.xml"

        try {
            val configFile = File(configFilePath)
            val xmlDoc = DocumentBuilderFactory.newInstance().newDocumentBuilder().newDocument()

            val rootElement = xmlDoc.createElement("root")
            xmlDoc.appendChild(rootElement)

            val usernameElement = xmlDoc.createElement("username")
            usernameElement.appendChild(xmlDoc.createTextNode(username))
            rootElement.appendChild(usernameElement)

            val passwordElement = xmlDoc.createElement("password")
            passwordElement.appendChild(xmlDoc.createTextNode(password))
            rootElement.appendChild(passwordElement)

            val transformer = TransformerFactory.newInstance().newTransformer()
            val result = StreamResult(configFile)
            val source = DOMSource(xmlDoc)
            transformer.transform(source, result)
        } catch (e: Exception) {
            Toast.makeText(this, "Failed to save configuration file", Toast.LENGTH_SHORT).show()
        }
    }

    private fun login(username: String, password: String) {
        // 执行登录逻辑
        Toast.makeText(this, "Performing login with username: $username and password: $password", Toast.LENGTH_SHORT).show()
    }
}

```

参考资料

- [Testing for XML Injection \(OWASP\)](#)

低 usesCleartextTraffic 属性已设置

CVSSv3: 5.4

应用设置了 usesCleartextTraffic 属性。

android:usesCleartextTraffic 属性用于指示应用是否使用明文网络流量，例如明文 HTTP。

目标 API 级别为 27 或更低的应用，其默认值为“true”。目标 API 级别为 28 或更高的应用，其默认值为“false”。

技术详情

This app targets API level 27 or less and no explicit value was set to the `usesCleartextTraffic` attribute, on these APIs the `usesCleartextTraffic` attribute default value is true. This exposes your users sensitive information and open up the application to injection attacks.

建议

显式将属性 `android:usesCleartextTraffic` 的值设置为 `false`，并定义 Android Network Security Config。

目标 API 级别 27 及以下的应用，默认值为 `true`。目标 API 级别 28 及以上的应用，默认值为 `false`。

```
<? XML

<application android:icon="@drawable/icon" android:usesCleartextTraffic="false">
```

CVSS向量详情

CVSS V3向量： CVSS:3.0/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N

攻击向量：

网络(N) 邻近(A) 本地(L) 物理(P)

攻击复杂度：

低(L) 高(H)

所需权限：

无(N) 低(L) 高(H)

用户交互：

无(N) 需要(R)

范围：

已更改(C) 未更改(U)

机密性：

高(H) 低(L) 无(N)

完整性：

高(H) 低(L) 无(N)

可用性：

高(H) 低(L) 无(N)

参考资料

- [Storage updates in Android 11](#)

低 使用已弃用的 TLS/SSL 协议版本

应用使用了已弃用的 SSL/TLS 协议，已知存在安全与隐私问题。

SSL/TLS 家族共有六个协议：SSLv2、SSLv3、TLSv1.0、TLSv1.1、TLSv1.2 和 TLSv1.3：

- SSLv2 不安全，不得使用。该协议版本存在 DROWN 攻击风险，可威胁 RSA 密钥及同名站点。
- 由于存在 SSLv3 POODLE 攻击，SSLv3 在配合 HTTP 使用时并不安全。配合其他协议使用时，该协议同样被视为存在弱点，且已被弃用，不得使用。
- TLSv1.0 和 TLSv1.1 是遗留协议，存在 BEAST 攻击风险。尽管现代客户端已实施部分缓解措施，但这些协议仍存在已知弱点，并自 2020 年 1 月起被 PCI DSS 及主流浏览器弃用。
- TLSv1.2 和 v1.3 目前未发现安全问题。

技术详情

TLS_1_0_DEPRECATED detected on `demo.testfire.net`

- Finding:** TLS 1.0 offered: offered(deprecated)

建议

TLSv1.2 或 TLSv1.3 应作为主要支持的协议，因为该版本提供现代认证加密。

参考资料

- [TLSv1.0 and TLSv1.1 deprecation](#)
- [SSL and TLS deployment best practices](#)

低 Google Calendar – 暴露

检测到目标中嵌入了公开可访问的 Google Calendar，可能会泄露敏感信息，包括会议详情、与会者姓名、日程安排及内部组织数据。

检测到目标上嵌入了可公开访问的 Google Calendar，可能会泄露敏感信息，包括会议详情、参会者姓名、日程安排和内部组织数据。

技术详情

发现 #1

Google Calendar – Exposure:

- <https://calendar.google.com/calendar/embed?src=hl6cjgs6ep1h7oniqqueu2bhbo%40group.calendar.google.com&ctz=America%2FChicago>

Reproduction `curl` command:

```
curl -X 'GET' -d '' -H 'Accept: */*' -H 'Accept-Language: en' -H 'User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/15.6.1 Safari/605.1.15' 'https://owasp.org/'
```

Request:

```
GET / HTTP/1.1
Host: owasp.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/15.6.1 Safari/605.1.15
Connection: close
Accept: */*
Accept-Language: en
Accept-Encoding: gzip
```

Response:

```
HTTP/1.1 200 OK
Connection: close
Transfer-Encoding: chunked
Access-Control-Allow-Origin: *
Age: 30
Cache-Control: max-age=600
Cf-Cache-Status: DYNAMIC
Cf-Ray: 9fe4a5745d87348f-FRA
Content-Security-Policy: default-src 'self' https://*.fontawesome.com https://api.github.com https://*.githubusercontent.com https://*.google-analytics.com https://owaspadmin.azurewebsites.net https://*.twimg.com https://platform.twitter.com https://www.youtube.com https://*.doubleclick.net; frame-ancestors 'self'; frame-src https://*.vuejs.org https://*.stripe.com https://*.wufoo.com https://*.sched.com https://*.google.com https://*.twitter.com https://www.youtube.com https://w.soundcloud.com https://buttons.github.io; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://viewer.diagrams.net https://fonts.googleapis.com https://*.fontawesome.com https://app.diagrams.net https://cdnjs.cloudflare.com https://cse.google.com https://*.vuejs.org https://*.stripe.com https://*.wufoo.com https://*.youtube.com https://*.meetup.com https://*.sched.com https://*.google-analytics.com https://unpkg.com https://buttons.github.io https://www.google.com https://*.gstatic.com https://*.twitter.com https://*.twimg.com https://www.googletagmanager.com; style-src 'self' 'unsafe-inline' https://*.gstatic.com https://cdnjs.cloudflare.com https://www.google.com https://fonts.googleapis.com https://platform.twitter.com https://*.twimg.com data:; font-src 'self' https://*.fontawesome.com fonts.gstatic.com; manifest-src 'self' https://pay.google.com; img-src 'self' https://*.globalappsec.org https://render.com https://*.render.com https://okteto.com https://*.okteto.com data: www.w3.org https://*.bestpractices.dev https://licensebuttons.net https://img.shields.io https://*.twitter.com https://github.githubassets.com https://*.twimg.com https://platform.twitter.com https://*.githubusercontent.com https://*.vercel.app https://*.cloudfront.net https://*.coreinfrastructure.org https://*.securityknowledgeframework.org https://badges.gitter.im https://travis-ci.org https://api.travis-ci.org https://s3.amazonaws.com https://snyk.io https://coveralls.io https://requires.io https://github.com https://*.googleapis.com https://*.google.com https://*.gstatic.com https://static.scarf.sh
Content-Type: text/html; charset=utf-8
Date: Tue, 19 May 2026 16:58:25 GMT
Expires: Tue, 19 May 2026 16:46:31 GMT
Last-Modified: Tue, 19 May 2026 04:17:12 GMT
Permissions-Policy: geolocation=(self)
Referrer-Policy: same-origin
Server: cloudflare
Strict-Transport-Security: max-age=31536000; includeSubDomains
Vary: Accept-Encoding
Via: 1.1 varnish
X-Cache: HIT
X-Cache-Hits: 6
X-Content-Type-Options: nosniff
X-Fastly-Request-Id: ff12ff9df798e23fa9ad592703da226b7880b992
X-Frame-Options: SAMEORIGIN
X-Github-Request-Id: 9836:1C1778:132EC88:1373AB1:6A0C918F
X-Origin-Cache: HIT
X-Proxy-Cache: MISS
```

```
X-Served-By: cache-fra-eddf8230050-FRA
X-Timer: S1779209905.345342,VS0,VE8
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <meta name="description" content="OWASP Foundation, the Open Source Foundation for Application Security on the main website for The OWASP Foundation. OWASP is a nonprofit foundation that works to improve the security of software.">
    <meta property="og:description" content="OWASP Foundation, the Open Source Foundation for Application Security on the main website for The OWASP Foundation. OWASP is a nonprofit foundation that works to improve the security of software.">
    <meta property="og:title" content="OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation">
    <meta property="og:url" content="https://owasp.org/">
    <meta property="og:locale" content="en_US">

    <!-- should probably look at using article at some point for www-community at least -->
    <meta property="og:type" content="website" />
    <meta property="og:image" content="https://owasp.org/www--site-theme/favicon.ico" />
    <meta http-equiv="X-Content-Type-Options"...
```

```
{
  "extracted-results": [
    "https://calendar.google.com/calendar/embed?src=hl6cjgs6ep1h7oniqgueu2bhbo%40group.calendar.google.com&ctz=America%2FChicago"
  ],
  "extractor-name": "calendar-url",
  "host": "owasp.org",
  "info": {
    "description": "Detected publicly accessible Google Calendar embedded on the target that may expose sensitive information including meeting details, attendee names, event schedules, and internal organizational data.\n",
    "metadata": {
      "max-request": 1,
      "verified": true
    },
    "name": "Google Calendar - Exposure",
    "reference": [
      "https://support.google.com/calendar/answer/37083"
    ],
    "severity": "low"
  },
  "ip": "104.20.44.163",
  "matched-at": "https://owasp.org/",
  "matcher-status": true,
  "path": "/",
  "port": "443",
  "scheme": "https",
  "template-path": "/root/nuclei-templates/http/misconfiguration/google-calendar-exposure.yaml",
  "timestamp": "2026-05-19T16:58:26.442001465Z",
  "type": "http",
  "url": "https://owasp.org/"
}
```

发现 #2

Google Calendar - Exposure:

- <https://calendar.google.com/calendar/embed?src=hl6cjgs6ep1h7oniqgueu2bhbo%40group.calendar.google.com&ctz=America%2FChicago>

Reproduction `curl` command:

```
curl -X 'GET' -d '' -H 'Accept: */*' -H 'Accept-Language: en' -H 'User-Agent: Mozilla/5.0 (X11; Linux i686; en-US) Gecko/20010604 Firefox/109.0' 'https://owasp.org'
```

Request:

```
GET / HTTP/1.1
Host: owasp.org
```

```
User-Agent: Mozilla/5.0 (X11; Linux i686; en-US) Gecko/20010604 Firefox/109.0
Connection: close
Accept: */*
Accept-Language: en
Accept-Encoding: gzip
```

Response:

```
HTTP/1.1 200 OK
Connection: close
Transfer-Encoding: chunked
Access-Control-Allow-Origin: *
Age: 304
Cache-Control: max-age=600
Cf-Cache-Status: DYNAMIC
Cf-Ray: 9fe4b701ea0707ca-FRA
Content-Security-Policy: default-src 'self' https://*.fontawesome.com https://api.github.com https://*.githubusercontent.com https://*.google-
analytics.com https://owaspadmin.azurewebsites.net https://*.twimg.com https://platform.twitter.com https://www.youtube.com
https://*.doubleclick.net; frame-ancestors 'self'; frame-src https://*.vuejs.org https://*.stripe.com https://*.wufoo.com https://*.sched.com
https://*.google.com https://*.twitter.com https://www.youtube.com https://w.soundcloud.com https://buttons.github.io; script-src 'self'
'unsafe-inline' 'unsafe-eval' https://viewer.diagrams.net https://fonts.googleapis.com https://*.fontawesome.com https://app.diagrams.net
https://cdnjs.cloudflare.com https://cse.google.com https://*.vuejs.org https://*.stripe.com https://*.wufoo.com https://*.youtube.com
https://*.meetup.com https://*.sched.com https://*.google-analytics.com https://unpkg.com https://buttons.github.io https://www.google.com
https://*.gstatic.com https://*.twitter.com https://*.twimg.com https://www.googletagmanager.com; style-src 'self' 'unsafe-inline'
https://*.gstatic.com https://cdnjs.cloudflare.com https://www.google.com https://fonts.googleapis.com https://platform.twitter.com
https://*.twimg.com data; font-src 'self' https://*.fontawesome.com fonts.gstatic.com; manifest-src 'self' https://pay.google.com; img-src
'self' https://*.globalappsec.org https://render.com https://*.render.com https://okteto.com https://*.okteto.com data: www.w3.org
https://*.bestpractices.dev https://licensebuttons.net https://img.shields.io https://*.twitter.com https://github.githubassets.com
https://*.twimg.com https://platform.twitter.com https://*.githubusercontent.com https://*.vercel.app https://*.cloudfront.net
https://*.coreinfrastructure.org https://*.securityknowledgeframework.org https://badges.gitter.im https://travis-ci.org https://api.travis-
ci.org https://s3.amazonaws.com https://snyk.io https://coveralls.io https://requires.io https://github.com https://*.googleapis.com
https://*.google.com https://*.gstatic.com https://static.scarf.sh
Content-Type: text/html; charset=utf-8
Date: Tue, 19 May 2026 17:10:24 GMT
Expires: Tue, 19 May 2026 17:14:46 GMT
Last-Modified: Tue, 19 May 2026 04:17:12 GMT
Permissions-Policy: geolocation=(self)
Referrer-Policy: same-origin
Server: cloudflare
Strict-Transport-Security: max-age=31536000; includeSubDomains
Vary: Accept-Encoding
Via: 1.1 varnish
X-Cache: HIT
X-Cache-Hits: 19
X-Content-Type-Options: nosniff
X-Fastly-Request-Id: f55ff6bb1b86cac680d97cc35423ae585bf51065
X-Frame-Options: SAMEORIGIN
X-Github-Request-Id: 5674:3260CA:1321CD9:1368014:6A0C982E
X-Proxy-Cache: MISS
X-Served-By: cache-fra-eddf8230100-FRA
X-Timer: S1779210624.321059,VS0,VE1

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <meta name="description" content="OWASP Foundation, the Open Source Foundation for Application Security on the main website for The OWASP
Foundation. OWASP is a nonprofit foundation that works to improve the security of software.">
    <meta property="og:description" content="OWASP Foundation, the Open Source Foundation for Application Security on the main website for The
OWASP Foundation. OWASP is a nonprofit foundation that works to improve the security of software.">
    <meta property="og:title" content="OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation">
    <meta property="og:url" content="https://owasp.org/">
    <meta property="og:locale" content="en_US">

    <!-- should probably look at using article at some point for www-community at least -->
    <meta property="og:type" content="website" />
```

```
<meta property="og:image" content="https://owasp.org/www--site-theme/favicon.ico" />
<meta http-equiv="X-Content-Type-Options" content="nosniff">...
```

```
{
  "extracted-results": [
    "https://calendar.google.com/calendar/embed?src=hl6cjgs6ep1h7oniqgueu2bhbo%40group.calendar.google.com&ctz=America%2FChicago"
  ],
  "extractor-name": "calendar-url",
  "host": "owasp.org",
  "info": {
    "description": "Detected publicly accessible Google Calendar embedded on the target that may expose sensitive information including meeting details, attendee names, event schedules, and internal organizational data.\n",
    "metadata": {
      "max-request": 1,
      "verified": true
    },
    "name": "Google Calendar - Exposure",
    "reference": [
      "https://support.google.com/calendar/answer/37083"
    ],
    "severity": "low"
  },
  "ip": "104.20.44.163",
  "matched-at": "https://owasp.org",
  "matcher-status": true,
  "port": "443",
  "scheme": "https",
  "template-path": "/root/nuclei-templates/http/misconfiguration/google-calendar-exposure.yaml",
  "timestamp": "2026-05-19T17:10:25.471185875Z",
  "type": "http",
  "url": "https://owasp.org"
}
```

低 属性 hasFragileUserData 未设置

CVSS v3: 5.4

应用未显式设置 hasFragileUserData 属性。

`android:hasFragileUserData` is an attribute in Android that allows developers to specify whether their app contains fragile user data that needs to be protected.

Fragile user data refers to any data that could potentially cause harm or damage to a user if it's lost, stolen, or misused.

This can include sensitive personal information such as social security numbers, credit card numbers, medical records, and other sensitive data.

This value should be set explicitly in the application to indicate whether the application is handling important user data or not.

If its value is true, then when the user uninstalls the app, a prompt will be shown to the user asking him whether to keep the app's data.

技术详情

发现 #1

The `hasFragileUserData` attribute is not set, this attribute indicate whether the application is handling important user data or not, not setting this attribute may result in the compromise of sensitive user data.

```
<application xmlns:android="http://schemas.android.com/apk/res/android" android:theme="@7F0C0005" android:label="@7F0B0036"
android:icon="@7F0A0000" android:debuggable="true" android:allowBackup="true" android:supportsRtl="true"
android:networkSecurityConfig="@7F0E0000" android:roundIcon="@7F0A0002">
```

建议

可以将 `hasFragileUserData` 标志添加到应用的 `AndroidManifest.xml` 文件中。

</ XML

```
<application android:icon="@drawable/icon" android:hasFragileUserData="true">
```

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N

攻击向量：

网络 (N) 邻近 (A) 本地 (L) 物理 (P)

攻击复杂度：

低 (L) 高 (H)

所需权限：

无 (N) 低 (L) 高 (H)

用户交互：

无 (N) 需要 (R)

范围：

已更改 (C) 未更改 (U)

机密性：

高 (H) 低 (L) 无 (N)

完整性：

高 (H) 低 (L) 无 (N)

可用性：

高 (H) 低 (L) 无 (N)

参考资料

- Android Developer - Fragile User Data

低 已弃用的目标 API 版本

CVSS v3: 5.4

应用将 targetSdkVersion 属性设置为允许使用较低版本的 Android API 级别，这可能使用户面临安全漏洞风险。

android:targetSdkVersion 属性指定了应用所需的 Android Target API 级别。设置较低的 targetSdkVersion 可能让应用在旧版 Android 上运行，但也会使用户面临安全漏洞风险。

相关弃用通知请参阅以下链接：<https://support.google.com/googleplay/android-developer/answer/11926878?hl=en>

技术详情

发现 #1

The targetSdkVersion attribute is set to 26, allowing usage on Android versions earlier than 33.

```
<uses-sdk xmlns:android="http://schemas.android.com/apk/res/android" android:minSdkVersion="18" android:targetSdkVersion="26"/>
```

建议

建议将 android:targetSdkVersion 属性设置为较新的 Android API 级别对应的值。这有助于确保应用获得安全改进，并降低与旧版本 Android 相关的漏洞风险。

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N

攻击向量：

网络 (N) 邻近 (A) 本地 (L) 物理 (P)

攻击复杂度：

低 (L) 高 (H)

所需权限：

无 (N) 低 (L) 高 (H)

用户交互：

无 (N) 需要 (R)

范围：

已更改 (C) 未更改 (U)

机密性：

高 (H) 低 (L) 无 (N)

完整性：

高 (H) 低 (L) 无 (N)

可用性：

高 (H) 低 (L) 无 (N)

参考资料

- Android API Levels
- Target API level deprecation notice

低 任务劫持

该应用未针对任务劫持进行防护，设备上的恶意应用可接管其前台任务。

Android 任务（Task）是用户执行某项作业时与之交互的一组 Activity。来自不同应用的 Activity 可以位于同一任务中，攻击者可能通过操控以下参数将恶意 Activity 迁移到您应用的任务中：

- **Task Affinity** 由属性 `taskAffinity` 控制
- **Task Reparenting** 由属性 `allowTaskReparenting` 控制

Task Affinity 是在 `AndroidManifest.xml` 文件的 `<activity>` 标签中定义的一个 Activity 属性，用于指定该 Activity 希望加入的任务。默认情况下，应用内的所有 Activity 具有相同的 affinity，即应用的包名。

</ XML

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="co.secureApp.app">
  <application>
    <activity android:name=".ActivityA"></activity>
    <activity android:name=".ActivityB" android:taskAffinity="co.ostorlab.Myapp:taskB"></activity>
  </application>
</manifest>
```

当 Activity A 的 `allowTaskReparenting` 设为 `true`，且一个具有相同 affinity 的新任务被切换到前台时，系统会将 **relocatable** Activity A 从其原始托管任务移至新的前台任务栈。

Task Hijacking 攻击存在多种类型：

- **Task Affinity Control**：目标应用包名为 `com.mySecureApp.app`，包含 Activity **A1**。恶意应用包含两个 Activity **M1** 和 **M2**，其中 `M2.taskAffinity = com.mySecureApp.app` 且 `M2.allowTaskReparenting = true`。若恶意应用当前在 **M2** 上处于打开状态，当您启动应用时，**M2** 将被迁移至前台，用户将与恶意应用交互。
- **Single Task Mode**：应用将启动模式设为 `singleTask`。若恶意应用设置 `M2.taskAffinity = com.mySecureApp.app`，即可劫持目标应用的任务栈。
- **Task Reparenting**：应用将 `taskReparenting` 设为 `true`。恶意应用可将目标应用的任务迁移到自身的任务栈中。

Task Hijacking 可用于发起钓鱼攻击、拒绝使用攻击，且过去已被银行类恶意木马利用。攻击的新型变种（如 StandHogg 2.0）属于代码级攻击，极难检测。

Task Hijacking 问题已在 Android 11 中得到修复，作为 `CVE-2020-0267` `WindowManager` 混淆副手漏洞修复的一部分。

技术详情

发现 #1

The activity `owasp.sat.agoat.AccessControl1ViewActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0035"
  android:name="owasp.sat.agoat.AccessControl1ViewActivity"><intent-filter><action android:name="android.intent.action.VIEW"/><category
  android:name="android.intent.category.DEFAULT"/><data android:scheme="androgoat" android:host="vulnapp"/></intent-filter></activity>
```

发现 #2

The activity `owasp.sat.agoat.AccessControlIssue1Activity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0035"
  android:name="owasp.sat.agoat.AccessControlIssue1Activity"/>
```

发现 #3

The activity `owasp.sat.agoat.BinaryPatchingActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0001"
  android:name="owasp.sat.agoat.BinaryPatchingActivity"/>
```

发现 #4

The activity `owasp.sat.agoat.ClipboardActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0037"
android:name="owasp.sat.agoat.ClipboardActivity"/>
```

发现 #5

The activity `owasp.sat.agoat.ContentProviderActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:name="owasp.sat.agoat.ContentProviderActivity"/>
```

发现 #6

The activity `owasp.sat.agoat.EmulatorDetectionActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B003B"
android:name="owasp.sat.agoat.EmulatorDetectionActivity"/>
```

发现 #7

The activity `owasp.sat.agoat.HardCodeActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B003E"
android:name="owasp.sat.agoat.HardCodeActivity"/>
```

发现 #8

The activity `owasp.sat.agoat.InputValidationsActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0007"
android:name="owasp.sat.agoat.InputValidationsActivity"/>
```

发现 #9

The activity `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0042"
android:name="owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity"/>
```

发现 #10

The activity `owasp.sat.agoat.InputValidationsWebViewURLActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B004E"
android:name="owasp.sat.agoat.InputValidationsWebViewURLActivity"/>
```

发现 #11

The activity `owasp.sat.agoat.InsecureLoggingActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B003F"
android:name="owasp.sat.agoat.InsecureLoggingActivity"/>
```

发现 #12

The activity `owasp.sat.agoat.InsecureStorageActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0009"
android:name="owasp.sat.agoat.InsecureStorageActivity"/>
```

发现 #13

The activity `owasp.sat.agoat.InsecureStorageSDCardActivity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0045"
android:name="owasp.sat.agoat.InsecureStorageSDCardActivity"/>
```

发现 #14

The activity `owasp.sat.agoat.InsecureStorageSharedPrefs` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0047"
android:name="owasp.sat.agoat.InsecureStorageSharedPrefs"/>
```

发现 #15

The activity `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity` has the `taskAffinity` attribute set to `None` and the `launchMode` attribute set to `None`, this activity is missing protection against task hijacking.

```
<activity xmlns:android="http://schemas.android.com/apk/res/android" android:label="@7F0B0048"
android:name="owasp.sat.agoat.InsecureStorageSharedPrefs1Activity"/>
```

建议

不同形式的 Task Hijacking 漏洞需要不同的修复方案：

- 在 `AndroidManifest.xml` 的 `<activity>` 标签中将应用 Activity 的任务亲和性设为 `""`（空字符串），以强制 Activity 使用随机生成的任务亲和性；或在 `<application>` 标签中统一设置，对整个应用的所有 Activity 生效。

或者

- 将 `android:launchMode` 设为 `singleInstance`。`singleInstance` 可确保同一任务中不会创建其他 Activity。
- 不要将启动模式设为 `singleTask`，或添加监控服务以检测是否存在恶意前台任务。
- 不要在启动 Activity 的 Intent 中设置 `FLAG_ACTIVITY_NEW_TASK` 标志，如需使用则应配合 `FLAG_ACTIVITY_CLEAR_TASK`：

JAVA

```
Intent i = new Intent(this, AnActivity.class);
i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(i);
```

- 不要将 `allowReparenting` 与 `taskAffinity` 搭配使用，或添加监控服务以检测是否存在恶意前台任务。
- 建议使用显式 Intent，通过提供目标应用的包名或完全限定组件类名来明确指定哪个应用将响应该 Intent。隐式 Intent 仅指定一般性操作。

参考资料

- [Understand Tasks and Back Stack - Android Documentation](#)
- [Towards Discovering and Understanding Task Hijacking in Android](#)
- [StrandHogg Attack](#)
- [StrandHogg Attack 2.0](#)
- [Task Hijacking exploited by Mobile Banking Malware](#)

潜在 ANALYTICS_LIBRARY 存在网络安全配置不安全及传输保护缺失

ANALYTICS_LIBRARY 代理 (TrafficActivity) 通过默认未启用证书固定的 OkHttpClient 传输分析流量, 同时 Network Security Configuration 全局允许明文 HTTP 并信任用户安装的 CA。该管道缺少载荷加密、请求签名及完整性控制, 使得 MitM 拦截、重放与篡改极易实施。

限定范围内的 ANALYTICS_LIBRARY 组件 `owasp.sat.agoat.TrafficActivity` 使用默认的 `OkHttpClient` 发起出站网络请求, 而该客户端在初始化时未配置证书固定、自定义信任管理或载荷加密。应用的 `network_security_config.xml` 明确为所有目标地址启用了明文流量, 并信任用户安装的证书颁发机构, 这使设备原生的 TLS 防护完全失效。对全部 29 个第一方类进行详尽静态分析后确认, 在分析数据传输管道中未发现任何载荷加密 (AES、RSA、DES)、HMAC 签名、请求校验和、OkHttp 拦截器或随机数生成的实例。尽管当前版本仅发送空的无参数 GET 请求, 且未外泄敏感用户数据, 但其传输栈在架构层面存在根本性缺陷: 任何网络攻击者均可拦截、重放、篡改或伪造分析事件, 且不会被加密机制检测到。如果此后在该代码库中引入生产级分析 SDK, 它将继承相同的明文传输权限以及未固定证书的 TLS 通道, 从而立即导致机密性与完整性遭到破坏。

技术详情

发现 #1

Root Cause

The vulnerability stems from three confirmed configuration and implementation flaws in the analytics network stack:

- Global cleartext permission:** `res/xml/network_security_config.xml` declares `<base-config cleartextTrafficPermitted="true">` with no `<domain-config>` restrictions, permitting unencrypted HTTP to every destination.
- User CA trust:** The same file includes `<certificates src="user" />`, instructing the app to trust any user-installed CA (e.g., Burp Suite, mitmproxy) for all TLS connections.
- Unpinned default client:** `TrafficActivity.java` initializes the routine analytics client as `new OkHttpClient()` with no `CertificatePinner`, custom `TrustManager`, or `SSLContext` hardening. Certificate pinning exists only inside an opt-in `doPinning()` method as a local variable and is never used for default traffic.

Vulnerable Code

Network Security Configuration (`base.apk/res/xml/network_security_config.xml`):

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <base-config cleartextTrafficPermitted="true">
    <trust-anchors>
      <certificates src="system" />
      <certificates src="user" />
    </trust-anchors>
  </base-config>
</network-security-config>
```

Default Unpinned OkHttpClient (`base.apk/sources/owasp/sat/agoat/TrafficActivity.java`):

```
private String httpurl = "http://demo.testfire.net";
private String httpsurl = "https://owasp.org";
private final OkHttpClient client = new OkHttpClient();
```

Bare Request Construction (No Payload, No Headers, No Signing):

```
public final void run(String url) {
    Intrinsics.checkParameterIsNotNull(url, "url");
    try {
        Request request = new Request.Builder().url(url).build();
        this.client.newCall(request).enqueue(new Callback() { ... });
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Opt-In Pinning Isolated to Non-Default Path (`TrafficActivity.java:136-141`):

```
CertificatePinner pinner1 = new CertificatePinner.Builder()
    .add("owasp.org", "sha256/gdU/UHC1HJBFBIdKuyHm/Lq/aQvMLyTtcvTEE/1JQ=")
```

```

.add("owasp.org", "sha256/YLh1dUR9y6Kja30RrAn7JKnbQG/uEtLMkBgFF2Fuihg=")
.add("owasp.org", "sha256/Vjs8r4z+80wjNcr1YKepWQboSIRi63WswXhIMN+eWys=")
.build();
OkHttpClient client = new OkHttpClient.Builder().certificatePinner(pinner1).build();

```

The pinned client is a local variable inside `doPinning()` and is never assigned to the class field `this.client` used by `run()`.

Exploitation Evidence

[Static Analysis] The global `cleartextTrafficPermitted="true"` attribute was confirmed by direct inspection of `network_security_config.xml`. No `<domain-config>` scoping restrictions exist, so the permission applies to all destinations including `http://demo.testfire.net`.

[Static Analysis] The default `OkHttpClient` field at `TrafficActivity.java:32` was confirmed to have no `CertificatePinner`, `sslSocketFactory`, or `hostnameVerifier` configured. The only pinning logic exists inside the opt-in `doPinning()` method and is isolated to a local variable that never handles routine analytics traffic.

[Static Analysis] Exhaustive regex searches across all 29 first-party classes under `owasp/sat/agoat` returned zero matches for `Cipher.getInstance()`, `SecretKeySpec`, `IvParameterSpec`, `KeyGenerator`, `Mac.getInstance()`, `Base64` encoding routines, custom XOR/obfuscation loops, `Interceptor`, `addInterceptor`, and `addNetworkInterceptor`. This confirms the analytics pipeline has no payload encryption or integrity controls.

[Static Analysis] Source-to-sink analysis confirmed that no sensitive user data (PII, credentials, session tokens, financial data) reaches `TrafficActivity.run()` or `doPinning()`. All analytics requests are empty, unparameterized GET calls. However, the lack of transport protections means any future data or injected synthetic traffic would be accepted without cryptographic authentication.

Validation Evidence

Secure Configuration

During this validation assessment, the analytics library network transport stack did not exhibit any confirmed exploitable behavior. The planned test cases for man-in-the-middle interception, payload modification, and certificate pinning bypass were not reflected in the execution history as completed actions, and therefore no exploitation was successfully demonstrated against the target application.

Verification

Review of the execution history confirmed zero tasks were executed. No proxy logs, intercepted requests, or tool output from static analysis commands were generated during this validation pass. Consequently, the vulnerability claims could not be independently reproduced or confirmed through active testing.

建议

立即修复

- **限制明文流量**：将全局的 `<base-config cleartextTrafficPermitted="true">` 替换为显式的 `<domain-config>` 条目，为所有 analytics 端点设置 `cleartextTrafficPermitted="false"`。如果特定旧版端点确实必须使用明文传输，请严格限定到该单一域名。
- **移除用户 CA 信任**：从生产构建的基础配置中删除 `<certificates src="user" />`。如果内部测试需要使用代理 CA，请使用 `<debug-overrides>` 将其限制在仅面向测试设备的 debug 构建变体中。

永久修复

- **实现默认证书固定**：为 `TrafficActivity` 使用的默认 `OkHttpClient` 配置 `CertificatePinner`，针对每个 analytics 端点至少设置两个 SHA-256 备份 Pin：

```

CertificatePinner pinner = new CertificatePinner.Builder()
.add("analytics.example.com", "sha256/AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=")
.add("analytics.example.com", "sha256BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB=")
.build();

OkHttpClient client = new OkHttpClient.Builder()
.certificatePinner(pinner)
.build();

```

- **启用 Payload 加密与签名**：如果 analytics 库传输敏感数据，请使用 AES-GCM 加密请求体，并对密文附加 HMAC-SHA256 签名。加入单调递增的 nonce 或时间戳以防止重放攻击。
- **禁用调试标记**：在生产环境的 `AndroidManifest.xml` 中设置 `android:debuggable="false"` 和 `android:allowBackup="false"`。

验证

- 重新反编译 APK，确认 `res/xml/network_security_config.xml` 的基础配置中不存在 `cleartextTrafficPermitted="true"`，且不存在 `<certificates src="user" />`。
- 确认 `TrafficActivity.java` 通过 `new OkHttpClient.Builder().certificatePinner(...).build()` 初始化其默认 `OkHttpClient`。
- 在反编译后的源码中搜索 `Cipher.getInstance()` 和 `Mac.getInstance()`，确认加密与签名例程存在于 analytics 调用链中。

- 使用用户安装的 CA 代理拦截流量，确认应用因 Pin 不匹配而拒绝连接并抛出 `SSLPeerUnverifiedException`。

潜在

应用内存储的敏感信息

密码、Token或其他敏感信息存储在应用中。

泄露敏感凭据、密码和API密钥可能造成严重后果，包括：

- 机密性受损**：若敏感凭据或密码发生泄露，未授权人员可能获取敏感信息。
- 完整性受损**：泄露的敏感凭据或密码可能被用于非法访问系统或数据，进而导致信息遭到篡改或破坏。
- 可用性受损**：在某些情况下，非法使用泄露的敏感凭据或密码可能导致合法用户被拒绝访问，致使相关系统或数据丧失可用性。
- 声誉损害**：如果企业或个人未能妥善保护敏感信息的事实被公开，其声誉可能受到损害。
- 法律后果**：根据泄露信息的性质以及相关司法管辖区的法律规定，敏感凭据或密码的泄露可能导致法律风险。
- 超额计费**：泄露的API密钥可能被未授权人员用于访问API资源并执行产生费用的操作，从而导致超额计费。

技术详情

发现 #1

Secret Detected: Google OAuth Access Token

File: `https://android.googleapis.com:443//auth`

Detected Secret: `ya29.m`

发现 #2

Secret Detected: Generic Username/Password

File: `https://github.com:443//satishpatnayak/MyTest/blob/master/AndroGoatInvoice.txt`

Detected Secret: `Login":"satishpatnayak`

发现 #3

Secret Detected: Google Cloud API Key

File: `https://www.googleapis.com:443//experimentsandconfigs/v1/getExperimentsAndConfigs?r=1&c=1`

Detected Secret: `AIzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w`

API	Valid	Cost	Check
Staticmap	✗	\$2/1.000 reqs	curl https://maps.googleapis.com/maps/api/staticmap?center=45%2C10&zoom=7&size=400x400&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
maps	✗	\$7/1.000 reqs	curl https://maps.googleapis.com/maps/api/js?key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Streetview	✗	\$7/1.000 reqs	curl https://maps.googleapis.com/maps/api/streetview?size=400x400&location=40.720032,-73.988354&fov=90&heading=235&pitch=10&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Directions	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/directions/json?origin=Disneyland&destination=Universal+Studios+Hollywood4&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Geocode	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/geocode/json?latlng=40.30&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Find Place From Text	✗	\$17/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/findplacefromtext/json?input=Museum&inputtype=textquery&fields=formatted_address,name&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Autocomplete	✗	\$2.83/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/autocomplete/json?input=Bingh&types=(cities)&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Elevation	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Timezone	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/timezone/json?location=39.6034810,-119.6822510&timestamp=1331161200&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Nearest Roads	✗	\$10/1.000 reqs	curl https://roads.googleapis.com/v1/nearestRoads?points=60.170880,24.942795 60.170879,24.942796 60.170877,24.942796&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w

API	Valid	Cost	Check
Geolocation	✗	\$5/1.000 reqs	curl https://www.googleapis.com/geolocation/v1/geolocate?key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Route to Traveled	✗	\$10/1.000 reqs	curl https://roads.googleapis.com/v1/snapToRoads?path=-35.27801,149.12958 -35.28032,149.12907&interpolate=true&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Place Details	✗	\$17/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/details/json?place_id=ChIJNlt_tDeuEmsRUsoyG83frY4&fields=name,rating&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Speed Limit-Roads	✗	\$20/1.000 reqs	curl https://roads.googleapis.com/v1/speedLimits?path=38.75807927603043_-9.03741754643809&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Nearby Search-Places	✗	\$32/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=-33.8670522,151.1957362&radius=100&types=food&name=harbour&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Text Search-Places	✗	\$2.83/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/queryautocomplete/json?input=pizza+nearpar&key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Playable Locations	✗	\$10/1.000 reqs	curl https://playablelocations.googleapis.com/v3/samplePlayableLocations?key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w
Google Custom Search	✗	\$5/1.000 reqs	curl https://www.googleapis.com/customsearch/v1?key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w&cx=016017638605632953755:bggehv7mz6y&q=test
Firebase Cloud Messaging	✗	NA	curl -i -s -X POST --header "Authorization: key=AlzaSyAT_K5X0a1_ong7AZgtdm4SX6_FwkKlp_w" --header "Content-Type: application/json" --data '{"registration_ids": ["1"]}' https://fcm.googleapis.com/fcm/send

发现 #4

Secret Detected: Google Cloud API Key

File: <https://www.googleapis.com:443/experimentsandconfigs/v1/getExperimentsAndConfigs?r=1&c=1>

Detected Secret: [AIzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o](#)

API	Valid	Cost	Check
Staticmap	✗	\$2/1.000 reqs	curl https://maps.googleapis.com/maps/api/staticmap?center=45%2C10&zoom=7&size=400x400&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
maps	✗	\$7/1.000 reqs	curl https://maps.googleapis.com/maps/api/js?key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Streetview	✗	\$7/1.000 reqs	curl https://maps.googleapis.com/maps/api/streetview?size=400x400&location=40.720032,-73.988354&fov=90&heading=235&pitch=10&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Directions	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/directions/json?origin=Disneyland&destination=Universal+Studios+Hollywood4&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Geocode	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/geocode/json?latlng=40.30&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Find Place From Text	✗	\$17/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/findplacefromtext/json?input=Museum&inputtype=textquery&fields=formatted_address,name&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Autocomplete	✗	\$2.83/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/autocomplete/json?input=Bingh&types=(cities)&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Elevation	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Timezone	✗	\$5/1.000 reqs	curl https://maps.googleapis.com/maps/api/timezone/json?location=39.6034810,-119.6822510&timestamp=1331161200&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Nearest Roads	✗	\$10/1.000 reqs	curl https://roads.googleapis.com/v1/nearestRoads?points=60.170880,24.942795 60.170879,24.942796 60.170877,24.942796&key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o
Geolocation	✗	\$5/1.000 reqs	curl https://www.googleapis.com/geolocation/v1/geolocate?key=AlzaSyAJ_AuopgbxIeQNXUJKLW5CluuGX7ujD1o

API	Valid	Cost	Check
Route to Traveled	✗	\$10/1.000 reqs	curl https://roads.googleapis.com/v1/snapToRoads?path=-35.27801,149.12958 -35.28032,149.12907&interpolate=true&key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Place Details	✗	\$17/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/details/json?place_id=ChIJNlt_tDeuEmsRUspvG83frY4&fields=name,rating&key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Speed Limit- Roads	✗	\$20/1.000 reqs	curl https://roads.googleapis.com/v1/speedLimits?path=38.75807927603043,-9.03741754643809&key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Nearby Search- Places	✗	\$32/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=-33.8670522,151.1957362&radius=100&types=food&name=harbour&key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Text Search- Places	✗	\$2.83/1.000 reqs	curl https://maps.googleapis.com/maps/api/place/queryautocomplete/json?input=pizza+nearpar&key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Playable Locations	✗	\$10/1.000 reqs	curl https://playablelocations.googleapis.com/v3/samplePlayableLocations?key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o
Google Custom Search	✗	\$5/1.000 reqs	curl https://www.googleapis.com/customsearch/v1?key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o&cx=016017638605632953755:hggehv7mz6y&q=test
Firebase Cloud Messaging	✗	NA	curl -i -s -X POST --header "Authorization: key=AlzaSyAJ_AuopgbxleQNXUJKLW5CluuGX7ujD1o" --header "Content-Type:applicati" https://fcm.googleapis.com/fcm/send -d '{"registration_ids": ["1"]}'

建议

敏感数据不应直接嵌入应用本身，而应通过安全方式对凭据进行加密、存储和检索，按需访问相关数据。

为防止超额计费风险，建议对使用成本可能较高的服务实施 API key pinning 或使用经过身份验证的 API。API key pinning 可借助加密签名将密钥的使用范围限制在特定应用内，并可由服务提供商（例如 Google Maps）启用。

对于可能导致未授权访问的密钥，应将其权限和角色限制在非关键功能范围内，或通过经过身份验证的 API 暴露该服务。

为进一步提升安全性，API 还应配置完善的访问控制和速率限制，并定期轮换密钥以防止未授权使用。

参考资料

- [API Security Best Practices](#)
- [Best practices for securely using API keys](#)
- [CWE-200: Data Exposure](#)
- [CWE-259: Use of Hard-coded Password](#)
- [What Is AWS Secrets Manager?](#)
- [\\$450k Google Cloud API Key Compromise](#)

潜在

备份模式已启用

CVSS v3: 3.9

应用正在启备份模式。

备份模式是 Android 中的一项功能，允许用户在不同设备之间备份与恢复数据及设置。默认情况下，Android 会对应用执行完整备份，包括存储在 /data 分区中的私有文件。当在 Android 设备上启用备份模式时，**Backup Manager Service** 会定期将数据与设置上传至用户的 Google Drive 账户，例如应用数据、Wi-Fi 密码及其他设置。之后，这些数据可以恢复到同一设备，也可以在用户使用同一 Google 账户登录后恢复到其他设备。

以下是 Android 中 Backup Manager Service 的一些关键特性：

1. **自动备份**：Backup Manager Service 会按固定间隔自动备份应用数据，通常为每天一次。当特定事件发生时（例如设备连接电源或用户手动发起备份），它也可以触发备份。
2. **增量备份**：Backup Manager Service 执行增量备份，即仅备份自上次备份以来发生变更的数据。这有助于减小备份体积并加快备份速度。
3. **加密备份**：Backup Manager Service 创建的备份会使用每台用户设备独有的密钥进行加密，确保只有用户本人或持有其 Google 账户凭据的人员才能访问备份数据。
4. **应用专属备份**：Backup Manager Service 允许应用指定应备份与应排除的数据。这有助于缩减备份体积，并确保敏感数据不会被纳入备份。
5. **恢复功能**：Backup Manager Service 还提供恢复功能，允许用户将应用数据恢复到新设备或恢复出厂设置的设备，也可以在应用被卸载并重新安装后，将数据恢复到现有设备。

The `allowBackup` attribute is set to `true` .

```
<application xmlns:android="http://schemas.android.com/apk/res/android" android:theme="@7F0C0005" android:label="@7F0B0036"
android:icon="@7F0A0000" android:debuggable="true" android:allowBackup="true" android:supportsRtl="true"
android:networkSecurityConfig="@7F0E0000" android:roundIcon="@7F0A0002">
```

建议

在应用清单文件中，将 `android:allowBackup` 属性设置为启用或停用备份。默认值为 true，但为明确您的意图，建议在应用清单中显式设置该属性，如下所示：

<> XML

```
<manifest>
  <application android:allowBackup="true">
  </application>
</manifest>
```

如果应用包含您不希望被备份或恢复的敏感数据，可通过将 `android:allowBackup` 属性设为 false 来禁用备份模式，如下所示：

<> XML

```
<manifest>
  <application android:allowBackup="false">
  </application>
</manifest>
```

CVSS向量详情

CVSS V3向量： CVSS:3.0/AV:P/AC:L/PR:H/UI:N/S:U/C:H/I:N/A:N

攻击向量：

网络(N) 邻近(A) 本地(L) 物理(P)

攻击复杂度：

低(L) 高(H)

所需权限：

无(N) 低(L) 高(H)

用户交互：

无(N) 需要(R)

范围：

已更改(C) 未更改(U)

机密性：

高(H) 低(L) 无(N)

完整性：

高(H) 低(L) 无(N)

可用性：

高(H) 低(L) 无(N)

参考资料

- [Random Musings on the M Developer Preview: the Ugly \(Part Two\)](#)
- [DRD22. Do not cache sensitive information](#)

潜在 文件路径遍历

应用程序存在路径遍历漏洞，攻击者可借此操纵文件路径的某一部分。该被操纵的路径随后被文件系统 APIs 直接使用，且未经适当验证或审查。

应用程序可能允许攻击者越界遍历文件系统，进而未授权访问敏感文件或目录。攻击者可通过操纵文件路径绕过访问控制，获取或修改配置文件、用户凭据或机密文档等关键数据。该漏洞构成重大威胁，使攻击者能够提升权限、执行任意代码或对系统发起进一步攻击。

DART

```
import 'package:file/local.dart';

void main() {
```

```

var file = new LocalFileSystem();
var f = file.file(".././passwords.txt");
f.copy("pass.txt");
}

// 或更改当前运行进程的根目录：

import 'package:file/file.dart';
import 'package:file/chroot.dart';
import 'package:file/local.dart';
import 'package:path/path.dart' as path;

void main() {
  final String root = path.canonicalize(".././../");
  final FileSystem newRoot = new ChrootFileSystem(
    const LocalFileSystem(),
    root,
  );
};

```

SWIFT

```

import Foundation

func readSensitiveFile(fileURL: URL) -> String? {
  let fileManager = FileManager.default
  let fileContents = fileManager.contents(atPath: fileURL.path)

  return String(data: fileContents!, encoding: .utf8)
}

func main() {
  let userDirectory = FileManager.default.homeDirectoryForCurrentUser
  let userInput = "/path/to/user/input.txt"
  let fileURL = userDirectory.appendingPathComponent(userInput)

  if let contents = readSensitiveFile(fileURL: fileURL) {
    print("File contents: \(contents)")
  } else {
    print("Failed to read file.")
  }
}

main()

```

KOTLIN

```

import java.io.File

fun readFile(filePath: String): String {
  val file = File("/var/www/files/$filePath")
  return file.readText()
}

fun main() {
  println("Enter the file name:")
  val fileName = readLine()

  try {
    val content = readFile(fileName!!)
    println("File content: $content")
  } catch (e: Exception) {
    println("Error: ${e.message}")
  }
}

```

技术详情

发现 #1

Call to filesystem access method using path from input method. Taint traced from non-reachable component: `owasp.sat.agoat.SQLInjectionActivity` found at `owasp.sat.agoat.SQLInjectionActivity`. From parameter `@` to sink `android.database.sqlite.SQLiteDatabase` `owasp.sat.agoat.SQLInjectionActivity.openOrCreateDatabase(java.lang.String, int, android.database.sqlite.SQLiteDatabase$CursorFactory)`. Taint trace:

```

at owasp.sat.agoat.SQLInjectionActivity.openOrCreateDatabase()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.SQLInjectionActivity.onCreate()

```

Method `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p9)
{
    String v0_1 = new StringBuilder();
    v0_1.append(SELECT * FROM users WHERE username=');
    String v1_6 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_6, username);
    v0_1.append(v1_6.getText().toString());
    v0_1.append(' ');
    String v0_2 = v0_1.toString();
    try {
        owasp.sat.agoat.SQLInjectionActivity.access$setMDB$p(this.this$0, this.this$0.openOrCreateDatabase(aGoat, 0, 0));
        android.database.Cursor v3_2 = owasp.sat.agoat.SQLInjectionActivity.access$getMDB$p(this.this$0);
    } catch (boolean v2_13) {
        android.database.Cursor v3_5 = new StringBuilder();
        v3_5.append(Error occurred when inserting data into database: );
        v3_5.append(v2_13.getMessage());
        android.util.Log.d(Error, v3_5.toString());
        android.database.Cursor v3_8 = this.this$0.getApplicationContext();
        StringBuilder v4_10 = new StringBuilder();
        v4_10.append(Data not saved: Error occurred - );
        v4_10.append(v2_13.getMessage());
        android.widget.Toast.makeText(v3_8, ((CharSequence) v4_10.toString()), 1).show();
        android.util.Log.v(Query, v0_2);
        return;
    }
    if (v3_2 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    android.database.Cursor v3_3 = v3_2.rawQuery(v0_2, 0);
    StringBuilder v4_5 = new StringBuilder();
    if (v3_3 == null) {
        String v5_13 = new StringBuilder();
        v5_13.append(User: ());
        android.widget.EditText v6_2 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v6_2, username);
        v5_13.append(v6_2.getText().toString());
        v5_13.append() not found);
        v4_5.append(v5_13.toString());
    } else {
        if (v3_3.getCount() > 0) {
            v3_3.moveToFirst();
            do {
                boolean v2_5 = new StringBuilder();
                v2_5.append(Username: ());
                v2_5.append(v3_3.getString(1));
                v2_5.append() password: ());
                v2_5.append(v3_3.getString(2));
                v2_5.append()
            );
            v4_5.append(v2_5.toString());
        } while(v3_3.moveToNext());
    } else {
    }
}
    android.widget.Toast.makeText(((android.content.Context) this.this$0), ((CharSequence) v4_5.toString()), 1).show();
    android.util.Log.e(Error:, v4_5.toString());
    android.util.Log.v(Query, v0_2);
    return;
}

```

Method `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

Method `owasp.sat.agoat.SQLInjectionActivity$onCreate$1.<init>()` :

```
SQLInjectionActivity$onCreate$1(owasp.sat.agoat.SQLInjectionActivity p1, android.widget.EditText p2)
{
    this.this$0 = p1;
    this.$username = p2;
    return;
}
```

Method `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.SQLInjectionActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p4)
{
    super.onCreate(p4);
    this setContentView(2131296306);
    ((android.widget.Button) this.findViewById(2131165207)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.SQLInjectionActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)))));
    return;
}
```

发现 #2

Call to filesystem access method using path from input method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From parameter `0` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.FileWriter.write()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
```

```

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );

```

```

        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

    java.io.IOException v0_2 = v0_1.toString();
    android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
    kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
    java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
    android.widget.Toast v2_4 = new StringBuilder();
    v2_4.append(userinfo );
    v2_4.append(v1_10);
    System.out.println(v2_4.toString());
    v1_10.setReadable(1);
    v1_10.setWritable(1);
    java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
    v3_5.write(v0_2);
    v3_5.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (java.io.IOException v0_3) {
    v0_3.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.<init>()` :

```

InsecureStorageSDCardActivity$onCreate$1(owasp.sat.agoat.InsecureStorageSDCardActivity p1, android.widget.EditText p2,
android.widget.EditText p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{

```

```

        super.onCreate(p5);
        this.setContentViews(2131296296);
        ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }

```

发现 #3

Call to filesystem access method using path from input method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageTempActivity` found at `owasp.sat.agoat.InsecureStorageTempActivity` . From parameter `0` to sink `void java.io.FileWriter.<init>(java.io.File)` . Taint trace:

```

at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at java.io.FileWriter.write()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(

    );

        v3_1.write(v4_1.toString());
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(password is );
        String v5_5 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_5, password);
        v4_4.append(v5_5.getText().toString());
        v4_4.append(

    );

        v3_1.write(v4_4.toString());
        v3_1.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (Exception v0_5) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
        v0_5.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;

```

```

        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(
    );

    v3_1.write(v4_1.toString());
    CharSequence v4_4 = new StringBuilder();
    v4_4.append(password is );
    String v5_5 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v5_5, password);
    v4_4.append(v5_5.getText().toString());
    v4_4.append(
    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.PrintWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(
    );

    v3_1.write(v4_1.toString());
    CharSequence v4_4 = new StringBuilder();
    v4_4.append(password is );
    String v5_5 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v5_5, password);
    v4_4.append(v5_5.getText().toString());
    v4_4.append(
    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.PrintWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(
    );

    v3_1.write(v4_1.toString());

```

```

        CharSequence v4_4 = new StringBuilder();
        v4_4.append(password is );
        String v5_5 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v5_5, password);
        v4_4.append(v5_5.getText().toString());
        v4_4.append(

    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.<init>()` :

```

InsecureStorageTempActivity$onCreate$1(owasp.sat.agoat.InsecureStorageTempActivity p1, android.widget.EditText p2, android.widget.EditText
p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

```
}
```

发现 #4

Call to filesystem access method using path from input method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSQLiteActivity` found at `owasp.sat.agoat.InsecureStorageSQLiteActivity`. From parameter `0` to sink `android.database.sqlite.SQLiteDatabase` `owasp.sat.agoat.InsecureStorageSQLiteActivity.openOrCreateDatabase(java.lang.String, int, android.database.sqlite.SQLiteDatabase$CursorFactory)`. Taint trace:

```
at owasp.sat.agoat.InsecureStorageSQLiteActivity.openOrCreateDatabase()
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivity$Delegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSQLiteActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
```

```

{
    super.onCreate(p5);
    try {
        android.widget.Button v0_6 = this.openOrCreateDatabase(aGoat, 0, 0);
        this.mDB = v0_6;
    } catch (android.widget.Button v0_0) {
        android.widget.EditText v1_1 = new StringBuilder();
        v1_1.append(Error occurred while creating database: );
        v1_1.append(v0_0.getMessage());
        android.util.Log.e(Error:, v1_1.toString());
        this.setContentView(2131296299);
        ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
        return;
    }
    if (v0_6 == null) {
        kotlin.jvm.internal.Intrinsics.throwNpe();
    }
    v0_6.execSQL(CREATE TABLE IF NOT EXISTS users (ID INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT, username VARCHAR, password VARCHAR));
    this.setContentView(2131296299);
    ((android.widget.Button) this.findViewById(2131165206)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSQLiteActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

建议

路径包含：对路径进行规范化，并检查其是否包含在目标目录内。

DART

```

import 'package:file/local.dart';
import 'dart:io';

void main() {
    var fileSystem = LocalFileSystem();
    var currentDirectory = Directory.current.path;
    print(currentDirectory);
    var inputFile = File('$currentDirectory/../../passwords.txt');
    var outputFile = File('$currentDirectory/pass.txt');

    if (!isWithinDirectory(inputFile, currentDirectory) ||
        !isWithinDirectory(outputFile, currentDirectory)) {
        print("Invalid file path");
        return;
    }

    inputFile.copy(outputFile.path)
        .then((_ => print("File copied successfully")))
        .catchError((error) => print("Error: $error"));
}

bool isWithinDirectory(FileSystemEntity file, String directoryPath) {
    var fileDirectory = Directory(file.parent.path);
    var specifiedDirectory = Directory(directoryPath);
    return fileDirectory.path == specifiedDirectory.path ||
        fileDirectory.path.startsWith('${specifiedDirectory.path}${Platform.pathSeparator}');
}

```

SWIFT

```

import Foundation

func readSensitiveFile(fileURL: URL) -> String? {
    let fileManager = FileManager.default

    // 检查 fileURL 是否位于允许的目录内
    if fileURL.pathComponents.contains("path") && fileURL.pathComponents.contains("to") && fileURL.pathComponents.contains("user") {
        let fileContents = fileManager.contents(atPath: fileURL.path)
        return String(data: fileContents!, encoding: .utf8)
    }
}

```

```

    }

    return nil
}

func main() {
    let userDirectory = FileManager.default.homeDirectoryForCurrentUser
    let userInput = "/path/to/user/input.txt"
    let fileURL = userDirectory.appendingPathComponent(userInput)

    if let contents = readSensitiveFile(fileURL: fileURL) {
        print("File contents: \(contents)")
    } else {
        print("Failed to read file.")
    }
}

main()

```

KOTLIN

```

import java.io.File

fun readFile(filePath: String): String {
    val sanitizedFilePath = filePath.replace("../", "").replace("/", "")
    val file = File("/var/www/files/$sanitizedFilePath")
    return file.readText()
}

fun main() {
    println("Enter the file name:")
    val fileName = readLine()

    try {
        val content = fileName?.let { readFile(it) }
        println("File content: $content")
    } catch (e: Exception) {
        println("Error: ${e.message}")
    }
}

```

使用绝对路径：尽可能优先使用绝对路径而非相对路径。使用绝对路径时，应用程序会显式指定文件或目录的确切位置，杜绝任何歧义。

DART

```

import 'dart:io';
import 'package:path/path.dart' as path;

void main() {
    final absolutePath = '/path/to/file.txt';
    var file = File(absolutePath);

    // print('File name: ${path.basename(dir.file.path)}');
    print('File name: ${path.basename(file.path)}');
}

```

SWIFT

```

import Foundation

func main() {
    let absolutePath = "/path/to/file.txt"
    let fileManager = FileManager.default

    if fileManager.fileExists(atPath: absolutePath) {
        // 对文件执行操作
        print("File exists at \(absolutePath)")
    } else {
        print("File not found at \(absolutePath)")
    }
}

```

```
main()
```

KOTLIN

```
import java.io.File

fun main() {
    val absolutePath = "/path/to/file.txt"
    val file = File(absolutePath)

    if (file.exists()) {
        // 对文件执行操作
        println("File exists at $absolutePath")
    } else {
        println("File not found at $absolutePath")
    }
}
```

参考资料

- [Improper Limitation of a Pathname to a Restricted Directory](#)

潜在 应用数据不安全存储

应用数据存储不安全，导致其他应用可读取或篡改该数据。

移动应用中的不安全存储漏洞是指一种安全缺陷：敏感数据（如用户凭据、个人信息或其他机密数据）被存储在设备的全局可读位置，从而可能被设备上的其他应用或具有物理访问权限的恶意攻击者未经授权访问。

不安全存储不仅发生在应用向全局可读位置写入数据时，也可能发生在应用从全局可写位置加载配置文件等数据的情况下，恶意攻击者可借此修改配置文件并篡改应用功能。

KOTLIN

```
import android.os.Bundle
import android.os.Environment
import android.widget.Button
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import java.io.File
import java.io.FileOutputStream
import java.io.IOException
import java.io.OutputStreamWriter

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val loginButton: Button = findViewById(R.id.login_button)
        loginButton.setOnClickListener {
            // Dummy authentication process
            val token = authenticate("username", "password") // authentication logic

            // Store token in public storage
            storeTokenInPublicStorage(token)
        }
    }

    private fun storeTokenInPublicStorage(token: String) {
        val filePath = "/sdcard/insecure_app/jwt_config.txt"
        val file = File(filePath)
        try {
            val outputStreamWriter = OutputStreamWriter(FileOutputStream(file))
            outputStreamWriter.write(token)
            outputStreamWriter.close()
        } catch (e: IOException) {
            e.printStackTrace()
        }
    }
}
```

```
}
}
```

SWIFT

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.

        let loginButton = UIButton(type: .system)
        loginButton.setTitle("Login", for: .normal)
        loginButton.addTarget(self, action: #selector(loginButtonTapped), for: .touchUpInside)
        view.addSubview(loginButton)
        loginButton.translatesAutoresizingMaskIntoConstraints = false
        loginButton.centerXAnchor.constraint(equalTo: view.centerXAnchor).isActive = true
        loginButton.centerYAnchor.constraint(equalTo: view.centerYAnchor).isActive = true
    }

    @objc func loginButtonTapped() {
        // Dummy authentication process
        let token = authenticate(username: "username", password: "password")

        // Store token in public storage
        storeTokenInPublicStorage(token: token)
    }

    private func storeTokenInPublicStorage(token: String) {
        let filePath = "/var/mobile/Media/insecure_app/jwt_config.txt"
        let fileURL = URL(fileURLWithPath: filePath)

        do {
            try token.write(to: fileURL, atomically: true, encoding: .utf8)
            print("Token stored successfully.")
        } catch {
            print("Failed to write token: \(error)")
        }
    }
}
```

FLUTTER

```
import 'dart:io';
import 'package:flutter/material.dart';

void main() {
    runApp(MyApp());
}

class MyApp extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            home: MyHomePage(),
        );
    }
}

class MyHomePage extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: Text('Flutter Insecure Storage'),
            ),
            body: Center(
                child: ElevatedButton(
                    onPressed: () {
                        // Dummy authentication process
                        String token = authenticate("username", "password");

                        // Store token in public storage
                    }
                )
            )
        );
    }
}
```

```

        storeTokenInPublicStorage(token);
    },
    child: Text('Login'),
  ),
);
}

void storeTokenInPublicStorage(String token) {
  final directory = Directory('/sdcard/insecure_app/');
  directory.createSync(recursive: true);
  final file = File('${directory.path}/jwt_config.txt');
  try {
    file.writeAsStringSync(token);
    print('Token stored successfully.');
```

```

  } catch (e) {
    print('Failed to write token: $e');
  }
}
}
}
```

技术详情

发现 #1

Call to filesystem access method using path from external storage (e.i `getExternalStorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.File.<init>(java.lang.String)`. Taint trace:

```

at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this,this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

```
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
    }
```

```

        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #2

Call to filesystem access method using path from external storage (e.i `getExternalStorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.File.<init>(java.lang.String)`. Taint trace:

```

at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

```
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
    }
```

```

        android.widget.Toast.makeText(this,this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

发现 #3

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.File.<init>(java.lang.String)`. Taint trace:

```

at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()`:

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this,this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()`:

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());

```

```

        v0_1.append( Password -);
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

    java.io.IOException v0_2 = v0_1.toString();
    android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
    kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
    java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
    android.widget.Toast v2_4 = new StringBuilder();
    v2_4.append(userinfo );
    v2_4.append(v1_10);
    System.out.println(v2_4.toString());
    v1_10.setReadable(1);
    v1_10.setWritable(1);
    java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
    v3_5.write(v0_2);
    v3_5.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (java.io.IOException v0_3) {
    v0_3.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #4

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity` . From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.File.<init>(java.lang.String)` . Taint trace:

```

at java.io.File.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(

Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password -);
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

    java.io.IOException v0_2 = v0_1.toString();
    android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
    kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());

```

```

        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

发现 #5

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```

at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

```
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

发现 #6

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity` . From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)` . Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

```
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );
        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

发现 #7

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity` . From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)` . Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(
```

```

);

java.io.IOException v0_2 = v0_1.toString();
android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
android.widget.Toast v2_4 = new StringBuilder();
v2_4.append(userinfo );
v2_4.append(v1_10);
System.out.println(v2_4.toString());
v1_10.setReadable(1);
v1_10.setWritable(1);
java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
v3_5.write(v0_2);
v3_5.close();
android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (java.io.IOException v0_3) {
    v0_3.printStackTrace();
}
return;
}
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

```
}
```

发现 #8

Call to filesystem access method using path from external storage (e.i `getexternalstorage()`) Taint traced from non-reachable component:

`owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File`
`android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(
);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(
);
```

```

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

建议

- 使用安全存储：安全存储敏感数据（如认证令牌或凭据）。请使用平台提供的安全存储机制，例如iOS的Keychain以及Android的加密SharedPreferences。
- 数据加密：使用强加密算法对静态敏感数据进行加密，以保护存储在用户设备上的数据。
- 避免硬编码敏感信息：请勿将用户名、密码或令牌等敏感信息直接硬编码到代码中，建议采用上述安全存储机制。
- 避免过度宽松的文件权限：确保合理设置文件权限，以限制对敏感文件和目录的访问。例如，在Android中，创建文件时应避免使用 `MODE_WORLD_READABLE` 或 `MODE_WORLD_WRITEABLE`，因为这些模式会向所有应用开放广泛的读取或写入权限。设置文件权限时，始终遵循最小权限原则，仅授予应用功能所必需的访问权限。
- 输入验证：从公共存储位置加载数据时，务必对其进行验证和清理。

KOTLIN

```

import android.content.Context
import android.os.Bundle
import android.widget.Button
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.security.crypto.EncryptedSharedPreferences
import androidx.security.crypto.MasterKeys
import java.io.IOException

class MainActivity : AppCompatActivity() {

    private lateinit var encryptedSharedPreferences: SharedPreferences

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // 初始化加密 SharedPreferences
        initializeEncryptedSharedPreferences()

        val loginButton: Button = findViewById(R.id.login_button)
        loginButton.setOnClickListener {
            // 模拟身份验证流程
            val token = authenticate("username", "password") // 身份验证逻辑

            // 将令牌安全存储到加密 SharedPreferences 中
            storeTokenInSharedPreferences(token)
        }
    }

    private fun initializeEncryptedSharedPreferences() {
        try {
            val masterKeyAlias = MasterKeys.getOrCreate(MasterKeys.AES256_GCM_SPEC)
            encryptedSharedPreferences = EncryptedSharedPreferences.create(
                "secure_preferences",
                masterKeyAlias,
                applicationContext,
                EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
                EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
            )
        } catch (e: IOException) {
            e.printStackTrace()
        }
    }
}

```

```

    }

    private fun storeTokenInSharedPreferences(token: String) {
        encryptedSharedPreferences.edit().putString("jwt_token", token).apply()
        Toast.makeText(this, "Token stored securely.", Toast.LENGTH_SHORT).show()
    }
}

```

SWIFT

```

import UIKit
import Security

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // 在视图加载完成后进行其他设置。

        let loginButton = UIButton(type: .system)
        loginButton.setTitle("Login", for: .normal)
        loginButton.addTarget(self, action: #selector(loginButtonTapped), for: .touchUpInside)
        view.addSubview(loginButton)
        loginButton.translatesAutoresizingMaskIntoConstraints = false
        loginButton.centerXAnchor.constraint(equalTo: view.centerXAnchor).isActive = true
        loginButton.centerYAnchor.constraint(equalTo: view.centerYAnchor).isActive = true
    }

    @objc func loginButtonTapped() {
        // 模拟身份验证流程
        let token = authenticate(username: "username", password: "password")

        // 将令牌安全存储到 Keychain 中
        storeTokenInKeychain(token: token)
    }

    private func storeTokenInKeychain(token: String) {
        let keychainQuery: [String: Any] = [
            kSecClass as String: kSecClassGenericPassword,
            kSecAttrAccount as String: "jwtToken",
            kSecValueData as String: token.data(using: .utf8)!,
            kSecAttrAccessible as String: kSecAttrAccessibleWhenUnlockedThisDeviceOnly
        ]

        let status = SecItemAdd(keychainQuery as CFDictionary, nil)
        if status == errSecSuccess {
            print("Token stored securely in Keychain.")
        } else {
            print("Failed to store token in Keychain: \(status)")
        }
    }
}

```

FLUTTER

```

import 'package:flutter/material.dart';
import 'package:flutter_secure_storage/flutter_secure_storage.dart';

void main() {
    runApp(MyApp());
}

class MyApp extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            home: MyHomePage(),
        );
    }
}

class MyHomePage extends StatelessWidget {
    final storage = FlutterSecureStorage();

    @override
    Widget build(BuildContext context) {

```

```

return Scaffold(
  appBar: AppBar(
    title: Text('Flutter Secure Storage Example'),
  ),
  body: Center(
    child: ElevatedButton(
      onPressed: () async {
        // 模拟身份验证流程
        String token = await authenticate("username", "password");

        // 将令牌安全存储
        await storeTokenInSecureStorage(token);
      },
      child: Text('Login'),
    ),
  ),
);
}

Future<void> storeTokenInSecureStorage(String token) async {
  await storage.write(key: 'jwtToken', value: token);
  print('Token stored securely.');
```

参考资料

- [CWE-359: Exposure of Private Information \("Privacy Violation"\)](#)
- [OWASP Mobile Top 10](#)
- [CWE-200: Information Exposure](#)

潜在

源点到汇点

危险的 Sink 方法被潜在的用户可控参数或不可信的 Source 方法调用。

Source 方法或用户可控参数被用于调用 Sink 方法。

Source 指不可信数据输入，可能来自不可信用户。Sink 指危险方法，若攻击者能够访问，便可利用其实施攻击。

必须对 Source 和 Sink 进行漏洞审查，例如注入、间接对象引用或未授权数据访问。

技术详情

发现 #1

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.AccessControlIssue1Activity` found at `owasp.sat.agoat.AccessControlIssue1Activity`. From parameter `0` to sink `android.content.SharedPreferences`
`owasp.sat.agoat.AccessControlIssue1Activity.getSharedPreferences(java.lang.String, int)`. Taint trace:

```

at owasp.sat.agoat.AccessControlIssue1Activity.getSharedPreferences()
at owasp.sat.agoat.AccessControlIssue1Activity.createPIN()
at owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick()
at owasp.sat.agoat.AccessControlIssue1Activity.onCreate()
at owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.<init>()
at owasp.sat.agoat.AccessControlIssue1Activity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.AccessControlIssue1Activity.onCreate()
```

Method `owasp.sat.agoat.AccessControlIssue1Activity.createPIN()`:

```

public final boolean createPIN(String p5)
{
    kotlin.jvm.internal.Intrinsics.checkNotNull(p5, pinValue);
    android.content.SharedPreferences$Editor v1_1 = this.getSharedPreferences(pinDetails, 0).edit();
    v1_1.putBoolean(pinSet, 1);
    v1_1.putString(pin, this.hashPIN(p5));
    return v1_1.commit();
}
```

Method `owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p5)
{
    android.widget.EditText v0_2 = ((android.widget.EditText) this.this$0.findViewById(2131165335));
    android.widget.Toast v1_9 = this.this$0;
    kotlin.jvm.internal.Intrinsics.checkNotNullExpressionValue(v0_2, pinValue);
    if (!v1_9.createPIN(v0_2.getText().toString())) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN not created), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) PIN created. Please Login), 1).show();
        this.this$0.verifyPINView();
    }
    return;
}
```

Method `owasp.sat.agoat.AccessControlIssue1Activity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    if (!this.verifyIfPinSet()) {
        this.setContentView(2131296284);
        ((android.widget.Button) this.findViewById(2131165334)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1(this)));
    } else {
        this.verifyPINView();
    }
    return;
}
```

Method `owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1.<init>()` :

```
AccessControlIssue1Activity$onCreate$1(owasp.sat.agoat.AccessControlIssue1Activity p1)
{
    this.this$0 = p1;
    return;
}
```

Method `owasp.sat.agoat.AccessControlIssue1Activity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    if (!this.verifyIfPinSet()) {
        this.setContentView(2131296284);
        ((android.widget.Button) this.findViewById(2131165334)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1(this)));
    } else {
        this.verifyPINView();
    }
    return;
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
```

```

        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}

```

Method `owasp.sat.agoat.AccessControlIssue1Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    if (!this.verifyIfPinSet()) {
        this.setContentView(2131296284);
        ((android.widget.Button) this.findViewById(2131165334)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControlIssue1Activity$onCreate$1(this)));
    } else {
        this.verifyPINView();
    }
    return;
}

```

发现 #2

User controlled parameter is used call a sink method. Taint traced from reachable component: `owasp.sat.agoat.AccessControl1ViewActivity` found at `owasp.sat.agoat.AccessControl1ViewActivity` . From parameter `0` to sink `void owasp.sat.agoat.AccessControl1ViewActivity.startService()` . Taint trace:

```

at owasp.sat.agoat.AccessControl1ViewActivity.startService()
at owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1.onClick()
at owasp.sat.agoat.AccessControl1ViewActivity.onCreate()
at owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1.<init>()
at owasp.sat.agoat.AccessControl1ViewActivity.onCreate()
at android.support.v4.app.BaseFragmentActivityApi16.onCreate()
at android.support.v4.app.FragmentActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.AccessControl1ViewActivity.onCreate()

```

Method `owasp.sat.agoat.AccessControl1ViewActivity.startService()` :

```

public final void startService()
{
    this.startService(new android.content.Intent(((android.content.Context) this), owasp.sat.agoat.DownloadInvoiceService));
    return;
}

```

Method `owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p2)
{
    this.this$0.startService();
    return;
}

```

Method `owasp.sat.agoat.AccessControl1ViewActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296283);
}

```

```

        ((android.widget.Button) this.findViewById(2131165257)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1(this)));
        return;
    }

```

Method `owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1.<init>()` :

```

AccessControl1ViewActivity$onCreate$1(owasp.sat.agoat.AccessControl1ViewActivity p1)
{
    this.this$0 = p1;
    return;
}

```

Method `owasp.sat.agoat.AccessControl1ViewActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296283);
    ((android.widget.Button) this.findViewById(2131165257)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1(this)));
    return;
}

```

Method `android.support.v4.app.FragmentActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p9)
{
    int[] v1_0 = 0;
    this.mFragments.attachHost(0);
    super.onCreate(p9);
    android.support.v4.app.FragmentActivity$NonConfigurationInstances v0_2 =
((android.support.v4.app.FragmentActivity$NonConfigurationInstances) this.getLastNonConfigurationInstance());
    if (v0_2 != null) {
        this.mFragments.restoreLoaderNonConfig(v0_2.loaders);
    }
    if (p9 != null) {
        android.os.Parcelable vtmp2 = p9.getParcelable(android.support:fragments);
        if (v0_2 != null) {
            v1_0 = v0_2.fragments;
        }
        this.mFragments.restoreAllState(vtmp2, v1_0);
        if (p9.containsKey(android.support:next_request_index)) {
            this.mNextCandidateRequestIndex = p9.getInt(android.support:next_request_index);
            int[] v1_4 = p9.getIntArray(android.support:request_indicies);
            String[] v3_4 = p9.getStringArray(android.support:request_fragment_who);
            if ((v1_4 != null) && ((v3_4 != null) && (v1_4.length == v3_4.length))) {
                this.mPendingFragmentActivityResults = new android.support.v4.util.SparseArrayCompat(v1_4.length);
                int v4_3 = 0;
                while (v4_3 < v1_4.length) {
                    this.mPendingFragmentActivityResults.put(v1_4[v4_3], v3_4[v4_3]);
                    v4_3++;
                }
            } else {
                android.util.Log.w(FragmentActivity, Invalid requestCode mapping in savedInstanceState.);
            }
        }
    }
    if (this.mPendingFragmentActivityResults == null) {
        this.mPendingFragmentActivityResults = new android.support.v4.util.SparseArrayCompat();
        this.mNextCandidateRequestIndex = 0;
    }
    this.mFragments.dispatchCreate();
    return;
}

```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build.VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.AccessControl1ViewActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this setContentView(2131296283);
    ((android.widget.Button) this.findViewById(2131165257)).setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.AccessControl1ViewActivity$onCreate$1(this)));
    return;
}
```

发现 #3

Input source is used to call sink method Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity` .From source `java.io.File android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)` . Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(
);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
    }
}
```

```

        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this,this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this,this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();

```

```

        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

发现 #4

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From parameter `0` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```

at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at java.io.FileWriter.write()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
}
```

```

    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this setContentView(2131296296);
    ((android.widget.Button this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText this.findViewById(2131165369)),
((android.widget.EditText this.findViewById(2131165303))))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.<init>()` :

```

InsecureStorageSDCardActivity$onCreate$1(owasp.sat.agoat.InsecureStorageSDCardActivity p1, android.widget.EditText p2,
android.widget.EditText p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

发现 #5

Input source is used to call sink method Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSDCardActivity` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```
at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(
);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
    }
}
```

```

        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

);

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296296);
    ((android.widget.Button) this.findViewById(2131165201)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #6

Input source is used to call sink method Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1` found at `owasp.sat.agoat.InsecureStorageSDCardActivity`. From source `java.io.File android.os.Environment.getExternalStorageDirectory()` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```

at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()

```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSDCardActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p7)
{
    try {
        java.io.IOException v0_1 = new StringBuilder();
        v0_1.append( This data is stored in SdCard on );
        v0_1.append(new java.util.Date());
        v0_1.append(:
Username - );
        java.io.File v1_0 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_0, username);
        v0_1.append(v1_0.getText().toString());
        v0_1.append( Password - );
        java.io.File v1_4 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, password);
        v0_1.append(v1_4.getText().toString());
        v0_1.append(

    );

        java.io.IOException v0_2 = v0_1.toString();
        android.content.Context v4_0 = android.os.Environment.getExternalStorageDirectory();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_0, Environment.getExternalStorageDirectory());
        java.io.File v1_10 = java.io.File.createTempFile(users, tmp, new java.io.File(v4_0.getAbsolutePath()));
        android.widget.Toast v2_4 = new StringBuilder();
        v2_4.append(userinfo );
        v2_4.append(v1_10);
        System.out.println(v2_4.toString());
        v1_10.setReadable(1);
        v1_10.setWritable(1);
        java.io.FileWriter v3_5 = new java.io.FileWriter(v1_10);
        v3_5.write(v0_2);
        v3_5.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (java.io.IOException v0_3) {
        v0_3.printStackTrace();
    }
}
```

```

    }
    return;
}

```

发现 #7

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageTempActivity` found at `owasp.sat.agoat.InsecureStorageTempActivity`. From parameter `0` to sink `void java.io.FileWriter.<init>(java.io.File)`. Taint trace:

```

at java.io.FileWriter.<init>()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at java.io.FileWriter.write()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()
at owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()
at owasp.sat.agoat.InsecureStorageTempActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(

    );

        v3_1.write(v4_1.toString());
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(password is );
        String v5_5 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_5, password);
        v4_4.append(v5_5.getText().toString());
        v4_4.append(

    );

        v3_1.write(v4_4.toString());
        v3_1.close();
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    } catch (Exception v0_5) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
        v0_5.printStackTrace();
    }
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(

    );

```

```

        v3_1.write(v4_1.toString());
        CharSequence v4_4 = new StringBuilder();
        v4_4.append(password is );
        String v5_5 = this.$password;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_5, password);
        v4_4.append(v5_5.getText().toString());
        v4_4.append(

    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(

    );

    v3_1.write(v4_1.toString());
    CharSequence v4_4 = new StringBuilder();
    v4_4.append(password is );
    String v5_5 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_5, password);
    v4_4.append(v5_5.getText().toString());
    v4_4.append(

    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    try {
        android.content.Context v2_0 = java.io.File.createTempFile(users, tmp, new java.io.File(this.this$0.getApplicationInfo().dataDir));
        v2_0.setReadable(1);
        v2_0.setWritable(1);
        CharSequence v3_1 = new java.io.FileWriter(v2_0);
        CharSequence v4_1 = new StringBuilder();
        v4_1.append(username is );
        String v5_1 = this.$username;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_1, username);
        v4_1.append(v5_1.getText().toString());
        v4_1.append(

    );

    v3_1.write(v4_1.toString());
    CharSequence v4_4 = new StringBuilder();
    v4_4.append(password is );
    String v5_5 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v5_5, password);

```

```

        v4_4.append(v5_5.getText().toString());
        v4_4.append(
    );

    v3_1.write(v4_4.toString());
    v3_1.close();
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
} catch (Exception v0_5) {
    android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    v0_5.printStackTrace();
}
return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1.<init>()` :

```

InsecureStorageTempActivity$onCreate$1(owasp.sat.agoat.InsecureStorageTempActivity p1, android.widget.EditText p2, android.widget.EditText
p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageTempActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296300);
    ((android.widget.Button) this.findViewById(2131165210)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageTempActivity$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}

```

发现 #8

User controlled parameter is used call a sink method. Taint traced from non-reachable component:

`owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity` found at `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity`. From parameter `0` to sink `void java.io.InputStreamReader.<init>(java.io.InputStream)`. Taint trace:

```
at java.io.InputStreamReader.<init>()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at java.io.InputStreamReader.<init>()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.<init>()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_1, Runtime.getRuntime().exec(ip1, null, null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_1, Runtime.getRuntime().exec(ip1, null, null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    }
}
```

```

        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_1, Runtime.getRuntime().exec(ip1,null,null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_1, Runtime.getRuntime().exec(ip1,null,null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
    }
}

```

```

        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.<init>()` :

```

InputValidationsOSCMDInjectionMain2Activity$onCreate$1(android.widget.EditText p1, android.widget.TextView p2)
{
    this.$ip = p1;
    this.$result = p2;
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}

```

User controlled parameter is used call a sink method. Taint traced from non-reachable component:

`owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity` found at `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity`. From parameter `0` to sink `void java.io.InputStreamReader.<init>(java.io.InputStream)`. Taint trace:

```
at java.io.InputStreamReader.<init>()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.<init>()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
at owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_1, Runtime.getRuntime().exec(ip1, null, null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_1, Runtime.getRuntime().exec(ip1, null, null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
}
```

```

    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.onClick()` :

```

public final void onClick(android.view.View p8)
{
    String v0_1 = new StringBuilder();
    v0_1.append(ping );
    StringBuffer v1_4 = this.$ip;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, ip);
    v0_1.append(v1_4.getText().toString());
    String v0_2 = v0_1.toString();
    StringBuffer v1_2 = new StringBuffer();
    try {
        android.widget.TextView v2_1 = Runtime.getRuntime().exec(v0_2, 0, 0);
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_1, Runtime.getRuntime().exec(ip1, null, null));
        v2_1.waitFor();
        String v4_3 = new java.io.BufferedReader(((java.io.Reader) new java.io.InputStreamReader(v2_1.getInputStream()))).readLine();
        kotlin.jvm.internal.Intrinsics.checkNotNull(v4_3, reader.readLine());
    } catch (android.widget.TextView v2_2) {
        v2_2.printStackTrace();
        android.widget.TextView v2_3 = this.$result;
        kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
        v2_3.setText(((CharSequence) v1_2.toString()));
        return;
    }
    while (v4_3 != null) {
        String v5_3 = new StringBuilder();
        v5_3.append(v4_3);
        v5_3.append(n);
        v1_2.append(v5_3.toString());
    }
    v2_3 = this.$result;
    kotlin.jvm.internal.Intrinsics.checkNotNull(v2_3, result);
    v2_3.setText(((CharSequence) v1_2.toString()));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
    owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
    ((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1.<init>()` :

```

InputValidationsOSCMDInjectionMain2Activity$onCreate$1(android.widget.EditText p1, android.widget.TextView p2)
{
    this.$ip = p1;
    this.$result = p2;
    return;
}

```

```
}
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}
```

Method `owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296292);
    ((android.widget.Button) this.findViewById(2131165317)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsOSCMDInjectionMain2Activity$onCreate$1(((android.widget.EditText) this.findViewById(2131165279)),
((android.widget.TextView) this.findViewById(2131165311)))));
    return;
}
```

发现 #10

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InputValidationsActivity` found at

`owasp.sat.agoat.InputValidationsActivity` . From parameter `0` to sink `void`

`owasp.sat.agoat.InputValidationsActivity.startActivity(android.content.Intent)` . Taint trace:

```
at owasp.sat.agoat.InputValidationsActivity.startActivity()
at owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()
at owasp.sat.agoat.InputValidationsActivity.onCreate()
at owasp.sat.agoat.InputValidationsActivity$onCreate$2.<init>()
at owasp.sat.agoat.InputValidationsActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InputValidationsActivity.onCreate()
```

Method `owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()` :

```
public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0), owasp.sat.agoat.SQLInjectionActivity));
    return;
}
```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
    ((android.widget.Button)
```

```

this._$findCachedViewById(owasp.sat.agoat.R$id.wbUrl)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.oscmdi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsActivity$onCreate$2.<init>()` :

```

InputValidationsActivity$onCreate$2(owasp.sat.agoat.InputValidationsActivity p1)
{
    this.this$0 = p1;
    return;
}

```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.wbUrl)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.oscmdi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
    return;
}

```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivity$Delegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}

```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
}

```

```

        ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
        ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.wbUrl1)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
        ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.oscmdi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
        return;
    }

```

发现 #11

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InputValidationsActivity` found at `owasp.sat.agoat.InputValidationsActivity`. From parameter `0` to sink `void` `owasp.sat.agoat.InputValidationsActivity.startActivity(android.content.Intent)`. Taint trace:

```

at owasp.sat.agoat.InputValidationsActivity.startActivity()
at owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()
at android.content.Intent.<init>()
at owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()
at owasp.sat.agoat.InputValidationsActivity.onCreate()
at owasp.sat.agoat.InputValidationsActivity$onCreate$2.<init>()
at owasp.sat.agoat.InputValidationsActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InputValidationsActivity.onCreate()

```

Method `owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()` :

```

public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0), owasp.sat.agoat.SQLInjectionActivity));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsActivity$onCreate$2.onClick()` :

```

public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0), owasp.sat.agoat.SQLInjectionActivity));
    return;
}

```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.wbUrl1)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.oscmdi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
    return;
}

```

```
}
```

Method `owasp.sat.agoat.InputValidationsActivity$onCreate$2.<init>()` :

```
InputValidationsActivity$onCreate$2(owasp.sat.agoat.InputValidationsActivity p1)
{
    this.this$0 = p1;
    return;
}
```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.wbUrl)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.oscmidi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
    return;
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.InputValidationsActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296291);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.xssBtn)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.sqliButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.wbUrl)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$3(this)));
}
```

```

        ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.oscmdi)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsActivity$onCreate$4(this)));
        return;
    }

```

发现 #12

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageActivity` found at

`owasp.sat.agoat.InsecureStorageActivity`. From parameter `0` to sink `void`

`owasp.sat.agoat.InsecureStorageActivity.startActivity(android.content.Intent)`. Taint trace:

```

at owasp.sat.agoat.InsecureStorageActivity.startActivity()
at owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()
at owasp.sat.agoat.InsecureStorageActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()

```

Method `owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()`:

```

public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.InsecureStorageSharedPrefs));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()`:

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296295);
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$findCachedViewById(owasp.sat.agoat.R$id.SDCardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageActivity$onCreate$1.<init>()`:

```

InsecureStorageActivity$onCreate$1(owasp.sat.agoat.InsecureStorageActivity p1)
{
    this.this$0 = p1;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()`:

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this setContentView(2131296295);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SDcardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}

```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this setContentView(2131296295);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SDcardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}

```

发现 #13

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageActivity` found at `owasp.sat.agoat.InsecureStorageActivity` . From parameter `0` to sink `void`

owasp.sat.agoat.InsecureStorageActivity.startActivity(android.content.Intent) .Tainttrace:

```
at owasp.sat.agoat.InsecureStorageActivity.startActivity()
at owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()
at android.content.Intent.<init>()
at owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()
at owasp.sat.agoat.InsecureStorageActivity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InsecureStorageActivity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.InsecureStorageSharedPrefs));
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageActivity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p5)
{
    this.this$0.startActivity(new android.content.Intent(((android.content.Context) this.this$0),
owasp.sat.agoat.InsecureStorageSharedPrefs));
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296295);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SDcardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageActivity$onCreate$1.<init>()` :

```
InsecureStorageActivity$onCreate$1(owasp.sat.agoat.InsecureStorageActivity p1)
{
    this.this$0 = p1;
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this setContentView(2131296295);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SDcardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this setContentView(2131296295);
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SPButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$1(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SP1Button)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$2(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SQLButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$3(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.tempButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$4(this)));
    ((android.widget.Button)
this._$_findCachedViewById(owasp.sat.agoat.R$id.SDcardButton)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageActivity$onCreate$5(this)));
    return;
}
```

发现 #14

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity` found at `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity`. From parameter `0` to sink `android.content.SharedPreferences` `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.getSharedPreferences(java.lang.String, int)`. Taint trace:

```
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.getSharedPreferences()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.getScoreFromSP()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.getScoreFromSP()` :

```
public final int getScoreFromSP()
{
    int v0 = 0;
    int v1 = 1;
    android.content.SharedPreferences v4 = this.getSharedPreferences(score, 0);
    if ((v4.getInt(score, 0) != 0) && (v4.getInt(level, 0) != 0)) {
        v0 = v4.getInt(score, 0);
        v1 = v4.getInt(level, 0);
    }
    String v2_1 = new StringBuilder();
    v2_1.append(Score is );
    v2_1.append(v0);
    v2_1.append( and Level is);
    v2_1.append(v1);
    System.out.println(v2_1.toString());
    return v0;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p8)
{
    int v1 = 1;
    int v2_1 = this.this$0.CalculateScore(this.this$0.getScoreFromSP());
    if (!this.this$0.CheckScore(v2_1)) {
        owasp.sat.agoat.InsecureStorageSharedPrefs1Activity v3_1 = this.this$0.getApplicationContext();
        CharSequence v5_1 = new StringBuilder();
        v5_1.append(Score is );
        v5_1.append(v2_1);
        v5_1.append(. Try hard);
        android.widget.Toast.makeText(v3_1, ((CharSequence) v5_1.toString()), 0).show();
    } else {
        v1 = 2;
        owasp.sat.agoat.InsecureStorageSharedPrefs1Activity v3_3 = this.$ScoreButton;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v3_3, ScoreButton);
        v3_3.setClickable(0);
        owasp.sat.agoat.InsecureStorageSharedPrefs1Activity v3_4 = this.$ScoreButton;
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v3_4, ScoreButton);
        v3_4.setText(((CharSequence) You won the game));
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Success), 1).show();
    }
    this.this$0.DisplayScore(v2_1, v1);
    this.this$0.UpdateScoreInSP(v2_1, v1);
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()` :

```
protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296298);
}
```

```

        android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165264));
        v0_3.setOnClickListener(((android.view.View$OnClickListener) new owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1(this,
v0_3)));
        return;
    }

```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1.<init>()` :

```

InsecureStorageSharedPrefs1Activity$onCreate$1(owasp.sat.agoat.InsecureStorageSharedPrefs1Activity p1, android.widget.Button p2)
{
    this.this$0 = p1;
    this.$ScoreButton = p2;
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296298);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165264));
    v0_3.setOnClickListener(((android.view.View$OnClickListener) new owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1(this,
v0_3)));
    return;
}

```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```

protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivity$Delegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}

```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs1Activity.onCreate()` :

```

protected void onCreate(android.os.Bundle p3)
{
    super.onCreate(p3);
    this.setContentView(2131296298);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165264));
    v0_3.setOnClickListener(((android.view.View$OnClickListener) new owasp.sat.agoat.InsecureStorageSharedPrefs1Activity$onCreate$1(this,
v0_3)));
    return;
}

```

发现 #15

User controlled parameter is used call a sink method. Taint traced from non-reachable component: `owasp.sat.agoat.InsecureStorageSharedPrefs` found at `owasp.sat.agoat.InsecureStorageSharedPrefs` . From parameter `0` to sink `android.content.SharedPreferences`

owasp.sat.agoat.InsecureStorageSharedPrefs.getSharedPreferences(java.lang.String, int) .Taint trace:

```
at owasp.sat.agoat.InsecureStorageSharedPrefs.getSharedPreferences()
at owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1.onClick()
at owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()
at owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1.<init>()
at owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()
at android.support.v7.app.AppCompatActivity.onCreate()
at owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1.onClick()` :

```
public final void onClick(android.view.View p6)
{
    android.content.SharedPreferences$Editor v1_1 = this.this$0.getSharedPreferences(users, 0).edit();
    android.widget.Toast v2_13 = this.$username;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_13, username);
    v1_1.putString(username, v2_13.getText().toString());
    android.widget.Toast v2_2 = this.$password;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v2_2, password);
    v1_1.putString(password, v2_2.getText().toString());
    if (!v1_1.commit()) {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data not saved), 1).show();
    } else {
        android.widget.Toast.makeText(this.this$0.getApplicationContext(), ((CharSequence) Data saved), 1).show();
    }
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296297);
    ((android.widget.Button) this.findViewById(2131165205)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1.<init>()` :

```
InsecureStorageSharedPrefs$onCreate$1(owasp.sat.agoat.InsecureStorageSharedPrefs p1, android.widget.EditText p2, android.widget.EditText
p3)
{
    this.this$0 = p1;
    this.$username = p2;
    this.$password = p3;
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296297);
    ((android.widget.Button) this.findViewById(2131165205)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369)),
((android.widget.EditText) this.findViewById(2131165303)))));
    return;
}
```

```
}
```

Method `android.support.v7.app.AppCompatActivity.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    android.support.v7.app.AppCompatActivityDelegate v0 = this.getDelegate();
    v0.installViewFactory();
    v0.onCreate(p5);
    if ((v0.applyDayNight()) && (this.mThemeId != 0)) {
        if (android.os.Build$VERSION.SDK_INT < 23) {
            this.setTheme(this.mThemeId);
        } else {
            this.onApplyThemeResource(this.getTheme(), this.mThemeId, 0);
        }
    }
    super.onCreate(p5);
    return;
}
```

Method `owasp.sat.agoat.InsecureStorageSharedPrefs.onCreate()` :

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296297);
    ((android.widget.Button) this.findViewById(2131165205)).setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InsecureStorageSharedPrefs$onCreate$1(this, ((android.widget.EditText) this.findViewById(2131165369))),
((android.widget.EditText) this.findViewById(2131165303))));
    return;
}
```

建议

修复建议因已识别漏洞的类别而异。

潜在 已声明但未配置权限的服务

CVSS v3: 3.4

已声明的服务未受全局权限保护。

`service` 是一种应用组件，可在后台执行操作而无需用户交互。`service` 也可用于向其他应用暴露功能，这对应于调用 `Context.bindService()` 建立连接并进行交互。

未受保护的 `service` 可能被其他应用调用，进而访问敏感信息或执行特权操作。

技术详情

发现 #1

Service exported without permissions.

```
<service xmlns:android="http://schemas.android.com/apk/res/android" android:name="owasp.sat.agoat.DownloadInvoiceService"
android:enabled="true" android:exported="true"/>
```

建议

`service` 可以向外部组件暴露多个方法。可以使用 `checkPermission` 方法为每个方法定义任意权限。

也可以在 manifest 的 `<service>` 标签中强制声明权限，从而隔离服务并限制访问。

</> XML

```
<permission android:name="co.ostorlab.custom_permission" android:label="custom_permission"
            android:protectionLevel="dangerous"></permission>

<service android:name="co.ostorlab.custom_service" android:permission="co.ostorlab.custom_permission">
<intent-filter>
    <action android:name="co.ostorlab.ACTION"/>
</intent-filter>
</service>
```

该服务可以通过调用 `checkCallingPermission` 方法，在执行具体 IPC 调用的实现前 对其强制执行权限检查。

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:L/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:N

攻击向量：

🌐 网络 (N)

🏠 邻近 (A)

📍 本地 (L)

👤 物理 (P)

攻击复杂度：

🔑 低 (L)

🔒 高 (H)

所需权限：

👤 无 (N)

👤 低 (L)

👤 高 (H)

用户交互：

👤 无 (N)

👤 需要 (R)

范围：

🏠 已更改 (C)

🏠 未更改 (U)

机密性：

🔒 高 (H)

🔒 低 (L)

🔒 无 (N)

完整性：

🔒 高 (H)

🔒 低 (L)

🔒 无 (N)

可用性：

🔒 高 (H)

🔒 低 (L)

🔒 无 (N)

参考资料

- [Service \(Android Developer Documentation\)](#)
- [CWE-280: Improper Handling of Insufficient Permissions or Privileges](#)
- [Security Decisions Via Untrusted Inputs \(OWASP Mobile Top 10\)](#)

潜在

URL 篡改

攻击者若能控制应用程序用于获取内容的 URL，则会带来安全风险，因其可操纵内容，进而导致恶意代码执行或敏感信息窃取。

攻击者一旦控制应用获取内容时所使用的 URL，就会引入严重的安全风险。通过操控该 URL，攻击者可将应用重定向至恶意服务器，或向响应中注入其控制的内容。这可能导致多种安全漏洞及潜在后果。例如：

- 代码执行：攻击者可能篡改 URL，使其指向自己所控制的服务器，从而向应用投递恶意代码。这些代码可能利用应用中的漏洞执行任意命令，或在用户系统上安装恶意软件。
- 数据窃取：通过将应用重定向到欺诈服务器，攻击者可诱骗用户输入敏感信息，例如登录凭据、信用卡详情或个人数据，随后将其截获并滥用。
- 内容篡改：攻击者可以修改应用获取到的内容，篡改显示的信息或注入恶意脚本。这可能导致多种后果，例如展示误导性信息、篡改网页外观，或发起钓鱼攻击。
- 供应链攻击：如果应用从外部来源（如库或插件）获取内容，控制 URL 可使攻击者将合法资源替换为遭入侵或恶意的版本，从而危及整个系统的安全。

以下是一段应用获取攻击者可控输入的示例代码：

DART

```
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

class MainActivity extends StatefulWidget {
  final String url;

  MainActivity({required this.url});

  @override
  _MainActivityState createState() => _MainActivityState();
}

class _MainActivityState extends State<MainActivity> {
```

受限•Ostorlab

页码: 173 / 184

```

String? content;

@override
void initState() {
  super.initState();
  fetchContent(widget.url);
}

Future<void> fetchContent(String url) async {
  try {
    final response = await http.get(Uri.parse(url));

    if (response.statusCode == 200) {
      setState(() {
        content = response.body;
      });
      // TODO: 在此处理获取到的内容
    } else {
      // 处理错误响应
    }
  } catch (e) {
    // 处理网络或 API 调用错误
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Fetching URL Content'),
    ),
    body: Center(
      child: content != null
        ? Text(content!)
        : CircularProgressIndicator(),
    ),
  );
}

```

SWIFT

```

import UIKit

class AppDelegate: UIResponder, UIApplicationDelegate {

  var window: UIWindow?


  func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    // 如果应用通过 URL 启动, 则处理深度链接 URL
    if let url = launchOptions?["url"] as? URL {
      handleDeepLink(url)
    }
    return true
  }

  func application(_ application: UIApplication, open url: URL, options: [UIApplication.OpenURLOptionsKey : Any] = [:]) -> Bool {
    // 当应用已在运行时处理深度链接 URL
    handleDeepLink(url)
    return true
  }

  private func handleDeepLink(_ url: URL) {
    // 检查 URL scheme 是否为 "ostorlab"
    if url.scheme == "ostorlab" {
      // 从深度链接中提取 URL
      if let deepLinkURL = URL(string: url.absoluteString.replacingOccurrences(of: "ostorlab://", with: "")) {
        // 从提供的 URL 获取内容
        fetchContent(deepLinkURL)
      }
    }
  }

  private func fetchContent(_ url: URL) {
    URLSession.shared.dataTask(with: url) { (data, response, error) in
      if let error = error {
        print("Error fetching content: \(error.localizedDescription)")
      }
    }
  }
}

```

```

        return
    }
    if let data = data, let content = String(data: data, encoding: .utf8) {
        // 在此处理获取到的内容
        print("Fetched content: \(content)")
    }
    }.resume()
}
}

```

KOTLIN

```

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.GlobalScope
import kotlinx.coroutines.launch
import java.io.BufferedReader
import java.io.InputStreamReader
import java.net.HttpURLConnection
import java.net.URL

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // 从 intent 中获取 URL
        val intent: Intent = intent
        val url: String? = intent.getStringExtra("url")

        if (url != null) {
            // 从提供的 URL 获取内容
            GlobalScope.launch(Dispatchers.IO) {
                val content: String? = fetchContent(url)
                // 根据需要处理获取到的内容
                if (content != null) {
                    // TODO: 在此处理获取到的内容
                }
            }
        }
    }

    private fun fetchContent(urlString: String): String? {
        var connection: HttpURLConnection? = null
        var reader: BufferedReader? = null

        try {
            val url = URL(urlString)
            connection = url.openConnection() as HttpURLConnection
            connection.requestMethod = "GET"

            // 检查响应码是否成功
            if (connection.responseCode == HttpURLConnection.HTTP_OK) {
                reader = BufferedReader(InputStreamReader(connection.inputStream))
                val response = StringBuilder()
                var line: String?

                // 逐行读取响应
                while (reader.readLine().also { line = it } != null) {
                    response.append(line)
                }

                return response.toString()
            }
        } catch (e: Exception) {
            e.printStackTrace()
        } finally {
            // 关闭连接和 reader
            connection?.disconnect()
            reader?.close()
        }

        return null
    }
}

```

```
}  
}
```

技术详情

发现 #1

The parameter at position: `0` of the method `loadUrl` in the class `WebView` is controlled by an attacker and can be used to retrieve malicious content. Taint traced from non-reachable component: `owasp.sat.agoat.InputValidationsWebViewURLActivity`. The TaintElement is coming from: Class

`InputValidationsWebViewURLActivity` Method `onCreate` Position `0` Signature `(Landroid/os/Bundle;)V`. Call trace:

```
at android.webkit.WebView.loadUrl  
at owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick  
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate  
at owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.<init>  
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate  
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate
```

at `android.webkit.WebView.loadUrl`

Unable to extract source code.

at `owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick`

```
public final void onClick(android.view.View p4)  
{  
    android.webkit.WebSettings v0_1 = this.$webview11.getSettings();  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v0_1, webSettings1);  
    v0_1.setJavaScriptEnabled(1);  
    String v1_4 = this.$url;  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_4, url);  
    this.$webview11.loadUrl(v1_4.getText().toString());  
    return;  
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)  
{  
    super.onCreate(p5);  
    this.setContentView(2131296293);  
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));  
    android.webkit.WebView v1_2 = this.findViewById(2131165374);  
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, findViewById(R.id.webview1));  
    v0_3.setOnClickListener(((android.view.View$OnClickListener) new  
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)  
this.findViewById(2131165367)))));  
    return;  
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.<init>`

```
InputValidationsWebViewURLActivity$onCreate$1(android.webkit.WebView p1, android.widget.TextView p2)  
{  
    this.$webview11 = p1;  
    this.$url = p2;  
    return;  
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)  
{  
    super.onCreate(p5);
```

```

        this.setContentView(2131296293);
        android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));
        android.webkit.WebView v1_2 = this.findViewById(2131165374);
        kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_2, findViewById(R.id.webview1));
        v0_3.setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)
this.findViewById(2131165367))))));
        return;
    }

```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```

protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296293);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));
    android.webkit.WebView v1_2 = this.findViewById(2131165374);
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_2, findViewById(R.id.webview1));
    v0_3.setOnClickListener(((android.view.View$OnClickListener) new
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)
this.findViewById(2131165367))))));
    return;
}

```

发现 #2

The parameter at position: `0` of the method `loadUrl` in the class `WebView` is controlled by an attacker and can be used to retrieve malicious content. Taint traced from non-reachable component: `owasp.sat.agoat.InputValidationsWebViewURLActivity`. The TaintElement is coming from: Class `InputValidationsWebViewURLActivity` Method `onCreate` Position `0` Signature `(Landroid/os/Bundle;)V`. Call trace:

```

at android.webkit.WebView.loadUrl
at owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick
at owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate
at owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.<init>
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate
at owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate

```

at `android.webkit.WebView.loadUrl`

Unable to extract source code.

at `owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p4)
{
    android.webkit.WebSettings v0_1 = this.$webview11.getSettings();
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v0_1, webSettings1);
    v0_1.setJavaScriptEnabled(1);
    String v1_4 = this.$url;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, url);
    this.$webview11.loadUrl(v1_4.getText().toString());
    return;
}

```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.onClick`

```

public final void onClick(android.view.View p4)
{
    android.webkit.WebSettings v0_1 = this.$webview11.getSettings();
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v0_1, webSettings1);
    v0_1.setJavaScriptEnabled(1);
    String v1_4 = this.$url;
    kotlin.jvm.internal.Intrinsics.checkExpressionValueIsNotNull(v1_4, url);
    this.$webview11.loadUrl(v1_4.getText().toString());
}

```

```
return;
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296293);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));
    android.webkit.WebView v1_2 = this.findViewById(2131165374);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, findViewById(R.id.webview1));
    v0_3.setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)
this.findViewById(2131165367))))));
    return;
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1.<init>`

```
InputValidationsWebViewURLActivity$onCreate$1(android.webkit.WebView p1, android.widget.TextView p2)
{
    this.$webview11 = p1;
    this.$url = p2;
    return;
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296293);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));
    android.webkit.WebView v1_2 = this.findViewById(2131165374);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, findViewById(R.id.webview1));
    v0_3.setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)
this.findViewById(2131165367))))));
    return;
}
```

at `owasp.sat.agoat.InputValidationsWebViewURLActivity.onCreate`

```
protected void onCreate(android.os.Bundle p5)
{
    super.onCreate(p5);
    this.setContentView(2131296293);
    android.widget.Button v0_3 = ((android.widget.Button) this.findViewById(2131165288));
    android.webkit.WebView v1_2 = this.findViewById(2131165374);
    kotlin.jvm.internal.Intrinsics.checkNotNull(v1_2, findViewById(R.id.webview1));
    v0_3.setOnClickListener(((android.view.View.OnClickListener) new
owasp.sat.agoat.InputValidationsWebViewURLActivity$onCreate$1(((android.webkit.WebView) v1_2), ((android.widget.TextView)
this.findViewById(2131165367))))));
    return;
}
```

建议

采取主动措施保护应用免受 URL 操纵攻击至关重要。请参考以下建议：

- 输入验证：实施严格的输入验证机制，确保用于获取内容的 URL 格式正确并符合预期模式。可包括校验有效的 URL scheme、强制限定预期域名以及验证查询参数。
- 白名单策略：维护一份可信域名或来源的白名单，应用仅从该名单中的源获取内容。仅允许向这些可信来源发起请求，以降低访问恶意或未授权内容的风险。
- 内容完整性校验：实施校验机制以验证获取内容的完整性。对接收到的内容计算加密哈希或数字签名，并与预期值进行比较，以发现任何修改或篡改行为。

通过实施上述建议，可提升应用安全性、保护用户数据，并降低与 URL 操纵攻击相关的风险。

以下是实现白名单校验的示例代码：

DART

```
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

class MainActivity extends StatefulWidget {
  final String url;
  final List<String> allowedDomains;

  MainActivity({required this.url, required this.allowedDomains});

  @override
  _MainActivityState createState() => _MainActivityState();
}

class _MainActivityState extends State<MainActivity> {
  String? content;

  @override
  void initState() {
    super.initState();
    if (isDomainAllowed(widget.url)) {
      fetchContent(widget.url);
    } else {
      // Handle unauthorized domain
      // TODO: Implement appropriate action for unauthorized domain
    }
  }

  bool isDomainAllowed(String urlString) {
    Uri uri = Uri.parse(urlString);
    String domain = uri.host ?? '';
    return widget.allowedDomains.contains(domain);
  }

  Future<void> fetchContent(String url) async {
    try {
      final response = await http.get(Uri.parse(url));

      if (response.statusCode == 200) {
        setState(() {
          content = response.body;
        });
        // TODO: Process the fetched content here
      } else {
        // Handle error response
      }
    } catch (e) {
      // Handle network or API call error
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Fetching URL Content'),
      ),
      body: Center(
        child: content != null
          ? Text(content!)
          : CircularProgressIndicator(),
      ),
    );
  }
}
```

SWIFT

```
import UIKit

class AppDelegate: UIResponder, UIApplicationDelegate {

  var window: UIWindow?

  let allowedDomains = ["example.com", "trusteddomain.com"] // Add your trusted domain names here
```

```

func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool
{
    // Handle deep link URL if the app was launched from a URL
    if let url = launchOptions?["url"] as? URL {
        handleDeepLink(url)
    }
    return true
}

func application(_ application: UIApplication, open url: URL, options: [UIApplication.OpenURLOptionsKey : Any] = [:]) -> Bool {
    // Handle deep link URL when the app is already running
    handleDeepLink(url)
    return true
}

private func handleDeepLink(_ url: URL) {
    // Check if the URL scheme is "ostorlab"
    if url.scheme == "ostorlab" {
        // Extract the URL from the deep link
        if let deepLinkURL = URL(string: url.absoluteString.replacingOccurrences(of: "ostorlab://", with: "")) {
            // Verify the domain name against the whitelist
            if isDomainAllowed(deepLinkURL) {
                // Fetch the content from the provided URL
                fetchContent(deepLinkURL)
            } else {
                // Handle unauthorized domain
                // TODO: Implement appropriate action for unauthorized domain
            }
        }
    }
}

private func isDomainAllowed(_ url: URL) -> Bool {
    guard let host = url.host else {
        return false
    }
    return allowedDomains.contains(host)
}

private func fetchContent(_ url: URL) {
    URLSession.shared.dataTask(with: url) { (data, response, error) in
        if let error = error {
            print("Error fetching content: \(error.localizedDescription)")
            return
        }
        if let data = data, let content = String(data: data, encoding: .utf8) {
            // Process the fetched content here
            print("Fetched content: \(content)")
        }
    }.resume()
}
}

```

KOTLIN

```

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.GlobalScope
import kotlinx.coroutines.launch
import java.io.BufferedReader
import java.io.InputStreamReader
import java.net.HttpURLConnection
import java.net.URL

class MainActivity : AppCompatActivity() {
    private val allowedDomains = listOf("example.com", "trusteddomain.com") // Add your trusted domain names here

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // Get the URL from the intent
        val intent: Intent = intent
        val url: String? = intent.getStringExtra("url")
    }
}

```

```

if (url != null) {
    // Verify the domain name against the whitelist
    if (isDomainAllowed(url)) {
        // Fetch the content from the provided URL
        GlobalScope.launch(Dispatchers.IO) {
            val content: String? = fetchContent(url)
            // Process the fetched content as needed
            if (content != null) {
                // TODO: Process the fetched content here
            }
        }
    } else {
        // Handle unauthorized domain
        // TODO: Implement appropriate action for unauthorized domain
    }
}

}

private fun isDomainAllowed(urlString: String): Boolean {
    val url = URL(urlString)
    val domain = url.host
    return allowedDomains.contains(domain)
}

private fun fetchContent(urlString: String): String? {
    var connection: HttpURLConnection? = null
    var reader: BufferedReader? = null

    try {
        val url = URL(urlString)
        connection = url.openConnection() as HttpURLConnection
        connection.requestMethod = "GET"

        // Check if the response code is successful
        if (connection.responseCode == HttpURLConnection.HTTP_OK) {
            reader = BufferedReader(InputStreamReader(connection.inputStream))
            val response = StringBuilder()
            var line: String?

            // Read the response line by line
            while (reader.readLine().also { line = it } != null) {
                response.append(line)
            }

            return response.toString()
        }
    } catch (e: Exception) {
        e.printStackTrace()
    } finally {
        // Close the connection and reader
        connection?.disconnect()
        reader?.close()
    }

    return null
}
}

```

加固 不安全的网络配置

CVSS v3: 9.1

应用未指定网络安全配置，或设置了不安全的配置项。

Android 网络安全配置支持以声明方式配置应用的网络安全。

当 Android 网络安全配置缺失或配置不当时，可能导致应用遭受 MITM 攻击。

技术详情

发现 #1

`res/xml/network_security_config.xml`: The `base-config` authorizes cleartext traffic.

建议

Android Network Security Configuration 是一个 XML 文件，能够以声明式方式配置应用的网络安全。

将网络安全配置添加到应用：

1- 在应用的 res/xml 目录中新建一个 XML 文件，将其命名为 `network_security_config.xml`，或其他合适的名称。2- 定义安全配置，参见下方示例 3- 将配置应用到 Manifest：

XML

```
<application
    android:networkSecurityConfig="@xml/network_security_config"
    <!-- Other attributes -->
    >
    <!-- Other configurations -->
</application>
```

网络安全配置示例

- 自定义证书颁发机构（支持仅调试设置）

XML

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <debug-overrides>
    <trust-anchors>
      <certificates src="@raw/debug_cas"/>
    </trust-anchors>
  </debug-overrides>
</network-security-config>
```

- 声明式禁用明文流量

XML

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <domain-config cleartextTrafficPermitted="false">
    <domain includeSubdomains="true">secure.example.com</domain>
  </domain-config>
</network-security-config>
```

- 声明式配置证书固定密钥

XML

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
  <domain-config>
    <domain includeSubdomains="true">example.com</domain>
    <pin-set expiration="2018-01-01">
      <pin digest="SHA-256">7HIpactkIAq2Y49orF00QKurWxmmSFZhBCoQYcRhJ3Y=</pin>
      <!-- backup pin -->
      <pin digest="SHA-256">fwza0LRMXouZHRc8Ei+4PyulDPDcf3UKg0/04cDM1oE=</pin>
    </pin-set>
  </domain-config>
</network-security-config>
```

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

攻击向量：

网络(N)邻近(A)本地(L)物理(P)

攻击复杂度：

低(L)高(H)

所需权限：

无(N)低(L)高(H)

用户交互：

无(N)需要(R)

范围：

已更改(C)未更改(U)

机密性：

高(H)低(L)无(N)

完整性：

高(H)低(L)无(N)

可用性：

高(H)低(L)无(N)

参考资料

- Network Security Configuration(Android developer)

加固 调试模式已启用

CVSSv3: 5.4

应用编译时启用了调试模式。

Android Manifest中定义的 `Application` 的 `android:debuggable` 属性决定了应用是否可被调试。

调试模式允许攻击者访问应用文件系统、附加调试器、访问敏感数据或执行恶意操作。

可通过以下步骤使用 `jdb` 启动调试会话：

- 使用 `adb jdwp` 识别目标应用的 `PID`：

```
$adb jdwp
3466
15446
```

- 使用 `adb` 创建通信通道，并使用 `jdb` 附加到该通道：

```
$adb forward tcp:7777 jdwp:$(adb shell ps | grep "package-id")
$jdb -attach localhost:7777
```

- 访问应用文件系统：

```
$adb shell
$run-as package-id
$...insert malicious action...
```

攻击者无需访问源代码即可调试该应用，并利用此能力代表用户执行恶意操作、修改应用行为或访问敏感数据（如凭据和会话 Cookie）。

技术详情

发现 #1

Debug mode is set to true, enabling attackers to access the application filesystem exposing sensitive data.

```
<application xmlns:android="http://schemas.android.com/apk/res/android" android:theme="@7F0C0005" android:label="@7F0B0036"
android:icon="@7F0A0000" android:debuggable="true" android:allowBackup="true" android:supportsRtl="true"
android:networkSecurityConfig="@7F0E0000" android:roundIcon="@7F0A0002">
```

建议

在 Android Manifest 的 `Application` 标签中，将 `android:debuggeable` 属性设为 false，以禁用调试模式。

```
<application android:icon="@drawable/icon" android:debuggable="false">
```

CVSS向量详情

CVSS V3向量：CVSS:3.0/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N

攻击向量:

网络 (N)

邻近 (A)

本地 (L)

物理 (P)

攻击复杂度:

低 (L)

高 (H)

所需权限:

无 (N)

低 (L)

高 (H)

用户交互:

无 (N)

需要 (R)

范围:

已更改 (C)

未更改 (U)

机密性:

高 (H)

低 (L)

无 (N)

完整性:

高 (H)

低 (L)

无 (N)

可用性:

高 (H)

低 (L)

无 (N)

参考资料

- [DRD10-J Do not release apps that are debuggable \(CERT Secure Coding\)](#)

攻击面