Anatsa's Latest Updates | ThreatLabz

By Himanshu Sharma

Published: 2025-08-21 · Archived: 2026-04-05 19:00:48 UTC

Technical Analysis

Unlike in previous campaigns, the latest Anatsa campaigns implement various anti-analysis techniques. The parent installer now decrypts each string at runtime using a dynamically generated Data Encryption Standard (DES) key, making it more resistant to static analysis tools. Furthermore, Anatsa has enhanced its evasion strategies by performing emulation checks and verifying device models to bypass dynamic analysis environments.

After confirming that the C2 server is active and the device meets the necessary criteria, the installer proceeds to download Anatsa as an update. If these conditions are not met, the application displays a file manager view to the user, maintaining the appearance of a legitimate application, as shown in the figure below.

Figure 2: Example behavior of the Anatsa installer depending on the result of anti-analysis checks.

To evade detection across infected systems, the application package name and installation hash are periodically altered.

The core payload has been updated to incorporate a new keylogger variant of Anatsa. Additionally, the malware utilizes a well-known Android APK ZIP obfuscator for enhanced evasion. The DEX payload is concealed within a JSON file, which is dynamically dropped at runtime and promptly deleted after being loaded.

The APK uses a corrupted archive to hide a DEX file, which is deployed during runtime. This archive has invalid compression and encryption flags, making it hard for static analysis tools to detect. Since these tools depend on standard ZIP header checks in Java libraries, they fail to process the application. Despite this, the application will run on standard Android devices.

The figure below shows a malformed archive used by Anatsa to evade analysis.

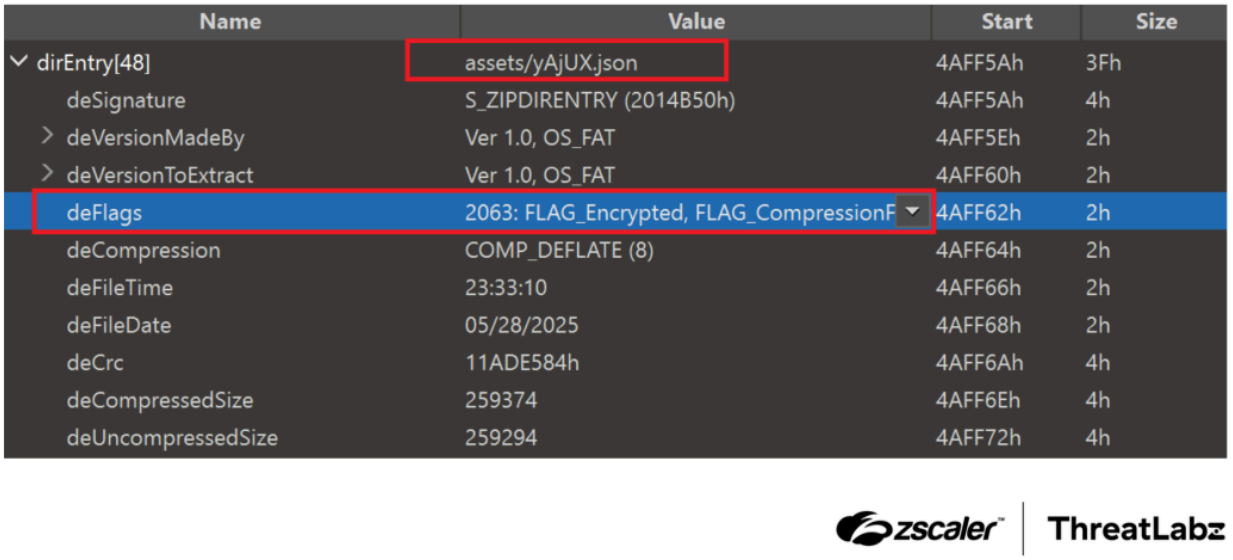

Figure 3: Example headers of a malformed archive used by Anatsa to evade analysis.

Once installed, Anatsa requests accessibility permissions from the user. If granted, the malware automatically enables all the permissions specified in its manifest file, which include the following:

- `SYSTEM_ALERT_WINDOW`
- `READ_SMS`
- `RECEIVE_SMS`
- `USE_FULL_SCREEN_INTENT`

Anatsa connects to the server to request specific commands and encrypts C2 communication using a single byte XOR key (66 in decimal). The following JSON structure contains an example of Anatsa's configuration data.

```
 {
 "hide_sms": null,
 "gauth_confirm": null,
 "lock_device": null,
 "extensive_logging": null,
 "injects_version": 254,
 "keyloggers_version": 403,
 "commands": null,
 "installed_apps_count": 37,
 "domains": [
   "http://185.215.113.108:85/api/",
   "http://193.24.123.18:85/api/",
   "http://162.252.173.37:85/api/"
 ],
 "active_injects": null
}
```


Anatsa primarily exfiltrates credentials by displaying fake banking login pages, which are downloaded from its C2 server. These pages are tailored based on the financial institution applications detected on the user's device.

The list of financial institutions and corresponding injection pages targeted by Anatsa appears to be a work in progress and continues to evolve. Out of the 831 applications targeted for keylogging, many of these injection pages were incomplete or unavailable. For example, the injection content at the time of analysis for the Robinhood application is shown below:


```
 {
 "application": "com.robinhood.android",
 "html": "Scheduled maintenance We're working on enhancements and will have  things back up and running soon. Th
 "inj_type": "bank"
}
```


Explore more Zscaler blogs
