10 January 2017 Study · Mobile Security · Web & Cloud

# Automated Analysis of Windows Phone Applications

The full text is currently only available in the Deutsch version. A translation will follow.

**Archived.** This content is no longer updated. It remains available for reference.

**Source** https://technology2.a-sit.at/en/publications/automated-analysis-of-windows-phone-applications/

We carried out an automated analysis of about 65.000 Windows Phone apps. The idea was to automatically identify potential implementation errors that are security relevant. About 8.5 percent of the applications appear to have such errors.

The objective of this project was to analyze a number of Windows Phone Apps on security relevant issues using the .NET Compiler Platform. The found issues correspond to four groups:

- Wrong use of  cryptographic functions
- Insufficient TLS Security
- Exposed API keys
- Other exposed client secrets

The metadata of 360.000 apps was downloaded from the Windows Phone Store. 65.000 apps of these 360.000 apps are in a format that can be analyzed. The remaining (older) apps are encrypted and cannot be decompiled. The extraction and decompilation phase resulted in about 6 million unique source code files. 8.5% of all analyzed apps contain at least one potential security relevant issue. 4.5% of the analyzed apps use a hard-coded seed for random number generators. The following list shows the number of affected applications per issue group:

- 181 apps use hard-coded cryptographic keys,
- 293 apps use hard-coded initialization vectors for cryptographic operations,
- 138 apps use 1000 or less iterations for key derivations functions,

- 500 apps expose their client secrets,
- 516 apps ignore HTTPS related certificate validation errors,
- 4919 apps expose their Google API keys,
- 73 apps expose their Amazon Web Service keys and
- 2980 apps use random number generators with hard-coded seeds.

The impact of these issues cannot be determined automatically, but it ranges from endangered user's privacy in case of broken cryptography or insufficient transport layer security, to monetary losses for the developer, when an attacker abuses the developer's billing enabled API key.