Analyzing Windows Telemetry component

Pablo Agustín Artuso
LU: 282/11
artusopablo@gmail.com

University of Buenos Aires
Faculty of Exact and Natural Sciences
Computer Science department

August 9, 2018

Director: Rodolfo Baader
rbaader@dc.uba.ar

Co-Director: Aleksandar Milenkoski
amilenkoski@ernw.de

Technology evolves fast. Humans cannot adapt to it with the same speed. Even though software which gathers information has been in use for several years, most of the people are not aware about the impact of the loose of privacy of their personal information. During the last years, lot of research has been carried out to improve the state of the art in this topic [1], [2], [3].

Windows, one of the most popular OS, has a component called Telemetry. It collects information from the system with the goal of analyzing and fixing software & hardware problems, improving the user experience, among others. The kind of information that can be obtained by this component is partially configurable in four different levels: security, basic, enhanced and full, being "security" the level where less information is gathered and "full" the opposite case. Although the information is collected from the system itself, the processing phase is not carried out inside it. Once the information is extracted, there exists some components which are in charge of sending that information, using a secure channel, to the Microsoft backend server where all the analysis is finally performed.

How Telemetry stores/process/administrates the information extracted? It employs a widely used framework called Event Tracer for Windows (ETW) [4]. Embedded not only in userland application but also in the kernel modules, the

ETW framework has the goal of providing a common interface to log events and therefore help to debug and log system operations, by instrumenting it.

The working principles of Telemetry are not documented in depth. Furthermore, it is a closed source project, making it harder to develop an analysis of the component. Previous works on the same topic were based on either online documentation or just in the analysis of the network traffic sent to Microsoft.

In this work, we are going to analyze a part of the Windows kernel to better understand how Telemetry works from an internal perspective. This work will make windows analysts, IT admins or even windows users, more aware about the functionality of the Telemetry component helping to deal with privacy issues, bug fixing, knowledge of collected data, etc. Our analysis implies performing reverse engineering [5],[6] on the Telemetry component, which involves challenging processes such as kernel debugging, dealing with undocumented kernel internal structures, reversing of bigger frameworks (i.e: ETW), binary libraries which lack symbols, etc. As part of the analysis we will also develop an in depth comparison between the differences among each level of Telemetry, stressing the contrast in the amount of events written, verbosity of information, etc. Finally, we will study how the communication between the Windows instance and the Microsoft backend servers is carried out.

References

[1] Bolin Ding, Janardhan Kulkarni and Sergey Yekhanin. Collecting Telemetry Data Privately. In proceedings of Neural Information Processing Systems Conference (NIPS), 2017.

[2] Vasyl Pihur, lfar Erlingsson and Aleksandra Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In CCS, pages 10541067, 2014

[3] Microsoft Corporation. Dynamic collection analysis and reporting of telemetry data. US 9,590,880 B2, 2017.

[4] Tarik Soulami. Inside Windows debugging. Chapter 12, 2012

[5] Hausi A. Miiller, Jens H. Jahnke, Kenny Wong ,Dennis B. Smith , Scott R. Tilley , Margaret-Anne Storey. Reverse Engineering: A Roadmap. In Proceedings of the Conference on The Future of Software Engineering, Pages 47-60, 2000.

[6] Bruce Dang, Alexandre Gazet, Sbastien Josse, Elias Bachaalany. Practical Reverse Engineering: x86, x64, ARM, Windows Kernel, Reversing Tools, and Obfuscation. 2014.