Windows Phone 7
Internals and Exploitability

(abridged white paper)

Fourteenforty Research Institute, Inc.
Tsukasa Oi | Research Engineer

目次

1. Abstract

Windows Phone 7 is a modern mobile operating system developed by Microsoft. This operating system – based on Windows Embedded CE 6 – protects the system and the user by modern sandbox and secure application model. We have analyzed the internals of this operating system to evaluate security of the Windows Phone 7 operating system.

We also focused on customizations by third-party vendors. Windows Phone 7-based devices by some vendors have special interfaces for OEM use but some of those makes subverting sandbox and chamber model easier because of various design/implementation issues such as directory traversal and improper open privileged operations.

2. Introduction: Windows Phone 7 and Analysis

Windows Phone 7 is based by Windows Embedded CE 6.0R3 and some features (including security loader) are ported from Windows Embedded Compact 7. However, many components in this operating system are veiled and this is why we decided to analyze security in Windows Phone 7 operating system.

We used native code access and reverse engineering to unveil operating system components and analyzed some components and mechanisms to evaluate its security.

3. Security Analysis – Windows Phone 7

3.1. Application Model

Windows Phone 7 only allows Windows Phone Marketplace applications which are signed by Microsoft. Almost all applications are written in .NET (except some OEM applications and applications from selected vendors) and minimize attack surface.

3.2. Process Memory and ASLR

Address Space Layout Randomization (ASLR) is one of the exploit mitigation features which make executing arbitrary code harder by randomizing memory addresses and layout in the process. If randomization is not enough, it could make attackers to estimate memory addresses easier and could lead to arbitrary code execution.

We analyzed randomness of ASLR using native code access and most of memory areas are randomized well (estimated entropy is approximately 10-bits). However, we found the fact that only base addresses are randomized. This makes heap spraying or similar techniques easier and more reliable.

3.3. Memory uses in .NET

We also found insecure memory uses which is related to dynamic string allocation in .NET. More specifically, dynamic Unicode strings are allocated in executable (RWX) memory. This could be used by attackers to spray executable code in the memory.

3.4. Application Security System

In Windows Phone 7 operating system, each application is isolated and sandboxed. This seems to be well-designed and conforming “principle of least privilege”. Execution domain is separated by chambers which make elevating privileges much harder. We analyzed the Policy Engine and base policy data and we could not find any design flaw.

4. Security Analysis – OEM customizations

Windows Phone 7 can be customized by OEMs. OEMs can install their original drivers and applications. However, OEMs’ code is relatively vulnerable and some of OEM customizations (driver interfaces) make possible to break Windows Phone 7’s strong sandbox. Here are some details.

Some OEMs’ and non-OEMs’ applications declare undocumented capability ID_CAP_INTEROPSERVICES which allows native code and “Interop Services” access. If these applications are vulnerable, the attacker can exploit vulnerabilities and access OEMs' driver interfaces.

For example, HTCFileUtility.dll driver allows some file system access. If this driver is used by the attacker, he/she can steal critical and sensitive information from the device, including their e-mails. HTC has recently modified the driver to restrict "accessible paths" but this modification was not enough. We found latest version of this DLL has directory traversal vulnerability and the attacker is still possible to access almost every file in the system.

Some of those device drivers seem to be unconcerned about security and this might be a big difference between device drivers for Android, which are relatively well-designed.

5. Conclusions

Windows Phone 7's sandbox is very strong and conforming "principle of least privilege". But some insufficient memory protection features and OEM driver interfaces might threaten users and devices. We think ID_CAP_INTEROPSERVICES is a design flaw in this operating system. Microsoft and OEMs will need to make Interop Services access more granularized.

We hope these issues to be fixed in Windows Phone 8 and we guess... they *will*.