Hacking Android: My Journey in Google Bug Bounty

Alena Skliarova

November 13, 2024

TenguCon

Tokyo, Japan

#whoami

Alena Skliarova

Security researcher

Reverse engineer

Bug hunter

Ghidra SRE contributor

in/askliarova

Nalen98

Agenda

- Introduction to the Google Bug Bounty platform
- Android OS: How to research and find security bugs
- Creating a high-quality report
- Android Security Team resolutions
- Discovered CVEs: technical details and impact


Google Bug Hunters

Where do you want to report your vulnerability?

- ( ) Google VRP – Report security issues affecting any Google-owned or Alphabet (Bet) subsidiary web service that handles reasonably sensitive user data. **If you are not sure where to report, use this VRP**. See program rules
- ( ) Chrome VRP – Report security issues affecting the Chrome browser. See program rules
- ( ) ChromeOS VRP – Report security issues affecting the ChromeOS ecosystem. See program rules
- (•) Android & Devices VRP – Report security issues affecting Pixel, Google Nest, Pixel Watch, and Fitbit devices and their latest operating systems. See program rules
- ( ) Mobile VRP – Report security issues affecting first-party Android applications. See program rules
- ( ) Cloud VRP – Report security issues affecting any Google Cloud product or web service that handles reasonably sensitive user data. See program rules
- ( ) OSS VRP – Report security issues affecting open source software stored in the public repositories of Google-owned GitHub organizations and selected repositories hosted on other platforms. See program rules
- ( ) Abuse VRP – Report security issues that identify abuse-related methodologies. See program rules
- ( ) Chrome Extensions VRP – Report security issues in first-party Chrome extensions. See program rules

OUR MISSION <!--OSS Patch Rewards-->

Patch Rewards

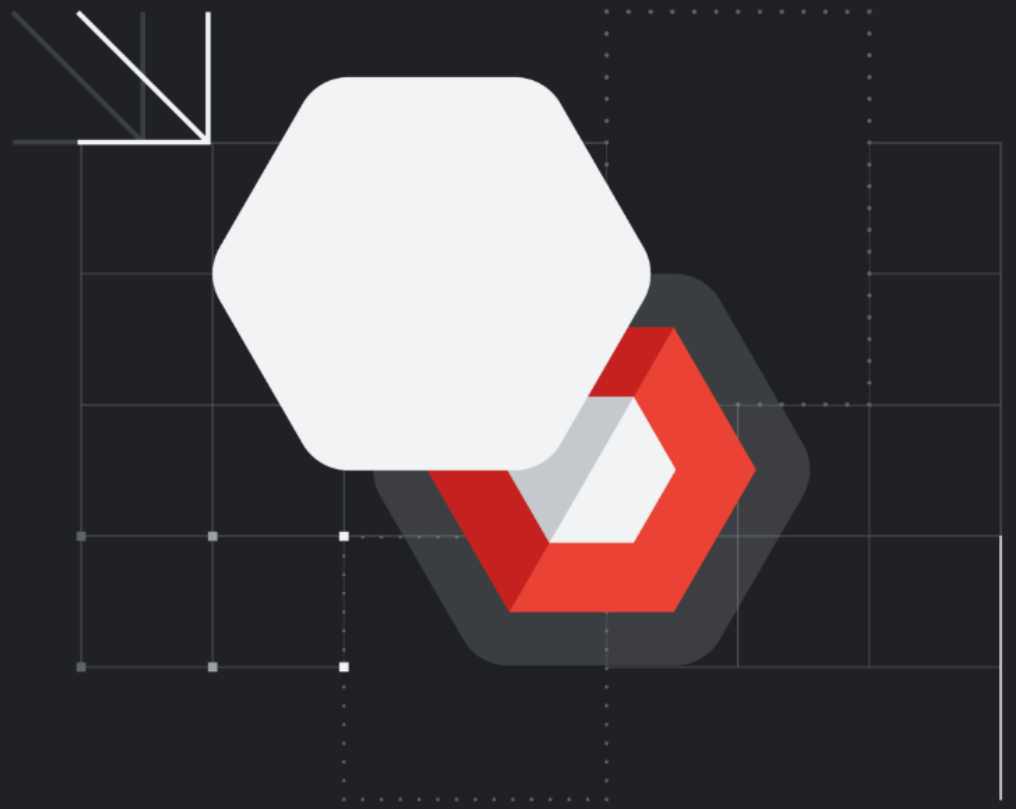
Through the Patch Rewards program, you can claim rewards for proactive improvements you've made to security in open source projects.

Any patch (typically a merged GitHub pull request) that you can demonstrate to have improved the security of an in-scope project will be considered for a reward.

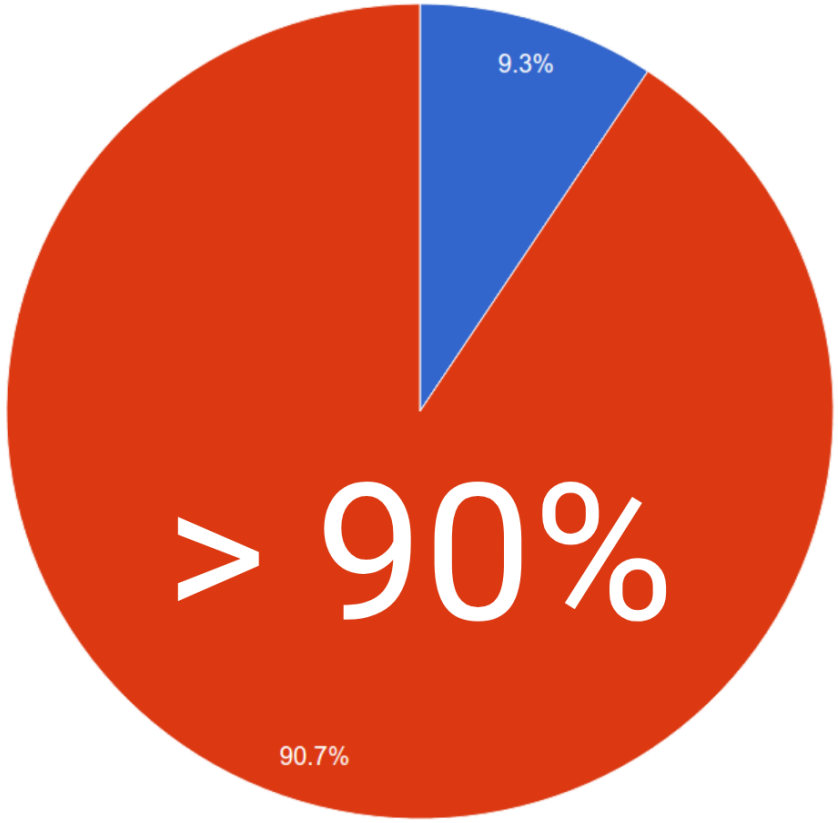
For more details on in-scope projects and qualifying submissions, see the information on this page and the program rules.

SUBMIT PATCH

VIEW RULES



Security Report Statistics



do not result in bugs

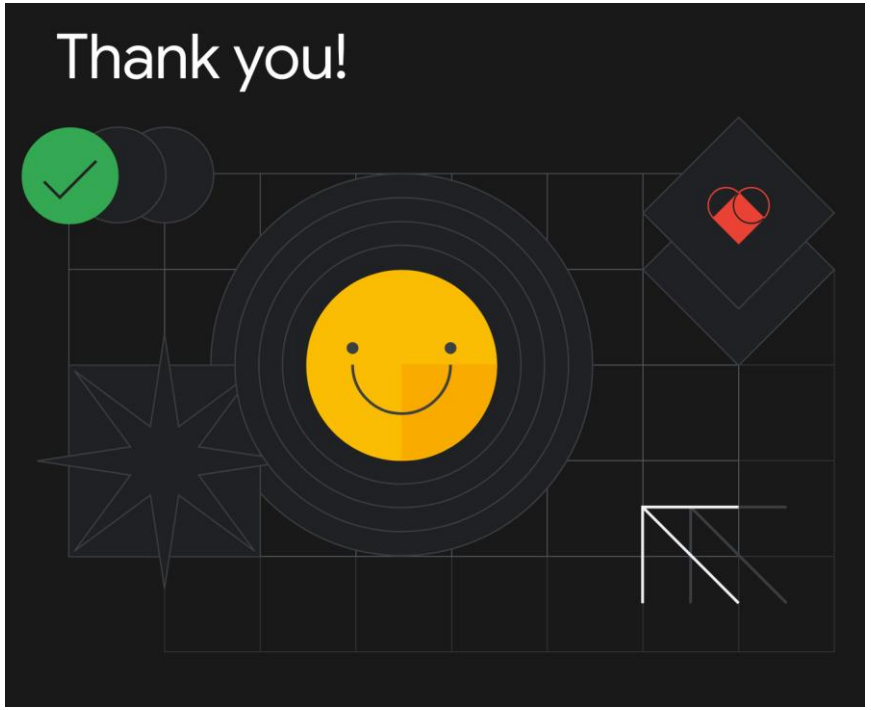
Most popular issues

- foobar@gmail.com receives e-mails to foo.bar@gmail.com
- XSS in translate.googleusercontent.com

Both are invalid :/

«Secrets of Google VRP», Google Security Team

Security Report Statistics



Avoiding invalid reports

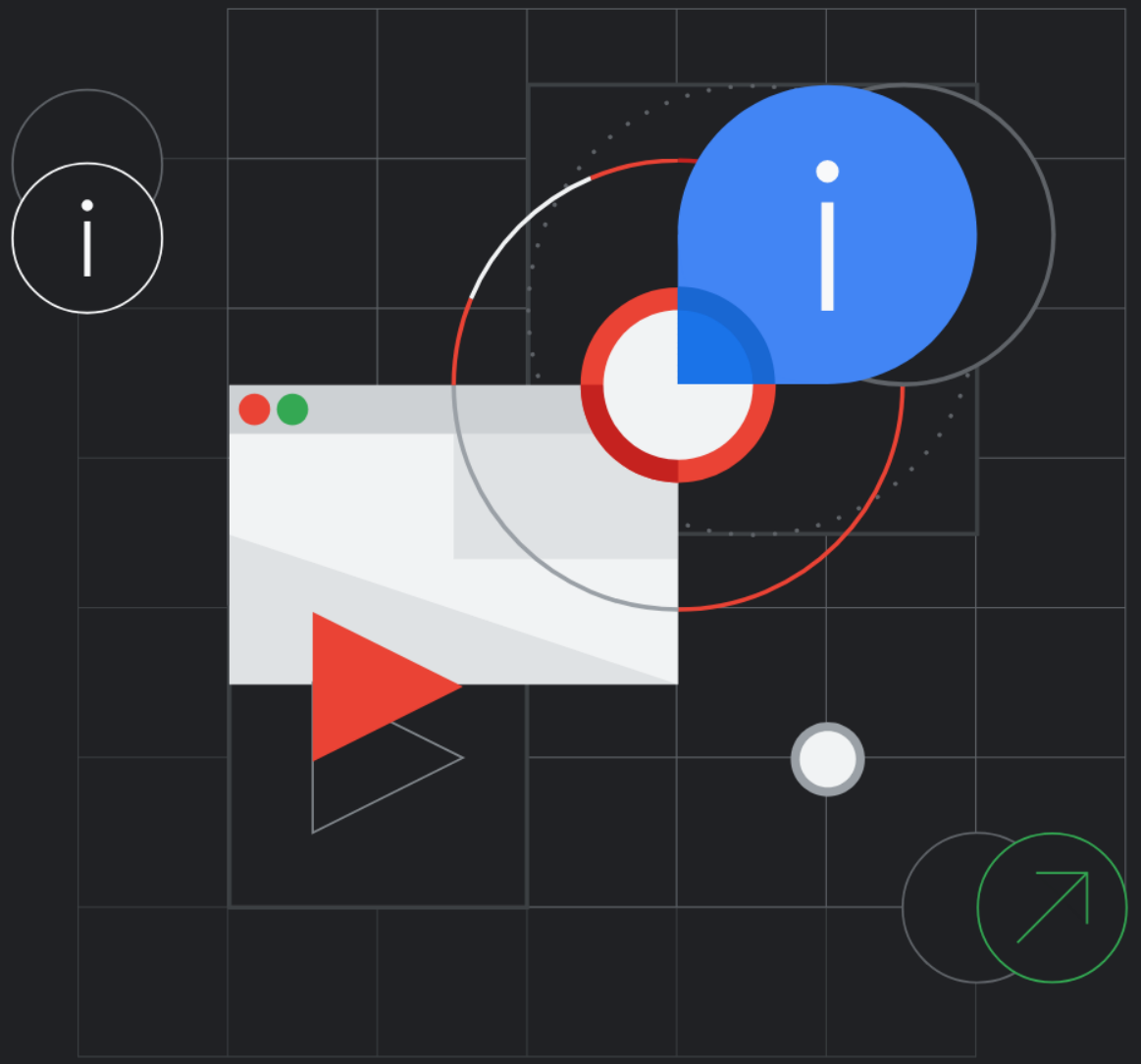
Around 90% of reports we receive describe issues that are not security vulnerabilities, despite looking like one. See our articles at **Learn | Invalid Reports** to learn more.

IMPROVING YOUR REPORTS <!--discoverySection---------------------------->

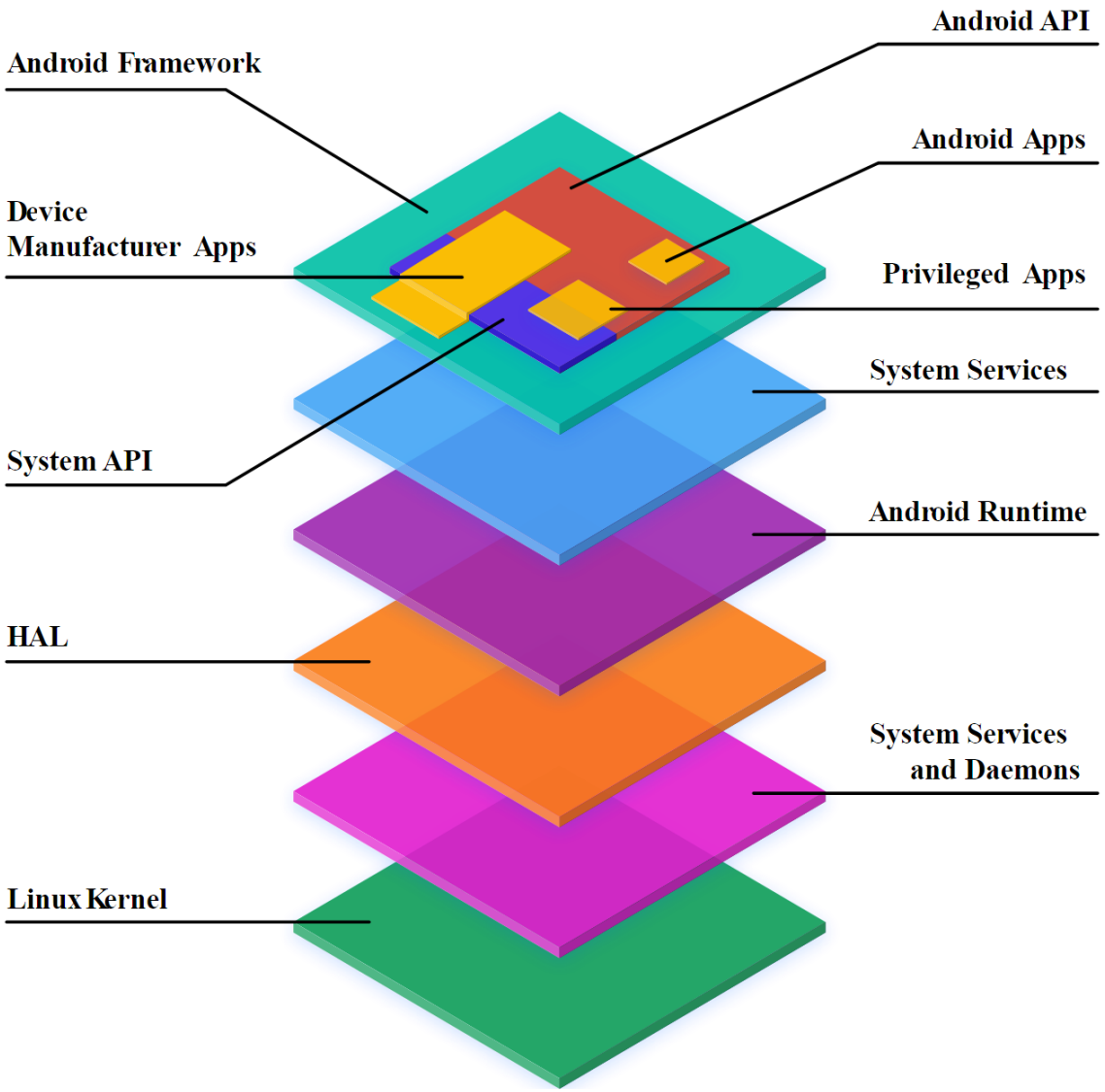
Welcome to Bug Hunter University

Here you'll find all you need to sharpen your ability, whether you're an advanced hunter or just starting out.

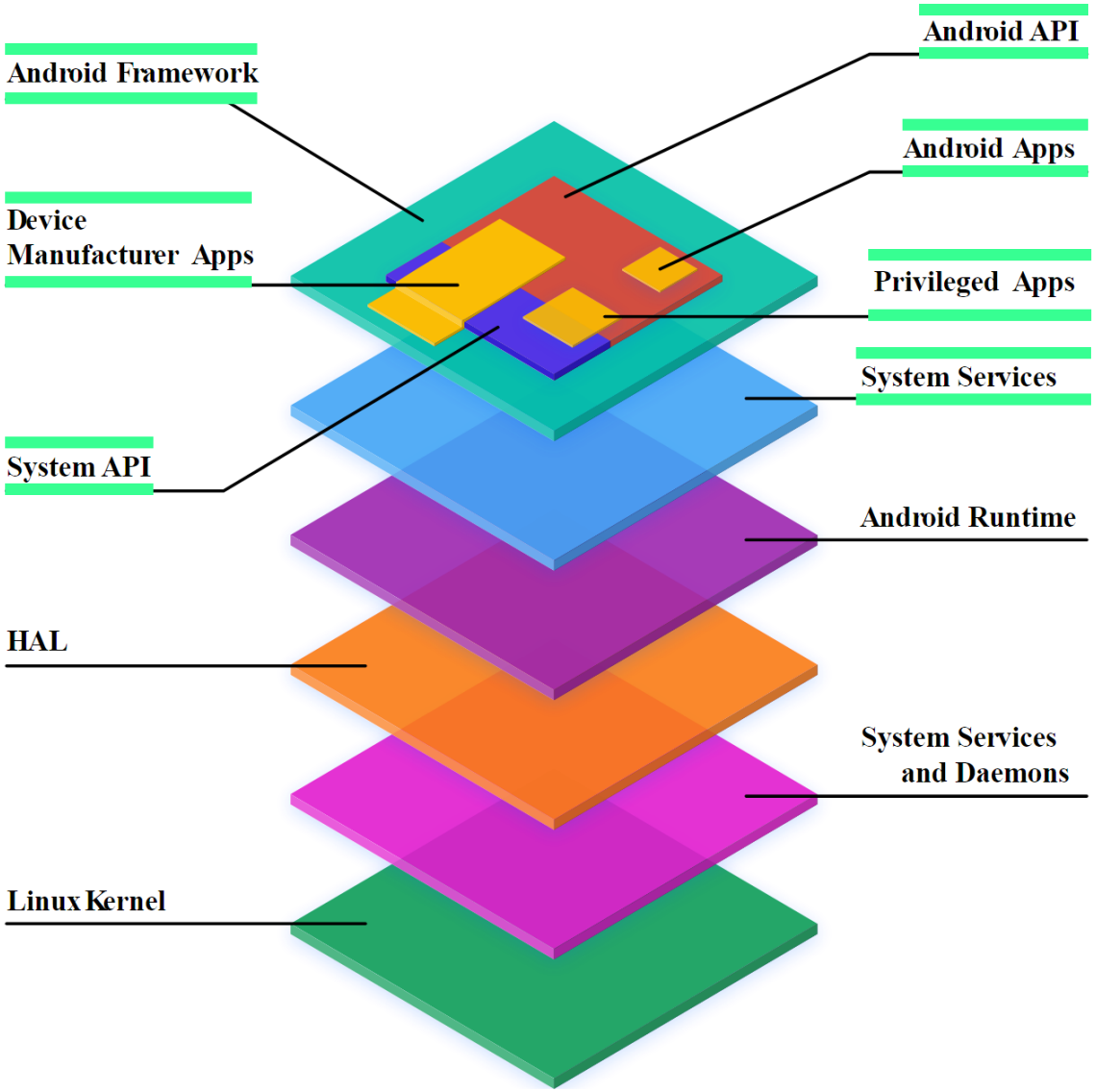
Search

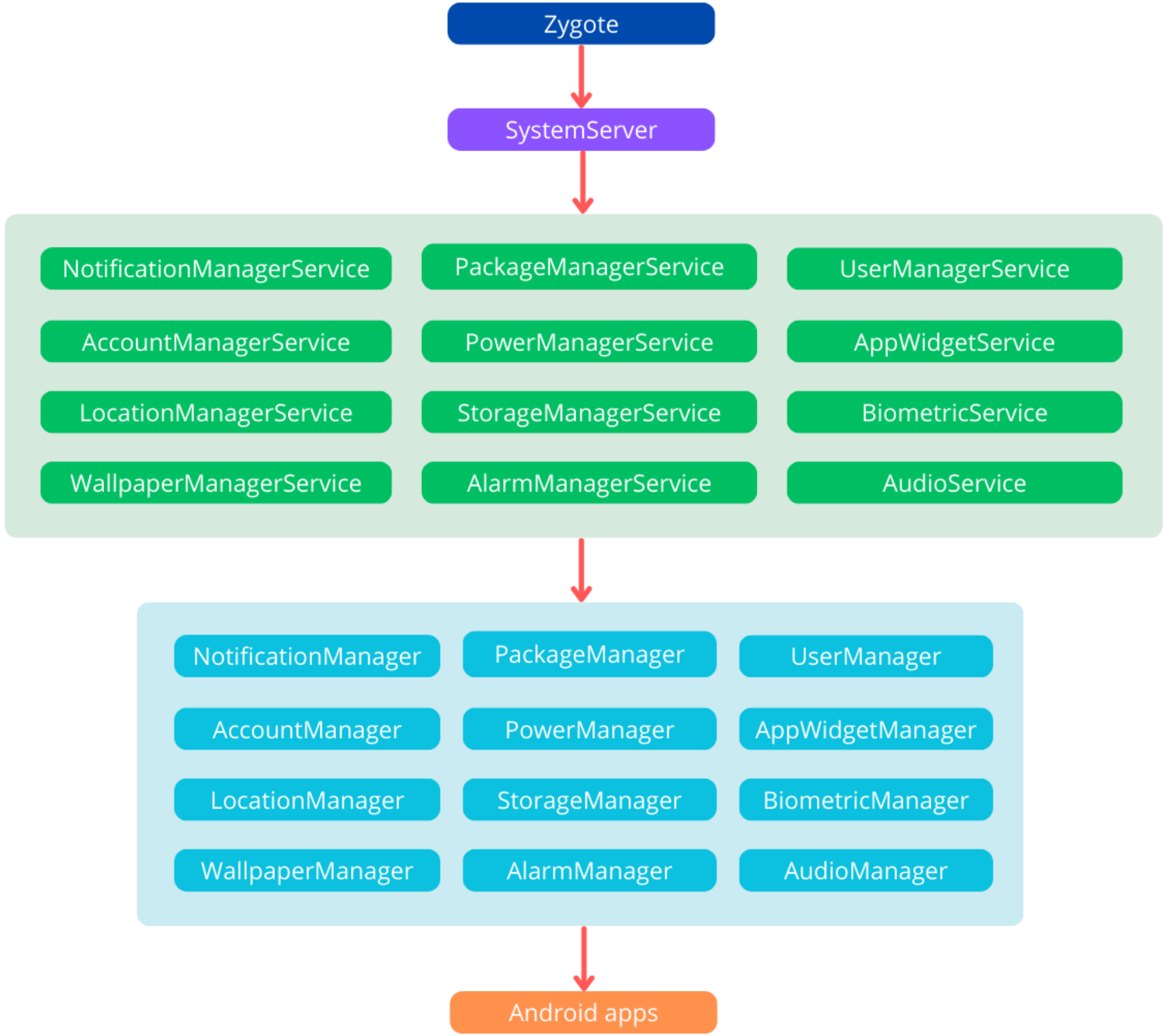


AOSP Architecture



AOSP Architecture




Zygote
SystemServer
NotificationManagerService
PackageManagerService
UserManagerService
AccountManagerService
PowerManagerService
AppWidgetService
LocationManagerService
StorageManagerService
BiometricService
WallpaperManagerService
AlarmManagerService
AudioService
NotificationManager
PackageManager
UserManager
AccountManager
PowerManager
AppWidgetManager
LocationManager
StorageManager
BiometricManager
WallpaperManager
AlarmManager
AudioManager
Android apps

Bug Hunting Strategies

One-to-many: Scan OS for one vulnerability type

VS

Many-to-one: Research a specific OS component

Types of Security Vulnerabilities

RCE	Remote code execution
EoP	Elevation of privilege
ID	Information disclosure
DoS	Denial of service

Maximum Exploit Rewards

Code execution reward amounts

Description	Maximum Reward
Pixel Titan M with Persistence, Zero click	Up to $1,000,000
Pixel Titan M without Persistence, Zero click	Up to $500,000
Local App to Pixel Titan M without Persistence	Up to $300,000
Secure Element	Up to $250,000
Trusted Execution Environment	Up to $250,000
Kernel	Up to $250,000
Privileged Process	Up to $100,000

Bypass reward amounts

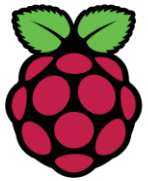
Description	Maximum Reward
Lockscreen bypass [1]	Up to $100,000
Device Policy Controller bypass [2]	Up to $75,000

Data exfiltration reward amounts

Description	Maximum Reward
High value data secured by Pixel Titan M	Up to $500,000
High value data secured by a Secure Element	Up to $250,000

Useful Tools

FRIDA

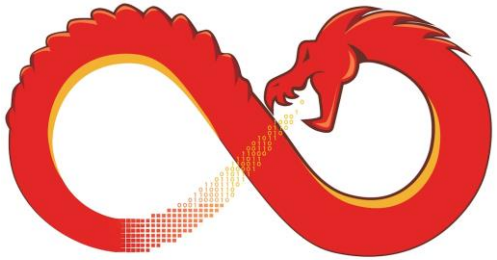


Raspberry Pi

Semgrep

ATEST

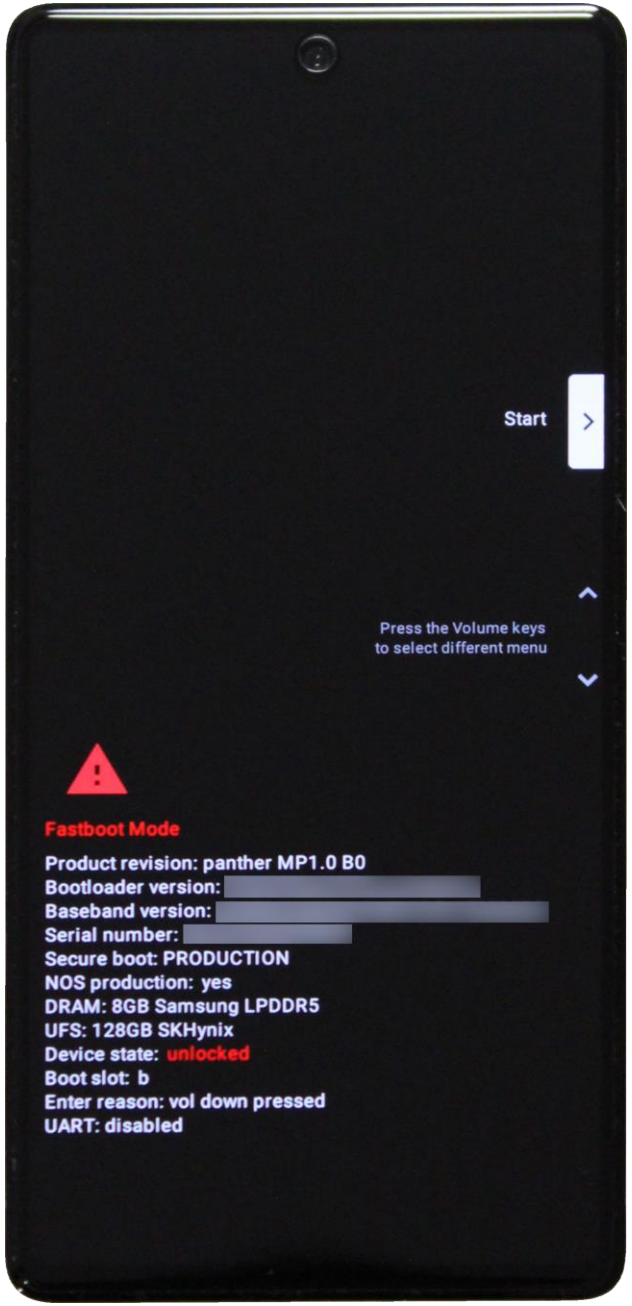
ADB

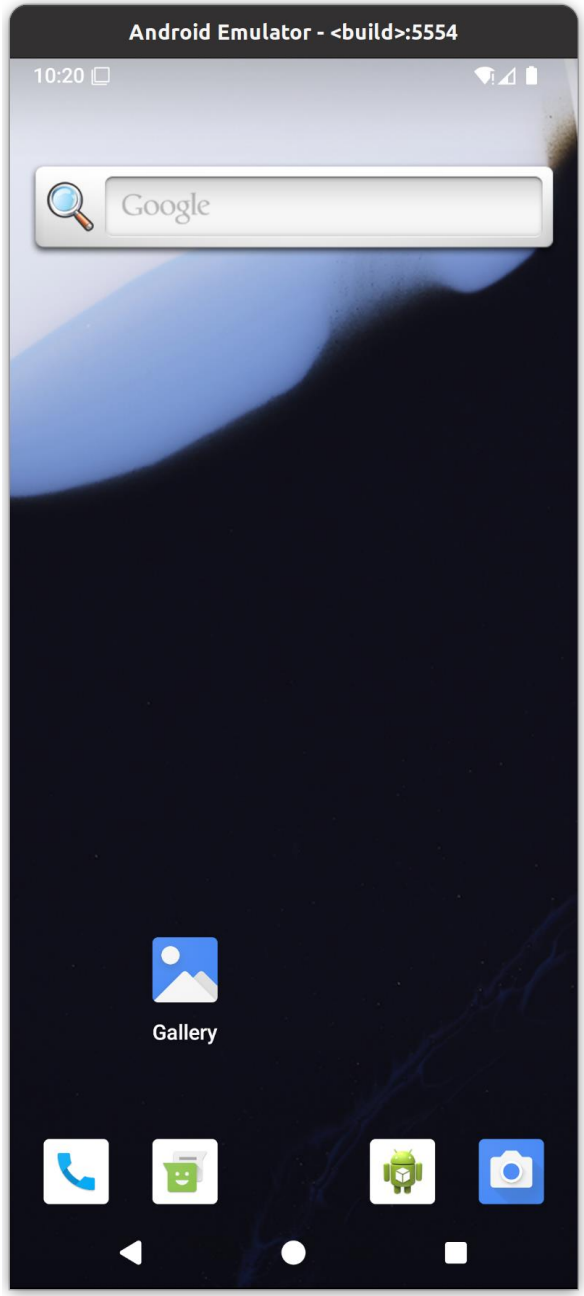


GHIDRA

Atest

```
nale@nale:~/usb/new$ atest FrameworksServicesTests:com.android.server.accounts.AccountManagerServiceTest#testCheckAddAccountLongName

Finding Tests...
Found 'FrameworksServicesTests:com.android.server.accounts.AccountManagerServiceTest#testCheckAddAccountLongName' as CACHE
(Test info has been cached for speeding up the next run, if test info needs to be updated, please add -c to clean the old cache.)

Building Dependencies...
compatibility-tradefed-host, atest_script_help.sh-host, atest-tradefed-host, tradefed-host, atest_tradefed.sh-host, compatibility-host
-util-host, out/target/product/emu64x/module-info.json, aapt-host, adb-host, FrameworksServicesTests-target, aapt2-host

Running Tests...

x86_64 FrameworksServicesTests
------------------------------
com.android.frameworks.servicestests (1 Test)
[1/1] com.android.server.accounts.AccountManagerServiceTest#testCheckAddAccountLongName: PASSED (3.904s)

Summary (Test executed with 1 devices.)
-------
x86_64 FrameworksServicesTests: Passed: 1, Failed: 0, Ignored: 0, Assumption Failed: 0

All tests passed!
```


Start
Press the Volume keys
to select different menu
Fastboot Mode
Product revision: panther MP1.0 B0
Bootloader version:
Baseband version:
Serial number:
Secure boot: PRODUCTION
NOS production: yes
DRAM: 8GB Samsung LPDDR5
UFS: 128GB SKHynix
Device state: unlocked
Boot slot: b
Enter reason: vol down pressed
UART: disabled


Android Emulator - <build>:5554
10:20
Google
Gallery

Invalid Reports

- Non-eligible devices
- Old Android OS builds
- Rooted devices
- Exploitation is only possible on an emulator
- Local device denial of service attack
- PoC application requests excessive permissions
- Hardware attacks (JTAG, soldering)
- Enabled developer mode

Android Severity Assessment Matrix

Severity

The severity of a bug generally reflects the potential harm that could occur if a bug was successfully exploited. Use the following criteria to determine the severity.

Rating	Consequence of successful exploitation
Critical	• Arbitrary code execution in the TEE or SE • Bypass of software mechanisms designed to prevent safety-related software or hardware components from malfunctioning (for example, thermal protections) • Remote access to sensitive credentials used for remote service authentication (for example, account passwords or bearer tokens) • Remote arbitrary code execution within the cellular baseband context with no user interaction (for example, exploiting a bug in the cellular radio service) • Remote arbitrary code execution in a privileged context, the bootloader chain, THB, or the OS Kernel • Remote bypass of user interaction requirements on package installation or equivalent behavior • Remote bypass of user interaction requirements for core developer, security, or privacy settings • Remote persistent denial of service (permanent, requiring reflashing the entire operating system, or a factory reset) • Remote secure boot bypass

Creating a High-Quality Report

Technical Details

The problem

Please describe the technical details of the vulnerability

bold *italics* ~strike~ `code` ```preformatted``` >quote Markdown Cheatsheet

WRITE PREVIEW

1. Brief description of the vulnerability and its security impact.
2. Describe how you discovered the vulnerability in the code, quoting AOSP where applicable.
3. Include stack traces and crash cases if you conducted fuzz testing.
4. Be sure to analyze the malicious impact.
5. Note the probability of successful exploitation and the resources required.
6. Explain how the PoC works.

Reproduction Steps

Please specify the steps to reproduce the issue, including sample code where appropriate. Please be as detailed as possible.

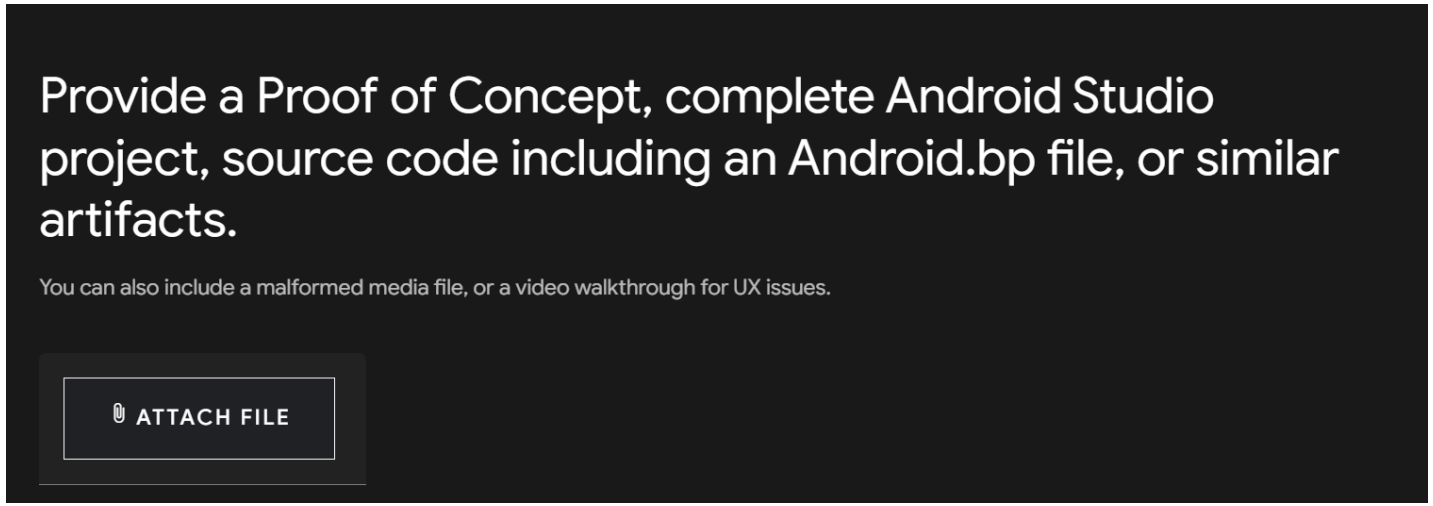
bold *italics* ~strike~ `code` ```preformatted``` >quote Markdown Cheatsheet

WRITE PREVIEW

Also, provide the characteristics of the devices or emulators on which you tested the PoC. Include configurations and describe the environment that needs to be set up.

Record a screen capture video of the device and include it in the report. Make sure to show all the necessary steps to reproduce the vulnerability.

PoC (Proof of Concept)



Key guidelines:

PoC format: Android Studio project, script, executable, payload, malformed file

Reproducibility: Should be reproducible on the latest AOSP build

No pre-compiled APKs: Provide an Android Studio project instead of pre-compiled APK files

Minimize dependencies: Avoid external libraries to ensure PoC acceptance

Attach Security Patch

Alena Skliarova #20 Feb 1, 2023 04:06PM

Hello!

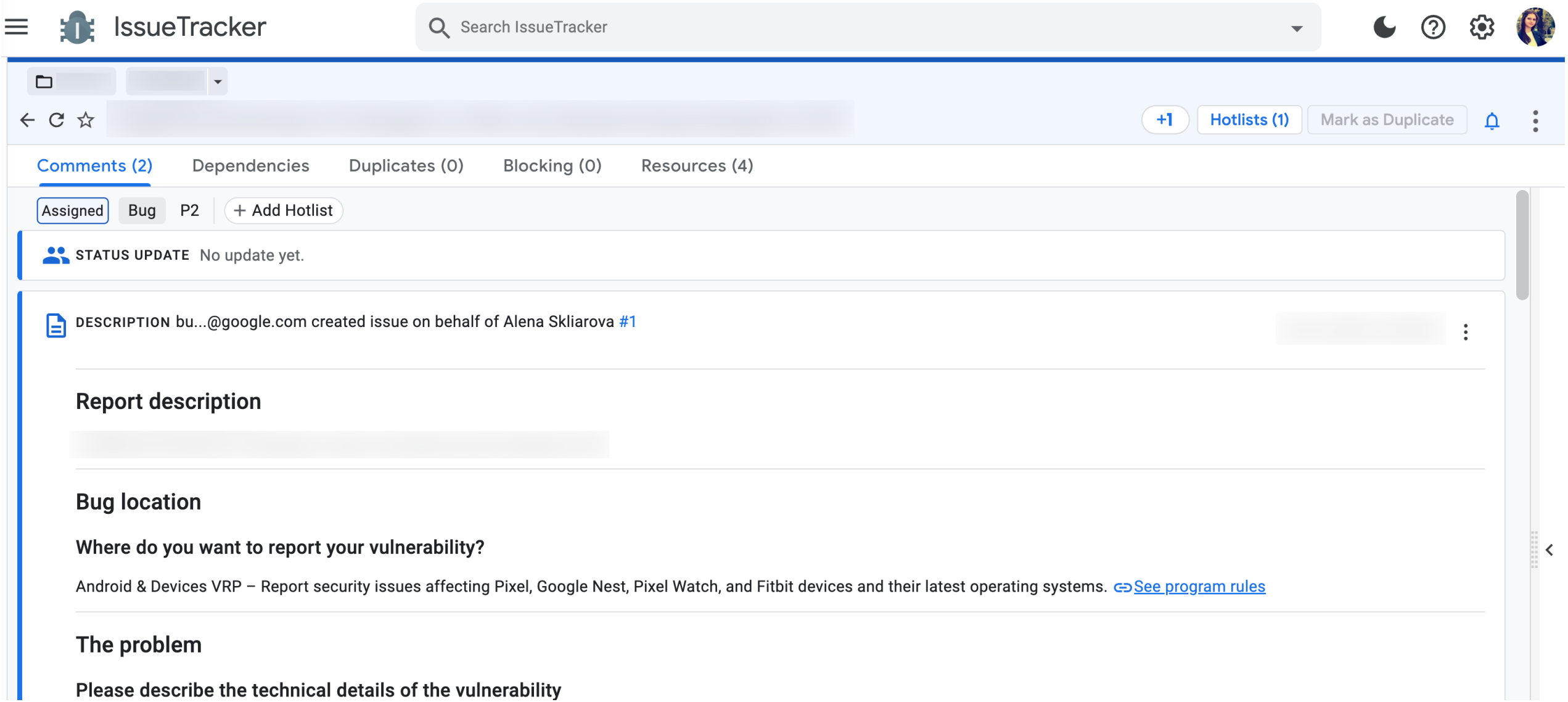
I have attached a patch (file `patch.diff`) to fix the vulnerability described in this issue.

---

patch.diff

721 B View Download

Successfully Submitted Report



Android Security Team Resolutions

Low

Moderate

High

Critical

Severity

Duplicate

NSBC (Not Security Bulletin Class)

Security Bug Lifecycle

Security Bug Lifecycle

Android security acknowledgements

The Android Security Team would like to thank the following people and parties for helping to improve Android security. They have done this either by finding and responsibly reporting security vulnerabilities through the AOSP bug tracker Security bug report template or by committing code that has a positive impact on Android security, including code that qualifies for the Patch Rewards program.

2024

October

Researchers	CVEs
Alena Skliarova (https://www.linkedin.com/in/askliarova)	CVE-2024-40674

Not a Bug / Duplicate

NSBC (Not Security Bulletin Class)

- Won’t fix (Not Reproducible)
- Won’t fix (Intended behavior)
- Won’t fix (Infeasible)
- Won’t fix (Obsolete)

Duplicate – someone submitted the same vulnerability report before you

No CVE for Low and Moderate Bugs

New Android & Google Device Vulnerability Reward Program Initiatives

May 17, 2023

Additionally, starting May 15th, 2023, Android will no longer assign Common Vulnerabilities and Exposures (CVEs) to most moderate severity issues. CVEs will continue to be assigned to critical and high severity vulnerabilities.

source: Google Security Blog

Rating Modifiers

Reason	Effect
Requires running as a privileged context to execute the attack (not applicable to TEE, SE, and hypervisors such as pKVM)	-1 Severity
Vulnerability-specific details limit the impact of the issue	-1 Severity
Biometric authentication bypass that requires biometric information directly from the device owner	-1 Severity
Compiler or platform configurations mitigate a vulnerability in the source code	Moderate Severity if the underlying vulnerability is Moderate or higher
Requires physical access to device internals and is still possible if the device is off or hasn't been unlocked since being powered on	-1 Severity
Requires physical access to device internals while the device is on and has previously been unlocked	-2 Severity

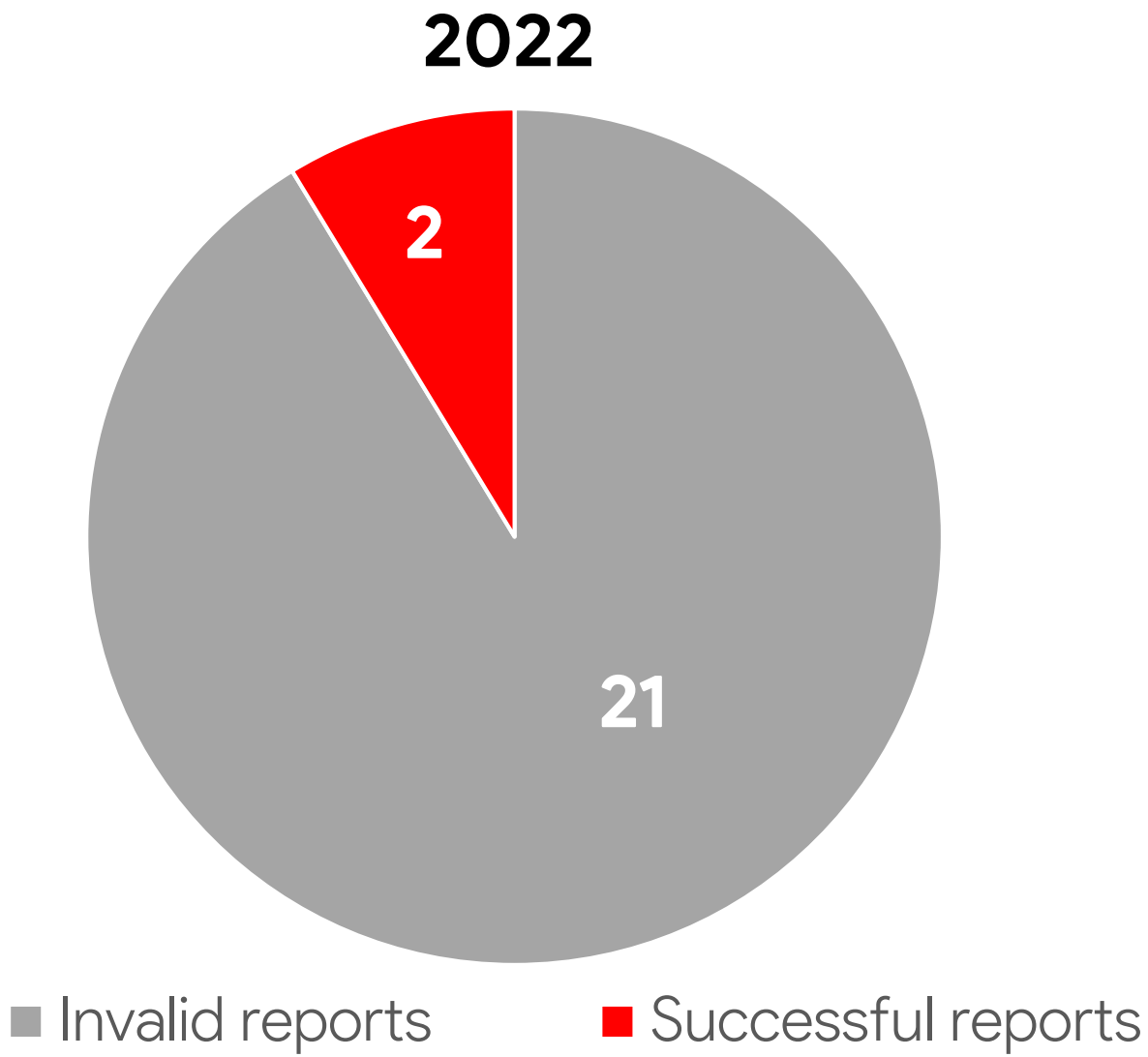
Rating Modifiers in Action

As with [redacted], this is reduced from High severity to Moderate severity because it depends on [redacted], which is only accessible to apps targeting SDK versions that are out of support, or ProfileOwner/DeviceOwner apps.

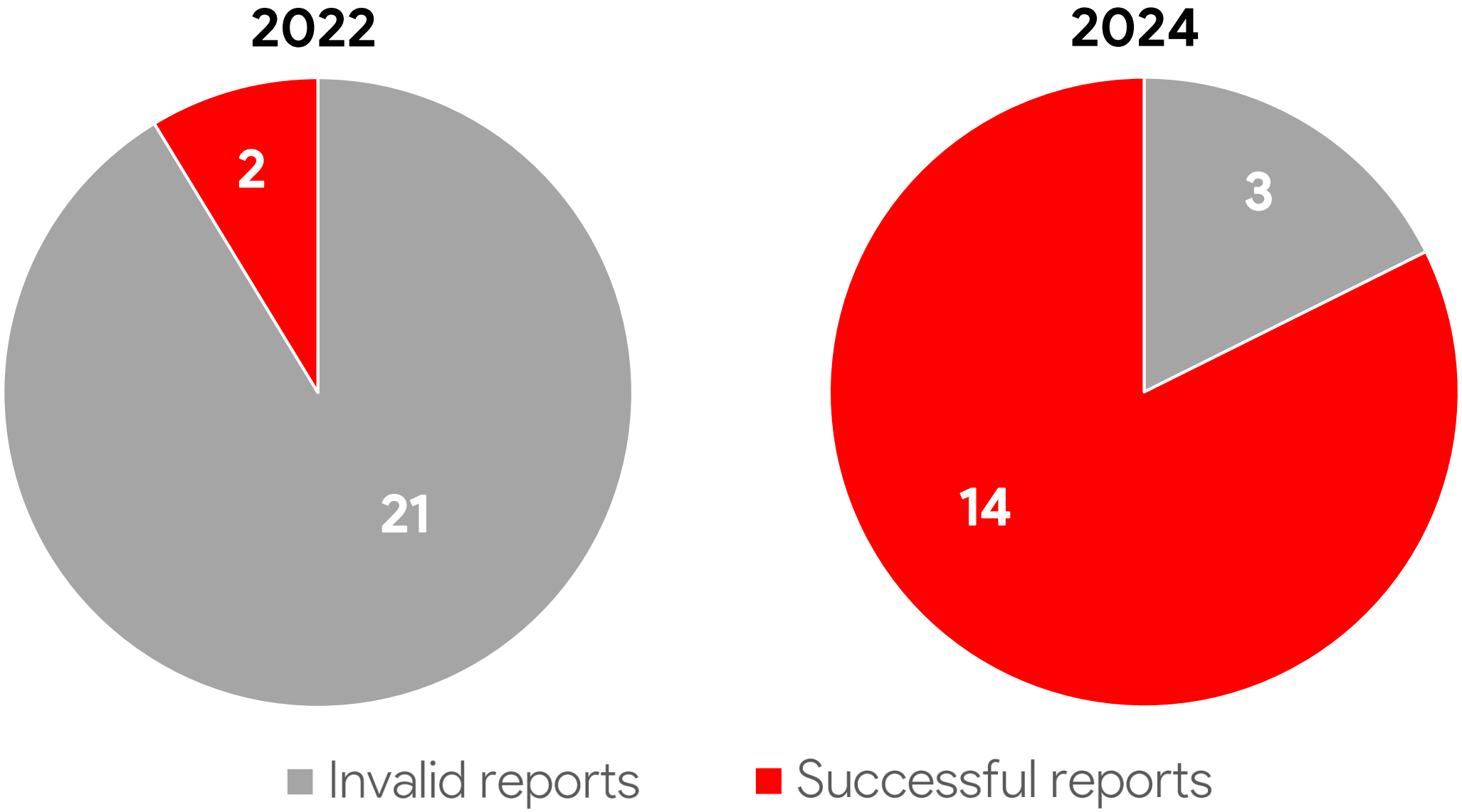
Thank you,
Android Security Team

(1) Severity Matrix: https://source.android.com/security/overview/updates-resources#severity

My Bug Hunting Journey: 2022 vs 2024



My Bug Hunting Journey: 2022 vs 2024



Discovered CVEs

CVE-2023-21143

CVE-2023-21240

CVE-2023-21243

CVE-2023-21252

CVE-2023-21348

CVE-2023-21350

CVE-2024-0026

CVE-2024-0027

CVE-2024-23712

CVE-2024-23713

CVE-2024-40674

CVE-2024-43083

CVE-2023-21348

```
<!-- Allows query of any normal app on the device, regardless of manifest declarations.
    <p>Protection level: normal -->
<permission android:name="android.permission.QUERY_ALL_PACKAGES"
            android:label="@string/permlab_queryAllPackages"
            android:description="@string/permdesc_queryAllPackages"
            android:protectionLevel="normal" />
<uses-permission android:name="android.permission.QUERY_ALL_PACKAGES"/>
```

CVE-2023-21348

Use of the broad package (App) visibility (QUERY_ALL_PACKAGES) permission

Google Play restricts the use of high-risk or sensitive permissions, including the `QUERY_ALL_PACKAGES` permission, which gives visibility into the inventory of installed apps on a given device. Play regards the inventory of installed apps queried from a user's device as personal and sensitive information, and the use of the permission is only permitted when your app's core user-facing functionality or purpose requires broad visibility into installed apps on the user's device.

If your app does not meet the requirements for acceptable use below, you must remove it from your app's manifest in order to comply with Play policy. Suggestions for policy-compliant alternative implementations are also detailed below.

If your app meets the policy requirements for the acceptable use of the `QUERY_ALL_PACKAGES` permission, you will be required to declare this and any other high-risk permissions using the Permissions Declaration Form in Play Console.

Apps that fail to meet the policy requirements or do not submit the Permissions Declaration Form may be removed from Google Play.

source: support.google.com

CVE-2023-21348

Bypass of **QUERY_ALL_PACKAGES** permission, side channel attack

```
private boolean doesAddToastWindowRequireToken(String packageName, int callingUid,
        WindowState attachedWindow) {
    // Try using the target SDK of the root window
    if (attachedWindow != null) {
        return attachedWindow.mActivityRecord != null
                && attachedWindow.mActivityRecord.mTargetSdk >= Build.VERSION_CODES.O;
    } else {
        // Otherwise, look at the package
        try {
            ApplicationInfo appInfo = mContext.getPackageManager()
                    .getApplicationInfoAsUser(packageName, 0,
                            UserHandle.getUserId(callingUid));
            if (appInfo.uid != callingUid) {
                throw new SecurityException("Package " + packageName + " not in UID "
                        + callingUid);
            }
            if (appInfo.targetSdkVersion >= Build.VERSION_CODES.O) {
                return true;
            }
        } catch (PackageManager.NameNotFoundException e) {
            /* ignore */
        }
    }
```

CVE-2023-21348

Bypass of **QUERY_ALL_PACKAGES** permission, side channel attack

```
private boolean doesAddToastWindowRequireToken(String packageName, int callingUid,
        WindowState attachedWindow) {
    // Try using the target SDK of the root window
    if (attachedWindow != null) {
        return attachedWindow.mActivityRecord != null
                && attachedWindow.mActivityRecord.mTargetSdk >= Build.VERSION_CODES.O;
    } else {
        // Otherwise, look at the package
        try {
            ApplicationInfo appInfo = mContext.getPackageManager()
                    .getApplicationInfoAsUser(packageName, 0,
                            UserHandle.getUserId(callingUid));
            if (appInfo.uid != callingUid) {
                throw new SecurityException("Package " + packageName + " not in UID "
                        + callingUid);
            }
            if (appInfo.targetSdkVersion >= Build.VERSION_CODES.O) {
                return true;
            }
        } catch (PackageManager.NameNotFoundException e) {
            /* ignore */
        }
    }
```

CVE-2024-23713

Retaining access to all posted notifications after permission is revoked, elevation of privilege

```
public void migrateNotificationFilter(INotificationListener token, int defaultTypes,
        List<String> disallowedApps) {
    final long identity = Binder.clearCallingIdentity();
    try {
        synchronized (mNotificationLock) {
            final ManagedServiceInfo info = mListeners.checkServiceTokenLocked(token);

            Pair key = Pair.create(info.component, info.userid);

            NotificationListenerFilter nlf = mListeners.getNotificationListenerFilter(key);
            if (nlf == null) {
                nlf = new NotificationListenerFilter();
            }
            if (nlf.getDisallowedPackages().isEmpty() && disallowedApps != null) {
                for (String pkg : disallowedApps) {
                    // block the current user's version and any work profile versions
                    for (int userId : mUm.getProfileIds(info.userid, false)) {
                        try {
                            int uid = getUidForPackageAndUser(pkg, UserHandle.of(userId));
                            VersionedPackage vp = new VersionedPackage(pkg, uid);
                            nlf.addPackage(vp);
                        } catch (Exception e) {
                            // pkg doesn't exist on that user; skip
                        }
```

CVE-2024-23713

Retaining access to all posted notifications after permission is revoked, elevation of privilege

```
public void migrateNotificationFilter(INotificationListener token, int defaultTypes,
        List<String> disallowedApps) {
    final long identity = Binder.clearCallingIdentity();
    try {
        synchronized (mNotificationLock) {
            final ManagedServiceInfo info = mListeners.checkServiceTokenLocked(token);

            Pair key = Pair.create(info.component, info.userid);

            NotificationListenerFilter nlf = mListeners.getNotificationListenerFilter(key);
            if (nlf == null) {
                nlf = new NotificationListenerFilter();
            }
            if (nlf.getDisallowedPackages().isEmpty() && disallowedApps != null) {
                for (String pkg : disallowedApps) {
                    // block the current user's version and any work profile versions
                    for (int userId : mUm.getProfileIds(info.userid, false)) {
                        try {
                            int uid = getUidForPackageAndUser(pkg, UserHandle.of(userId));
                            VersionedPackage vp = new VersionedPackage(pkg, uid);
                            nlf.addPackage(vp);
                        } catch (Exception e) {
                            // pkg doesn't exist on that user; skip
                        }
```

Any string!

Will never go this way

CVE-2024-40674

WiFi **SSID** value with arbitrary length, permanent device denial of service

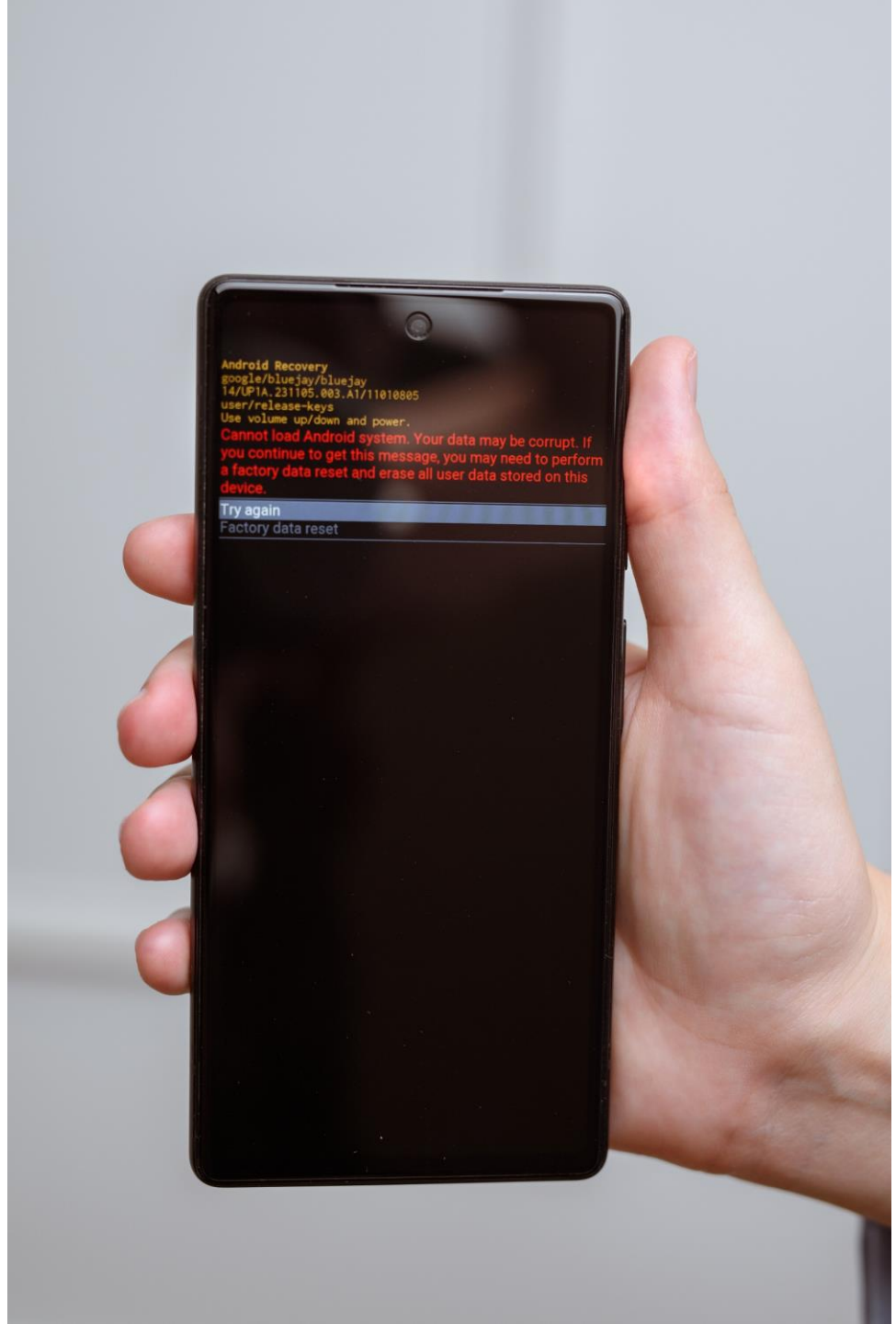
```
private static boolean validateSsid(String ssid, boolean isAdd) {
    if (isAdd) {
        if (ssid == null) {
            Log.e(TAG, "validateSsid : null string");
            return false;
        }
    } else {
        if (ssid == null) {
            // This is an update, so the SSID can be null if that is not being changed.
            return true;
        }
    }
    if (ssid.isEmpty()) {
        Log.e(TAG, "validateSsid failed: empty string");
        return false;
    }
    if (!ssid.startsWith("\"") && ssid.length() > SSID_HEX_MAX_LEN) {
        Log.e(TAG, "validateSsid failed: hex ssid " + ssid + " longer than 32 bytes");
        return false;
    }
```

CVE-2024-40674

WiFi **SSID** value with arbitrary length, permanent device denial of service

```
private static boolean validateSsid(String ssid, boolean isAdd) {
    if (isAdd) {
        if (ssid == null) {
            Log.e(TAG, "validateSsid : null string");
            return false;
        }
    } else {
        if (ssid == null) {
            // This is an update, so the SSID can be null if that is not being changed.
            return true;
        }
    }
    if (ssid.isEmpty()) {
        Log.e(TAG, "validateSsid failed: empty string");
        return false;
    }
    if (!ssid.startsWith("\"") && ssid.length() > SSID_HEX_MAX_LEN) {
        Log.e(TAG, "validateSsid failed: hex ssid " + ssid + " longer than 32 bytes");
        return false;
    }
```

Doesn't work!


Android Recovery
google/bluejay/bluejay
14/UP1A.231105.003.A1/11010885
user/release-keys
Use volume up/down and power.
Cannot load Android system. Your data may be corrupt. If you continue to get this message, you may need to perform a factory data reset and erase all user data stored on this device.
Try again
Factory data reset

Good luck with finding bugs!