**ANÁLISIS ESTÁTICO SOBRE EL CÓDIGO APK Y SUS DIRECTORIOS**

1. Tipo de archivo a analizar **app-arm64-v8a-release.apk**
2. Se procede a descompilar el APK con la herramienta **apktool**  dando como resultado el directorio **app-arm64-v8a-release/** con sus subdirectorios y archivos.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

1. Una vez dentro del **app-arm64-v8a-release/** se procede a listar su contenido, Dando como resultado :

* 8 Directorios
* 2 Archivos

Imagen de la pantalla de un computador

El contenido generado por IA puede ser incorrecto.

1. Se procede a buscar permisos sensibles en el archivo **AndroidManifest.xml** dando el siguiente resultado:

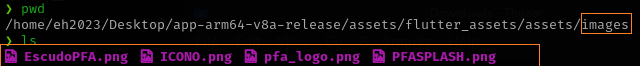
* android.permission.INTERNET
* Riesgo alto al utilizar Internet. Recomendación utilizar bajo protocolo HTTPS
* android.permission.USE\_BIOMETRIC y android.permission.USE\_FINGERPRINT
* Permisos de autenticación biométrica. Su implementación debe ser robusta
* android:exported="true"
* La actividad principal ( io.flutter.embedding.android.FlutterFragmentActivity) tiene android:exported="true". Esto es necesario para que la aplicación se inicie. Sin embargo, si esta actividad maneja datos confidenciales o realiza acciones críticas, deberá asegurarse de que esté debidamente protegida contra posibles ataques externos.
* **android:exported="true" en el ProfileInstallReceiver:**
* También androidx.profileinstaller.ProfileInstallReceiver exporta y tiene el permiso android.permission.DUMP. Si bien android.permission.DUMP es un permiso a nivel de sistema, la exportación de un receptor podría ser un punto a investigar. Asegúrese de que las acciones que gestiona ( androidx.profileinstaller.action.INSTALL\_PROFILE, SKIP\_FILE, SAVE\_PROFILE, BENCHMARK\_OPERATION) no sean susceptibles de abuso por parte de otras aplicaciones, incluso con el requisito de permiso.
* La aplicación define un permiso personalizado: com.example.app\_sifam.DYNAMIC\_RECEIVER\_NOT\_EXPORTED\_PERMISSIONcon protectionLevel="signature". Esto significa que solo las aplicaciones firmadas con el mismo certificado pueden usar este permiso.

1. El archivo **apktool.yml** contiene metadatos sobre cómo se construyó y descompiló el APK:

* **compressionType: false**:
* Indica que el APK no está comprimido. Si bien esto no es una vulnerabilidad en sí misma, un APK no comprimido es más grande y podría facilitar ligeramente el análisis de los archivos internos.
* **forcedPackageId: '127'**:
* Forzar un ID de paquete específico no es común. Si bien no es una vulnerabilidad directa, podría indicar una configuración inusual o un proceso de construcción personalizado

1. Se procede a listar alguna información del contenido:

Algunas imágenes:

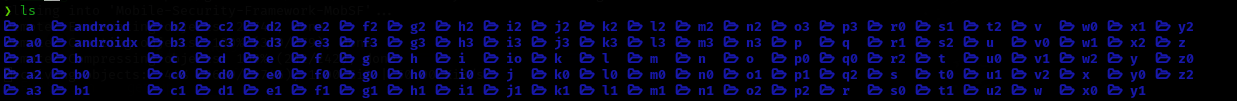


Palabras claves como API, PASS, KEY

Texto

El contenido generado por IA puede ser incorrecto.

Código ofuscado:



Conclusión:

Con base al análisis realizado en términos de seguridad sobre el APK, utilizando herramientas de código abierto y la distribución Kali Linux, se puede concluir por el momento que en términos generales la aplicación no presenta vulnerabilidades críticas.

No obstante re recomienda realizar un análisis más exhaustivo sobre el código y el comportamiento de la aplicación en tiempos de ejecución.