Бабак Н.Г., Крюков А.Ф. Защита информации в операционной системе Android // Международный журнал информационных технологий и энергоэффективности. – 2019. – Т.4 №1(11) с. 21-26

Международный журнал информационных технологий и энергоэффективности

Сайт журнала:
http://www.openaccessscience.ru/index.php/ijcse/

УДК 004.457

ЗАЩИТА ИНФОРМАЦИИ В ОПЕРАЦИОННОЙ СИСТЕМЕ ANDROID

[1]Бабак Н.Г., [2]Крюков А.Ф.

Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ», Россия (111250, г.Москва, ул. Красноказарменная, д. 14); e-mail: [1]nikita.enrollee@gmail.com, [2]KriukovAF@mpei.ru

Описывается реализация аппаратных методов защиты данных в операционной системе Android. Сравнивается полнодисковое и файловое шифрование, приводятся преимущества и недостатки каждого метода. Описываются способы аутентификации пользователей в Android.

Ключевые слова: защита информации, Android, криптография, шифрование.

INFORMATION SECURITY IN THE ANDROID OPERATION SYSTEM

[1]Babak N.G., [2]Kryukov A.F.

National Research University “Moscow Power Engineering Institute”, Russia (111250, Moscow, Krasnokazarmennaya street, 14); e-mail: [1]nikita.enrollee@gmail.com, [2]KriukovAF@mpei.ru

The paper describes the implementation of hardware data protection methods in the Android operating system. Compares full-disk and file-based encryption, gives advantages and disadvantages of each method. Describes how to authenticate users in the Android.

Keywords: data protection, Android, cryptography, encryption.

Для людей всегда актуальна проблема защиты их личных данных. Поскольку мобильные устройства имеют большую популярность, то в предлагаемой статье рассмотрена реализация защиты данных в операционной системе Android.

Android – операционная система с открытым исходным кодом, основана на ядре Linux и собственной реализации виртуальной машины Java от Google. До версии Android 5.0 использовалась виртуальная машина Dalvik, а после – ART (Android Runtime) [1].

Существуют различные способы защиты информации на Android устройстве. К ним относятся, например, установка пароля, использование антивирусных программ, шифрование всего устройства, использование шифрующих приложений для хранения определённых данных. Наличие пароля в том или ином виде подразумевается во всех методах защиты информации.

Шифрование – это процесс кодирования данных пользователя на Android устройстве с помощью симметричных шифров. При записи данные сначала автоматически шифруются, а при чтении расшифровываются. Шифрование гарантирует, что злоумышленник не сможет прочесть данные даже при получении доступа к ним. [2]

В Android существует два метода шифрования:

- полнодисковое шифрование (full-disk encryption);
- файловое шифрование (file-based encryption).

Полнодисковое шифрование (FDE) Android устройства основано на модуле dm-crypt, входящем в функционал ядра Linux и обеспечивающем возможность шифрования на любом блочном устройстве хранения данных.

Полноценное полнодисковое шифрование стало возможным с выходом версии 5.0, что связано с появлением 64-битных процессоров.

При первом включении устройство генерирует случайный 128-битный мастер-ключ (device encryption key или DEK), а затем хэширует его паролем по умолчанию и солью. Пароль задаётся пользователем, а соль – сгенерированное устройством 128-битное случайное число.

С помощью DEK шифруются все данные. Пользователь не видит и не использует этот ключ. Когда пользователь меняет пароль, то заново шифруется только хранимый DEK ключ, а не все данные.

Если при первоначальном шифровании устройство потеряет питание, то имеющиеся данные будут утеряны и потребуется сброс к заводским настройкам.

Алгоритм шифрования ключа DEK перед его сохранением в криптографические метаданные представлен на рисунке 1 [3].

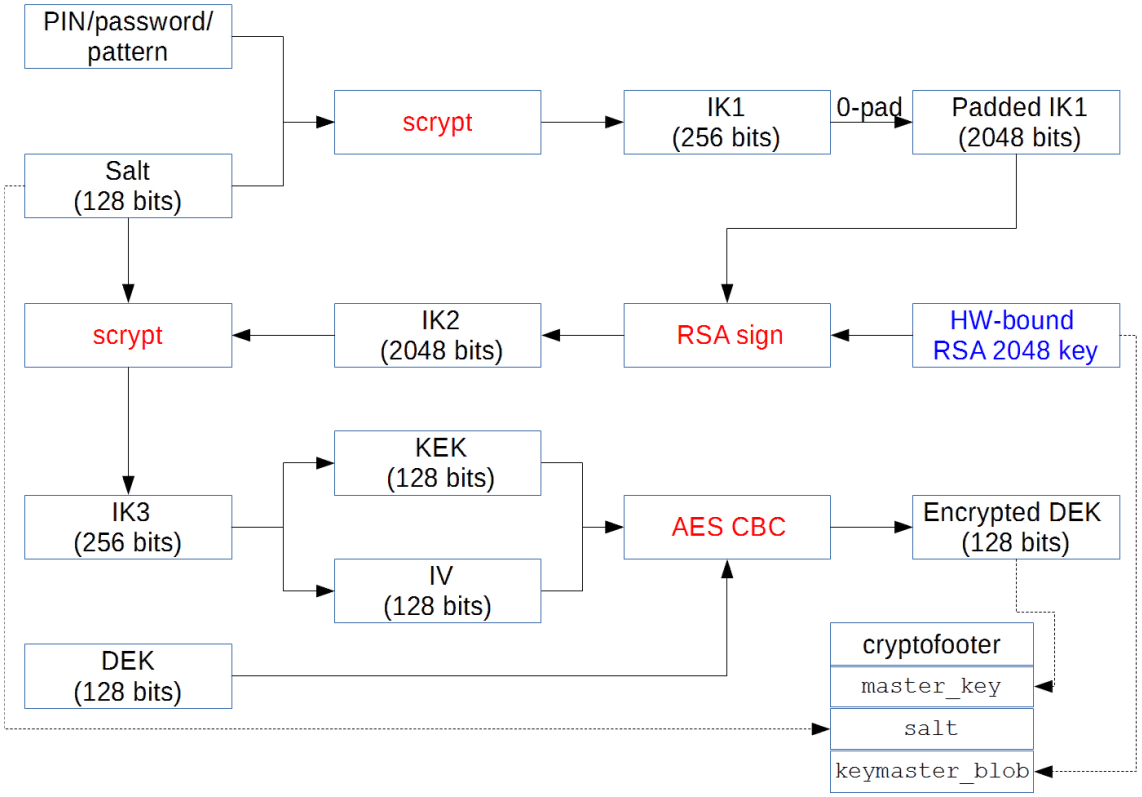

Рисунок 1 – Алгоритм шифрования ключа DEK

AES (Advanced Encryption Standard) – симметричный алгоритм блочного шифрования, принятый правительством США на основе результатов проведенного конкурса в качестве стандарта шифрования. Основу алгоритма составляют замены, подстановки и линейные преобразования, каждое из которых выполняется блоками по 128 бит.

CBC (Cipher block chaining) – режим сцепления блоков шифротекста – один из режимов шифрования для симметричного блочного шифра с использованием механизма обратной связи. Каждый блок открытого текста (кроме первого) побитно складывается по модулю два с предыдущим результатом. Одна ошибка в бите блока шифротекста влияет на расшифровку всех последующих блоков. Перестройка порядка блоков зашифрованного текста вызывает повреждения результата дешифрования [4]. На рисунке 2 представлена схема работы режима CBC.

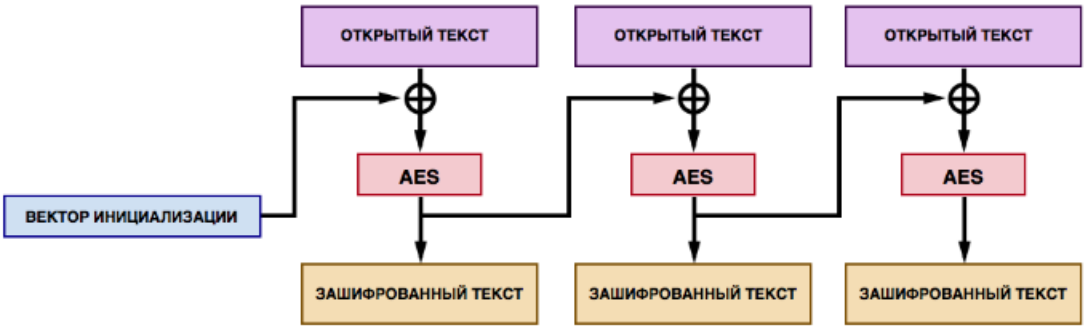

Рисунок 2 – Режим CBC

Важно отметить, что криптографическая схема AES-CBC считается уязвимой к утечке данных, так как допускает определение точки их изменения. Она позволяет выполнять атаки по типу подмены и перемещения.

На данный момент Android уходит от поддержки полнодискового шифрования и советует использовать файловое шифрование. Одна из причин такого решения – невозможность осуществления экстренного вызова после перезагрузки устройства, пока пользователь не введёт пароль.

Файловое шифрование (FBE) стало доступно, начиная с версии Android 7.0. Файловое шифрование, которое выполняется с использованием возможностей файловой системы ext4, позволяет шифровать различные файлы различными ключами и расшифровывать их независимо.

Файловое шифрование добавляет новую функцию Direct Boot [5] (прямая загрузка). Прямая загрузка позволяет работать приложениям, когда устройство было включено, но не разблокировано пользователем. Режим Direct Boot по умолчанию не включен в приложении и при включении удаляет все имеющиеся данные. Ранее в Full-disk encryption было необходимо ввести пароль, прежде чем получить доступ к каким-либо функциям устройства.

С введением файлового шифрования приложения могут работать в зашифрованном режиме, что позволяет защитить личные данные пользователя именно там, где это действительно необходимо.

На устройствах с File-based encryption имеется два доступных для приложений хранилища:

- шифрование на уровне учётных данных – Credential Encrypted (CE);

- шифрование на уровне устройства – Device Encrypted (DE).

CE хранилище используется по умолчанию и доступно только после разблокировки устройства. DE хранилище доступно в режиме Direct Boot и после того, как пользователь разблокировал устройство.

При отключенном файловом шифровании хранилища DE и CE всегда находятся в разблокированном состоянии. Direct Boot позволяет приложениям обращаться к каждому из этих хранилищ.

Такое разделение хранилищ делает рабочие профили более безопасными, поскольку шифрование больше не основано исключительно на пароле загрузки, как в Full-disk encryption.

Содержимое файлов шифруется с помощью шифра AES-256 в режиме XTS. Имена файлов шифруются шифром AES-256 в режиме CBC-CTS. Режим XTS разрабатывался специально для шифрования на блочных устройствах и не имеет типичных для режима CBC уязвимостей. В частности, XTS не позволяет определить точку изменения данных, не подвержен утечке данных, устойчив к атакам подмены и перемещения.

Android поддерживает следующие способы аутентификации пользователя:

- ПИН-код;
- пароль;
- графический ключ;
- отпечаток пальца.

При первом запуске устройства пользователь вводит ПИН-код, пароль или графический ключ. Эта начальная регистрация создаёт случайно сгенерированный 64-разрядный идентификатор безопасности пользователя (SID – user secure identifier). SID привязан к паролю.

После настройки пользователем учётных данных, он получает идентификатор SID и может приступить к аутентификации, которая начинается с ввода ПИН-кода, пароля, графического ключа или с помощью отпечатка пальца.

Все компоненты безопасной среды исполнения (trusted execution environment - TEE) имеют общий секретный ключ, используемый для аутентификации.

На рисунке 3 представлена схема процесса проверки подлинности [6].

Процесс аутентификации.

1. Пользователь вводит ПИН, пароль, графический ключ или отпечаток пальца. И в зависимости от метода проверки отправляется запрос в gatekeeperd или fingerprintd, называемые деймон (daemon).
2. Деймон посылает данные своей дочерней части в TEE, которая генерирует токен аутентификации AuthToken.
3. Деймон получает подписанный AuthToken и передаёт его службе хранилища ключей.
4. Служба хранилища ключей передаёт AuthToken мастеру ключей keymaster и проверяет подлинность ключа.

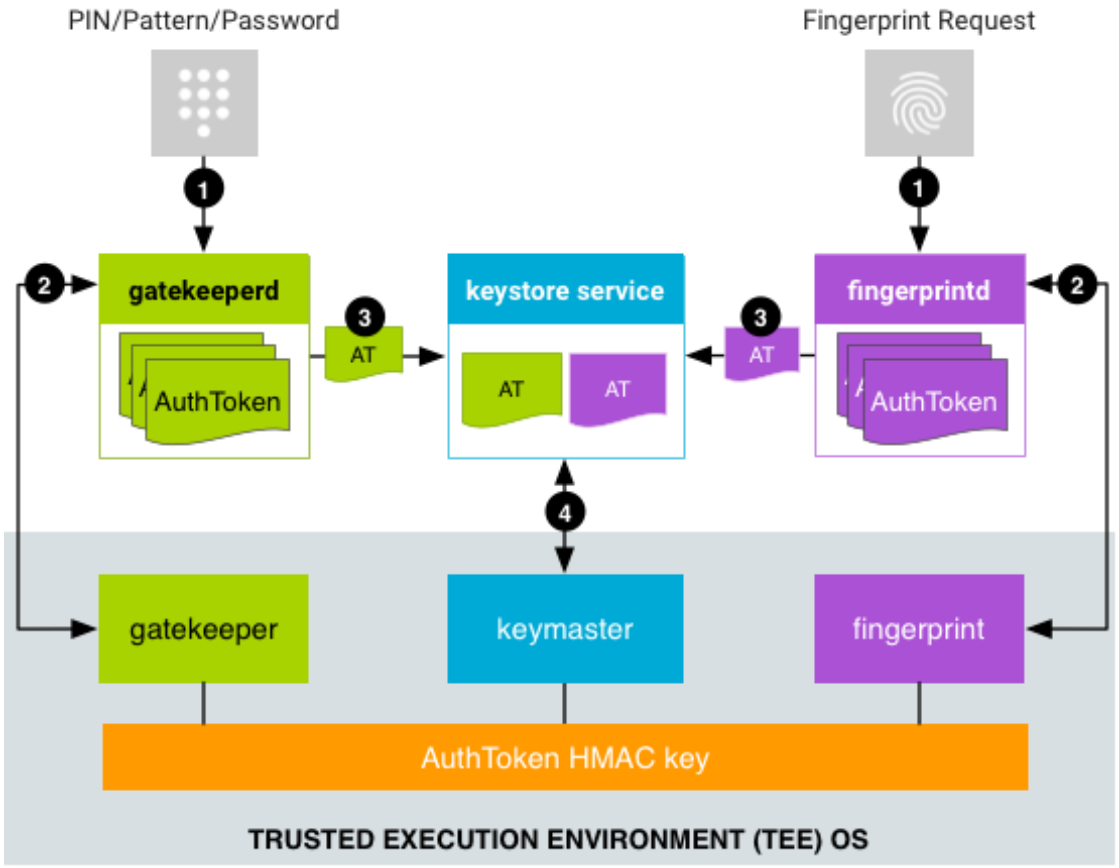

Рисунок 3 – Процесс проверки подлинности

AuthToken после перезагрузки устройства становится недействительным. Формат AuthToken'а унифицирован и состоит из следующих компонентов:

- версия токена аутентификации – 1 байт;
- идентификатор операции – 64 бита;
- неповторяющийся идентификатор пользователя, привязанный ко всем ключам – 64 бита;
- идентификатор проверки подлинности ASID – 64 бита;
- тип аутентификатора (Gatekeeper или Fingerprint) – 32 бита;
- временная метка, время в миллисекундах с момента последней загрузки системы – 64 бита;
- AuthToken HMAC – 256 бит.

При каждой загрузке устройства случайным образом генерируется ключ AuthToken HMAC, который сообщается всем компонентам безопасной среды исполнения TEE (Gatekeeper, Fingerprint). Данный ключ не должен быть доступен за пределами безопасной среды исполнения [6].

Файловое шифрование не лишено недостатков. Данный метод уязвим к side channel [7] атакам, так как, несмотря на шифрование файлов и их имен, он оставляет открытыми метаданные, которые можно использовать для выяснения типа хранимой информации и идентификации пользователя устройства. Также встроенные методы шифрования приводят к снижению производительности. Но лучше использовать их, чем хранить данные в сторонних непроверенных приложениях.

Учитывая преимущества и недостатки каждого метода, сделан вывод, что при поддержке устройством файлового шифрования следует использовать его. Полнодисковое шифрование имеет смысл только на устройствах с операционной системой Android версии меньше 7.0. При необходимости защиты определённых данных можно использовать проверенные, желательно сертифицированные, приложения, созданные специально для защиты данных. Это позволит уменьшить нагрузку на устройство, а значит увеличить его производительность, также это даёт возможность учитывать специфику шифрования тех или иных типов данных.

Список литературы

1. Android. URL: https://ru.wikipedia.org/wiki/Android
2. Encryption. URL: https://source.android.com/security/encryption
3. Извлечение аппаратного ключа полнодисковой защиты в телефонах Android на процессорах Qualcomm. URL: https://sohabr.net/habr/post/395643
4. AES шифрование и Android клиент. URL: https://habr.com/company/rambler-co/blog/279835
5. Режим Direct Boot. URL: https://developer.android.com/training/articles/direct-boot
6. Authentication. URL: https://source.android.com/security/authentication
7. Атака по сторонним каналам. URL: https://ru.wikipedia.org/wiki/Атака_по_сторонним_каналам

References

1. Android. URL: https://ru.wikipedia.org/wiki/Android
2. Encryption. URL: https://source.android.com/security/encryption
3. Извлечение аппаратного ключа полнодисковой защиты в телефонах Android на процессорах Qualcomm. URL: https://sohabr.net/habr/post/395643
4. AES шифрование и Android клиент. URL: https://habr.com/company/rambler-co/blog/279835
5. Режим Direct Boot. URL: https://developer.android.com/training/articles/direct-boot
6. Authentication. URL: https://source.android.com/security/authentication
7. Атака по сторонним каналам. URL: https://ru.wikipedia.org/wiki/Атака_по_сторонним_каналам