Detection Strategy for Debugger Evasion (T1622), Detection Strategy DET0371

Archived: 2026-04-05 14:39:32 UTC

AN1045

Monitor for suspicious use of Windows API calls such as IsDebuggerPresent() and NtQueryInformationProcess(), or processes manually checking the BeingDebugged flag in the Process Environment Block (PEB). Detect sequences of OutputDebugStringW() calls in short intervals that may indicate debugger flooding attempts.

Log Sources

Mutable Elements

Field	Description
ApiCallFrequencyThreshold	Number of repeated debug-related API calls allowed before raising an alert
ProcessAllowList	Legitimate debuggers or developer tools that may trigger similar behaviors

AN1046

Monitor access to /proc/self/status where TracerPID field is queried, as this is a common technique for debugger detection. Detect processes that attempt to trigger exceptions intentionally and monitor whether exception handling indicates presence of a debugger.

Log Sources

Data Component	Name	Channel
File Access (DC0055)	auditd:SYSCALL	open/read: Access to /proc/self/status with focus on TracerPID field

Mutable Elements

Field	Description
MonitoredPaths	Set of /proc paths to monitor for suspicious access
SyscallThreshold	Rate of syscalls (open/read) used to detect repeated probing for debug artifacts

AN1047

Detect suspicious calls to sysctl or ptrace API used to determine if a process is being debugged. Monitor for processes that flood OutputDebugString equivalents or generate abnormal exceptions to evade analysis.

Log Sources

Data Component	Name	Channel
OS API Execution (DC0021)	macos:unifiedlog	ptrace: Processes invoking ptrace with PTRACE_TRACEME flag

Mutable Elements

Field	Description
PtraceInvocationThreshold	Number of ptrace calls in a time window that should raise suspicion
DevToolExclusionList	Exclude known developer tools and monitoring agents

---

Source: https://attack.mitre.org/detectionstrategies/DET0371#AN1046