**JAVA-мидлеты**

Крис Касперски

**Будьте бдительны**

Многие компании активно используют сотовые телефоны в служебных целях, доверяя им не только связь, но и конфиденциальную информацию, совершенно не задумываясь о возможных последствиях. Практически все модели телефонов содержат серьезные дефекты в системе безопасности. Угрозу представляют java-мидлеты, скачиваемые пользователями из ненадежных источников.

За последние несколько лет сотовые телефоны резко «поумнели». Сначала они обзавелись быстрыми процессорами и емкой энергонезависимой памятью, затем в них появились встроенные фото- и видеокамеры и протоколы, ориентированные на передачу больших объемов данных через Интернет. Фактически телефон превратился в миниатюрный компьютер, а компьютеру, как известно, нужны программы. Игры, записные книжки, органайзеры, платежные системы, картографические приложения, средства навигации... А вслед за программами подтянулись вирусы, черви и троянские кони.

**Чего опасаться**

Сотовый телефон—это своеобразный электронный кошелек. Микроплатежи через SMS уже давно стали обыденностью, и, чтобы пополнить свой счет, злоумышленнику достаточно просто зарегистрировать платный номер и создать мидлет, рассылающий SMS без ведома владельца аппарата. Информация из записной книжки представляет огромный интерес как для журналистов, так и для лиц, занимающихся промышленным шпионажем, поскольку позволяет не только установить телефоны влиятельных лиц, но и узнать некоторые интимные подробности ихжизни, существенно упрощающие психологические атаки типа социального инжиниринга.

Фотографии, снятые встроенной камерой,— это настоящая кладезь информации. Очень часто сотовый телефон используется для оперативной съемки интересных событий или технологических процессов, не предназначенных для посторонних глаз. Наличие высокоскоростных сетевых протоколов позволяет мидлетам скрытно передавать содержимое памяти телефона на какой-нибудь анонимный ftp/smtp-сервер или на другой сотовый телефон.

Наконец, аппарат может быть «порабощен» спаммером, рассылающим SMS-сообщения или даже обыкновенные письма через протокол GPRS. И не так важно, что GPRS медленный. Ведь трафик все равно не свой, а система защиты в сотовых телефонах на порядок слабее, чем в персональных компьютерах. Вы видели сотовый телефон с антивирусом? А заплатки когда последний раз устанавливали? И брандмауэра, надо полагать, тоже нет. Одна надежда на встроенную систему безопасности.

**Откуда ноги растут**

Для начала несколько пояснений, касающихся используемой терминологии. Сокращение J2ME расшифровывается как «Java 2 Platform Micro Edition», т. е. Java-платформа версии 2—специальная редакция для микроустройств. CLDC, в свою очередь, означает «Connected Limited Device Configuration» — конфигурации коммуникационных устройств с ограниченными ресурсами, другими словами, Java для мобильных телефонов, КПК и смартфонов.

J2ME-пpилoжeниe, называемое мидлетом (от английского midlet), начинает свой жизненный путь с предверификатора (preverifier), анализирующего байт-код на предмет выявления подставных вещей. Проверенный байт-код передается в MIDP (Mobile Information Device Profile —профиль для мобильного устройства с информационными функциями), в сферу компетенции которого входят следующие функции:

1. Жизненный цикл мидлета (инсталляция, удаление, обновление, запуск, остановка, пауза, возобновление работы);

2. Безопасность и авторизация доступа к защищенным функциям (например, коммуникационным);

3. Графический интерфейс (в том числе низкоуровневый, пригодный для программирования игр);

4. Ввод команд пользователя (например, посредством клавиатуры).

Для абстрагирования от конкретного оборудования MIDP обращается к CLDC-библи-отеке, экспортирующей все жизненно важные функции (кстати говоря, CLDC версии 1.0 не поддерживала плавающей арифметики и ее приходилось эмулировать непосредственно самому мидлету). Байт-код, включая код библиотеки CLDC, исполняется на виртуальной Java-машине, обозначаемой аббревиатурой KVM (K-Virtual Machine). А сама виртуальная машина опирается на API-функции операционной системы (Operating System), «натянутой» поверх аппаратного обеспечения (Hardware). Для увеличения производительности Java-машина оптимизирует код (Java Code Compact, сокращенно JCC), применяя вполне традиционные алгоритмы оптимизации, общие для всех языков. Собственного названия удостоилась лишь одна технология—ROMizing («ромизация»), обеспечивающая упреждающую предзагрузку и связывание (prelink) классов, что замедляет загрузку, но увеличивает производительность мидлета во время его выполнения.

**Рассылка SMS**

Разработчики сотовых телефонов позволили мидлетам рассылать SMS-сообщения путем вызова специальной API-функции, на аппаратном уровне выводящей запрос на подтверждение. Его никак нельзя отключить, что вполне логично.

Известное изречение гласит: защита, как столб,—ее трудно перепрыгнуть, но легко обойти. Хакерской группе «The Phenoelit Hackers group» удалось найти брешь в обороне Siemens S55. Вывести запрос на подтверждение («allow SMS15142467980—Уеs/Nо») конструкторы вывели, а вот заблокировать дисплей забыли, в результате чего мидлеты получили возможность перезаписать оригинальный текст, заменив ее каким-нибудь безобидным вопросом в стиле «Wanna Play?» Пользователь нажимает «Yes» и SMS-сообщение скрытно уходит адресату. Проверка остальных телефонов показала, что данная уязвимость затрагивает весь модельный ряд, распространяясь не только на SMS, но и на другие запросы на подтверждение. У телефонов Motorola V600 и Nokia 3600 в этом отношении все нормально, поскольку они блокируют дисплей перед выводом любых запросов на подтверждение.

Фрагмент Java-кода, обеспечивающий защиту запроса на подтверждение отправки SMS

Void PermissioriDialog(...) // запрос на подтверждение отправки SMS

Permi ssionForm.setCommandListener (this);

// блокировка дисплея от изменений

displayManager.preemptDisplay (token, this, form, true);

private void SetAnswer(...) //

отправка SMS

// снятие блокировки с дисплея displayManager.donePreempting(preemp tToken); notifyO;

**Хранилища**

Когда мидлету требуется сохранить какие-либо данные в энергонезависимой памяти, то он это делает не напрямую, а через специальный интерфейс, обеспечивающий управление хранилищами (Record Stores). Каждый мидлет может иметь одно или более хранилищ, а начиная с MIDP 2.0 появилась поддержка разделяемых хранилищ, к которым могут обращаться сразу несколько мидлетов. Физически хранилища представляют собой группы записей (Record), а записи есть не что иное как обыкновенные массивы байтов. Доступ к хранилищам осуществляется по уникальному имени, состоящему из имен производителя (Vendor Name), мидлета (MIDIet Suite Name) и записи (Record Store Name). Записи, принадлежащие различным мидлетам, могут иметь совпадающие имена (также, как разные каталоги могут содержать одноименные файлы).

Хранилищами заведует система управления записями (Record Management System, RMS), гарантирующая, что данные одного мидлета не попадут к остальным без его согласия. Программисты, уверенные в непогрешимости защиты, сплошь и рядом помещают в хранилища секретные данные.

В действительности большинство производителей реализуют RMS-системы на языке Java, предоставляя высокоуровневые API-функции (High level MIDP API), вызываемые мидлетами и выполняющие все необходимые проверки. Однако, если спуститься на одну ступеньку вглубь, мы обнаружим низкоуровневые API-функции (Low level API), также доступные мидлетам и позволяющие обращаться к любому хранилищу без проверок вообще (в первую очередь следует обратить внимание на класс RecordStoreFile, название которого говорит за себя).

Причем хранилища обычно размещаются в одном из файлов и доступны для просмотра через штатный проводник или функции файлового ввода/вывода. Теоретически, хранилища должны быть зашифрованы, чтобы никакой мидлет не смог добраться к ним через файловую систему, но в некоторых телефонах они хранятся в открытом виде. В частности, на Nokia 3650 хранилища сосредоточены в файле rms.db, находящемся в текущем каталоге мидлета (т. е. лежат там же, где и Jad/Jar-файлы), что делает ихлегкой жертвой для атаки.

**Заключение**

Мы рассмотрели только две уязвимости мобильных устройств, на деле их больше. Но не спешите выкидывать свой телефон на помойку, если в нем обнаружена уязвимость. Где гарантия, что другой аппарат окажется лучше? Более надежный выход—скачивать мидлеты только из надежных источников. Это не защит от атак на 100%, но существенно снизит их вероятность.

**Список литературы**

IT спец № 07 ИЮЛЬ 2007