Breaking Payloads with Runtime Code Stripping and Image Freezing

Collin Mulliner
collin[at]mulliner.org

Matthias Neugschwandtner
matthias.neugeschwandtner[at]gmail.com

August 2015

This document is the accompanying white paper for the presentation *Breaking Payloads with Runtime Code Stripping and Image Freezing* at Black Hat USA 2015 in Las Vegas, NV, USA.

Fighting off attacks based on memory corruption vulnerabilities is hard and a lot of research was and is conducted in this area. In our recent work we take a different approach and looked into breaking the payload of an attack. Current attacks assume that they have access to every piece of code and the entire platform API. In this talk we present a novel defensive strategy that targets this assumption. We built *CodeFreeze* a system that removes unused code from an application process to prevent attacks from using code and APIs that would otherwise be present in the process memory but normally are not used by the actual application. Our system is only active during process creation time, and, therefore, incurs no runtime overhead and thus no performance degradation. Our system does not modify any executable files or shared libraries as all actions are executed in memory only. We implemented our system for Windows 8.1 and tested it on real world applications. Besides presenting our system we also show the results of our investigation into code overhead present in current applications. *CodeFreeze* was able to remove %28 of the code introduced by DLLs loaded by Adobe Reader 9.4.

Material

Website with updated material:
`http://www.mulliner.org/security/codefreeze/`