Для анализа я выбрал намеренно уязвимое приложение damn-vulnerable bank:
https://github.com/rewanthtammana/Damn-Vulnerable-Bank

PDF отчет из mobSF можно найти по ссылке на мой github (в папке task_2/mobsf_report.pdf):
https://github.com/numoworld/aton

При статическом анализе было выявлено несколько серьезных уязвимостей:

1. Используется настройка usesClearTextTraffic=True, которая допускает использование незащищенных протоколов для обмена информацией с сервером. Эта настройка отключена по умолчанию начиная с Android 9. Даже если данные шифруются внутри кода, реверс-инжиниринг позволит реконструировать функции шифрования и дешифрования. Это приведет к тому, что любой злоумышленник, прослушивающий трафик, будет иметь возможность читать обмениваемые данные между клиентом и сервером.
   Для устранения можно рекомендовать несколько действий:
   1) Установить usesClearTextTraffic=False, либо же обновить минимальную версию API до 28 (Android 9)
   2) Использовать для обмена сообщениями исключительно защищенные протоколы, например HTTPS через TLS >= 1.2
   3) Усилить обфускацию кода, чтобы усложнить реверс-инжиниринг функций шифрования и дешифрования

2. Используется настройка доверия сертификатам пользователя по умолчанию (trust-anchors -> certificate src='user'). Такая настройка является дефолтной для Android API < v24. Доверие установленным пользователям сертификатам может позволить злоумышленнику установить свой сертификат и провести атаку mitm, подменив сертификаты.
   Для устранения можно рекомендовать следующие действия:
   1) Убрать доверие сертификатам пользователя по умолчанию в networkSecurityConfig.
   2) Добавить CA Pinning, усложнив подмену сертификатов.

3. Несколько Activity не защищены разрешениями (на самом деле, там ещё есть активити, позволяющее посмотреть баланс пользователя). Рекомендуется защитить их разрешениями.

4. Рекомендуется запретить бэкап данных для приложения.

5. Допускаются логи с содержанием чувствительных данных. Например, можно найти файл, в котором логается “access_token” запроса к банку. Рекомендуется “отфильтровать” информацию, помещаемую в логи.

6. Имеются hard-coded секреты, например google-api token. Рекомендуется либо помещать в отдельно зашифрованный файл, либо получать токен через https.

7. Приложение может записывать данные в External Storage, доступ к которому имеют все приложения. Рекомендуется убрать возможность и функционал записи данных во внешнее хранилище.