УДК 004.056.5

О СКРЫТЫХ КАНАЛАХ ПО ВРЕМЕНИ В ОС ANDROID

Т. И. Милованов

Исследованы два различных скрытых канала. Первый из них найден в используемых в настоящее время версиях ОС Android. Измерены его основные характеристики, такие, как пропускная способность и процент ошибок. Канал основан на изменении количества свободного места в файловой системе и может быть использован двумя вредоносными приложениями для обмена данными. Написаны два тестовых приложения, реализующих информационный поток через найденный канал. Второй исследованный скрытый канал использует разницу во времени работы некоторой системной функции с фиксированными секретными и разными открытыми (контролируемыми любым пользователем) данными. В работе такой информационный поток реализуется от легального приложения к вредоносному без ведома легального в файловой системе `ext2` в ОС Android.

Ключевые слова: *скрытые каналы, вредоносные приложения, Android.*

В настоящее время наблюдается бурное распространение вредоносного программного обеспечения на платформе Android. Программы типа «троянский конь» составляют существенную часть современных вредоносных программ. Обычно такие программы запрашивают у пользователя критичные привилегии (например, возможность доступа в интернет), что позволяет реализовать утечку критичных данных пользователя. В то же время набор запрашиваемых приложением привилегий используется антивирусными средствами для обнаружения вредоносной активности.

Использование скрытых каналов позволяет вредоносным программам получать доступ к критичным данным, не имея полного набора привилегий. Одним из известных троянов, использующим скрытый канал по времени, является `Soundcomber` [1]. Данный троян для уменьшения набора привилегий не требует доступа в интернет, но устанавливает парное приложение с данной привилегией и для общения с ним использует скрытый канал, реализованный через изменение общедоступных настроек звука/вибрации. Недостаток такого канала в том, что процесс передачи данных легко сбивается пользовательскими действиями.

В результате проведённых исследований обнаружен скрытый канал по времени, основанный на изменении количества свободного места в файловой системе. Пусть имеются два вредоносных приложения A и B, и A имеет доступ к некоторым секретным данным, но не имеет доступа к сети, а B имеет доступ к сети, но не имеет доступа к секретным данным. При этом у приложений нет общих ресурсов для обмена данными, но у каждого приложения есть собственная директория, доступная для записи и чтения ему и только ему. Для передачи бита 1 приложение A изменяет количество свободного места в файловой системе, создавая в собственной директории файл с произвольным содержимым. Приложение B измеряет количество доступного места в файловой системе, фиксирует его уменьшение и делает вывод, что передан бит 1. Для передачи следующего единичного бита приложение A удаляет созданный им файл, а приложение B фиксирует увеличение свободного места. Размер файла подбирается эмпирическим путём так, чтобы действия пользователя и других приложений не спровоцировали ошибку приложения B. Приложение A передаёт бит 0, не изменяя ничего.

Основной фактор, снижающий скорость работы данного скрытого канала, — это синхронизация между приложениями. Рассмотрим два следующих способа её реализации.

1) Синхронизация посредством поочерёдного ожидания. После каждой передачи бита передающее приложение ожидает в течение времени, необходимого на считывание вторым приложением этого бита информации. Аналогично второе приложение ожидает, пока первое не передаст следующий бит. Проблема этого способа заключается в том, что время выполнения системного вызова записи в файл сильно зависит от загруженности системы и может увеличиваться в 2–3 раза. Поэтому для безошибочной передачи данных необходимо значительно увеличивать время ожидания приложений, что снижает пропускную способность канала. Максимальная скорость передачи данных, полученная при таком способе синхронизации, равна 12 бит/с.

2) Синхронизация посредством ещё одного скрытого канала. Этот способ быстрее предыдущего, но требует наличия ещё одного канала. При тестировании в качестве канала использовались общедоступные системные настройки звука/вибрации. Но пользователь своими действиями может легко вмешаться в передачу информации и сбить синхронизацию. Максимальная скорость передачи данных, полученная при таком способе синхронизации, равна 20 бит/с.

Ещё один найденный канал основан на атаке по времени. Атака по времени, или «таймминговая атака», использует время выполнения некоторой функции, которая зависит от произвольных пользовательских данных и секретных данных. Подобную функцию будем называть *уязвимой к таймминговой атаке*. Изучая время выполнения этой функции на разных пользовательских данных, можно сделать некоторые выводы о секретных данных. В файловой системе `ext2` существует функция стандартной библиотеки `stat`, которая, в свою очередь, использует функцию `strcmp`, являющуюся уязвимой к таймминговой атаке. Функция `stat` принимает в качестве аргумента имя файла и выполняет поиск требуемого файла в директории, выполняя сравнение переданного имени со всеми, хранящимися внутри.

Пусть есть приложение-злоумышленник A и приложение B, не являющееся злоумышленником. Приложение B хранит секретные данные в собственной директории, к которой имеет доступ на чтение и запись только оно. Приложение A может через скрытый канал, основанный на использовании уязвимой к таймминговой атаке функции, узнать секретные имена файлов в директории приложения B методом посимвольного подбора. Первым этапом приложение A узнаёт длину некоторого существующего файла в директории. Для этого оно выполняет в цикле вызов функции `stat`, передавая ей в качестве аргумента строки длины 1, 2, 3, ... Если цикл вызовов со строкой длины l выполняется дольше, чем хотя бы один из предыдущих циклов, то в директории существует хотя бы один файл с именем длины l. На этом первый этап считается завершённым. Вторым этапом приложение A начинает посимвольно подбирать имя файла, изменяя сначала первый символ в строке длиной l, затем второй и так далее. Символ считается подобранным верно, если цикл вызовов с этим символом на данной позиции выполняется дольше, чем с другим символом. После подбора последнего символа второй этап можно считать завершённым. Таким образом, скрытый канал от приложения B к приложению A без ведома приложения B и без непосредственного доступа приложения A к файлам реализуется посредством измерения приложением A разницы времени работы системной функции `stat` с фиксированными секретными и разными открытыми (контролируемыми любым пользователем) данными.

ЛИТЕРАТУРА

1. *Schlegel R., Zhang K., Zhou X., et al.* Soundcomber: a stealthy and context-aware sound Trojan for smartphones // Proc. 18th Annual Network and Distributed System Security Symposium (NDSS '11), San Diego, CA, February 6–9, 2011. P. 17–33.

УДК 004.94

ИСПОЛЬЗОВАНИЕ ЭЛЕКТРОННЫХ СЕРТИФИКАТОВ ДЛЯ АВТОРИЗАЦИИ ПО ДОВЕРЕННОСТИ В ОС LINUX

В. И. Рыжков

Предлагается решение для делегирования некоторого набора прав от одного пользователя (доверителя) операционной системы другому (доверенному лицу) на определённый промежуток времени. Для этого предложено использовать «доверенности» — объекты, содержащие в себе такие поля, как идентификатор доверителя, идентификатор доверенного лица, время действия доверенности, а также набор прав, делегируемых доверенному лицу доверителем. Доверенность должна содержать также цифровую подпись на закрытом ключе доверителя под всеми вышеперечисленными полями. Предложенное решение реализовано для операционной системы Linux с помощью криптографического инструмента OpenSSL и подключаемых модулей аутентификации (PAM). В качестве доверенностей здесь выступают цифровые сертификаты стандарта X.509 v3, а делегируемые полномочия указываются по определённому формату в поле «Расширения» этих сертификатов. Сам функционал авторизации по доверенности реализован в виде модуля PAM.

Ключевые слова: *электронные сертификаты, X.509, Linux, PAM, OpenSSL.*

В операционных системах привилегии пользователя можно задавать, используя группы, в которые он входит.

Пусть некоторый пользователь (доверитель) хочет передать некоторые свои права другому пользователю (доверенному лицу), который изначально этими правами не обладает. Такая схема полезна в случае, когда доверителю приходится отсутствовать по той или иной причине и он хочет передать свои полномочия своему доверенному лицу. Самое очевидное решение: доверитель может добавить доверенное лицо в некоторую группу, которая обладает этими правами. При таком подходе возникают следующие проблемы:

1) Право переводить пользователя из группы в группу есть, как правило, далеко не у каждого.
2) Пусть право переводить пользователей из группы в группу у доверителя есть. Допустим, доверитель будет отсутствовать в течение месяца, но эти права необходимо делегировать на неделю. Следующие три недели доверенное лицо будет находиться в привилегированной группе, не имея в этом потребности.

Таким образом, возникает задача построения системы делегирования некоторого набора прав доступа, которыми обладает некий пользователь-доверитель (не обязательно «привилегированный» в системе), другому пользователю — доверенному лицу, который ими изначально может не обладать, на некоторый (определённый пользователем-доверителем) промежуток времени.

При этом, очевидно, должны выполняться следующие требования: