Albiriox Exposed: A New RAT Mobile Malware Targeting Global Finance and Crypto Wallets

By Gianluca Scotti, Simone Mattia

Archived: 2026-04-23 02:51:37 UTC

Key Points:

- **Emergence of a new MaaS**: Albiriox is a newly identified Android malware family offered as a Malware-as-a-Service (MaaS), showing signs of active development and rapid iteration. Evidence suggests the operation is managed by Russian-speaking Threat Actors (TAs).
- **Two-stage deployment chain**: The malware leverages dropper applications distributed through social engineering lures, combined with packing techniques, to evade static detection and deliver its payload.
- **On-Device Fraud capabilities**: Albiriox exhibits the core features of modern Android Banking Trojans, enabling TAs to perform On-Device Fraud through remote control, screen manipulation, and real-time interaction with the infected device.
- **Global targeting of financial and crypto institutions**: Hardcoded targets indicate a broad target spectrum, encompassing major banking and cryptocurrency applications worldwide (over 400).
- **Overlay and RAT combined**: Beyond remote device takeover with screen streaming and UI manipulation, Albiriox supports targeted overlay attacks for credential harvesting, covering the full spectrum of Android banking fraud techniques.

Executive Summary

Over the past few months, the **Cleafy Threat Intelligence team** has identified and analyzed **Albiriox**, a newly emerging Android malware family promoted as a Malware-as-a-Service (MaaS) within underground cybercrime forums. First observed in September 2025 during a limited recruitment phase targeting high-reputation forum members, the project transitioned to a publicly available MaaS offering in October 2025. Forum activity, linguistic patterns, and infrastructure analysis indicate that Russian-speaking Threat Actors (TAs) are behind the operation.

Despite its recent emergence, Albiriox already demonstrates a well-structured architecture explicitly designed for **On-Device Fraud (ODF)**, a tactic in which attackers take control of a victim's mobile device and execute fraudulent actions directly within legitimate banking or cryptocurrency apps. The malware's design clearly reflects this objective, prioritizing **Full Device Takeover**, **Real-Time Interaction**, and the ability to perform unauthorized operations while remaining undetected by the user.

Albiriox combines two core attack vectors: a **VNC-based Remote Access** module for real-time device control, and an **Overlay Attack** mechanism for credential harvesting. While the remote control functionality is fully operational, the overlay component is under active development, with generic templates currently in place rather than application-specific phishing pages.

Even in its early stage, Albiriox already exhibits the defining traits of the latest generation of **ODF-oriented Android banking malware**, including stealthy delivery, evasion techniques, dynamic device manipulation, and broad targeting across the financial sector. Its MaaS business model and ongoing development suggest that Albiriox may rapidly gain traction among TAs seeking efficient and scalable tools for high-impact mobile fraud.

From private beta to public MaaS

The first traces of **Albiriox emerged in late September 2025** within a specific Telegram channel, where the suspected author was discussing the project with a small community of followers. In these early conversations, **TAs announced plans to release Albiriox as a MaaS offering**, mentioning that a beta phase would be made available exclusively to high-reputation members of the underground forums where the malware would be promoted.

Date	Actor	Translated Message (EN)	Original (RU)
21.09.2025 19:25:48 UTC+01:00	[redacted]	Albiriox Bot - Will be a MaaS project ;) Details coming soon on [redacted] ([redacted] most likely not)	Albiriox Bot - Будет MaaS проектом ;)Подробности уже скоро на [redacted] ([redacted] скорее всего нет)
21.09.2025 19:27:01 UTC+01:00	[redacted]	will they be here? otherwise I'm not on those sites	а тут будут? а то меня нет на тех площадках
21.09.2025 19:27:27 UTC+01:00	[redacted]	Will :)	Будет :)
21.09.2025 19:33:14 UTC+01:00	[redacted]	Can you tell me the approximate rental price?	приблизительную цену аренды можешь озвучить?
21.09.2025 19:35:04 UTC+01:00	[redacted]	>2000 month?	>2000 месяц?
21.09.2025 19:35:55 UTC+01:00	[redacted]	I haven't decided on this yet :) I think 600-800$~ month	Ещё не определился с этим :)Думаю 600-800$~ месяц
21.09.2025 19:36:15 UTC+01:00	[redacted]	But I'm thinking of doing a beta test for respected people on the forum	Но думаю сделать бета тест для уважаемых людей на форуме
21.09.2025 19:36:34 UTC+01:00	[redacted]	Rega from 2023 or from 24 with a good history	Рега от 2023 года или от 24 с хорошей историей
21.09.2025 19:37:17 UTC+01:00	[redacted]	Moderators, people with reputation	Модерам, людям с репутацией

Figure 1 - Translated Messages from TG

A few days later, the official beta announcement appeared on two well-known Russian-speaking cybercrime forums. The initial post provided the first technical details about the malware, revealing a fully-featured RAT with all the capabilities required to perform On-Device Fraud attacks. Among the advertised functionalities, one stands out: **AcVNC (Accessibility VNC).**

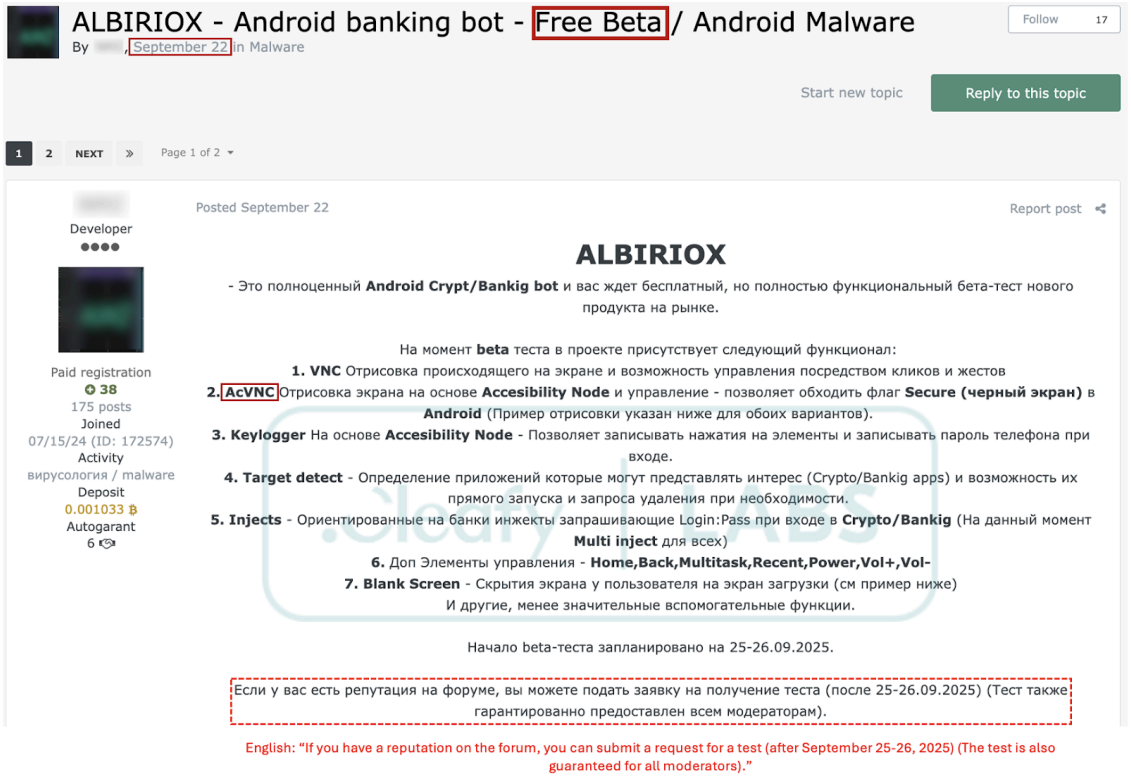


Figure 2 - Albiriox Free Beta Post on the Forum

The terminology around this capability (AcVNC) is not standardized across the underground ecosystem. In a later exchange observed on the same Telegram channel, a potential affiliate expressed confusion about the different names used to describe this technique. The Albiriox developer(s) clarified that terms like **hVNC, screen reader, skeleton VNC, AcVNC,** and **accessibility view are essentially interchangeable**, *"purely marketing, and everyone has different names."* To address this confusion and demonstrate the malware's capabilities in action, the author subsequently released a series of promotional videos showcasing the core functionalities, providing prospective affiliates with a clearer understanding of how Albiriox operates.

Date	Actor	Translated Message (EN)	Original (RU)
04.11.2025 20:28:03 UTC+01:00		confusion in terminologies	путаница в терминологиях
04.11.2025 20:28:10 UTC+01:00		+	+
04.11.2025 20:28:13 UTC+01:00		Confused	Перепутали
04.11.2025 20:28:31 UTC+01:00		purely marketing and everyone has different names	чисто маркетинг и у всех разные названия
04.11.2025 20:28:45 UTC+01:00		This is an access node tree	Это древ нод аксесс
04.11.2025 20:28:50 UTC+01:00		I speak harshly	Грубо говоря
04.11.2025 20:28:56 UTC+01:00		Like without permission	Типо без разрешения
04.11.2025 20:29:04 UTC+01:00		hvnc/screen reader/sceleton vnc/ ac vnc / accesibility view	hvnc/screen reader/sceleton vnc/ ac vnc / accesibility view
04.11.2025 20:29:08 UTC+01:00		it's all the same	все это одно и тоже
04.11.2025 20:29:12 UTC+01:00		Yes	Да

Figure 3 - Other Translated Messages from TG

Following the beta period, the same TA published a more structured announcement promoting the first public release of the Albiriox MaaS. The announcement disclosed the MaaS pricing model: **$650 per month until October 21st, 2025, increasing to $720 afterwards**.

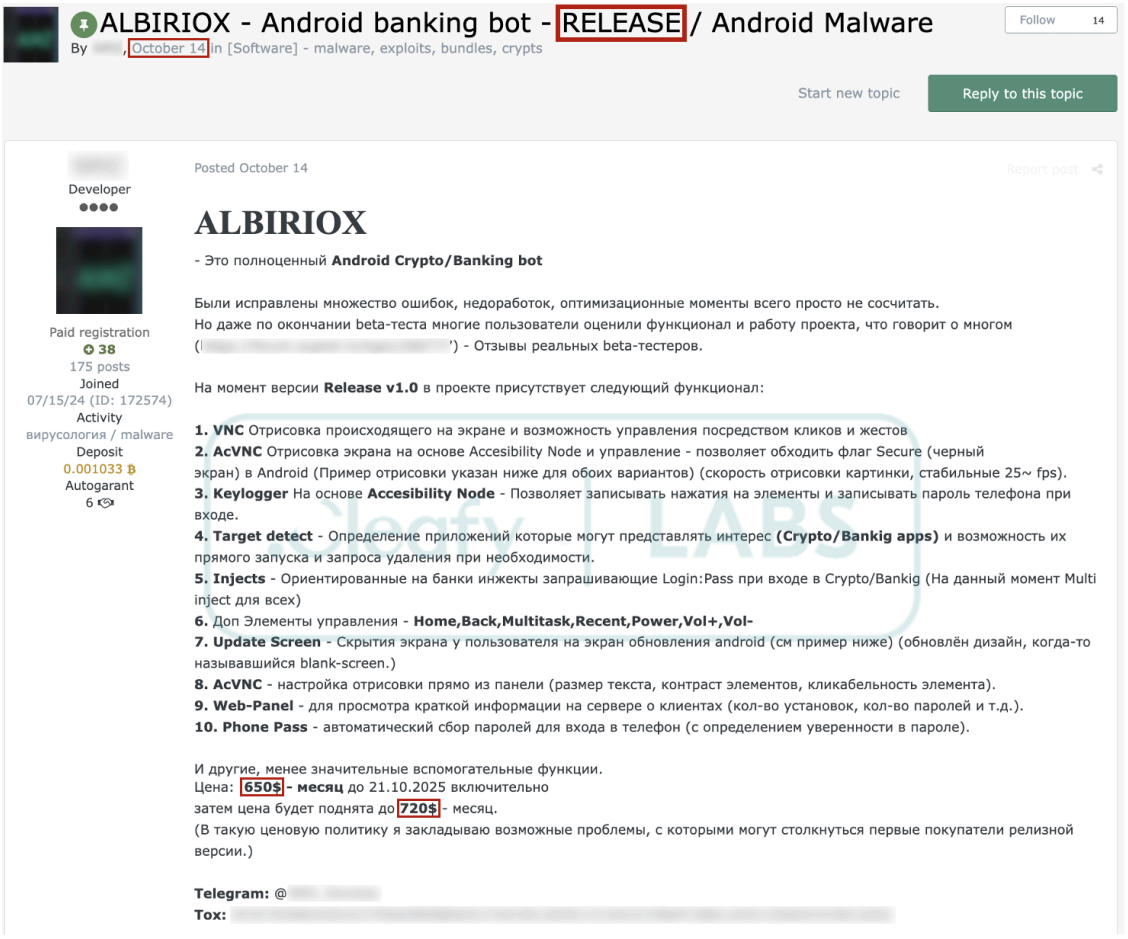


Figure 4 - Official Albiriox Release Announcement

In parallel, **our Threat Intelligence team identified multiple APK samples** that, based on their internal structure and functionalities, appear to be directly **tied to the initial development cycle of Albiriox**. The following sections detail the distribution campaigns and technical capabilities observed during this early phase. Given Albiriox's active development, this analysis represents a snapshot of a rapidly evolving threat that we expect to mature significantly in the coming months.

Early campaigns

During the early monitoring of Albiriox, **Cleafy intercepted one of the first distribution campaigns tied to this malware**. Given the timing—aligned with the beta phase—and the limited scope of the operation, we assess that this campaign is likely attributable to a single affiliate, potentially one of the high-reputation forum members granted early access to the MaaS platform.

The campaign targets Austrian victims explicitly, leveraging German-language lures and social engineering tactics consistent with the broader mobile banking threat landscape. Distribution relies on SMS messages containing shortened links that redirect to fraudulent landing pages impersonating legitimate services.

Fake Google Play Page

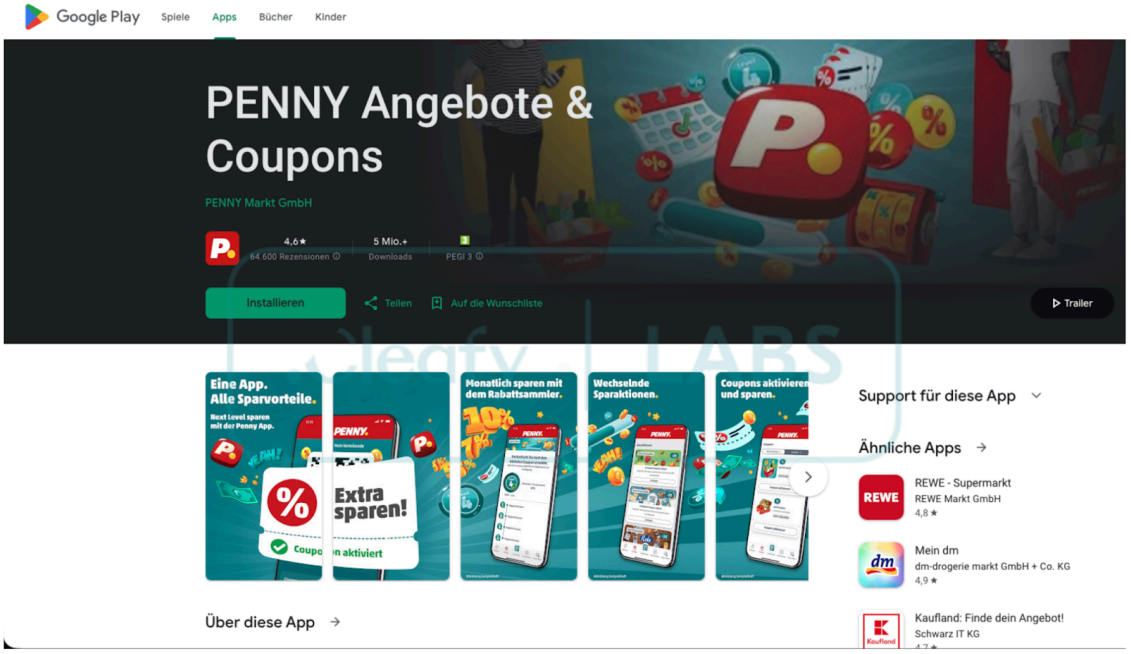


Figure 5 - First Delivery Method Detected

The initial delivery mechanism observed was straightforward: victims were directed to a fake Google Play page offering what appeared to be the official "Penny Market" application, a popular discount retail chain in the DACH region. The page faithfully reproduced Google's visual identity, displaying German-language elements, including app descriptions, ratings, and installation prompts. Once the user clicked "Install," the dropper APK was downloaded directly from attacker-controlled infrastructure, bypassing the official Play Store entirely.

Evolution: Phone Number Requirement

Shortly after the initial wave, we observed a notable shift in the distribution flow. The **phishing infrastructure has been updated**: the landing page no longer exposes a direct APK download. Instead, users were prompted to enter their mobile phone number, with the page instructing that the download link would be delivered "via WhatsApp".

Figure 6 - Second Delivery Method Detected

The updated flow unfolds in four stages: the victim selects a fuel provider, spins a promotional "wheel of fortune," enters their phone number, and receives confirmation that a representative will contact them shortly. Analysis of the underlying JavaScript reveals that only Austrian phone numbers are accepted (isValidAustrianNumber function), and submitted data is forwarded directly to a Telegram bot controlled by the TAs.

Figure 7 - JavaScript Code Snippet into Phishing Kit

Technical Analysis

This section provides a concise technical overview of the key findings derived from the analysis of the **Albiriox** malware. As with many well-known Android banking Trojans, its core functionality aligns with established patterns commonly observed in the mobile threat landscape, including VNC-based remote control and Overlay attacks.

Installation

In the campaign we examined, the initial attack stage leverages the fake Penny application (dropper), which has been used as a dedicated decoy for the main **Albiriox** payload. Our analysis revealed that this sample utilizes the JSONPacker technique, a form of code obfuscation and dynamic unpacking employed to deliver the payload.

Below are all the steps for installing the malware:

- **Upon installation and launch**, the initial dropper triggers a social engineering sequence immediately. Instead of behaving like a legitimate application, it displays a fraudulent System Update interface designed to pressure the victim into granting the requested permissions.
- **The dropper's** primary goal is to obtain the critical "Install Unknown Apps" permission, which enables out-of-store installations.
- **Once this permission is granted**, the application installs the final payload **Albiriox** on the compromised device.

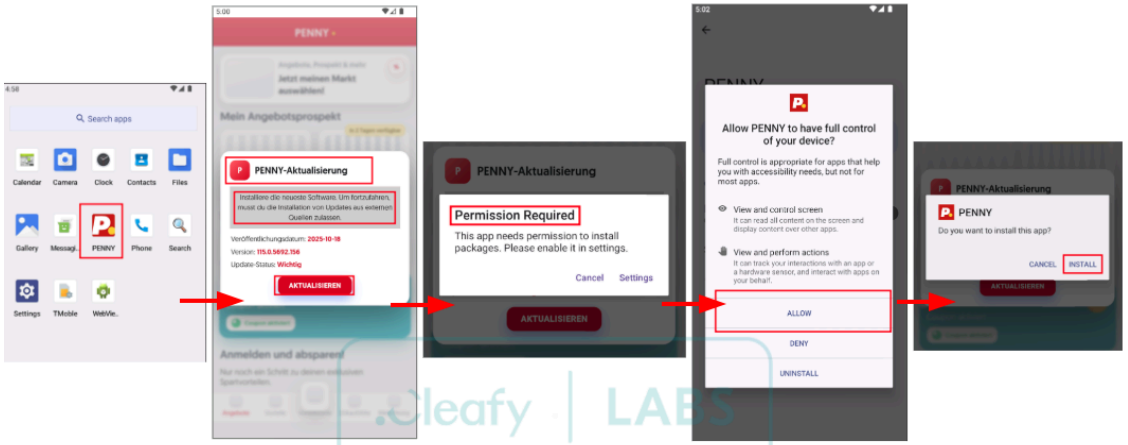
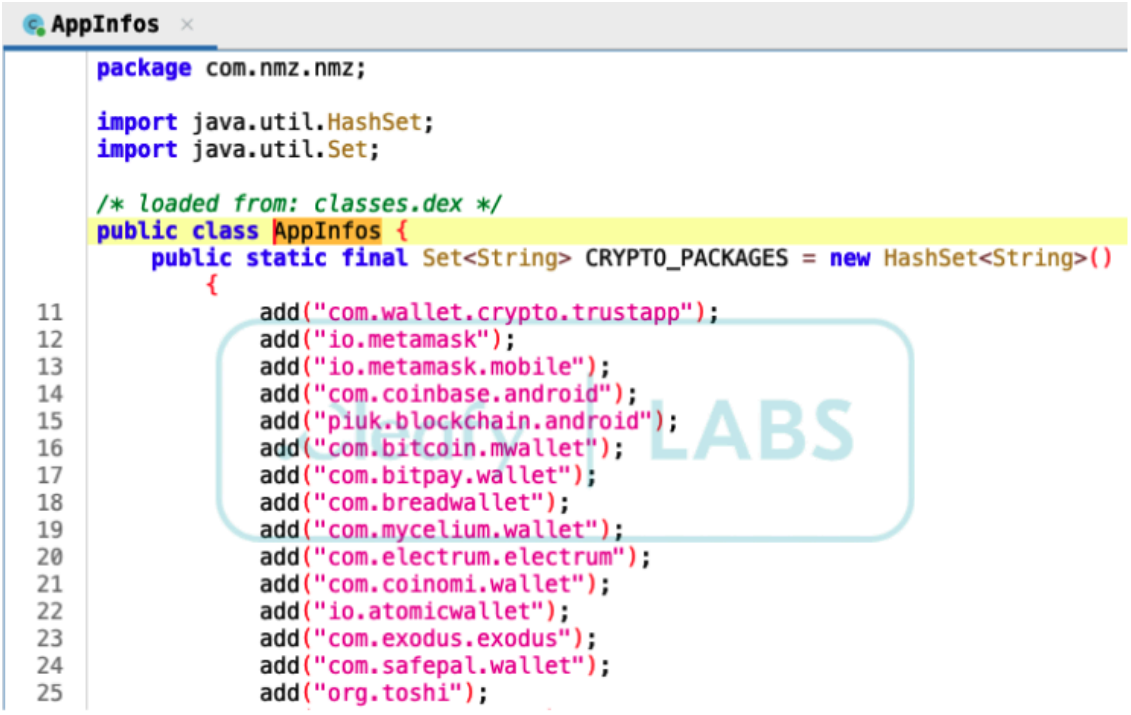


Figure 8 - Malware Installation

This intricate, staged deployment highlights the TA's efforts to evade static signature detection by delivering the malicious functionality dynamically after the initial dropper installation.

Malware Targets

Analyzing malware's source code, we successfully identified all potential application targets used for overlay attacks and credential harvesting. These targets are hardcoded within a dedicated class named AppInfos, which effectively acts as the malware's internal database for application monitoring and fraud execution.



```java
package com.nmz.nmz;

import java.util.HashSet;
import java.util.Set;

/* loaded from: classes.dex */
public class AppInfos {
    public static final Set<String> CRYPTO_PACKAGES = new HashSet<String>()
        {
            add("com.wallet.crypto.trustapp");
            add("io.metamask");
            add("io.metamask.mobile");
            add("com.coinbase.android");
            add("piuk.blockchain.android");
            add("com.bitcoin.mwallet");
            add("com.bitpay.wallet");
            add("com.breadwallet");
            add("com.mycelium.wallet");
            add("com.electrum.electrum");
            add("com.coinomi.wallet");
            add("io.atomicwallet");
            add("com.exodus.exodus");
            add("com.safepal.wallet");
            add("org.toshi");
```


Figure 9 - Targets

In total, over 400 applications were identified. The targets span a wide range of financial traditional banking, fintech, payment processors, cryptocurrency exchanges, digital wallets, and trading platforms. This broad spectrum of targets provides strong evidence that **Albiriox** is designed to operate as a fully-fledged banking Trojan, capable of supporting global ODF campaigns.

To better illustrate the breadth and intent of the TAs, we grouped all identified applications into several functional and geographic categories, as shown in the following graph. This classification helps highlight the malware's strategic targeting approach, which prioritizes high-value, globally recognized financial brands.

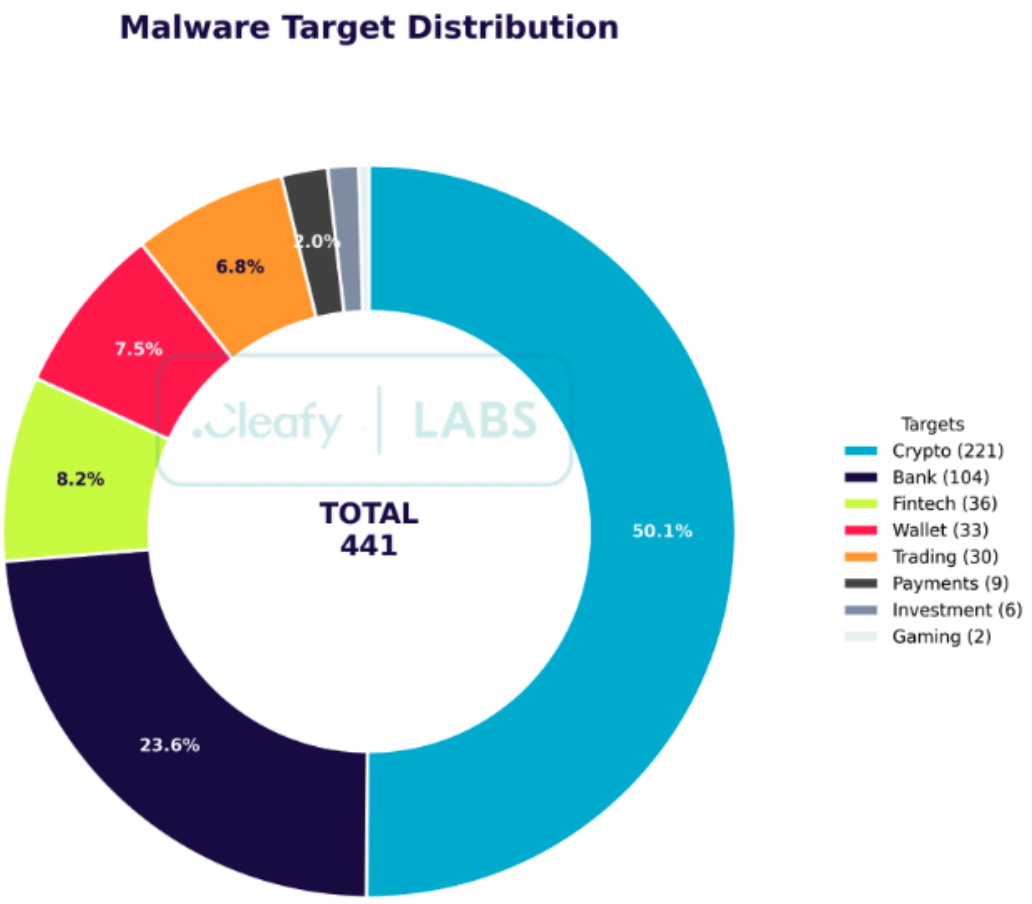


Figure 10 - Target Type

C2 Communication Protocol

The malware establishes a persistent communication channel with its C2 infrastructure using an unencrypted **TCP Socket connection**.

- **Handshake and Identification**: Immediately upon successful execution, the malware initiates a detailed handshake with the C2 server. This message contains device identifiers, including the Hardware ID (HWID), device model, and current Android OS version, effectively registering the victim device within the botnet.

```
private void sendHandshake() {
    try {
        String orCreatePersistentHWID = getOrCreatePersistentHWID();
        Log.d(TAG, "========================================");
        Log.d(TAG, "HWID Generation Details:");
        Log.d(TAG, "  Device: " + Build.MANUFACTURER + " " + Build.MODEL);
        Log.d(TAG, "  Android Version: " + Build.VERSION.RELEASE + " (API " + Build.VERSION.SDK_INT + ")");
        Log.d(TAG, "  Generated HWID: " + orCreatePersistentHWID);
        Log.d(TAG, "========================================");
        JSONObject jSONObject = new JSONObject();
        jSONObject.put("type", "handshake");
        jSONObject.put("build_id", "BETA");
        jSONObject.put("version", "2.1");
        jSONObject.put("deviceModel", Build.MANUFACTURER + " " + Build.MODEL);
        jSONObject.put("hwid", orCreatePersistentHWID);
        jSONObject.put("androidId", Settings.Secure.getString(getContentResolver(), "android_id"));
        jSONObject.put("manufacturer", Build.MANUFACTURER);
        jSONObject.put("androidVersion", Build.VERSION.RELEASE);
        jSONObject.put("apiLevel", Build.VERSION.SDK_INT);
        jSONObject.put("brand", Build.BRAND);
        jSONObject.put("device", Build.DEVICE);
        jSONObject.put("fps", 25);
        enqueueMessage(jSONObject.toString().getBytes(StandardCharsets.UTF_8));
        Log.d(TAG, "Handshake sent with HWID: " + orCreatePersistentHWID);
    } catch (Exception e) {
        Log.e(TAG, "Error sending handshake: " + e.getMessage());
    }
}
```

Figure 11 - sendHandShake Method

- **Protocol Management**: Communication is managed via structured JSON objects transmitted over a TCP data stream. This setup facilitates efficient command parsing and data exfiltration.
- **Heartbeat Mechanism**: A Ping/Pong heartbeat mechanism is implemented to ensure the persistence and stability of the TCP connection, allowing the TAs to maintain continuous control and readiness for remote operations.

Albiriox Functionalities

Upon examining the source code, we identified a comprehensive set of commands (listed in the Appendix) that outline the operational capabilities of **Albiriox**. These commands provide a clear view of the malware's design philosophy and confirm that the TAs have implemented all the core components typically found in modern Android banking Trojans. Although the codebase exposes a wide range of device-level functions, all capabilities ultimately converge toward a single goal: **achieving full remote control of the device to enable On-Device Fraud (ODF)**.

At the core of its operational model, **Albiriox** installs and activates a **VNC-based remote access module,** enabling real-time interaction with the compromised device. When combined with black-screen overlays, this allows attackers to execute fraudulent actions while remaining undetected by the victim.

Beyond remote access, the command set enables a broad spectrum of device manipulation features, including:

- **UI Interaction & Device Automation:**
  Commands such as click, swipe, text, back, home, recent, and power allow the operator to fully interact with the user interface and navigate across applications.
- **Fraud-Driven Controls:**
  Functions like get_phone_password, clear_phone_password, live_key, live_key_stop, and set_vnc_mode provide the attacker with mechanisms to extract sensitive information, maintain session control, and streamline ODF activity.

- **Stealth & Environment Control:**
  Commands, including blank_screen, black_blank_screen, volume_up, volume_down, and control support concealment and operational stealth during fraud execution.
- **Application Management:**
  Instructions such as launch_app and uninstall_app enable Albiriox to manage the lifecycle of installed applications, allowing for further social engineering, evasion, or cleanup.
- **C2 Synchronisation & Session Checks:**
  The presence of ping and pong commands highlights the constant communication flow with the command-and-control server to validate connectivity and operator presence.

While **Albiriox** exposes numerous capabilities across its command set, these functions collectively support a unified operational workflow, enabling persistent, covert, and fully interactive control over the victim's device. This allows attackers to perform fraudulent transactions directly from the user's legitimate session. This approach is strongly aligned with the most advanced **ODF-oriented** mobile malware currently observed in the threat landscape.

The Remote Control and Real-Time Device Streaming

The most prominent feature confirmed is **Albiriox's** ability to operate as a full **Remote Controller**. This capability enables TAs to have real-time, unauthorized access and visual monitoring of the victim's device. It mirrors legitimate remote access technologies (**such as VNC or similar services**), enabling a live stream of the device display and allowing the operator to interact with the device remotely. Such behavior is strongly indicative of a mobile Remote Access Trojan (RAT) or a highly sophisticated banking Trojan that relies on session hijacking and on-device fraud.

As shown in the following figure, we recovered an example of an active device infected with Albiriox, alongside the corresponding VNC session. The malware exposes two distinct VNC streaming modes within the panel:

- **AC VNC**: a stream sourced from Accessibility Services, displaying all UI nodes and accessibility elements present on the device screen.
- **VNC**: a standard real-time visual stream of the device's display, similar to traditional VNC implementations.

In the screenshot, the highlighted "**AC VNC**" tab (visible in the lower-right corner) confirms that the operator can switch between the two modes depending on the phase of the fraud operation and the level of interaction required.

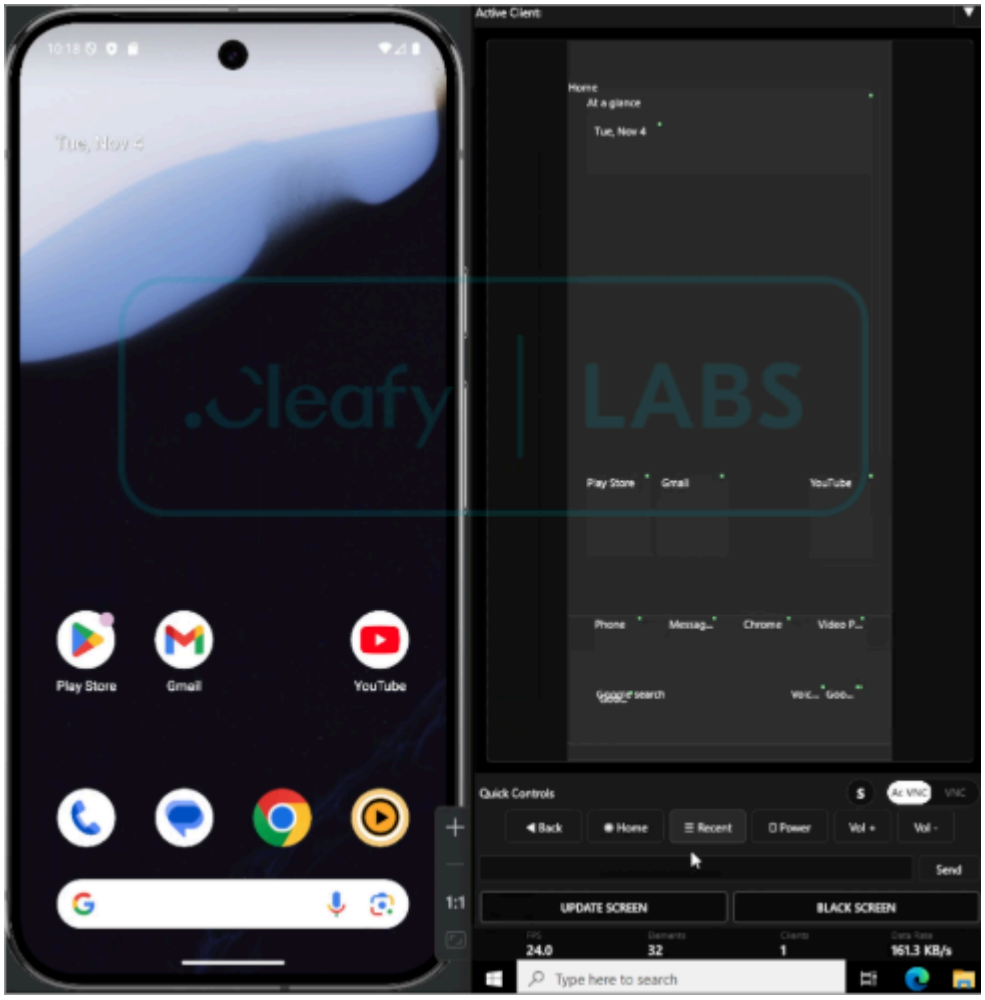


Figure 12 - Accessibility AC VNC Connection

This accessibility-based streaming mechanism is intentionally designed to bypass the limitations imposed by Android's **FLAG_SECURE** protection. Since many banking and cryptocurrency applications now block screen recording, screenshots, and display capture when this flag is enabled, leveraging Accessibility Services allows the malware to obtain a complete, node-level view of the interface without triggering any of the protections commonly associated with direct screen-capture techniques.

Overlay Attack Mechanism

A secondary, yet critical, functionality implemented by **Albiriox** is the use of the **Overlay Attack** technique, a staple of modern Android banking malware.
During the analysis, we successfully retrieved data indicating the deployment of at least three distinct types of overlay screens:

- **System Update Overlay**: An overlay designed to impersonate a legitimate System Update screen. This is used by the attackers when they have established the VNC connection, and they can decide to display this "Update" screen.

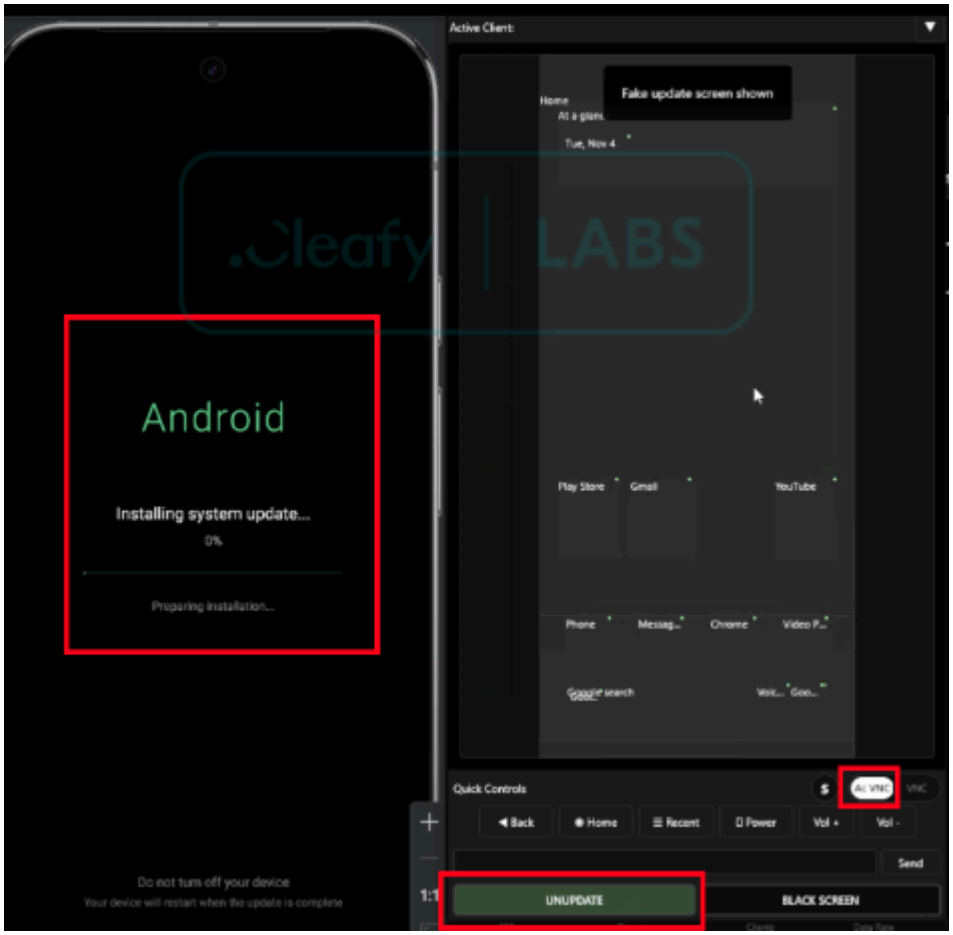


Figure 13 - System Update Overlay

- **Black Screen Overlay**: A full-screen black overlay is utilized, likely to obscure the victim's view. At the same time, the attacker executes unauthorized transactions or operations in the background using the remote control capability.

Figure 14 - Blank Screen Overlay

- **Targeted Application Overlay:** This third overlay is specifically deployed upon intercepting the execution of one of the hardcoded target applications monitored by the malware. This screen is not a typical fake login or data-entry form designed to harvest user credentials (e.g., banking passwords or crypto wallet seed phrases) that mimic the real application.

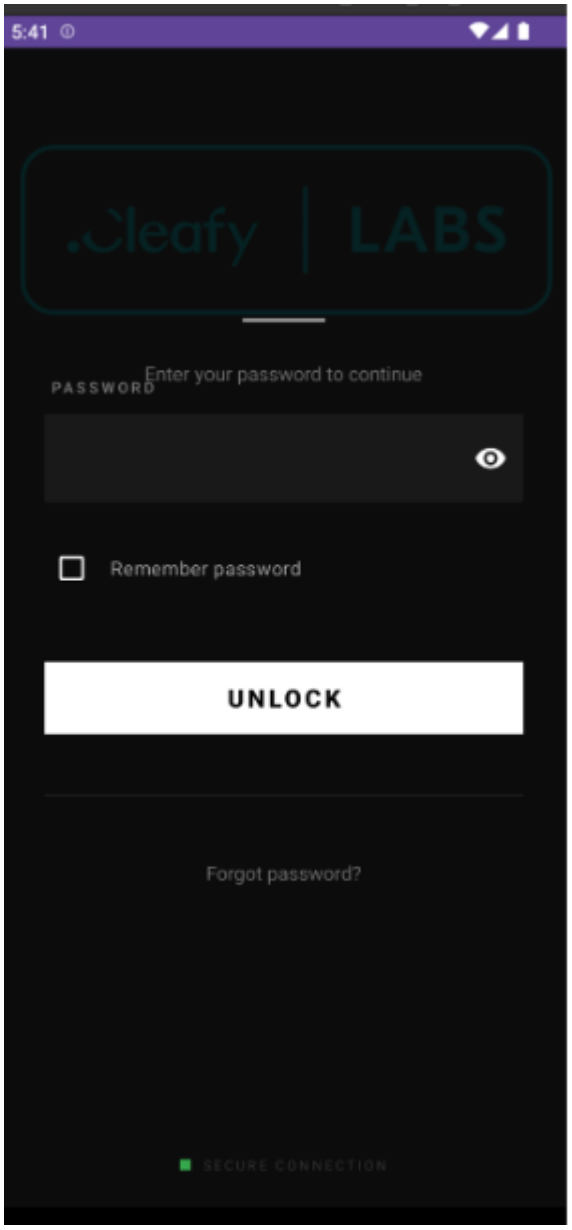


Figure 15 - Generic Target Overlay

Evading Detection

Beyond the recruitment messages and the initial beta-stage announcements presented in the "**From private beta to public MaaS"** chapter, Cleafy's monitoring activities uncovered an additional discussion thread tied to the **Albiriox** developers. In this conversation, a forum user explicitly asked whether the malware was ***FUD*** (Fully Undetectable), a common indicator of interest among TAs seeking tools capable of bypassing antivirus and mobile security solutions.
In response, the **Albiriox** developers clarified that they provide a **custom Builder** as part of their MaaS offering.

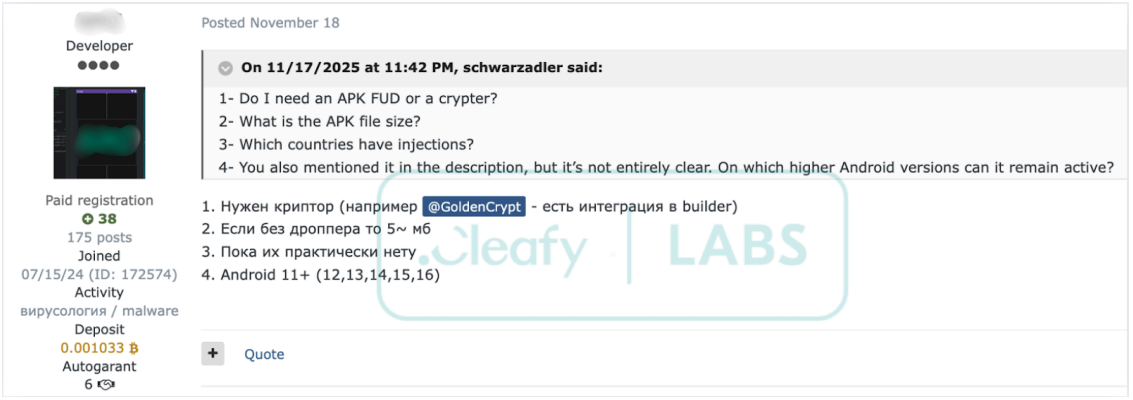


Figure 16 - User asking for FUD Capability of Albiriox

This builder reportedly integrates a third-party crypting service known as **Golden Crypt**, a well-established tool within cybercriminal markets and frequently advertised on the same forum. Notably, the developer of Golden Crypt is also an active member, reinforcing the tight ecosystem of TAs and service providers collaborating to enhance stealth and evasive capabilities.

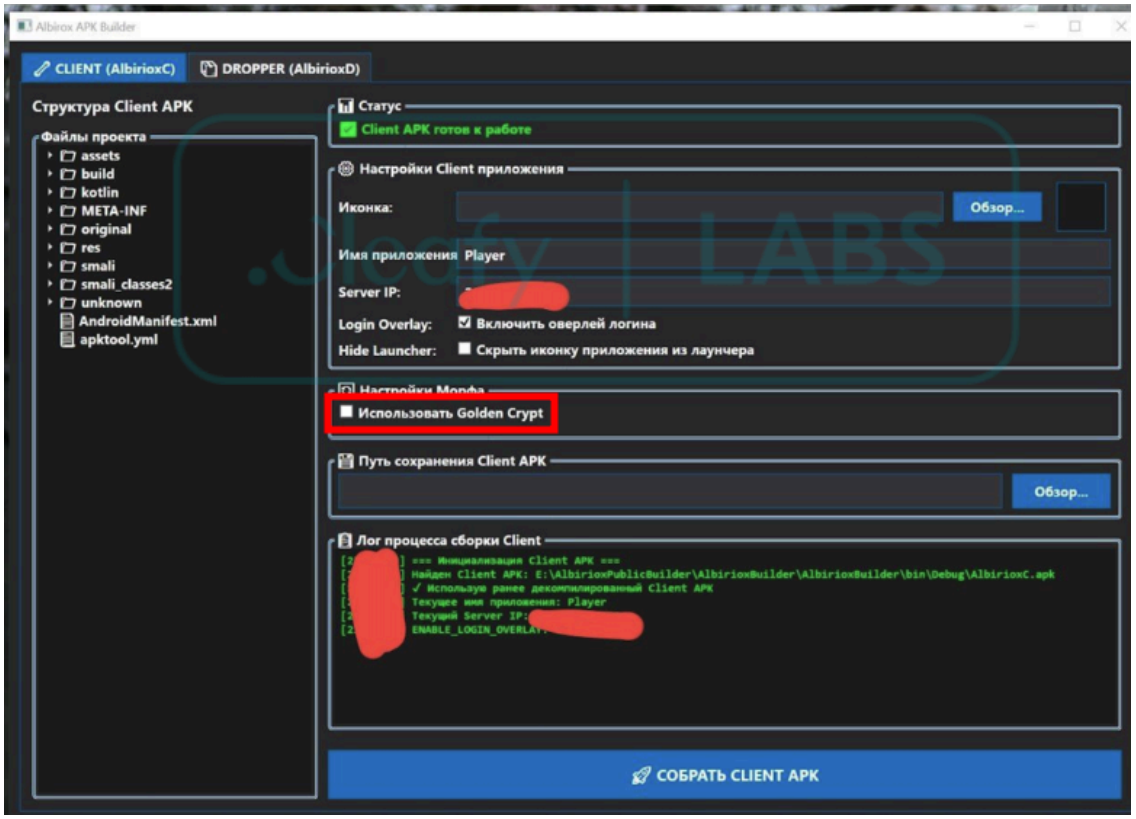


Figure 17 - Albiriox APK Builder

The inclusion of Golden Crypt within the builder pipeline suggests that the **Albiriox** operators are deliberately positioning the malware as a **stealth-optimized product**, aiming to evade static detection mechanisms and improve the likelihood of successful deployment during the early infection stages, especially relevant given the malware's reliance on the two-stage delivery and accessibility-based device takeover.

Conclusions

The analysis of the **Albiriox** malware, together with its dedicated Penny decoy, highlights the ongoing evolution and increasing sophistication of mobile banking threats. The evidence suggests that the TAs have adopted a two-stage, obfuscated delivery chain, specifically designed to evade detection while maintaining full control of compromised devices, particularly those running high-value financial or cryptocurrency applications.

Albiriox exhibits all core characteristics of modern **On-Device Fraud (ODF)** malware, including VNC-based remote control, accessibility-driven automation, targeted overlays, and dynamic credential harvesting. These capabilities enable attackers to bypass traditional authentication and fraud-detection mechanisms by operating directly within the victim's legitimate session.

In conclusion, Albiriox represents a rapidly evolving threat that exemplifies the broader shift toward ODF-focused mobile malware. Effectively countering such threats requires a layered defense approach that correlates client-side signals, behavioral patterns, and transactional anomalies in real-time. This multi-dimensional visibility enables financial institutions to detect compromise at the earliest stages of the attack chain and enforce precise, context-aware response policies before fraud is executed. As mobile banking threats continue to mature, the ability to orchestrate these indicators into actionable defenses will prove essential for staying ahead of this emerging class of Android malware.

Appendix - IOCs:

Malware Target Apps

Disclaimer: In our standard TLP:WHITE reports, we typically refrain from publishing detailed lists of targeted applications. Such information is often shared separately with financial CERTs through TLP:AMBER reports to facilitate timely distribution to associated financial institutions. If you are interested in which are the targets, please contact the labs[at]cleafy.com in order to get the list.

APKs:

package name	App name	MD5
com.example.myapplication	PENNY (dropper)	b6bae028ce6b0eff784de1c5e766ee33
com.example.myapplication	PENNY (dropper)	61b59eb41c0ae7fc94f800812860b22a
com.example.myapplication	PENNY (dropper)	f09b82182a5935a27566cdb570ce668f
com.nmz.nmz	nmz.nmz	f5b501e3d766f3024eb532893acc8c6c

C2 Server:

IP address	Port	Sample Related (MD5)
194.32.79.94	5555	f5b501e3d766f3024eb532893acc8c6c

Delivery:

Domains
google-app-download[.]download
google-get[.]download
google-aplication[.]download
play.google-get[.]store
google-app-get[.]com
google-get-app[.]com
google-app-install[.]com

C2 Command List

Command	Usage
clear_phone_password	Removes or resets the phone’s lockscreen password/PIN/pattern.
ping	Heartbeat request sent by the C2 to check if the device is online.
pong	Response sent by the malware to confirm it is connected and active.
get_phone_password	Retrieves and sends the device’s lockscreen password/PIN/pattern to the C2.
control	Enables remote control mode via Accessibility (attacker takes over UI).
click	Performs a tap gesture at a specified screen coordinate.
swipe	Performs a swipe gesture on the screen (scroll, navigation, etc.).
volume_up	Increases the device volume remotely.
recent	Opens the “Recent Apps” / Task Manager screen.
volume_down	Decreases the device volume remotely.
uninstall_app	Uninstalls a specified app from the device.
blank_screen	Displays a blank overlay screen to hide malicious activity.

Command	Usage
live_key_stop	Stops the “live control” or key/screen streaming session.
back	Simulates the Android “Back” button press.
home	Simulates the Android “Home” button press.
text	Inputs text into the currently focused text field.
power	Simulates the phone’s power button (screen on/off).
launch_app	Launches a specified application on the device.
set_vnc_mode	Enables or configures VNC-like remote viewing/streaming mode.
black_blank_screen	Displays a fully black overlay to hide all activity.
live_key	Starts live control mode (e.g., real-time screen/key streaming).
get_apps	Retrieves and sends the list of installed applications.
