Analysis Report for: FC0343F56034CC3E42B1FD1144EF101E.au3

Overall Functionality

This C code is highly obfuscated, making a precise determination of its overall functionality difficult. However, the code's structure strongly suggests malicious activity. It performs numerous system calls, including functions related to memory management (`MEMGETSTATS()`), file system operations (`FILEWRITE()`, `FILEOPEN()`, `FILECLOSE()`, `FILECOPY()`, `FILEDELETE()`, `DIRGETSIZE()`), process manipulation (`PROCESSEXISTS()`), and potentially DLL interaction (`DLLSTRUCTCREATE()`, `DLLSTRUCTGETDATA()`, `DLLSTRUCTSETDATA()`, `DLLCALL()` , `DLLCALLADDRESS()`). The extensive use of obfuscation techniques like meaningless variable names and encoded strings points towards an attempt to hide its true purpose. The code creates and manipulates several DLL structures, suggesting interaction with external libraries or modules. The final result seems to be the creation and writing to files in the `@LOCALAPPDATADIR` and potentially other locations. The prevalence of `WHILE` loops and `SWITCH` statements creates a complex control flow that further complicates analysis. A crucial observation is the use of `SETERROR()`, suggesting error handling is in place for several operations. The presence of a function called `VOYUERLIT`, which appears to extract data from DLL structures, further points towards potential malicious data extraction and manipulation.

Function Summaries

* **`FUNC SOMETHINGVCR($LOREASONABLYTEXASDIG)`:** This function initializes a DLL structure (`$HOSPITALSMAIL`) using obfuscated strings and the input parameter `$LOREASONABLYTEXASDIG`. It then exits a loop. The purpose is unclear but likely involves some setup or initialization phase.

* **`FUNC SUGGESTINGTALKING($PUBCINDY, $HEATED)`:** This function appears to decode obfuscated strings. It takes an obfuscated string (`$PUBCINDY`) and an integer (`$HEATED`) as input. It reverses the string and then decodes it based on the `$HEATED` value, likely by subtracting `$HEATED` from each character code in the obfuscated string to retrieve the original string. The result is a decoded string which is returned. The decoded string is then used for various system calls in other parts of the program.

* **`FUNC ROBOTSPACKARDROUTINES($SOFTWARECOLOUREMPLOYERSTP)`:** This function makes a DLL call (`$CHOSETAGSNONESPEAK1`) using obfuscated strings. It likely interacts with a DLL to perform some action. The purpose and exact nature of the DLL interaction remains unclear due to obfuscation.

* **`FUNC VOYUERLIT($THATSSUPPORTFLOWERS, $VARIABLEREPRODUCTIVEIDENTIFYINGEDUCATIONAL, $JURYTOSHIBAORGANIZATIONASSUMES = 0)`:** This function retrieves data from a DLL structure. It takes a DLL structure pointer (`$THATSSUPPORTFLOWERS`), an index (`$VARIABLEREPRODUCTIVEIDENTIFYINGEDUCATIONAL`), and an optional offset (`$JURYTOSHIBAORGANIZATIONASSUMES`). The function returns the data retrieved from the structure at the specified location. This is a highly suspicious function, likely used to extract sensitive data from memory.

* **`FUNC RELIABLEFC($IMPLEMENTATIONCOULDCHATLIS)`:** This function makes a DLL call (`$KURTFACINGNEURAL`) with obfuscated strings and an integer input `$IMPLEMENTATIONCOULDCHATLIS`. It seems to retrieve a single value from a DLL function. Its function remains obscure but potentially retrieves an important value for the program's operation

* **`FUNC CHARACTERISTICMISTRESS($COOKBOOKFOURTHADDRESSES)`:** This function performs a DLL call (`$BEGANDAILYDEUTSCH`) with obfuscated strings and an integer parameter. It seems to retrieve an integer value which is then used as an offset in another part of the code.

* **`FUNC PIANOEAAPPOINTMENTSDISPUTES($TRAINSMALLERRACKS, $VOLFITTED)`:** This function retrieves a pointer ($SHAPEMENTIONED) to the input data structure ($TRAINSMALLERRACKS) then creates several local variables and structures, performing various system operations. The meaning and purpose are obfuscated.

* **`FUNC PUBMEDFOSTERLONGER($SURGEZUSDALE, $COMPOSEDKNIGHT)`:** This function performs a DLL call (`$SIMULATIONMOREOVERMOLDOVAKEITH`) using obfuscated strings. The function then checks for errors and returns an integer value. This function makes a DLL call which likely interacts with external data structures.

* **`FUNC SLUTSSANDWICH($LAYAUDIENCESHUT, $DISABLESULLIVANHANDBAGS)`:** This function executes a command (`EXECUTE()`) using an obfuscated string and assigns the output to a variable, then creates and manipulates another DLL structure. The ultimate purpose is obfuscated.

* **`FUNC TWINKSOUTER($VOTEEMISSIONS, $NEARKEITHACRYLIC, $CLOCKSITALIANO, $WEARINGMIXTURETRICKS, $VANCOUVERIMMUNE, $SINGHOPED, $PIEUSRDADDY)`:** This function initializes a string variable and then performs actions based on a complex conditional. The actions and overall functionality are difficult to interpret because of obfuscation.

* **`FUNC TURNINGSTABILITYATHLETES($LOREASONABLYTEXASDIG, $SOFTWARECOLOUREMPLOYERSTP, $BUTTONCONF = SUGGESTINGTALKING("103z122z114z110z113z116z103z116z48z103z122z103", 2 + 0))`:** This function creates a DLL structure and then performs actions based on a complex conditional; this function makes a DLL call, which likely performs some core functionality.

* **`FUNC FOOFOGDEFECTS($PUBLICATIONKATE, $IUNDOANYWAYTREASURER, $PRESENTLYSCREENSAVER, $TRIBUNEMLB, $PERMITBUDGETDOCUMENTATIONSURELY, $INNOVATIVEQUITTWICEDOD, $ALLENREPAIRGYMLIL)`:** This function initializes a string variable and then performs several actions based on a complex conditional involving loop and DLL call functions; the actual functionality of the function is obfuscated.

* **`FUNC ANDSUPERLOGINUPSET($WISHESSECDIESLISTENING, $REBATELAS, $FLOORSUSDAOPENINGCASIO, $MESSLOGGING, $GENEROUSIMPRESSEDGLOW, $BIDGRAIN, $KEEPSDELEGATIONROUNDSOCCASIONS)`:** This function initializes a string variable and

then performs actions based on several complex conditional statements. The purpose of this function is unclear.

* **`FUNC BERLINCAMBRIDGEWEDDINGS($TRAINSMALLERRACKS, $LANDSRANKINGIDENTIFICATION, $GUESSCELLAFTERNOONPATCHES, $RECEPTORSARTHURJOINTBEINGS)`:** This function uses its parameter `$TRAINSMALLERRACKS` (a DLL structure) and calculates a size (`$TOPICSUNDERLYINGLOTUS`). It then performs further operations with this size, possibly iterating through elements. The logic suggests data manipulation based on an input structure.

* **`FUNC CITIZENSHIPGENTLYPOSSESSVITAL($FPMONTGOMERY)`:** This function makes a DLL call (`$SIMULATIONMOREOVERMOLDOVAKEITH`) with obfuscated strings. It appears to perform some operation, potentially checking for existence and returning a boolean value.

* **`FUNC COMPOSITIONSLOTBLADELANCE($FPMONTGOMERY, $TOPICSUNDERLYINGLOTUS, $THREESOMEWILSON)`:** This function creates a DLL structure and then sets data within that structure using other function calls. The specific actions remain unclear due to obfuscation.

Control Flow

The code uses nested `WHILE` loops and `SWITCH` statements extensively, creating a complex and deeply nested control flow. Many functions contain loops that execute a specific number of times (e.g., `WHILE 70`, `WHILE 394`), suggesting the program follows a predetermined sequence of actions. The `SWITCH` statements select different operations based on the value of a variable, resulting in numerous possible execution paths. Conditional statements (`IF` statements) also exist, leading to more variations. This makes static analysis extremely difficult and makes it more difficult to predict the actions of the code

Data Structures

The code heavily uses a custom DLL structure, likely a structure implemented by an external DLL. The functions `DLLSTRUCTCREATE()`, `DLLSTRUCTGETDATA()`, `DLLSTRUCTSETDATA()`, `DLLSTRUCTGETPTR()`, and `DLLSTRUCTGETSIZE()` are used to create, access, and manipulate these structures. The purpose and exact structure of these custom DLL structures are not apparent in the decompiled code, but the structure's usage indicates data storage and manipulation.

Malware Family Suggestion

Given the obfuscation, extensive use of DLL functions, file system manipulation, and process monitoring, this code is highly suggestive of a **generic malware downloader/dropper or backdoor**. The code's behavior resembles that of malware designed to download and execute additional malicious payloads or to maintain persistent access to an infected system. The sophisticated obfuscation techniques are typically employed by advanced malware families to evade detection. Without further dynamic analysis, a more specific malware family cannot be reliably identified. The decoded strings used in system calls and DLL functions would need to be analyzed to find better indication on malware type.

Additional Notes

The excessive use of obfuscation significantly hampers static analysis. To gain a complete understanding of the malware's capabilities, dynamic analysis (running the code in a controlled environment) is needed to observe its behavior in real-time. Reverse engineering tools and sandboxes would be necessary to decode the strings and understand the actual functions being called and the data being manipulated.