INCLUDE SECURITY

Remediation Test of Geph's Mobile and Web Applications

EXECUTIVE SUMMARY

Engagement Details

Client	Geph
Engagement Scope	Mobile and Web Applications
Original Assessment Schedule	Jan 16th, 2023 - Feb 6th, 2023
Remediation Test Dates	May 22nd, 2023 - May 22nd, 2023

Remediation Test Update: Technical Findings Summary

The information below summarizes the observations of the Includesec team during the course of the remediation test intended to reproduce the findings as originally reported. The team attempted to bypass any added mitigations or protections put in place to hinder exploitation of the findings.

Finding	Risk Rating	Status
M1	Medium	Closed
M2	Medium	Risk Accepted
M3	Medium	Risk Accepted
M4	Medium	Partially Mitigated
M5	Medium	Risk Accepted
M6	Medium	Closed
M7	Medium	Closed
L1	Low	Future Fix Planned

MEDIUM-RISK FINDINGS

M1: [binder] Passwords Cached in Plaintext

Status: Closed

Notes:

This finding was retested and found to be remediated. The password cache was removed in commit 5fb19e444; the highlighted line was removed at line 609 of **geph4-binder/src/bindercore_v2.rs** :

```
       if verify_libsodium_password(password.to_string(), pwdhash).await {
-           self.pwd_cache.insert(username.into(), password.into());
            Ok(true)
        } else {
```

A cache of hashed authentication tokens was added instead.

M2: [bridge] No Binary Authentication in Auto Update

Status: Risk Accepted

Notes:

The client accepted the risk with the following note:

"The bridge is not really a trusted party, while the B2 backblaze is a trusted party (since it also hosts all the binaries that users download) and pretty heavily guarded, so the former trusting the latter doesn't seem like a big issue."

M3: [Android] Application Executable Signed with v1 Signature Scheme (JANUS Vulnerability)

Status: Risk Accepted

Notes:

The client accepted the risk with the following note:

"JANUS is unavoidable because we must support very old, insecure Android devices, largely due to their popularity in places third-world countries like Iran."

M4: [Android] [iOS] Daemon Commands Exposed to All Applications on Device

Status: Partially Mitigated

Notes:

This finding was retested and found to be partially mitigated. Version 4.7.12 of the **Geph** Android application now generates a UUID the first time the daemon is started which acts as a key for the RPC service. This is shown in the following snippet from file **app/src/main/java/io/geph/android/MainActivity.kt**, lines 191-198:

```
"daemon_rpc" -> {
                val rpc_key = baseContext.getSharedPreferences("GEPH_RPC_KEY", Context.MODE_PRIVATE)
                    .getString("GEPH_RPC_KEY", "key-" + UUID.randomUUID().toString())!!;
                with (baseContext.getSharedPreferences("GEPH_RPC_KEY", Context.MODE_PRIVATE).edit()) {
                    android.util.Log.i("RPC_KEY_2", rpc_key);
                    putString("GEPH_RPC_KEY", rpc_key)
                    commit()
                }
```

As shown in the following command line snippet from the Android application, the daemon no longer responds to requests without the corresponding key:

```
# busybox wget -O- -q --post-data='{"jsonrpc":"2.0","method":"basic_stats","params":[],"id":1}' --header='Content-Type: application/x-www-form-urlencoded' http://127.0.0.1:9809

[...]

wget: server returned error: HTTP/1.1 500 Internal Server Error
```

However, the RPC key was found to be logged using the native Android logging functionality, which is exposed to system applications on the device, and to all applications on the device in Android versions before 4.1. The following output from the pidcat tool shows it was possible to recover the RPC key from the Android logs:

```
$ python pidcat.py io.geph.android

[...]

RPC_KEY_2  I  key-b07a0b2c-5135-4b8b-9883-714923956efd'
```

The assessment team recommends removing the logging call from the snippet above to fully remediate this vulnerability on Android. See the Android security documentation about logging security relevant information for more details.

This vulnerability was found to be remediated on iOS. The following snippet from **Geph/ViewController.swift**, lines 16-20, shows that a random password is generated to restrict access to the daemon when the application starts:

```
        let sharedDefaults = UserDefaults(suiteName: "group.geph.io")
        if sharedDefaults?.string(forKey: DAEMON_RPC_SECRET_PATH_KEY) == nil {
            let randomString = generateRandomString(length: 20)
            sharedDefaults?.set(randomString, forKey: DAEMON_RPC_SECRET_PATH_KEY)
        }
```

The following terminal output shows that the RPC daemon no longer accepted requests without the secret:

```
iPhone:~ root# wget -O-  --post-data='{"jsonrpc":"2.0","method":"is_connected","params":[],"id":1}' --header='Content-Type: application/x-www-form-urlencoded' http://127.0.0.1:9809 --2023-05-09 07:31:39--  http://127.0.0.1:9809/
Connecting to 127.0.0.1:9809... connected.
HTTP request sent, awaiting response... 500 Internal Server Error
2023-05-09 07:31:39 ERROR 500: Internal Server Error.
```

M5: [Android] Security Relevant User Data Stored in Clear Text

Status: Risk Accepted

Notes:

The client accepted the risk with the following note:

"We did not believe it would be worthwhile to put more effort into protecting passwords on Android, since a rooted phone can easily read passwords out of the WebView's local storage anyway."

M6: [client] Time-Based Client Deanonymization

Status: Closed

Notes:

This finding was retested and found to be remediated. Countermeasures were added in commit 3bfb74eab. The following was added to the end of the **get_auth_token()** function, causing the client to sleep for a random period between 3-8 seconds before making a second, potentially linkable request:

```
        smol::Timer::after(Duration::from_secs_f64(
            rand::thread_rng().gen_range(3.0, 8.0),
        ))
```

Additionally, the following line was added to the **E2eeHttpTransport.new()** function, to mitigate the client from making requests over the same long-lived HTTP connection:

```
            .pool_idle_timeout(Duration::from_secs(1)) // reduce linkability by forcing new connections
```

How far this mitigates the risk depends on the size of the anonymity set of the number of clients using each binder server. According to the client, binders process a significant number of requests, making the anonymity set large (see notes in the finding reproduction).

M7: [client] Client Did Not Validate Mizaru Keys

Status: Closed

Notes:

This finding was retested and found to be remediated. Commit e1bd469 now ensures that the hardcoded Mizaru public key of the binder is loaded in the geph4-protocol repository.

The **get_mizaru_pk()** function at line 200 of **src/binder/client.rs** now uses hardcoded public keys, instead of dynamically fetching the keys from the binder:

```
    match level {
        Level::Free => Ok(self.mizaru_free.clone()),
        Level::Plus => Ok(self.mizaru_plus.clone()),
    }
```

LOW-RISK FINDINGS

L1: [client] Registration Captcha Bypass

Status: Future Fix Planned

Notes:

The client indicated via the following comment that a more robust CAPTCHA system was being planned in a future release:

“We also did not fix any of the low-risk findings. We do plan on a better captcha system in the future, but that did not seem urgent.”