Hide Artifacts: VBA Stomping, Sub-technique T1564.007 - Enterprise

Archived: 2026-04-05 14:32:15 UTC

Adversaries may hide malicious Visual Basic for Applications (VBA) payloads embedded within MS Office documents by replacing the VBA source code with benign data.[1]

MS Office documents with embedded VBA content store source code inside of module streams. Each module stream has a `PerformanceCache` that stores a separate compiled version of the VBA source code known as p-code. The p-code is executed when the MS Office version specified in the `_VBA_PROJECT` stream (which contains the version-dependent description of the VBA project) matches the version of the host MS Office application.[2][3]

An adversary may hide malicious VBA code by overwriting the VBA source code location with zero's, benign code, or random bytes while leaving the previously compiled malicious p-code. Tools that scan for malicious VBA source code may be bypassed as the unwanted code is hidden in the compiled p-code. If the VBA source code is removed, some tools might even think that there are no macros present. If there is a version match between the `_VBA_PROJECT` stream and host MS Office application, the p-code will be executed, otherwise the benign VBA source code will be decompressed and recompiled to p-code, thus removing malicious p-code and potentially bypassing dynamic analysis.[4][1][5]

Source: https://attack.mitre.org/techniques/T1564/007