УДК 004.49

Агапова В.О.
Белорусский государственный университет
информатики и радиоэлектроники, Минск

Научный руководитель:
Лихачевский Д.В.
Белорусский государственный университет
информатики и радиоэлектроники, Минск

ИСПОЛЬЗОВАНИЕ ШИФРОВАНИЯ, ЗАЩИЩЕННОЙ ПАМЯТИ И KEYSTORE API ДЛЯ ЗАЩИТЫ ДАННЫХ В МОБИЛЬНЫХ ПРИЛОЖЕНИЯХ

Аннотация. В статье рассматриваются современные методы обеспечения безопасности данных в мобильных приложениях на платформе *Android*. Особое внимание уделено использованию шифрования, защищенной памяти устройств и возможностей *KeyStore API* для хранения криптографических ключей. Описаны преимущества каждого из подходов, а также приведены рекомендации по их комбинированному использованию для повышения общей защищенности приложений.

Ключевые слова: мобильная безопасность, шифрование данных, защищенная память, *KeyStore API*, *Android*.

Введение

Мобильные устройства повсеместно используются для хранения и обработки конфиденциальной информации. В связи с этим возрастает необходимость в надежных механизмах защиты данных как при их хранении, так и при передаче. Платформа *Android* предоставляет разработчикам ряд инструментов для обеспечения безопасности, включая механизмы шифрования, использование защищенных хранилищ и специализированные *API* для управления криптографическими ключами.

Основная часть

Применение шифрования данных

Шифрование позволяет сделать данные нечитаемыми для злоумышленников в случае их перехвата или несанкционированного доступа [1], [2]. В *Android* доступны как стандартные алгоритмы симметричного шифрования (например, *AES*-256 с режимами *GCM* или *CBC*), так и асимметричного (*RSA*, *ECC*) [3]. Кроме того, активно развивается использование алгоритмов постквантовой криптографии, обеспечивающих защиту от будущих угроз, связанных с появлением квантовых компьютеров [1].

Для повышения безопасности рекомендуется:

- применять современные и проверенные временем криптографические библиотеки (например, *BouncyCastle, Conscrypt*) [1];
- использовать надежные схемы управления ключами, включая генерацию уникальных сессионных ключей [2];
- реализовывать двойное шифрование особо чувствительных данных (двухступенчатую криптографию) [3].

Также важно обеспечивать защиту данных как «на покое» (*data at rest*), так и «в пути» (*data in transit*), применяя защищенные транспортные протоколы, такие как *TLS* 1.3 [2], [5].

Использование защищенной памяти

Современные мобильные устройства оснащаются доверенными средами исполнения (*Trusted Execution Environment*, *TEE*) или безопасными элементами (*Secure Element*, *SE*), обеспечивающими физическую и логическую защиту критически важных данных [1].

TEE, реализуемый, например, на базе *ARM TrustZone*, позволяет изолировать выполнение кода безопасности от основной операционной системы, обеспечивая:

- безопасное выполнение криптографических операций [1];
- проверку целостности приложений и среды выполнения [1];
- надежное хранение биометрических данных (отпечатков пальцев, сканов лица) [1].

Secure Elements представляют собой специализированные микросхемы, соответствующие банковским и государственным стандартам безопасности (например, *Common Criteria EAL* 5+), что делает их

оптимальным выбором для хранения особо важных данных, таких как ключи цифровых кошельков или идентификационные токены [2].

В *Android* реализована поддержка *Protected Confirmation API* и *Identity Credential API*, позволяющих использовать возможности защищенной памяти для безопасного подтверждения пользовательских операций и хранения цифровых документов [1].

Использование *KeyStore API*

KeyStore API Android предоставляет разработчикам возможность безопасной генерации, хранения и использования криптографических ключей без прямого доступа к ним со стороны приложения [1].

Особенности *KeyStore API*:

- ключи могут быть помечены как требующие аутентификацию пользователя (например, с помощью отпечатка пальца или распознавания лица) [1];
- возможность установки срока действия ключей и ограничений по числу применений [1];
- использование аппаратных модулей безопасности (*Hardware-Backed Keystore*) для предотвращения атак через извлечение из памяти [1].

Кроме того, в последних версиях Android (начиная с *Android 9 Pie*) доступны расширенные функции:

- *StrongBox Keymaster* – модуль, обеспечивающий еще более строгую аппаратную защиту ключей в отдельном чипе [1];
- поддержка безопасной генерации асимметричных пар ключей для подписания данных и аутентификации без передачи закрытого ключа [5].

Преимущества комплексного использования

Комбинирование технологий шифрования, защищенной памяти и *KeyStore API* позволяет достичь максимально возможного уровня защиты данных в мобильных приложениях [1], [2]:

- минимизация риска кражи данных даже при физическом доступе к устройству или взломе операционной системы [1];
- защита ключей и чувствительной информации от утечек через приложения, перехват трафика или эксплойты нулевого дня [1];
- повышение доверия пользователей благодаря соответствию приложения современным требованиям законодательства и стандартов безопасности (*GDPR, HIPAA, PCI DSS*) [4];
- устойчивость к новым угрозам, включая атаки с использованием квантовых вычислений и атаки по сторонним каналам (*side-channel attacks*) [1], [3].

В совокупности эти методы позволяют создать надежную архитектуру защиты мобильных приложений, минимизируя потенциальные риски и обеспечивая долгосрочную безопасность пользовательских данных [1].

Заключение

Обеспечение защиты данных в мобильных приложениях требует применения многоуровневых подходов. Использование шифрования,

защищенной памяти и *KeyStore API* позволяет значительно повысить безопасность пользовательских данных. В условиях растущего числа угроз разработчикам следует учитывать данные методы при проектировании архитектуры приложений и реализации механизмов хранения и обработки чувствительной информации.

Список использованных источников:

1. Stallings W. Cryptography and Network Security: Principles and Practice. 7th ed., Pearson, 2016.

2. Kaufman C., Perlman R., Speciner M. Network Security: Private Communication in a Public World. 2nd ed., Prentice Hall, 2002.

3. Menezes A.J., van Oorschot P.C., Vanstone, S.A. Handbook of Applied Cryptography. CRC Press, 1996.

4. Barkley J. Implementing Role-Based Access Control Using CORBA Security Service. –Proceedings of the Third ACM Workshop on Role-Based Access Control, 1998.

5. Housley R., Polk W., Ford W., Solo D. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC 5280, 2008.

Agapova V.O.
Belarusian State University of
Informatics and Radioelectronics, Minsk

Scientific supervisor:
Likhachevsky D.V.
Belarusian State University of
Informatics and Radioelectronics, Minsk

ANALYSIS OF METHODS FOR HIDING MALICIOUS APPLICATIONS FROM SANDBOXES AND ANTIVIRUS SYSTEMS

Abstract. The article examines modern methods of ensuring data security in mobile applications on the Android platform. Particular attention is paid to the use of encryption, secure device memory, and the capabilities of the KeyStore API for cryptographic key storage. The advantages of each approach are described, along with recommendations for their combined use to enhance the overall security of applications.

Keywords: mobile security, data encryption, secure memory, KeyStore API, Android.