Analysis Report for: 8B6F648C84EB8100A6072CC6D8555215.au3

Overall Functionality

This C code exhibits characteristics strongly suggestive of malware. It performs a series of obfuscated operations, including string decryption, DLL structure manipulation, and numerous calls to AutoIt functions (indicated by the `$` prefix on variables and function names). The code installs files ("indivisibility" and "Lymnaeidae"), interacts with the registry, executes external processes, and manipulates system files and directories in a way consistent with malicious activity. The core functionality seems to be centered around installing a payload (likely contained within the installed files) and executing it, alongside a large number of actions to evade detection and maintain persistence. The extensive use of nested loops and conditional statements further obscures the true purpose.

Function Summaries

* **`RMOATDF($mikebpdmop)`:** This function takes a space-delimited string of numbers (`$mikebpdmop`) as input. It splits the string, converts each number to its ASCII character equivalent, adds an offset (0xfffffff9), and concatenates the resulting characters. The result is a string; it seems likely this decrypts/decodes input strings.

* **`PKXQRORBF()`:** This function performs various actions using AutoIt functions, including setting GUI controls' colors and states, checking for the existence of processes ("dxrtm.exe", "command_line.exe"), and performing basic mathematical operations. These actions are seemingly unrelated but could be used to hinder reverse engineering or as indicators of malware functionality.

* **`LWOMONRQXX()`:** This function interacts with files, including getting shortcut information, creating directories, moving files, and reading and writing data. The file paths used suggest an attempt to blend in with legitimate system files, enhancing stealth.

* **`FCCYODFV()`:** This function uses AutoIt functions related to network operations (Ping), hotkey setting, window manipulation (WinWaitClose, WinSetOnTop), and file operations. The Pings to multiple URLs further suggest possible communication with a command-and-control (C&C) server.

* **`QDEBBLW()`:** Similar to other functions, this function mixes file operations (FileWrite, FileWriteLine, FileSetAttrib) with GUI control manipulation and process priority alteration. The actions are diverse and obfuscated.

* **`OBVKYNVMG()`:** This function interacts with files (FileSaveDialog, FileExists, FileMove, FileDelete), processes (ProcessExists, ProcessClose, ProcessWaitClose), and the registry (RegEnumVal). The `FileSaveDialog` suggests it might allow the malware to save configuration or other data.

Control Flow

The control flow is extremely complex due to deeply nested loops and conditional statements. These are used extensively to obfuscate the code's true purpose. The nested loops iterate thousands of times, performing incremental operations on global variables, potentially for obfuscation or to generate complex cryptographic keys or other obfuscated data. Conditional statements (If-ElseIf-Else) frequently alter the values of global variables based on their current values and arbitrary numerical thresholds, making static analysis incredibly difficult. The lack of comments makes this more difficult to reverse engineer.

Data Structures

The primary data structure is the AutoIt DLL structure created with `DllStructCreate`. The data contained within this structure (`$hgqqessgky`) seems to be the decrypted payload. The structure itself acts as a container for this data, possibly aiding in its execution within the malicious code. Global variables (`$siszqhctbb`, `$hgqqessgky`, `$iyhbuikhta`, `$cixbovvbyg`, etc.) store intermediate results and critical data throughout the execution. These global variables are extensively modified using arithmetic operations and conditional logic within nested loops, contributing to the obfuscation.

Malware Family Suggestion

Given the obfuscation techniques, file system manipulation, registry interaction, process manipulation, and network activity (pinging multiple external IPs), this code strongly resembles a **multi-component dropper or installer** for a more sophisticated piece of malware. The use of AutoIt suggests it is likely a piece of malware that may use it as a scripting language. The code could be a part of a larger malware family, potentially including information stealers, ransomware, or backdoors depending on the exact nature of the payload delivered by the "indivisibility" and "Lymnaeidae" files. The high degree of obfuscation is common in advanced malware families that utilize techniques to avoid being easily analyzed or detected. It's highly unlikely to be benign.

Important Note: Analyzing this code directly is highly risky. It should only be done in a carefully controlled and isolated virtual machine environment. Any attempt to run this code outside of such an environment could result in serious damage to your system.