Инфраструктура модуля полной разблокировки Windows Phone 7

Меньшиков М.А., аспирант кафедры системного программирования
СПбГУ, info [at] menshikov.org

Аннотация

Разблокировка (”Джейлбрейк”) различных закрытых операционных систем является нетривиальной задачей вследствие необходимости проводить реверс-инжиринг большого числа модулей. Подобная работа была проведена для Windows Phone 7 и привела к созданию «полной разблокировки» — набора модулей, обеспечивающего запуск нативных приложений и контроль прав доступа к системным ресурсам. В докладе рассмотрены описание механизма разблокировки с сохранением обновляемости операционной системы, компиляция файлов под данную ОС, восстановление функциональности отладчика.

Введение

Windows Phone 7 — операционная система от Microsoft, основанная на Windows Embedded Compact 7.x [1] и предназначенная для потребительских мобильных устройств. Отсутствие официальной возможности использовать нативный код при разработке программ вызвало интерес автора к изучению данной ОС с позиции энтузиаста. Результатами изысканий стали элементы программы сборки образов прошивки устройств (OSBuilder) и разработка модуля *полной разблокировки*, позволяющего запускать нативные приложения с максимальными правами доступа. Все связанное с ним можно назвать *инфраструктурой* модуля разблокировки, и ключевые его элементы описываются в данной работе. Данные сведения могут быть интересны широкому кругу исследователей в связи с общностью подходов к реверс-инжинирингу программ и операционных систем.

При описании инфраструктуры «полной разблокировки» делается допущение, что загрузчик целевых устройств уже был модифицирован для загрузки неподписанных образов ОС. Также предполагается, что существует возможность пересборки образа ОС целевого устройства.

Система безопасности

Интересующая исследователей часть системы безопасности Windows Phone 7 [2] строится на основе следующих модулей Execute-In-Place-памяти

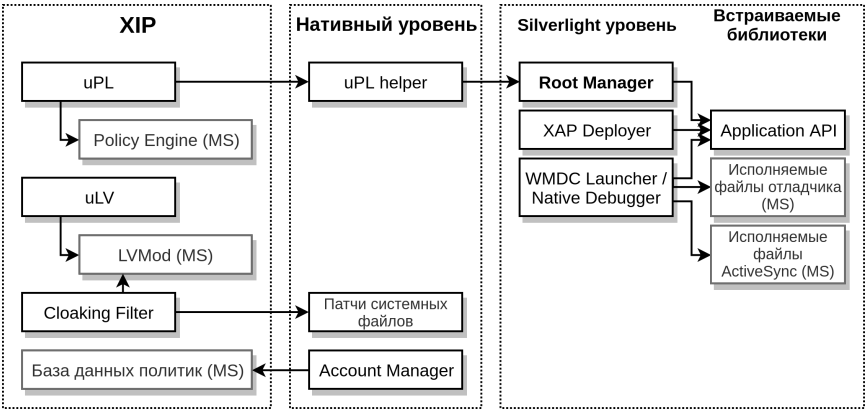

Рис. 1: Общая схема полной разблокировки

(XIP): Loader Verifier, Policy Engine. Также разблокировка предполагает корректировку системных исполняемых файлов, связанных с установкой приложений и их отладкой. С одной стороны, большую часть проблем можно решить сменой политик безопасности и переписыванием программ на более либеральные аналоги, с другой стороны — этот процесс сравнительно трудоемкий. Поэтому необходимы обертки для `lvmod.dll` и `policyengine.dll`, а также патчи библиотек. Весь процесс разблокировки представлен на рис. 1. Рассмотрим его в деталях.

Loader Verifier

Данный модуль отвечает за верификацию загружаемых модулей. В частности, в нем присутствует функция `LVModAuthenticateFile`, предоставляющая информацию об аутентификации для конкретного файла, т.е. метод служит как предварительный фильтр. Так как реализация данной функции нетривиальна, было решено пробрасывать её выполнение в стандартную библиотеку (новый модуль получил название uLV), за исключением проверки подписи «Менеджера прав доступа». Модуль менеджера подписан авторским ключом, сертификат встроен в Loader Verifier. Если осуществляется попытка его подменить, то есть выявлено несоответствие цифровой подписи файла публичному сертификату, то загрузка модуля останавливается. Это сделано с целью предотвращения несанкционированного доступа к базе данных пользовательских аккаунтов.

Функция `LVModRouting` проверяет, в рамках какого chamber [3] должен быть загружен файл. Здесь в любом случае выбирается аккаунт с SID

”S-1-5-112-0-0-1” (TCB). `LVModAuthorize` проверяет, может ли аккаунт загружать файл в контекст другого аккаунта. Здесь также было решено давать разрешение. Такой вариант позволяет загружать любые нативные исполняемые файлы, не привнося дополнительных проблем с безопасностью. Windows Phone 7 не позволяет осуществлять запуск нативных исполняемых файлов из безопасной среды, к которой относится вся система. Запуск возможен лишь через доверенное приложение, и, следовательно, доверие к дочернему процессу возникает транзитивно, а значит компрометация системы безопасности затруднена.

Policy Engine

Движок политик безопасности занимается определением уровня доступа к конкретному ресурсу вне текущей chamber процесса [3]. Конкретные реализации функций этой библиотеки определяются таблицей, получаемой через `GetFunctionTable()`. Все ресурсы адресуются по IRI (например, `/REGISTRY/HKLM/SOFTWARE/OEM`). Для правильной работы механизма полной разблокировки необходимо переопределить следующие методы: авторизация субъектов по классам политик — `PolicyCheck`, по каноническому имени — `PolicyCheckByCanonicalName`, по хэндлу — `PolicyCheckByHandle`, а также получение маски прав доступа — `PolicyGetAccessGranted`. Новый модуль, получивший название uPL, использует несколько проверок:

1. *Проверка для Zune*. Для правильной работы магазина приложений Zune и других встроенных программ строго необходимо закрывать им доступ к ветке с каноническим именем `UNSIGNEDNATIVEDLL_AUTHZ` (возможность загрузки неподписанных библиотек). При этом реальный пользователь ветки определяется эвристически по косвенным признакам (запрашиваемый уровень доступа, флаги, отсутствие дополнительного контекста). Такие проверки встроены лишь в функции `PolicyCheckByCanonicalNameDirect` и `PolicyCheckByCanonicalName`, так как именно их задействует Zune.

2. *Предварительная проверка*. Проводится проверка ветки с каноническим именем `REVOKED_CERTS/SHA1`, также по причине неработоспособности сервисов при её открытии. На этом этапе дается быстрый ответ о возможности использования нативной библиотеки. Если пользователь выдал расширенные права текущему аккаунту, то доступ разрешен. Если же нет, то запрос на авторизацию мгновенно отвергается,

а «Менеджеру прав доступа» отправляется информационное сообщение с Product ID заблокированной программы.

3. *Стандартная проверка*. Вызывается стандартный обработчик Policy Engine, и если он авторизовывает действие, то процесс заканчивается.

4. *Дополнительная проверка*. Данная проверка касается лишь сторонних приложений, для которых стандартная проверка не проходит. Если приложение авторизовано через «Менеджер прав доступа» (или включен режим полного доверия), то результат — положительный. Дополнительно следует отметить добавленную невозможность стороннему приложению использовать Account Database API или делать полную чистку памяти телефона (через безусловный запрет соответствующих действий).

Для удобства изменения прав доступа приложений, движок политик безопасности создает *очередь сообщений*, которую читает нативное приложение uPL helper, запущенное при запуске системы. При получении сообщения, программа определяет Product ID приложения по пользовательскому аккаунту, получает его локализованное название и создает уведомление через функцию `SHPostMessageToast`.

Новый менеджер политик безопасности попадает в отдельный пакет внутри XIP (Pkg_FullUnlock) и прописывается в реестре.

Важным элементом является «Менеджер прав доступа» — приложение на Silverlight, использующее API непосредственно для выдачи расширенных прав приложениям. Права хранятся как *свойства* аккаунтов приложений. Похожей деятельностью посвящен «Менеджер аккаунтов». Его задачей является добавление предустановленных программ в список доступа или включение режима полного доверия. Этот исполняемый файл помечен в файле политик безопасности как входящий в chamber TCB, и потому его цифровая подпись проверяется более строго. Такая многоступенчатая система позволяет избегать ошибок, связанных с неполным знанием устройства менеджера политик.

Скрывающий фильтр

«Скрывающий фильтр» — фильтр файловой системы. Его идея в том, что он *подменяет* пути загружаемых файлов на модифицированные их версии. Фильтр интегрирован в XIP основной ОС, но не в XIP SLDR — вторичного ядра, используемого для обновления системы. Таким образом, при нормаль-

ной работе ОС видит и загружает модифицированные версии файлов, а при обновлении — оригинальные.

Список файлов задается в реестре в области XIP. В первую очередь `lvmod` подменяется на `ulv` — путь к Loader Verifier жестко прописан в ядре ОС. Перенаправления также требуют `AppPreInstaller` (предустановщик программ), `PacmanInstaller` (непосредственно установщик пакетов приложений), `SirepServerAppDev` (подключение отладчика) — в них добавляются патчи, снимающие незначительные ограничения.

Нативные исполняемые файлы

Для компиляции нативных исполняемых файлов для Windows Phone 7 достаточно любого SDK с компилятором для архитектуры ARM, последним из которых Windows Mobile 6.5.3 SDK с поддержкой ARMv5. В новой системе не изменился формат файлов [4], но изменились стандартные библиотеки. В частности, обязательным оказалось получение новых .lib-файлов для `coredll` и `aygshell` при помощи dumpbin и lib.exe. Замена этих и других аналогичных статических библиотек решает вопрос совместимости получаемых исполняемых файлов с Windows Phone 7. Для XIP-модулей необходимо включить режим LARGEADDRESSAWARE и совместимость с Data Execution Prevention (DEP).

Проблемой стало формальное присутствие WINAPI по части оконных функций, но их фактическая неработоспособность. Это связано с переносом процесса отрисовки в отдельный Compositor, поддерживающий аппаратную акселерацию, а также с реализацией *страничного менеджера*. Путем реверсинжиниринга были найдены методы `SHPM*`[1], которые позволили создавать страницы, а главное — прикреплять к ним стандартные `HWND`. Открытие этих API привело к портированию графических приложений, таких как браузер Opera.

Подключение к компьютеру и нативная отладка

Возможность нативной отладки — важный аспект на пути к реализации полноценной разблокированной прошивки. В Windows Mobile 6.x работа с ActiveSync (Windows Mobile Device Center) реализуется посредством `rapiclnt.exe` и других связанных приложений

[1] `https://github.com/ultrashot/common/blob/master/PageManager.hpp`

Remote API (RAPI), интегрированных в систему. RAPI имеет большое число зависимостей. Фактически пришлось реализовать набор функций библиотек coredll (`SystemParametersInfoW`, подсистема Local Authentication Subsystem — LASS, `VerifyUserAsync`), aygshell (`RegistryGetDWORD`, `LoadDeviceLockTimeout`), ossvcs (`ProcessRoles`, `IsOTAPhoneNumber`, `SHRestricted`, `QueryPolicyImpl`, `WatsonReportFault`). В подавляющем числе случаев были использованы заглушки, возвращающие корректные коды результатов. Большое число методов для манипуляции путями из ceshell.dll (например, `CanonicalizePath`) было реализовано с нуля согласно спецификациям.

Отладка приложений выполняется через `edm2.exe`, `eDbgTL.dll` и прочие исполняемые файлы, присутствующие в составе Visual Studio. Исправление работы отладчика требует как портирования RAPI, так и реализации других отсутствующих функций. К примеру, функция переключения между режимами пользователя и ядра `SetKMode` более не актуальна, отображение адресов в разные адресные пространства через `CeZeroPointer` и `MapPtrToProcess` перестало быть нужным в связи с ликвидацией слотовой системы памяти, по той же причине излишней является и `SetProcPermissions`. Замена этих функций на заглушки позволяет использовать отладчик Visual Studio, вызываемый посредством RAPI. Также заработала отладка через IDA, почти все существующие редакторы реестра, CE Remote Tools.

Дополнительным результатом изысканий стала реализация установки[2] XAP-файлов за счет использования недокументированного API `PM*` из библиотеки PacmanClient.dll.

Тестирование

Проект полной разблокировки был протестирован в рамках разработки автором прошивки Dynamics, изначально предполагавшейся для устройств HTC. Программное обеспечение было портировано и под другие устройства — Nokia Lumia (RainbowMod), Samsung (Dynamics), а также адаптировано сторонними авторами. Среди выявленных проблем отмечались незначительные проблемы с функциональностью нативных приложений (при полной работоспособности приложений из Магазина), редкие потери обновляемости через сервис обновлений от Microsoft из-за допущенных при сборке

[2]Нативная часть представлена в репозитории `https://github.com/ultrashot/xapdeployer-native`

ошибок. Все проблемы были устранены, и итоговая версия не имела явных функциональных проблем. Можно отметить, что в общем сумме проектом полной разблокировки пользовались десятки тысяч человек.

Заключение

В работе рассмотрены принципы разблокировки нативных возможностей Windows Phone 7: авторизация исполняемых файлов, отключение проверок прав доступа, восстановление функциональности отладчика Visual Studio. Результатом работы стало создание пакета Full Unlock, не затрагивающего подсистему обновлений Windows Phone 7 (OTA). Проект был опубликован на профильных форумах, внедрен во многие модифицированные прошивки для данной системы и таким образом апробирован пользователями соответствующих устройств.

Благодарности

Автор благодарит коллег, внесших вклад в разработку проекта: AndrewSh, Barin, Cees Heim, compu829, Cotulla, feropont, GoodDayToDie, HD2Owner, Heathcliff74, Jaxbot, jessenic, LiquidStorm, OndraSter, Rafael Rivera, saintonotole, sh4d0w86, snickler, Vladimir1973, -W_O_L_F-, xboxmod и многих других.

Литература

[1] Phung S., Jones D., Joubert T. Professional Windows Embedded Compact 7. — John Wiley & Sons, 2011.

[2] Windows Phone 7 - Owned Every Mobile? // GitHub URL: `https://github.com/alexplaskett/Publications/blob/master/mwri_wp7-bluehat-technical_2011-11-08.pdf` (дата обращения: 21.03.2019).

[3] Schaefer T., Höfken H., Schuba M. Windows phone 7 from a digital forensics' perspective //International Conference on Digital Forensics and Cyber Crime. – Springer, Berlin, Heidelberg, 2011. – P. 62-76.

[4] PE Format // Microsoft Docs URL: `https://docs.microsoft.com/en-us/windows/desktop/debug/pe-format` (дата обращения: 21.03.2019).