Отчет о тестировании требований безопасности

|  |  |  |
| --- | --- | --- |
| Наименование приложения | | My MTS |
| Версия | | 2.4.7 |
| Дата проверки | | 04.03.2019 |
| Проверил | | Томашевич Константин |
| Конфиденциальная информация | |  |
| **Требования безопасности** | | **Результат** |
| 1 | Ненадежное хранение данных | - |
| 1.1 | Локальное хранение конфиденциальных данных  STORAGE-1 и STORAGE-2 | - |
| 1.1.1 | наличие разрешений AndroidManifest.xml для read/write во внешнем хранилище | Есть разрешения на чтение/запись в external storage. Я поискал файлы приложения на external storage и однозначно приложению принадлежит только кэш `MapKit`. Поэтому скорее всего ничего конфиденциального они в external storage не хранят. |
| 1.1.2 | наличие прав доступа к файлам MODE\_WORLD\_READABLE или MODE\_WORLD\_WRITABLE | Все SharedPreferences открываются с MODE\_PRIVATE. Есть один вызов getDir с MODE\_PRIVATE. Не нашёл других методов, которые могли бы использовать MODE\_WORLD\_\*. |
| 1.1.3 | наличие классов и функций:  SharedPreferences;  FileOutPutStream;  getExternal \*;  getWritableDatabase;  getReadableDatabase;  getCacheDir;  getExternalCacheDirs. | **SharedPreferences**  Используются в LocalStorage, ModelStorage, WidgetStorage.  В LocalStorage, по моим наблюдением, хранятся куки. При этом все изменения и запросы значений LocalStorage записываются в дебажный лог, что фактически делает настройки из LocalStorage равнозначными MODE\_WORLD\_READABLE настройкам. Пока не установил хранится ли там что-то действительно важное, но есть подозрения что там проскакивает как минимум телефонный номер пользователя.  WidgetStorage льёт многие изменения из SharedPreferences в ReportUtil, который льёт всё в YandexMetrica. Пока не установил важность информации и можно ли её перехватить из метрики яндекса.  **FileOutputStream**  Используется в WidgetStorage и StepsFragment для сохранения каких иконок. В том числе есть сохранение на external storage, но оно скорее всего инициируется именно пользователем, так как метод зовётся share.  **FileInputStream**  WidgetStorage грузит какие-то иконки, но они находятся во внутренней папке, открытой с помощью getDir с MODE\_PRIVATE.  **getExternal\***  StepsFragment использует getExternalFilesDir(Environment.DIRECTORY\_PICTURES) в методе share. Исходя из контекста это сохранение чего-либо пользователем, поэтому уязвимостью не является.  **getWritableDatabase**  Не нашёл использований.  **getReadableDatabase**  Не нашёл использований.  **getCacheDir**  Не нашёл использований.  **getExternalCacheDirs**  Не нашёл использований. |
| 1.1.4 | конфигурационные файлы build configs, такие как local.properties, gradle.properties | Нашёл кучу \*.properties файлов, но там только версии плагинов.  В манифесте нашёл такую интересную строку:  <meta-data android:name="io.fabric.ApiKey" android:value="94284c61ee7949c560f549ff5f7b374dd7c68371"/>  В MainActivity.java есть ещё такой интересный кусок кода:  MapKitFactory.setApiKey("0def7909-cf07-4c07-ad55-206c6821f5e0"); |
| 1.1.5 | файлы ресурсов | Нашёл ключ для работы с API Yandex: 8784accb-db3e-4026-bc75-dc2909ad90e1.  Нашёл ключ для работы с API Google: AIzaSyD\_swstDa1E57iRhcvDdZOwfxxwYBipX1Y.  Нашёл ключ для работы с Google Crash Reporting API: AIzaSyD\_swstDa1E57iRhcvDdZOwfxxwYBipX1Y.  Очень интересно выглядит ключ для API Google Maps, представляю полную строку из файла: <string name="google\_maps\_key">YOUR\_KEY\_HERE</string>.  Нашёл идентификатор сборки для Crashlytics (не знаю, насколько это важно): 286fcdfc-031c-4e6a-a74b-858cbf8a6f15.  Нашёл Google App Id: 1:255084819593:android:ff3f4088b8315f9a.  Пока не знаю, для чего эти строки, но выглядят они интересно:  <string name="path\_password\_eye">M12,4.5C7,4.5 2.73,7.61 1,12c1.73,4.39 6,7.5 11,7.5s9.27,-3.11 11,-7.5c-1.73,-4.39 -6,-7.5 -11,-7.5zM12,17c-2.76,0 -5,-2.24 -5,-5s2.24,-5 5,-5 5,2.24 5,5 -2.24,5 -5,5zM12,9c-1.66,0 -3,1.34 -3,3s1.34,3 3,3 3,-1.34 3,-3 -1.34,-3 -3,-3z</string>  <string name="path\_password\_eye\_mask\_strike\_through">M2,4.27 L19.73,22 L22.27,19.46 L4.54,1.73 L4.54,1 L23,1 L23,23 L1,23 L1,4.27 Z</string>  <string name="path\_password\_eye\_mask\_visible">M2,4.27 L2,4.27 L4.54,1.73 L4.54,1.73 L4.54,1 L23,1 L23,23 L1,23 L1,4.27 Z</string>  <string name="path\_password\_strike\_through">M3.27,4.27 L19.74,20.74</string> |
| 1.2 | Файлы отчетов (log files) STORAGE-3 | ActionManager.getData печает в лог всё что делает. Пока не проверил, есть ли там что-то важное.  Все вызовы Logix.getCookie летят в лог вместе с ключами и значениями. Пока не проверил, есть ли там что-то важное. Также куки хранятся в LocalStorage. В ScreenManager.java#L163 ищется значение кука phone, пока что не получилось проверить действительно ли это номер телефона пользователя.  Очень много других вызовов логинга (479+ только для дебажных сообщений), поэтому решил их все не разбирать, а использовать только динамический анализ.  При проведении динамического теста в логах ничего конфиденциального не нашёл несмотря даже на то что находил не очень хорошие методы логирования в коде (например логирование куков). Как раз таки куки почему-то logcat и не увидел, из чего я выдвигаю предположение о том, что дебаг-логирование приложения блокируется где-то на более высоком уровне. |
| 1.3 | Передача конфиденциальных данных третьей стороне STORAGE-4 | Вызывает сомнение буквально пронизавшее код приложения использования YandexMetrica: на неё логируется очень много различных событий, в том числе событие оплаты пользователем. Из-за не лучшего качества декомпилированного кода не могу быть на 100% уверенным в том, сливается через метрику что-то конфиденциальное или нет. |
| 1.4 | Поля ввода конфиденциальных данных STORAGE-5 | Во всех обнаруженных мной полях ввода отключены автоматические подсказки. |
| 1.5 | Безопасность межпроцессного взаимодействия STORAGE-6 | Очень много receiver’ов/провайдеров и сервисов, у некоторых exported стоит как true, а некоторые не эксортируются. Protection level нигде не указан, тем не менее у многих указан permission.  При более подробном рассмотрении обнаружилось, что самому приложению (а не сторонним библиотекам) принадлежит меньше четверти зарегистрированных объектов (не уверен, насколько это нормально). Если я ничего не упустил, то все единственный объект, через который могут утекать данные – сервис обновления виджетов, но он не является exported.  ContentProvider нигде не используется.  *BepaidFragment – что то странное с курсорами логингом? Следует пересмотреть.*  android.database.sqlite – не используется.  Update и Delete запросов не нашёл, только Query. |
| 1.6 | Безопасность данных в пользовательском интерфейсе STORAGE-7 | Поле ввода пароля нестандартное, но поддерживает скрытие символов пароля (и открытие по желанию пользователя). При сворачивании приложения ввод пароля автоматически не скрывается, что не очень хорошо.  В экране пополнения счёта не скрывается CVC код, что как бы совсем не очень. К слову, в поле ввода даты истечения срока действия карты можно ввести некорректный месяц (например 14-ый). В поле “имя держателя карты” включены подсказки, что не так критично, но и не очень хорошо.  Следует отметить довольно значительную дыру в авторизации – не вводить номер и пароль можно несколько дней подряд, а пользование приложением в этом промежутке ничем не ограничено (кроме пинкода, установка которого необязательна), то есть теоретически другой человек может взять телефон с приложением и переподключить телефон на другой тариф, к примеру. |
| 1.7 | Резервное копирование STORAGE-8 | У приложения установлено свойство allowBackup. При распаковке бэкапа были обнаружены следующие сохранённые конфеденциальные данные:   * Номер пользователя в plaintext. * Пароль пользователя в plaintext. * Почтовый ящик пользователя в plaintext. * Локальный пинкод, который можно установить для дополнительной защиты приложения, также хранится тут же в plaintext. |
| 1.8 | ПРИЛОЖЕНИЕ СКРЫВАЕТ КОНФИДЕНЦИАЛЬНЫЕ ДАННЫЕ С ЭКРАНА, КОГДА НАХОДИТСЯ В ФОНОВОМ РЕЖИМЕ STORAGE-9 | Использований FLAG\_SECURE не обнаружил. Динамическое тестирование также подверждает его отсутствие. Особенно радует возможность сделать скриншот в то время как пользователь вводит данные своей кредитной карточки для пополнения счёта. |
| 1.9 | Приложение не хранит конфиденциальные данные в памяти дольше, чем  необходимо STORAGE-10 | Как я ранее указывал, номер пользователя и пароль сохраняются в shared properties как строки. Вызовов сбора мусора после их сохранения я не нашёл, так что походу они зависают в памяти после использования.  Использований Arrays.fill и вывода в dev/null не нашёл.  Ввод данных карточки для пополнения счёта осуществляется через веб-форму. Скорее всего данные, введённые в веб-форму, также оседают в памяти как строки.  Динамический анализ провести не удалось, так как adb не даёт сделать снимок памяти приложения, не имеющего debug-метки. |
| 1.10 | Приложение требует от пользователя минимальную настройку доступа к  устройству  STORAGE 11 | Приложение не запрашивает дополнительных настроек защиты у пользователя. По задумке разработчиков пользователь должен защитить вход в приложение пинкодом, который надо вводить при каждом resume приложения. Но фичу с вводом пинкода можно легко отключить. |
| 2 | Ненадежная передача данных. |  |
| 2.1 | Проверка сертификатов X.509 NETWORK‑3 | Нашёл создание TrustManager’а в классе WebSocketOkhttp. Уже само по себе использование аннотации @SuppressLint({"TrustAllX509TrustManager"}) выдаёт с головой суть этого менеджера. На данный момент не смог найти чёткого подтверждения того, что через этот сокет производится пополнение баланса, а не только получение множества рекламных объявлений.  Нашёл замечательный метод UtilNetwork#getOkHttpClientForAnyRequest, в котором также создаётся TrustManager с @SuppressLint({"TrustAllX509TrustManager"}).  HostnameVerifier’ов не нашёл, да и врядли бы их использовали при учёте таких крутых TrustManager’ов. |
| 2.2 | Проверка SSL pinning NETWORK-4 | Не нашёл ресурсов с расширением BKS. Использования Network Security Configuration не обнаружил. |