An Empirical Study of Cryptographic Misuse in Android Applications

Дедков Сергей

2015

1 An Empirical Study of Cryptographic Misuse in Android Applications

Авторы данной статьи опросили разработчиков, используют ли они криптографические API, которые обеспечивают криптографическую безопасность. Разработали методы анализа и проанализировали Google Play рынок, обнаружив, что 10327 из 11748 приложений - используют криптографические интерфейсы. 88% допустили хотя бы одну ошибку.

Разработчики используют криптографические примитивы, такие как блочные шифры и код аутентфикации сообщения(MAC) для защиты данных и связи. Криптографы знают, что есть правильный способ и неправильный способ использовать эти примитивы, где правильный способ обеспечивает надежные гарантии безопасности, а неправильный способ неизменно приводит к уязвимостям.

Сосредоточились авторы на двух стандартах безопасности: атаки по выбранным открытым текстам (IND-CPA) и стойкости к уязвимостям.

При этом платформа Android была выбрана не случайно, для этого были следующие причины:

- Приложения запускаются на мобильных платформах, которые хранят множество персональной информации
- Приложения пишкться на Java. A Java's cryptographic API достаточно стабильна.
- Под Android написано очень много приложений, на основе которых формировалась база статисктики, в которой были отражены способы использования криптографических примитивов пользователями.

Для выявления дефектов был использован статический анализатор уязвимостей CryptoLint, основанный на Androguard Android. Результаты были приведены выше.

Авторы предлжили 6 правил. Любое приложение, которое нарушает одно из них не может быть безопасным:

- Не используйте режим ECB для шифрования.

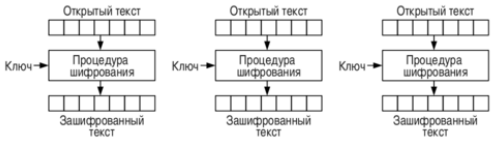

Рис. 1: Шифрование в режиме ECB (режиме электронной кодовой книги)

Используя режим ECB, разарботчики подвергают свое ПО уязвимостям. Такой режим уязвим, т.к. в нем одинаковые фрагменты шифруются в одинаковую последовательность(см. рисунок 1). Для предотвращения к блокам добавляют соль, для более сложного взлома.

- Не используйте неслучайный IV (вектор инициализации см. рисунок 2) для шифрования CBC.

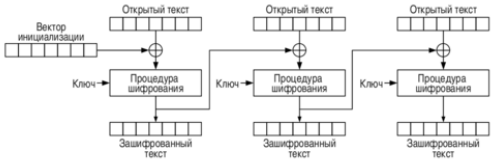

Рис. 2: Шифрование в режиме CBC (режиме сцепления блоков шифротекста)

Использование неслучайного вектора инициализации повышает криптостойкость системы.

- Не использовать постоянные ключи шифрования.

  Смысл такой же, как и у второго правила.

- Не используйте постоянные соли для PBE.

  Статическая соль и тому подобные конструкции могут служить достаточно хорошо, пока структура этих конструкций и соль хранятся в тайне. Если же злоумышленник вызнает секрет хэширования — он с легкостью сможет использовать в своих целях.

- Не используйте меньше 1000 итераций для PBE.

- Не использовать постоянные seed для получения псевдослучайных последовательностей SecureRandom.

  Правило 4 и 5 выведено чисто опытным путем для PBE схем.

CrytpoLint после анализа указывает соблюда.тся ли эти 6 правил.

В отличие от обычных java-приложений Android-приложение запускается на виртуальной машине Dalvik. С этим байкодом и работает инструмент авторов - CryptoLint. Приложения, которые запускаются на виртуальной машине Dalvik получают доступ к интерфейсу и подсистемам. Для получения доступа к алгоритмам шифрования вызывается метод Cipher.getInstance, которые зарегистрированы в cryptographic service providers (CSP), в подсистеме Java Cryptography Architecture (JCA).

В такой реализации по-умолчанию выбирается ECB шифрование. Авторами проводились исселдования графа потока управления приложения и то, как находились нарушения. В качестве примера было рассмотрено 3 популярных приложения с уязвимостями, при этом эти приложения имели миллионы скачиваний.