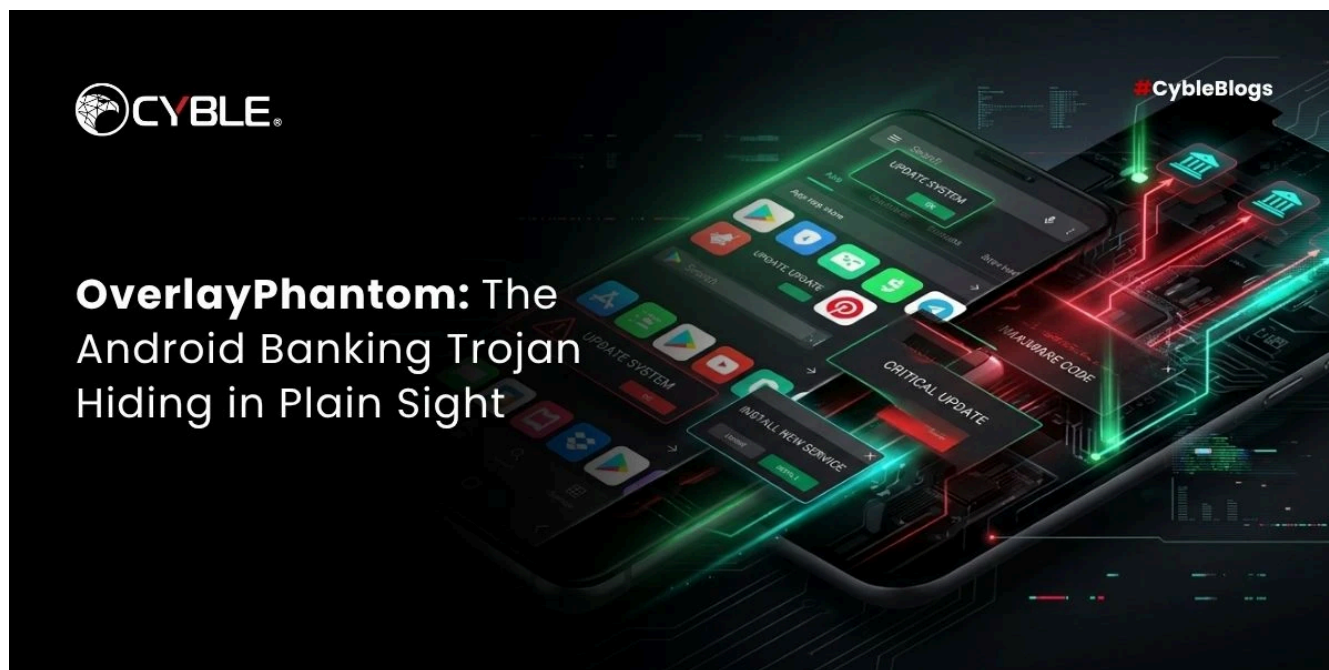


OverlayPhantom-android-banking-trojan-hiding In Plain Sight

 cyble.com/blog/overlayphantom-android-banking-trojan

May 27, 2026



Executive Summary

Cyble Research and Intelligence Labs (CRIL) has identified a novel Android banking trojan, dubbed OverlayPhantom, actively distributed in the wild via malicious URLs.

The [malware](#) employs a two-stage infection chain, using a dropper application that impersonates trusted platforms, including the official Austrian government identity application, ID Austria, and the widely used consumer platform TikTok, to deceive victims into installing it.

Once deployed, OverlayPhantom masquerades as “Google Play Services” and abuses Android’s Accessibility Service to gain persistent, elevated control of the infected device.

The malware is capable of executing over 30 remote commands, conducting real-time screen streaming, performing overlay attacks using embedded HTML phishing pages, and exfiltrating harvested credentials to a multi-port Command and Control (C&C) infrastructure.

Victimology

OverlayPhantom, active since May 2025, targets over 180 applications across banking, financial services, and cryptocurrency platforms, spanning 10 countries, including the United States, Australia, Germany, France, Belgium, Finland, the Netherlands, Italy, Spain, and the United Kingdom.

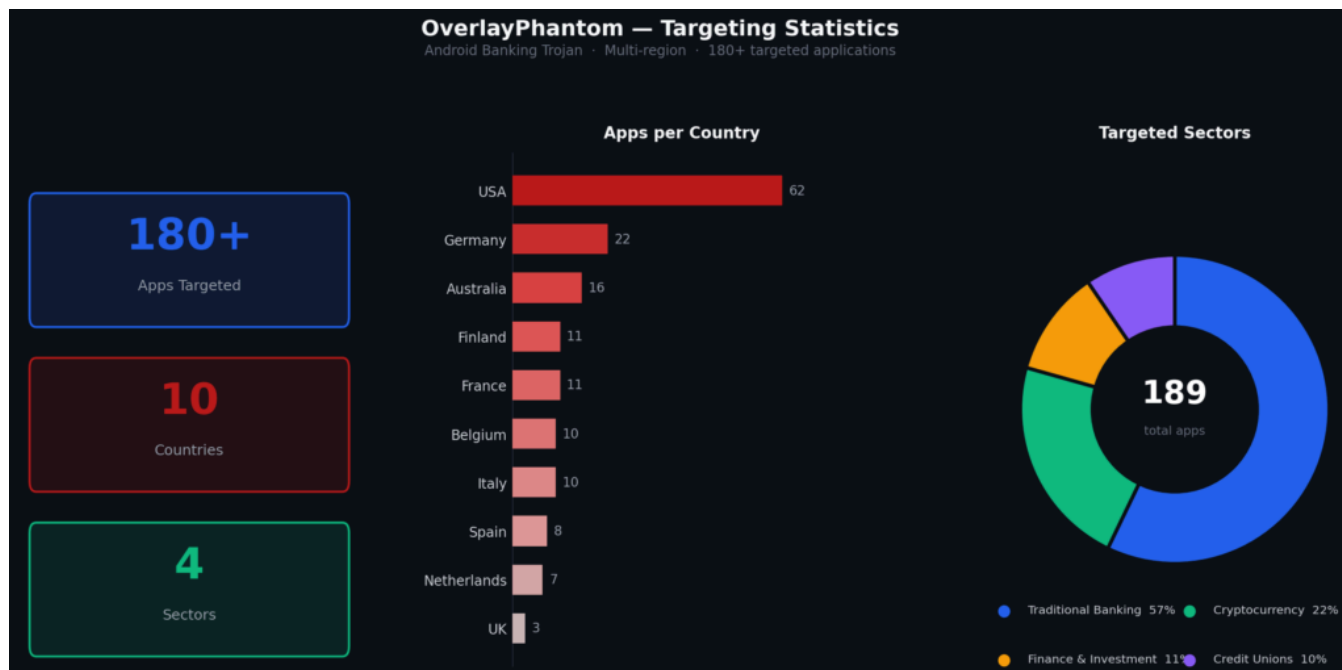


Figure 1 – OverlayPhantom's targets

The breadth of its targeting, combined with its operational sophistication, indicates a financially motivated [threat actor](#) with the capability and intent to conduct large-scale fraud across Western markets.

See What 2025 Really Looked Like Across Every Region
Global | APAC | Europe | North America | META | Australia & New Zealand
Get Your Free Reports Today!

Key Takeaways

- OverlayPhantom is a sophisticated Android banking trojan distributed via phishing URLs that impersonate high-trust applications.
- The malware deploys via a dropper application that simulates a fake Google Play service update and guides victims to enable the Accessibility Service.
- It abuses Android's Accessibility Service to silently monitor foreground app activity, intercept user input, simulate gestures, and maintain persistent control over the infected device.
- The malware currently targets over 180 banking, finance, and cryptocurrency applications across 10 countries using embedded WebView-based HTML phishing overlays that are visually indistinguishable from the legitimate apps they impersonate.
- C&C communication is handled over three dedicated non-standard ports — 9091 for command dispatch, 9092 for device status reporting, and 9090 for screen streaming.
- OverlayPhantom supports over 30 remote commands, enabling the threat actor to perform automated gestures, manipulate clipboard content, lock the device screen, display fake notifications, and capture PIN or password input via custom overlay windows.

- A built-in JPEG-based screen streaming capability, powered by Android's MediaProjection API, grants the threat actor near real-time visual access to the victim's device screen with minimal bandwidth overhead.

Overview

During an investigation into government-themed URL impersonation, Cyble Research and Intelligence Labs (CRIL) uncovered a previously undocumented Android banking trojan, dubbed **OverlayPhantom**.

The malware is being actively distributed in the wild through malicious URLs and masquerades as legitimate, high-trust applications to deceive users into installing it. CRIL's analysis indicates that OverlayPhantom has been active since early May 2025.

The initial sample discovered was hosted at `hxxps://bitlrewards-app[.]com/api/download/IDAustria`, distributing a malicious APK masquerading as ID Austria — the official Austrian government digital identity application.

The choice of this lure is significant, as impersonating a government identity service creates a strong [social engineering](#) pretext, particularly for victims who may be prompted to grant sensitive permissions under the guise of identity verification.

A second sample attributed to the same malware was identified impersonating TikTok and appeared to target users in Spain. The use of a high-popularity consumer application as a secondary lure indicates the threat actor is deliberately diversifying their distribution strategy across both institutional and consumer-facing decoys.

Although the distribution URL and observed sample appeared to target Austria, source code analysis revealed that OverlayPhantom is configured to target more than 180 applications across banking, financial services, and cryptocurrency platforms in multiple geographies, including the United States, Australia, Germany, France, Belgium, Finland, the Netherlands, Italy, Spain, and the United Kingdom. This wide targeting scope suggests a financially motivated threat actor operating a scalable and well-resourced campaign.

The malware abuses Android's Accessibility Service, a recurring technique among sophisticated Android banking trojans, to gain elevated control over the infected device.

Observed capabilities include overlay attacks to harvest credentials by displaying fraudulent screens over legitimate banking applications, real-time screen streaming to exfiltrate sensitive on-screen data, and automated action execution to perform unauthorized transactions and interactions without user awareness.

The combination of government and consumer app impersonation, wide geographic and sector targeting, and abuse of core Android accessibility features positions OverlayPhantom as a significant threat to both retail banking customers and cryptocurrency users across Western markets.

Technical Analysis

OverlayPhantom employs a two-stage delivery mechanism, utilizing a dropper application as the initial infection vector before deploying the core malware payload.

Upon execution, the dropper presents the victim with a convincing fake Google Play update screen, social-engineering the user into voluntarily installing what appears to be a legitimate system update. This technique effectively bypasses user suspicion by leveraging the inherent trust associated with the Google Play ecosystem.

Additionally, the dropper includes an interactive step-by-step tutorial that guides the victim through enabling the Accessibility Service.

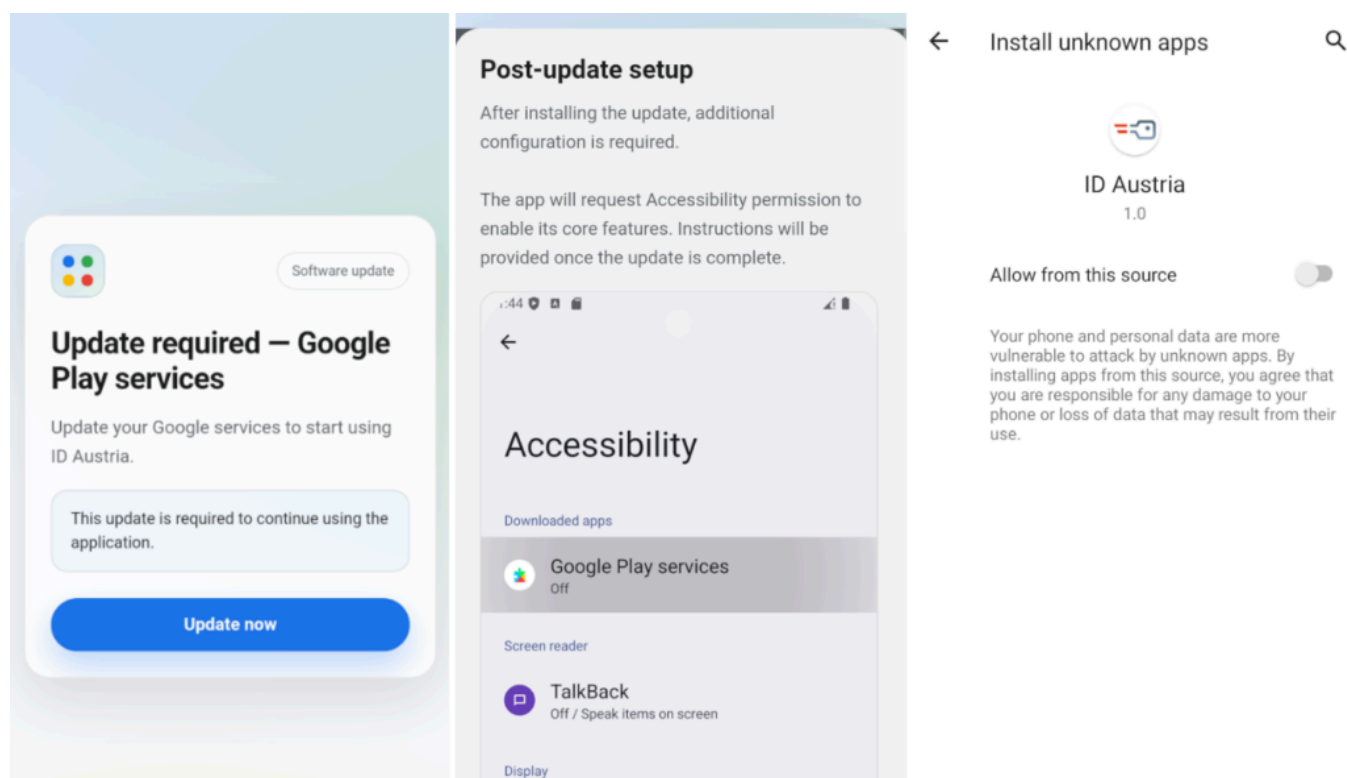


Figure 2 – Google Play Update lure to install OverlayPhantom

Once the victim completes the installation, the OverlayPhantom payload is installed onto the device. The malware immediately prompts the user to grant Accessibility Service permissions. Subsequently, it masquerades as “Google Play Services”, making it significantly harder for the victim to identify or remove the malicious application.

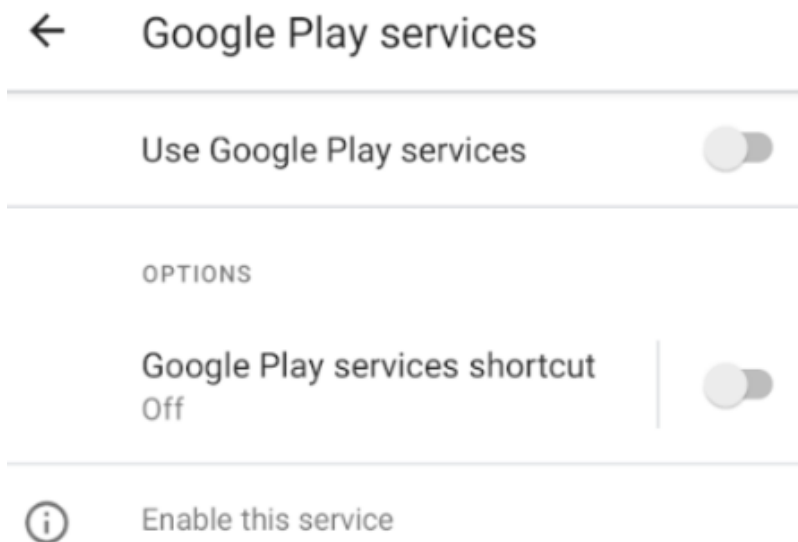


Figure 3 – Hiding itself as Google Play Services and prompting to enable Accessibility Service

Command & Control Communication

Once the victim grants the Accessibility Service permission, OverlayPhantom immediately establishes communication with its Command and Control (C&C) server at `hxxps://199.217[.]99[.]122`, utilizing a socket-based connection for real-time bidirectional communication between the infected device and the threat actor's infrastructure.

Notably, the malware does not rely on a single communication channel. Instead, it distributes its C&C traffic across three dedicated ports, as listed below:

Port	Description
9092	Used for device status and reporting
9091	Used as a Command and Control channel
9090	Used for screen streaming

Over OverlayPhantom, port 9091 receives operator-issued commands, executes them on the victim's device, and subsequently relays stolen data or execution status reports back to the server.

Analysis of the malware's source code reveals that OverlayPhantom can execute over 30 distinct commands, reflecting the breadth of control the threat actor can exert over a compromised device.

```

public final void ac9c6e8bab6443119db3294bf1b4a003(String str) throws JSONException, NumberFormatException {
    List list;
    try {
        JSONObject jsonObject = new JSONObject(str);
        String string = jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("NjsrQFk7MA=="));
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("ITU2")")) {
            String[] strArrSplit = jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("JTU/QVc0MA==")).split("-");
            float f = Float.parseFloat(strArrSplit[0]);
            float f2 = Float.parseFloat(strArrSplit[1]);
            DisplayMetrics displayMetrics = getResources().getDisplayMetrics();
            Path path = new Path();
            path.moveTo((f / 100.0f) * displayMetrics.widthPixels, (f2 / 100.0f) * displayMetrics.heightPixels);
            GestureDescription.StrokeDescription strokeDescription = new GestureDescription.StrokeDescription(path, 0L, 100L);
            GestureDescription.Builder builder = new GestureDescription.Builder();
            builder.addStroke(strokeDescription);
            dispatchGesture(builder.build(), null, null);
            return;
        }
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("MTszT1QwACdd")")) {
            String[] strArrSplit2 = jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("JTU/QVc0MA==")).split("-");
            c2846711d39d4dcf8aedc1f96945db2d(Float.parseFloat(strArrSplit2[0]), Float.parseFloat(strArrSplit2[1]));
            return;
        }
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("OTsoSmgnMTVe")")) {
            String[] strArrSplit3 = jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("JTU/QVc0MA==")).split("-");
            abf5aa5822074858892a45b78422aa06(Float.parseFloat(strArrSplit3[0]), Float.parseFloat(strArrSplit3[1]));
            return;
        }
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("JiMvXV0=")")) {
            String[] strArrSplit4 = jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("JTU/QVc0MA==")).split("-");
            f8454e4619eb4e75b92db4663bab56b7(Float.parseFloat(strArrSplit4[0]), Float.parseFloat(strArrSplit4[1]), Float.parseFloat(strArrSp:
            return;
        }
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("MSYnWg=")")) {
            bb817f1a112d4a95a3b6a8c4364938bf(jsonObject.getString("zi.feeaaef7c1284853b9bfb09e8c6c0228("JTU/QVc0MA==")));
            return;
        }
        if (string.equals("zi.feeaaef7c1284853b9bfb09e8c6c0228("OiQjQ2owNyNDTCY=")")) {
            performGlobalAction(3);
        }
    }
}

```

Figure 4 – Commands received from the server

The full command set is detailed in the table below.

Command	Description
tap	Performs a Tap Gesture
doubleTap	Performs a double-tap gesture
longPress	Performs a long-press gesture
swipe	Performs a swipe gesture
draw	Performs a custom gesture path
openRecents	Opens the Recent Apps screen
switchScreen	Keep the screen on from the locked state
volumeUp	Increases Audio Volume
volumeDown	Reduces the volume
power	Open the power menu
brightSettings	Open display settings
back	Performs back action
home	Performs home action
buf	Set the attacker-provided text to the clipboard content
target	Malware receives the list of target applications
startStreamJpeg	Initiates screen streaming
stopStreamJpeg	Stops screen streaming
startStreamACNode	Start sending Accessibility node information
stopStreamACNode	Stop sending Accessibility node information
ping	Maintaining the keepalive mechanism
Pong	Maintaining the keepalive mechanism
stub	Not implemented
register	Registers the device with BotID
resendInj	Reset target package injection list

rmResend	Deletes the file received from the server
switchOffScreen	Lock the device screen
blankScreen	Display a blank overlay screen
blankScreenRm	Removes the blank overlay screen
pinj	Display an overlay window to collect a PIN, a password or a draw pattern
notif	Displays a fake notification banner using the target app icon and name

Overlay Attack: Targeting Banking, Finance, and Cryptocurrency Applications

OverlayPhantom leverages the Accessibility Service to continuously monitor foreground application activity on the infected device. The malware maintains a hardcoded target application list embedded in its source code and includes a collection of counterfeit HTML [phishing](#) pages bundled directly into the APK's resources.

These pages are meticulously crafted to impersonate legitimate banking and financial applications, deceiving victims into submitting their credentials or payment card details.

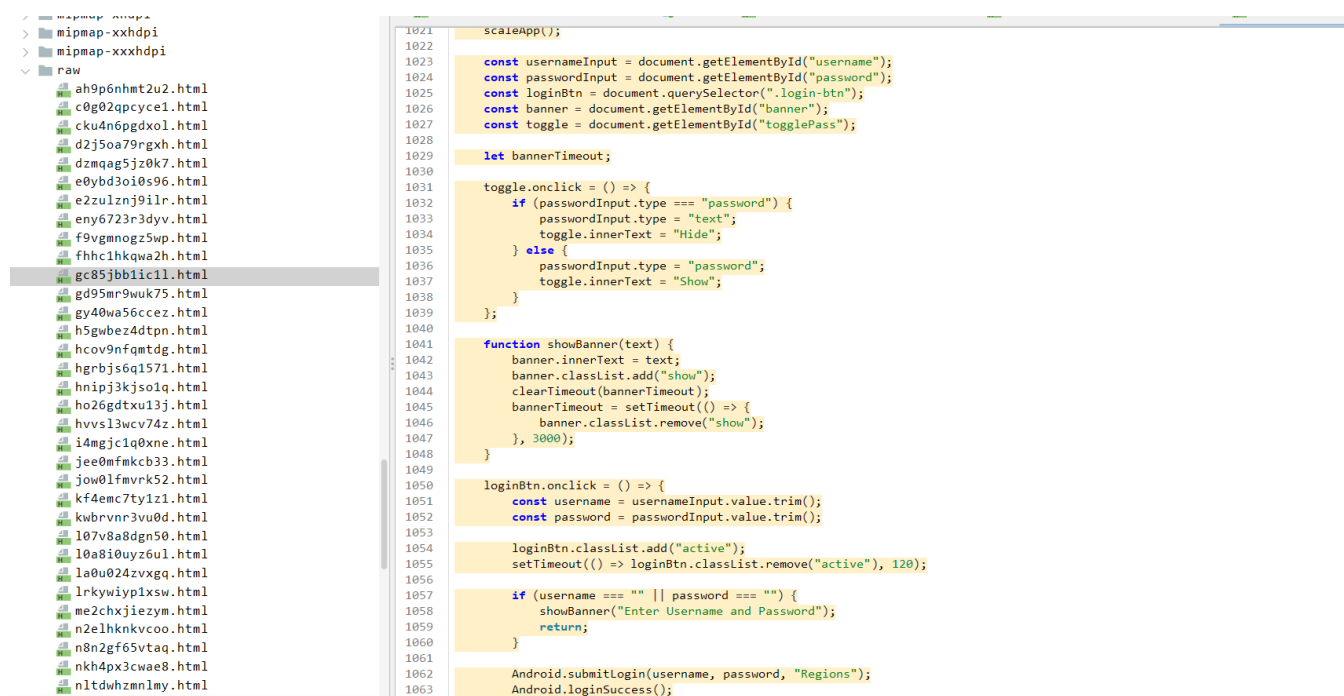


Figure 5 – Counterfeit HTML phishing pages in the APK file

When the victim launches a banking or financial application, OverlayPhantom silently checks whether the application's package name is present in its target list.

Upon a positive match, the malware retrieves the corresponding phishing page from its internal resources, renders it in an embedded WebView, and displays it as a seamless overlay window directly above the

legitimate application. From the victim's perspective, the experience is indistinguishable from interacting with the genuine application.

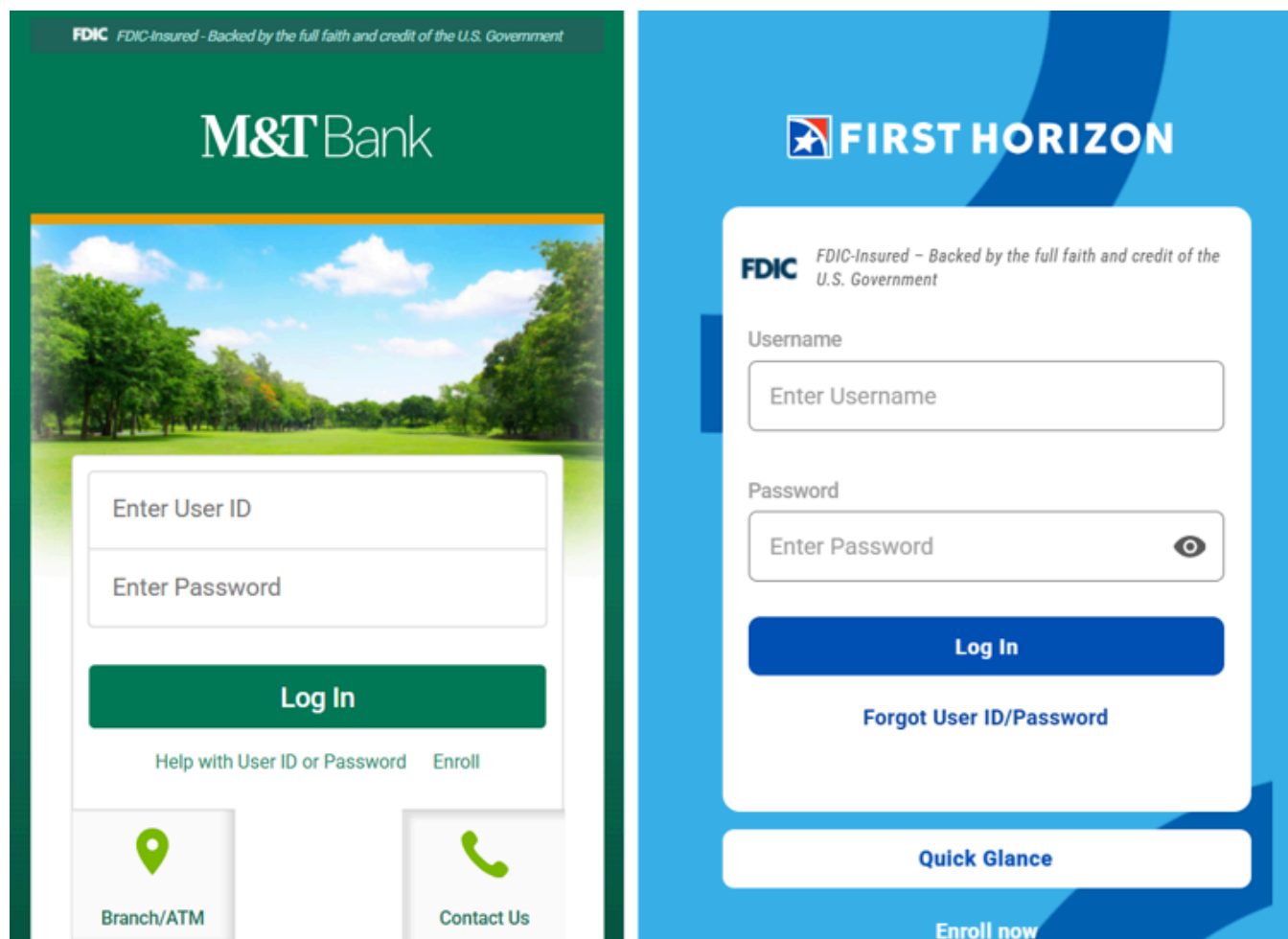


Figure 6 – Fake banking pages designed to steal banking credentials

Once the victim enters their credentials into the fraudulent overlay, OverlayPhantom harvests the submitted username, password, or card details and silently exfiltrates the stolen data to the C&C server, completing the credential theft cycle without raising any visible indication of compromise on the device.

Screen Streaming

OverlayPhantom provides real-time screen streaming via JPEG, which can be controlled remotely via the [startStreamJpeg](#) and [stopStreamJpeg](#) commands.

Upon receiving the [startStreamJpeg](#) command, the malware initiates a screen capture using Android's MediaProjection API, creating a VirtualDisplay instance named jpeg-stream and attaching it to an ImageReader to continuously capture the device's screen.

The output is resized to a fixed width of 540 pixels, with the height dynamically calculated to preserve the victim device's native screen aspect ratio.

```

, ... }

public class Ca2622a4fc6e5 extends Activity {
    @Override // android.app.Activity
    public final void onActivityResult(int i, int i2, Intent intent) {
        super.onActivityResult(i, i2, intent);
        if (i == 3001 && i2 == -1) {
            Intent intent2 = new Intent(this, (Class<?>) Ce899074c98a7.class);
            intent2.putExtra("zi.feeaaef7c1284853b9bfb09e8c6c0228("NjsiSA==)", i2);
            intent2.putExtra("zi.feeaaef7c1284853b9bfb09e8c6c0228("MTUyTA==)", intent);
            startForegroundService(intent2);
        }
        finish();
    }

    @Override // android.app.Activity
    public final void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        startActivityForResult(((MediaProjectionManager) getSystemService("media_projection")).createScreenCaptureIntent(), IAdLoadingError.LoadEi
    }
}

```

Figure 7 – Initiating Screen Capturing

While screen capture is active, the malware establishes a TCP connection to the C&C server on port 9090. Before transmitting any frames, it sends a bot and session identifier, derived from the malware's configured Bot ID and the device ID, to register the streaming session with the operator.

The malware then enters a continuous capture loop, calling *acquireLatestImage()* to fetch the latest screen frame, converting it into a Bitmap, compressing it as a JPEG, and writing the resulting bytes directly to the socket.

This provides the threat actor with near-real-time visibility into the victim's screen activity while keeping bandwidth consumption lower than that of raw frame transmission.

The streaming loop incorporates resilience logic to handle interruptions gracefully. If no frame is available, the malware briefly sleeps and resumes polling. In the event of a socket failure, it increments a retry counter, pauses for approximately two seconds, closes the active stream and socket, and attempts to re-establish the connection.

Once the retry threshold is exceeded, the streaming flag is disabled to prevent an indefinite number of reconnection attempts. The operator can terminate the stream at any time by issuing the *stopStreamJpeg* command, which flips the streaming state and invokes the corresponding service logic to cleanly shut down the capture session.

Conclusion

OverlayPhantom represents a mature and methodically engineered Android banking threat. From its deceptive dropper stage — which exploits user trust in the Google Play ecosystem — to its abuse of the Accessibility Service, multi-port C&C architecture, overlay-based credential harvesting, and real-time screen streaming, the malware demonstrates a high degree of operational sophistication.

Its broad targeting scope, encompassing over 180 banking, financial, and cryptocurrency applications across 10 countries at the time of this analysis, further underscores the scale of the threat actor's ambitions. Based on the observed functionality, we anticipate the threat actor's targeting scope and potential blast radius will continue to expand.

The techniques employed by OverlayPhantom are not novel in isolation, but their combination, particularly the use of government and consumer application lures, hardcoded phishing overlays, and granular remote-control capabilities, reflects a threat actor with both the technical capability and the strategic intent to conduct large-scale financial fraud across multiple regions.

Organizations and individuals operating in the targeted geographies should treat this threat with a high degree of urgency.

Our Recommendations

We have listed some essential [cybersecurity](#) best practices that serve as the first line of defense against attackers. We recommend that our readers follow the best practices given below:

- **Install Apps Only from Trusted Sources:**

Download apps exclusively from official platforms, such as the [Google Play Store](#). Avoid third-party app stores or links received via SMS, social media, or email.

- **Be Cautious with Permissions and Installs:**

Never grant permissions and install an application unless you're certain of an app's legitimacy.

- **Watch for Phishing Pages:**

Always verify the URL and avoid suspicious links and websites that ask for sensitive information.

- **Enable Multi-Factor Authentication (MFA):**

Use MFA for banking and financial apps to add an extra layer of protection, even if credentials are compromised.

- **Report Suspicious Activity:**

If you suspect you've been targeted or infected, report the incident to your bank and local authorities immediately. If necessary, reset your credentials and perform a factory reset.

- **Use Mobile Security Solutions:**

Install a mobile security application that includes real-time scanning.

- **Keep Your Device Updated:**

Ensure your Android OS and apps are updated regularly. Security patches often address vulnerabilities exploited by malware.

MITRE ATT&CK® Techniques

Tactic	Technique ID	Procedure
Initial Access (TA0027)	Phishing (T1660)	OverlayPhantom is distributed via phishing sites
Persistence (TA0028)	Event Triggered Execution: Broadcast Receivers (T1624.001)	OverlayPhantom implemented a broadcast receiver for screen capturing
Defense Evasion (TA0030)	Hide Artifacts: Suppress Application Icon (T1628.001)	OverlayPhantom hides its icon
Defense Evasion (TA0030)	Obfuscated Files or Information (T1406)	Malware uses obfuscated strings
Defense Evasion (TA0030)	Masquerading: Match Legitimate Name or Location (T1655.001)	OverlayPhantom masquerades as Google Play Service
Credential Access (TA0030)	Abuse Accessibility Features (T1453)	OverlayPhantom abuses Accessibility service
Discovery (TA0032)	Software Discovery (T1418)	OverlayPhantom checks the installed application list against the target list
Collection (TA0035)	Screen Capture (T1513)	OverlayPhantom captures screen content
Command & Control (TA0037)	Application Layer Protocol (T1437)	OverlayPhantom communicates with C2 over TCP
Command & Control (TA0037)	Non-Standard Port (T1509)	OverlayPhantom uses a non-standard port
Exfiltration (TA0036)	Exfiltration Over C2 Channel (T1646)	OverlayPhantom exfiltrates data to the C&C server

Indicators of Compromise (IOCs)

Indicators	Indicator type	Description
hxtps://bitlrewards-app[.]com/api/download/IDAustria	URL	Distribution URL
199.217[.]99[.]122	IP	C&C server
9ef37376bfaa18e193cc72218924ad8ebf56d2667d348f0eae5ae6ec45ab8775f8b614a2918378063d6e6655b676ceb52ae65b1510e2cc08087fcac31acb7aeb8ddc1f2a75f3d5b5bd054a5367bd5015ebc90f3453d63c7cce438c12dc2ae86a	FileHash-SHA256	OverlayPhantom Hash