

# OmniSaga CTF – Challenge Questions Guide

## Interactive Question-Based Solving Document


---




### 1. Shonen – Hardcoded Secrets

#### Questions:

- What key activities and classes are declared in the AndroidManifest.xml?
- Can you identify the main logic inside MainActivity2.java and SecretManager.java?
- Is there a method that uses Java reflection to access a secret? What does it invoke?
- Is a native library (libgetme.so) being loaded, and what method does it expose?
- Does the library contain any hardcoded encoded secrets (e.g., hex, Base64)?
- How is the encoded string being transformed? Are XOR operations involved?
- Can you reverse the encoding logic to extract the final flag?

 Tools: Jadx, strings, objdump

 Hint: XOR key is cd7862r==. Flag format: cdf\_Flag{...}


---



### 2. Isekai – Vulnerable Android IPC Components

#### Questions:

- What exported components are present in AndroidManifest.xml (services, receivers)?
- Does MyService.java expose a FileObserver on flag.txt?
- Can an external component interact with MyService using adb or a crafted intent?
- What kind of validation happens in MyObserver.validateFlag()?
- Does the flag get stored in a file and encoded with Base64?
- Can you extract and decode the file contents manually?

 Tools: Jadx, ADB, Base64 decoders


---



### 3. Seinen – Insecure Deeplinks

#### Questions:

- Which activity handles deep links? Can you find it in DeeplinkHandlerActivity.kt?
- What kind of hash (MD5?) is used in SecurityUtils.kt?
- Can you reverse or look up the hash value to get the secret key?
- What deep link URL can be constructed using this key?
- What is stored across .hiddenA, sysconfig.tmp, logcache.dat, and cache\_meta.bin?
- Can you combine and decode them to reveal the flag?

 Sample deep link:


**adb shell am start -a android.intent.action.VIEW -d 'ctfapp://flag?action=?&key=?'**



## 4. Slice of Life – Insecure Storage

### Questions:

- Are credentials hardcoded in InsecureActivity.kt?
- Which storage locations contain flag parts: SharedPreferences, SQLite, Internal files?
- How is the Base64-encoded flag stored in SharedPreferences?
- Can you dump the SQLite database and read stored flags?
- Can you reverse an XOR cipher used on a local file?

 Pro Tip: Combine all parts and check logs for potential hints.

---



## 5. Psych&Thrill – SQL Injection

### Questions:

- What does checkLogin() in LoginActivity.kt reveal about input validation?
- Is user input directly concatenated in SQL queries?
- Can you use ' OR '1'='1 to bypass authentication?
- Is there AES decryption of user data in AESUtils.kt?
- Are the key and salt hardcoded? Can you decrypt data using them?

 Tools: SQLite browser, AES decryptor (Python or online)


---



## 6. For the Glory of Humanity! – Steganography

### Questions:

- Are there any images provided in previous challenges?
- What tools can be used to extract hidden data from images (e.g., steghide, zsteg)?
- Can you reorder and reconstruct the hidden data to form a complete flag?
- What format does the final flag follow?

 Flag format: ShingekiNoKyojin{...}


---



## 7. Fight. Fight. – UI Fix & Hidden Functions

### Questions:

- Does the “Fight Fight” button route to the wrong activity? How was it fixed?
- Was the function checkFlag() replaced with checkProcess()?
- Is an encoded image being used to extract a flag?
- How does the flag validation work?

 Tools: Image encoders/decoders, log inspection

---



## 8. Frida Challenge-1 – Bypassing Security

### Questions:

- What classes contain root and debugger checks (isDeviceRooted, isDebuggerAttached)?
- Can you override them using Frida to always return false?
- How is the flag decryption implemented? Can you hook and log the output?
- What functions control UI navigation and flag verification?

 Tools: Frida, JavaScript hook scripts


---



## 9. Frida Challenge-2 – PIN Validation

### Questions:

- Where is validatePIN() implemented (native-lib)?
- Is the correct PIN hardcoded (e.g., 7357)?
- Can Frida be used to hook or bypass this check?

 Use native hooking with Frida

---



## 10. Frida Challenge-3 – License Verification

### Questions:

- How does checkLicense() behave in native C++ code?
- Is there a hardcoded key (e.g., VALID-123-SECRET)?
- Can you bypass this check by hooking checkLicense() to return true?
- Does system time affect encryption/decryption?

☒ Use Frida to force execution flow

---

**Note : Signature that support for this app is available in the assets folder in the name key2.jks**

**Or download link to the github :**

[OmniSaga](#)