Native API, Technique T1106 - Enterprise

Archived: 2026-04-05 16:27:29 UTC

S0045 ADVSTORESHELL

ADVSTORESHELL is capable of starting a process using CreateProcess.[16]

S1129 Akira

Akira executes native Windows functions such as `GetFileAttributesW` and `GetSystemInfo`.[17]

S1025 Amadey

Amadey has used a variety of Windows API calls, including `GetComputerNameA`, `GetUserNameA`, and `CreateProcessA`.[18]

S0622 AppleSeed

AppleSeed has the ability to use multiple dynamically resolved API calls.[19]

G0067 APT37

APT37 leverages the Windows API calls: VirtualAlloc(), WriteProcessMemory(), and CreateRemoteThread() for process injection.[20]

G0082 APT38

APT38 has used the Windows API to execute code within a victim's system.[21]

S0456 Aria-body

Aria-body has the ability to launch files using `ShellExecute`.[22]

S1087 AsyncRAT

AsyncRAT has the ability to use OS APIs including `CheckRemoteDebuggerPresent`.[23]

S0438 Attor

Attor's dispatcher has used CreateProcessW API for execution.[24]

S0640 Avaddon

Avaddon has used the Windows Crypto API to generate an AES key.[25]

S1053 AvosLocker

AvosLocker has used a variety of Windows API calls, including `NtCurrentPeb` and `GetLogicalDrives`.[26]

S0638 Babuk

Babuk can use multiple Windows API calls for actions on compromised hosts including discovery and execution.[27][28][29]

S0475 BackConfig

BackConfig can leverage API functions such as `ShellExecuteA` and `HttpOpenRequestA` in the process of downloading and executing files.[30]

S0606 Bad Rabbit

Bad Rabbit has used various Windows API calls.[31]

S1081 BADHATCH

BADHATCH can utilize Native API functions such as, `ToolHelp32` and `Rt1AdjustPrivilege` to enable `SeDebugPrivilege` on a compromised machine.[32]

S0128 BADNEWS

BADNEWS has a command to download an .exe and execute it via CreateProcess API. It can also run with ShellExecute.[33][34]

S0234 Bandook

Bandook has used the ShellExecuteW() function call.[35]

S0239 Bankshot

Bankshot creates processes using the Windows API calls: CreateProcessA() and CreateProcessAsUserA().[36]

S0534 Bazar

Bazar can use various APIs to allocate memory and facilitate code execution/injection.[37]

S0470 BBK

BBK has the ability to use the `CreatePipe` API to add a sub-process for execution via cmd.[38]

S0574 BendyBear

BendyBear can load and execute modules and Windows Application Programming (API) calls using standard shellcode API hashing.[39]

S0268 Bisonal

Bisonal has used the Windows API to communicate with the Service Control Manager to execute a thread.[40]

S0570 BitPaymer

BitPaymer has used dynamic API resolution to avoid identifiable strings within the binary, including `RegEnumKeyW`.[41]

S1070 Black Basta

Black Basta has the ability to use native APIs for numerous functions including discovery and defense evasion.[42][43][44][45][46]

S1180 BlackByte Ransomware

BlackByte Ransomware uses the `SetThreadExecutionState` API to prevent the victim system from entering sleep.[47]

G0098 BlackTech

BlackTech has used built-in API functions.[48]

S0521 BloodHound

BloodHound can use .NET API calls in the SharpHound ingestor component to pull Active Directory data.[49]

S1226 BOOKWORM

BOOKWORM has used various Windows API calls during execution and defense evasion.[50] [51] BOOKWORM has created a buffer on the heap using `HeapCreate` and `HeapAlloc` which allows for copying of shell code and then execution on the heap is initiated through callback function of legitimate API functions such as `EnumChildWindows` or `EnumSystemLanguageGroupsA`. [51]

S0651 BoxCaon

BoxCaon has used Windows API calls to obtain information about the compromised host.[52]

S1063 Brute Ratel C4

Brute Ratel C4 can call multiple Windows APIs for execution, to share memory, and defense evasion.[53][54]

S0471 build_downer

build_downer has the ability to use the `WinExec` API to execute malware on a compromised host.[38]

S1039 Bumblebee

Bumblebee can use multiple Native APIs.[55][56]

S0693 CaddyWiper

CaddyWiper has the ability to dynamically resolve and use APIs, including `SeTakeOwnershipPrivilege`.[57]

S1237 CANONSTAGER

CANONSTAGER has leveraged Native API calls to execute code within the victim's system including `GetCurrentDirectoryW`, `RegisterClassW` and `CreateWindowExW`.[58] CANONSTAGER also created a new overlapped window that initiates callback functions to a windows procedure that processes Windows messages until a designated message type of 0x0018 WM_SHOWWINDOW is observed which then initiates the deployment of a subsequent malicious payload.[58]

S0484 Carberp

Carberp has used the NtQueryDirectoryFile and ZwQueryDirectoryFile functions to hide files and directories.[59]

S0631 Chaes

Chaes used the `CreateFileW()` API function with read permissions to access downloaded payloads.[60]

G0114 Chimera

Chimera has used direct Windows system calls by leveraging Dumpert.[61]

S1149 CHIMNEYSWEEP

CHIMNEYSWEEP can use Windows APIs including `LoadLibrary` and `GetProcAddress`.[62]

S0667 Chrommme

Chrommme can use Windows API including `WinExec` for execution.[63]

S1236 CLAIMLOADER

CLAIMLOADER has used various Windows API calls during execution, when establishing persistence and defense evasion.[64][65] CLAIMLOADER has also leveraged the legitimate API functions to run its shellcode through the callback function, including `GetDC()` and `EnumFontsW()`.[64] CLAIMLOADER established persistence by utilizing the API `SHSetValue()`.[64] CLAIMLOADER has utilized APIs with callback functions such as `EnumpropsExW`, `EnumSystemLanguageGroupsA`, and `EnumCalendarInfoExW`.[65]

S0611 Clop

Clop has used built-in API functions such as WNetOpenEnumW(), WNetEnumResourceW(), WNetCloseEnum(), GetProcAddress(), and VirtualAlloc().[66][67]

S0154 Cobalt Strike

Cobalt Strike's Beacon payload is capable of running shell commands without `cmd.exe` and PowerShell commands without `powershell.exe`.[68][69][70]

S0126 ComRAT

ComRAT can load a PE file from memory or the file system and execute it with `CreateProcessW`.[71]

S0575 Conti

Conti has used API calls during execution.[72][73]

S0614 CostaBricks

CostaBricks has used a number of API calls, including `VirtualAlloc`, `VirtualFree`, `LoadLibraryA`, `GetProcAddress`, and `ExitProcess`.[74]

S0625 Cuba

Cuba has used several built-in API functions for discovery like GetIpNetTable and NetShareEnum.[75]

S0687 Cyclops Blink

Cyclops Blink can use various Linux API functions including those for execution and discovery.[76]

S1111 DarkGate

DarkGate uses the native Windows API `CallWindowProc()` to decode and launch encoded shellcode payloads during execution.[77] DarkGate can call kernel mode functions directly to hide the use of process hollowing methods during execution.[78] DarkGate has also used the `CreateToolhelp32Snapshot`, `GetFileAttributesA` and `CreateProcessA` functions to obtain a list of running processes, to check for security products and to execute its malware.[79]

S1066 DarkTortilla

DarkTortilla can use a variety of API calls for persistence and defense evasion.[80]

S1033 DCSrv

DCSrv has used various Windows API functions, including `DeviceIoControl`, as part of its encryption process.[81]

S1052 DEADEYE

DEADEYE can execute the `GetComputerNameA` and `GetComputerNameExA` WinAPI functions.[82]

S0354 Denis

Denis used the `IsDebuggerPresent`, `OutputDebugString`, and `SetLastError` APIs to avoid debugging. Denis used `GetProcAddress` and `LoadLibrary` to dynamically resolve APIs. Denis also used the `Wow64SetThreadContext` API as part of a process hollowing process.[83]

S0659 Diavol

Diavol has used several API calls like `GetLogicalDriveStrings`, `SleepEx`, `SystemParametersInfoAPI`, `CryptEncrypt`, and others to execute parts of its attack.[84]

S0695 Donut

Donut code modules use various API functions to load and inject code.[85]

S0694 DRATzarus

DRATzarus can use various API calls to see if it is running in a sandbox.[86]

S0384 Dridex

Dridex has used the `OutputDebugStringW` function to avoid malware analysis as part of its anti-debugging technique.[87]

S0554 Egregor

Egregor has used the Windows API to make detection more difficult.[88]

S1247 Embargo

Embargo has leveraged Windows Native API functions to execute its operations.[89]

S0367 Emotet

Emotet has used `CreateProcess` to create a new process to run its executable and `WNetEnumResourceW` to enumerate non-hidden shares.[90]

S0363 Empire

Empire contains a variety of enumeration modules that have an option to use API calls to carry out tasks.[91]

S0396 EvilBunny

EvilBunny has used various API calls as part of its checks to see if the malware is running in a sandbox.[92]

S1179 Exbyte

Exbyte calls `ShellExecuteW` with the `IpOperation` parameter `RunAs` to launch `explorer.exe` with elevated privileges.[93]

S0569 Explosive

Explosive has a function to call the OpenClipboard wrapper.[94]

S0512 FatDuke

FatDuke can call `ShellExecuteW` to open the default browser on the URL localhost.[95]

S0696 Flagpro

Flagpro can use Native API to enable obfuscation including `GetLastError` and `GetTickCount`.[96]

S0661 FoggyWeb

FoggyWeb's loader can use API functions to load the FoggyWeb backdoor into the same Application Domain within which the legitimate AD FS managed code is executed.[97]

S1044 FunnyDream

FunnyDream can use Native API for defense evasion, discovery, and collection.[98]

G0047 Gamaredon Group

Gamaredon Group malware has used `CreateProcess` to launch additional malicious components.[99][100]

S0666 Gelsemium

Gelsemium has the ability to use various Windows API functions to perform tasks.[63]

S0032 gh0st RAT

gh0st RAT has used the `InterlockedExchange`, `SeShutdownPrivilege`, and `ExitWindowsEx` Windows API functions.[101]

S0493 GoldenSpy

GoldenSpy can execute remote commands in the Windows command shell using the `WinExec()` API.[102]

S0477 Goopy

Goopy has the ability to enumerate the infected system's user name via `GetUserNameW`.[83]

G0078 Gorgon Group

Gorgon Group malware can leverage the Windows API call, CreateProcessA(), for execution.[103]

S0531 Grandoreiro

Grandoreiro can execute through the `WinExec` API.[104]

S0632 GrimAgent

GrimAgent can use Native API including `GetProcAddress` and `ShellExecuteW`.[105]

S0561 GuLoader

GuLoader can use a number of different APIs for discovery and execution.[106]

S0499 Hancitor

Hancitor has used `CallWindowProc` and `EnumResourceTypesA` to interpret and execute shellcode.[107]

S1229 Havoc

Havoc can use `NtAllocateVirtualMemory` and `NtCreateThreadEx` to aid process injection.[108]

S0391 HAWKBALL

HAWKBALL has leveraged several Windows API calls to create processes, gather disk information, and detect debugger activity.[109]

S0697 HermeticWiper

HermeticWiper can call multiple Windows API functions used for privilege escalation, service execution, and to overwrite random bites of data.[110][111][112][113]

S0698 HermeticWizard

HermeticWizard can connect to remote shares using `WNetAddConnection2W`.[112]

G0126 Higaisa

Higaisa has called various native OS APIs.[114]

S0431 HotCroissant

HotCroissant can perform dynamic DLL importing and API lookups using `LoadLibrary` and `GetProcAddress` on obfuscated strings.[115]

S0398 HyperBro

HyperBro has the ability to run an application (`CreateProcessW`) or script/file (`ShellExecuteW`) via API.[116]

S0537 HyperStack

HyperStack can use Windows API's `ConnectNamedPipe` and `WNetAddConnection2` to detect incoming connections and connect to remote shares.[117]

S0483 IcedID

IcedID has called `ZwWriteVirtualMemory` , `ZwProtectVirtualMemory` , `ZwQueueApcThread` , and `NtResumeThread` to inject itself into a remote process.[118]

S1152 IMAPLoader

IMAPLoader imports native Windows APIs such as `GetConsoleWindow` and `ShowWindow`.[119]

S0434 Imminent Monitor

Imminent Monitor has leveraged CreateProcessW() call to execute the debugger.[120]

S1139 INC Ransomware

INC Ransomware can use the API `DeviceIoControl` to resize the allocated space for and cause the deletion of volume shadow copy snapshots.[121]

S0259 InnaputRAT

InnaputRAT uses the API call ShellExecuteW for execution.[122]

S0260 InvisiMole

InvisiMole can use winapiexec tool for indirect execution of `ShellExecuteW` and `CreateProcessA`.[123]

S1190 Kapeka

Kapeka utilizes WinAPI calls to gather victim system information.[124]

S1020 Kevin

Kevin can use the `ShowWindow` API to avoid detection.[125]

S0607 KillDisk

KillDisk has called the Windows API to retrieve the hard disk handle and shut down the machine.[126]

S0669 KOCTOPUS

KOCTOPUS can use the `LoadResource` and `CreateProcessW` APIs for execution.[127]

S0356 KONNI

KONNI has hardcoded API calls within its functions to use on the victim's machine.[128]

S1160 Latrodectus

Latrodectus has used multiple Windows API post exploitation including `GetAdaptersInfo`, `CreateToolhelp32Snapshot`, and `CreateProcessW`.[129][130]

G0032 Lazarus Group

Lazarus Group has used the Windows API `ObtainUserAgentString` to obtain the User-Agent from a compromised host to connect to a C2 server.[131] Lazarus Group has also used various, often lesser known, functions to perform various types of Discovery and Process Injection.[132][133]

S0395 LightNeuron

LightNeuron is capable of starting a process using CreateProcess.[134]

S0680 LitePower

LitePower can use various API calls.[135]

S0681 Lizar

Lizar has used various Windows API functions on a victim's machine.[136]

S1202 LockBit 3.0

LockBit 3.0 has the ability to directly call native Windows API items during execution.[137][138]

S0447 Lokibot

Lokibot has used LoadLibrary(), GetProcAddress() and CreateRemoteThread() API functions to execute its shellcode.[139]

S1016 MacMa

MacMa has used macOS API functions to perform tasks.[140][141]

S1060 Mafalda

Mafalda can use a variety of API calls.[142]

S1169 Mango

Mango has the ability to use Native APIs.[143]

S0652 MarkiRAT

MarkiRAT can run the ShellExecuteW API via the Windows Command Shell.[144]

S0449 Maze

Maze has used several Windows API functions throughout the encryption process including IsDebuggerPresent, TerminateProcess, Process32FirstW, among others.[145]

G1051 Medusa Group

Medusa Group has leveraged Windows Native API functions to execute payloads.[146]

S1244 Medusa Ransomware

Medusa Ransomware has leveraged Windows Native API functions to execute payloads.[146]

S0576 MegaCortex

After escalating privileges, MegaCortex calls `TerminateProcess()`, `CreateRemoteThread`, and other Win32 APIs.[147]

G0045 menuPass

menuPass has used native APIs including `GetModuleFileName`, `lstrcat`, `CreateFile`, and `ReadFile`.[148]

S1059 metaMain

metaMain can execute an operator-provided Windows command by leveraging functions such as `WinExec`, `WriteFile`, and `ReadFile`.[142][149]

S0455 Metamorfo

Metamorfo has used native WINAPI calls.[150][151]

S0688 Meteor

Meteor can use `WinAPI` to remove a victim machine from an Active Directory domain.[152]

S1015 Milan

Milan can use the API `DnsQuery_A` for DNS resolution.[125]

S0084 Mis-Type

Mis-Type has used Windows API calls, including `NetUserAdd` and `NetUserDel`.[153]

S0083 Misdat

Misdat has used Windows APIs, including `ExitWindowsEx` and `GetKeyboardType`.[153]

S1122 Mispadu

Mispadu has used a variety of Windows API calls, including ShellExecute and WriteProcessMemory.[154][155]

S0256 Mosquito

Mosquito leverages the CreateProcess() and LoadLibrary() calls to execute files with the .dll and .exe extensions.[156]

G0129 Mustang Panda

Mustang Panda has used various Windows API calls during execution and defense evasion.[157][50][158][64][65][159][160][58][51][161][162][163]

S0630 Nebulae

Nebulae has the ability to use `CreateProcess` to execute a process.[164]

S0457 Netwalker

Netwalker can use Windows API functions to inject the ransomware DLL.[165]

S0198 NETWIRE

NETWIRE can use Native API including `CreateProcess` `GetProcessById` , and `WriteProcessMemory` .[166]

S1090 NightClub

NightClub can use multiple native APIs including `GetKeyState` , `GetForegroundWindow` , `GetWindowThreadProcessId` , and `GetKeyboardLayout` .[167]

S1100 Ninja

The Ninja loader can call Windows APIs for discovery, process injection, and payload decryption.[168][169]

S0385 njRAT

njRAT has used the ShellExecute() function within a script.[170]

S1170 ODAgent

ODAgent can pass commands using native APIs.[171]

S1172 OilBooster

OilBooster has used the `ShowWindow` and `CreateProcessW` APIs.[171]

C0022 Operation Dream Job

During Operation Dream Job, Lazarus Group used Windows API `ObtainUserAgentString` to obtain the victim's User-Agent and used the value to connect to their C2 server.[131]

C0006 Operation Honeybee

During Operation Honeybee, the threat actors deployed malware that used API calls, including `CreateProcessAsUser` .[172]

C0013 Operation Sharpshooter

During Operation Sharpshooter, the first stage downloader resolved various Windows libraries and APIs, including `LoadLibraryA()` , `GetProcAddress()` , and `CreateProcessA()` .[173]

C0014 Operation Wocao

During Operation Wocao, threat actors used the `CreateProcessA` and `ShellExecute` API functions to launch commands after being injected into a selected process.[174]

S1233 PAKLOG

PAKLOG has used Windows API `SetWindowsHookExW` with `idHook` set to `WH_KEYBOARD_LL` and a custom hook procedure to support its keylogging functions.[162]

S1050 PcShare

PcShare has used a variety of Windows API functions.[98]

S1145 Pikabot

Pikabot uses native Windows APIs to determine if the process is being debugged and analyzed, such as `CheckRemoteDebuggerPresent`, `NtQueryInformationProcess`, `ProcessDebugPort`, and `ProcessDebugFlags`.[175] Other Pikabot variants populate a global list of Windows API addresses from the `NTDLL` and `KERNEL32` libraries, and references these items instead of calling the API items to obfuscate execution.[176]

S0517 Pillowmint

Pillowmint has used multiple native Windows APIs to execute and conduct process injections.[177]

S0501 PipeMon

PipeMon's first stage has been executed by a call to `CreateProcess` with the decryption password in an argument. PipeMon has used a call to `LoadLibrary` to load its installer.[178]

S0435 PLEAD

PLEAD can use `ShellExecute` to execute applications.[179]

S0013 PlugX

PlugX can use the Windows API functions `GetProcAddress`, `LoadLibrary`, and `CreateProcess` to execute another process.[157][180][181]

S0518 PolyglotDuke

PolyglotDuke can use `LoadLibraryW` and `CreateProcess` to load and execute code.[95]

S0453 Pony

Pony has used several Windows functions for various purposes.[182]

S1058 Prestige

Prestige has used the `Wow64DisableWow64FsRedirection()` and `Wow64RevertWow64FsRedirection()` functions to disable and restore file system redirection.[183]

S0147 Pteranodon

Pteranodon has used various API calls.[184]

S1228 PUBLOAD

PUBLOAD has used various Windows API calls during execution, when establishing persistence and defense evasion.[158][65] PUBLOAD stager leveraged Windows API functions with callback including `GrayStringW`, `EnumDateFormatsA`, and `LineDDA` to bypass anti-virus monitoring. [160] PUBLOAD has also utilized other native windows API functions with callback functions such as `EnumChildWindows` and `EnumSystemLanguageGroupsA`. [51]

S0650 QakBot

QakBot can use `GetProcAddress` to help delete malicious strings from memory.[185]

S1242 Qilin

Qilin can attempt to log on to the local computer via `LogonUserW` and use `GetLogicalDrives()` and `EnumResourceW()` for discovery.[186][187]

S1076 QUIETCANARY

QUIETCANARY can call `System.Net.HttpWebRequest` to identify the default proxy configured on the victim computer.[188]

S0629 RainyDay

The file collection tool used by RainyDay can utilize native API including `ReadDirectoryChangeW` for folder monitoring.[164]

S0458 Ramsay

Ramsay can use Windows API functions such as `WriteFile`, `CloseHandle`, and `GetCurrentHwProfile` during its collection and file storage operations. Ramsay can execute its embedded components via `CreateProcessA` and `ShellExecute`.[189]

S0662 RCSession

RCSession can use WinSock API for communication including `WSASend` and `WSARecv`.[190]

S0416 RDFSNIFFER

RDFSNIFFER has used several Win32 API functions to interact with the victim machine.[191]

S0496 REvil

REvil can use Native API for execution and to retrieve active services.[192][193]

S0448 Rising Sun

Rising Sun used dynamic API resolutions to various Windows APIs by leveraging `LoadLibrary()` and `GetProcAddress()`.[173]

S0240 ROKRAT

ROKRAT can use a variety of API calls to execute shellcode.[194]

S1078 RotaJakiro

When executing with non-root permissions, RotaJakiro uses the the `shmget` API to create shared memory between other known RotaJakiro processes. RotaJakiro also uses the `execvp` API to help its dead process "resurrect".[195]

S1073 Royal

Royal can use multiple APIs for discovery, communication, and execution.[196]

S0148 RTM

RTM can use the `FindNextUrlCacheEntryA` and `FindFirstUrlCacheEntryA` functions to search for specific strings within browser history.[197]

S0446 Ryuk

Ryuk has used multiple native APIs including `ShellExecuteW` to run executables, `GetWindowsDirectoryW` to create folders, and `VirtualAlloc`, `WriteProcessMemory`, and `CreateRemoteThread` for process injection.[198]

S0085 S-Type

S-Type has used Windows APIs, including `GetKeyboardType`, `NetUserAdd`, and `NetUserDel`.[153]

S1210 Sagerunex

Sagerunex calls the `WaitForSingleObject` API function as part of time-check logic.[199]

S1018 Saint Bot

Saint Bot has used different API calls, including `GetProcAddress`, `VirtualAllocEx`, `WriteProcessMemory`, `CreateProcessA`, and `SetThreadContext`.[200][201]

S1099 Samurai

Samurai has the ability to call Windows APIs.[168]

G0034 Sandworm Team

Sandworm Team uses Prestige to disable and restore file system redirection by using the following functions: `Wow64DisableWow64FsRedirection()` and `Wow64RevertWow64FsRedirection()`.[183]

S1085 Sardonic

Sardonic has the ability to call Win32 API functions to determine if `powershell.exe` is running.[202]

S1089 SharpDisco

SharpDisco can leverage Native APIs through plugins including `GetLogicalDrives`.[167]

S0444 ShimRat

ShimRat has used Windows API functions to install the service and shim.[203]

S0445 ShimRatReporter

ShimRatReporter used several Windows API functions to gather information from the infected system.[203]

G1008 SideCopy

SideCopy has executed malware by calling the API function `CreateProcessW`.[204]

S0610 SideTwist

SideTwist can use `GetUserNameW`, `GetComputerNameW`, and `GetComputerNameExW` to gather information.[205]

G0091 Silence

Silence has leveraged the Windows API, including using CreateProcess() or ShellExecute(), to perform a variety of tasks.[206][207]

S0692 SILENTTRINITY

SILENTTRINITY has the ability to leverage API including `GetProcAddress` and `LoadLibrary`.[208]

S0623 Siloscape

Siloscape makes various native API calls.[209]

S0627 SodaMaster

SodaMaster can use `RegOpenKeyW` to access the Registry.[210]

S0615 SombRAT

SombRAT has the ability to respawn itself using `ShellExecuteW` and `CreateProcessW`.[74]

S1234 SplatCloak

SplatCloak has utilized Native Windows API calls dynamically through `ZwQuerySystemInformation`.[162]

S1232 SplatDropper

SplatDropper has utilized hashed Native Windows API calls.[162]

S1227 StarProxy

StarProxy has used native windows API calls such as `GetLocalTime()` to retrieve system data.[163]

S1200 StealBit

StealBit can use native APIs including `LoadLibraryExA` for execution and `NtSetInformationProcess` for defense evasion purposes.[211]

S1034 StrifeWater

StrifeWater can use a variety of APIs for execution.[212]

S0603 Stuxnet

Stuxnet uses the SetSecurityDescriptorDacl API to reduce object integrity levels.[213]

S0562 SUNSPOT

SUNSPOT used Windows API functions such as `MoveFileEx` and `NtQueryInformationProcess` as part of the SUNBURST injection process.[214]

S1064 SVCReady

SVCReady can use Windows API calls to gather information from an infected host.[215]

S0242 SynAck

SynAck parses the export tables of system DLLs to locate and call various Windows API functions.[216][217]

S0663 SysUpdate

SysUpdate can call the `GetNetworkParams` API as part of its C2 establishment process.[218]

G0092 TA505

TA505 has deployed payloads that use Windows API calls on a compromised host.[219]

S0011 Taidoor

Taidoor has the ability to use native APIs for execution including `GetProcessHeap`, `GetProcAddress`, and `LoadLibrary`.[220][221]

S0595 ThiefQuest

ThiefQuest uses various API to perform behaviors such as executing payloads and performing local enumeration.[222]

S0668 TinyTurla

TinyTurla has used `WinHTTP`, `CreateProcess`, and other APIs for C2 communications and other functions.[223]

G1022 ToddyCat

ToddyCat has used `WinExec` to execute commands received from C2 on compromised hosts.[169]

S1239 TONESHELL

TONESHELL has utilized Native Windows API functions such as `WriteProcessMemory` and `CreateRemoteThreadEx`.[159] TONESHELL has also utilized Windows API functions for creating seed values including `CoCreateGuid` and `GetTickCount`.[65][163] TONESHELL has leveraged the legitimate API function `EnumSystemLocalesA` to run its shellcode through the callback function.[51]

S0678 Torisma

Torisma has used various Windows API calls.[224]

S0266 TrickBot

TrickBot uses the Windows API call, CreateProcessW(), to manage execution flow.[225] TrickBot has also used `Nt*` API functions to perform Process Injection.[226]

G0081 Tropic Trooper

Tropic Trooper has used multiple Windows APIs including HttpInitialize, HttpCreateHttpHandle, and HttpAddUrl.[227]

G0010 Turla

Turla and its RPC backdoors have used APIs calls for various tasks related to subverting AMSI and accessing then executing commands through RPC and/or named pipes.[228]

S0022 Uroburos

Uroburos can use native Windows APIs including `GetHostByName`.[229]

S0386 Ursnif

Ursnif has used `CreateProcessW` to create child processes.[230]

S0180 Volgmer

Volgmer executes payloads using the Windows API call CreateProcessW().[231]

S0670 WarzoneRAT

WarzoneRAT can use a variety of API calls on a compromised host.[232]

S0612 WastedLocker

WastedLocker's custom crypter, CryptOne, leveraged the VirtualAlloc() API function to help execute the payload.[233]

S0579 Waterbear

Waterbear can leverage API functions for execution.[234]

S0689 WhisperGate

WhisperGate has used the `ExitWindowsEx` to flush file buffers to disk and stop running processes and other API calls.[235][236]

S0466 WindTail

WindTail can invoke Apple APIs `contentsOfDirectoryAtPath`, `pathExtension`, and (string) `compare`.[237]

S0141 Winnti for Windows

Winnti for Windows can use Native API to create a new process and to start services.[238]

S1065 Woody RAT

Woody RAT can use multiple native APIs, including `WriteProcessMemory`, `CreateProcess`, and `CreateRemoteThread` for process injection.[239]

S0161 XAgentOSX

XAgentOSX contains the execFile function to execute a specified file on the system using the NSTask:launch method.[240]

S0653 xCaon

xCaon has leveraged native OS function calls to retrieve victim's network adapter's information using GetAdapterInfo() API.[52]

S1207 XLoader

XLoader uses the native Windows API for functionality, including defense evasion.[241]

S1151 ZeroCleare

ZeroCleare can call the `GetSystemDirectoryW` API to locate the system directory.[62]

S0412 ZxShell

ZxShell can leverage native API including `RegisterServiceCtrlHandler` to register a service.RegisterServiceCtrlHandler

S1013 ZxxZ

ZxxZ has used API functions such as `Process32First`, `Process32Next`, and `ShellExecuteA`.[242]
