Analysis Report for: F0DA118AB49F7E0138B7BDE65AF94889.exe.c

Overall Functionality

This C code appears to be part of a malicious program, likely a rootkit or backdoor. Its primary function is to write a registry key ("PanCar") under `HKEY_LOCAL_MACHINE\Software\PaloAlto` with a value of 1. If it fails to create or write to this key, it logs the error to files ("C:\\KeyOpenFailed.txt" and "C:\\KeyValueFailed.txt"). The code also extensively uses function pointers, likely to load and use functions from `USER32.DLL` dynamically, obfuscating its actions. The numerous other functions suggest additional functionality beyond simple registry manipulation, potentially involving process and window manipulation, and exception handling. The use of encoded pointers further enhances the obfuscation.

Function Summaries

* **`sub_401000()`**: This function attempts to create a registry key under `HKEY_LOCAL_MACHINE\Software\PaloAlto` with the value name "PanCar" and a DWORD value of 1. It logs errors to files if the key creation or value setting fails. It uses Windows API functions (`RegCreateKeyExW`, `RegSetValueExW`, `RegCloseKey`). Returns an `LSTATUS` code indicating success or failure.

* **`main()`**: The entry point of the program. It simply calls `sub_401000()` and exits.

* **`sub_401937()`**: This function appears to call a weak symbol `flsall(1)`, potentially related to fiber local storage. Its purpose is unclear without more context about `flsall`. Returns an integer.

* **`sub_401940()`**: This function assigns the input pointer `a1` to a global variable `Ptr`. This suggests potential pointer manipulation or data hiding. Returns a void pointer.

* **`sub_401B65()`**: Returns a pointer to a global variable `off_40D180`, likely for data access or indirect function calls.

* **`sub_4025B8()`**: This function seems to write data from `a2` (likely a character array) to a location using a function pointer `write_char`. It handles potential errors, returning an error code or the number of characters written.

* **`sub_403211()`**: Sets a custom unhandled exception filter `__CxxUnhandledExceptionFilter`. This suggests an attempt to control how exceptions are handled within the program.

* **`sub_403E6F()`**: Returns a pointer to a global array `dword_40BAF0`, possibly used for storing data.

* **`sub_403E95()`**: An empty function, possibly a placeholder or remnant from code modifications.

* **`sub_4043EA()`**: Sets the global variable `dword_40EB48` to 0. Its significance is unclear without further context.

* **`sub_4071C4()`**: Decodes a pointer using `DecodePointer` and returns the result. This points to pointer encoding/decoding being used for obfuscation.

* **`sub_407374()` and `sub_407383()`**: These functions assign their integer arguments to global variables `dword_40EB10` and `dword_40EB14` respectively.

* **`sub_407392()`**: Assigns its void pointer argument to the global variable `dword_40EB18`.

* **`sub_407503()`**: This function is the most complex, dynamically loading functions from `USER32.DLL` (`MessageBoxW`, `GetActiveWindow`, `GetLastActivePopup`, `GetUserObjectInformationW`, `GetProcessWindowStation`). It appears to gather information about the active window and potentially the user's session. Uses encoded function pointers. Returns an integer.

* **`sub_407BD5()`**: Checks for a processor feature (likely related to CPU architecture) using `IsProcessorFeaturePresent` and stores the result.

* **`sub_408645()`**: This function appears to either set the value of a provided pointer to `dword_40EB3C` or return an error if the provided pointer is NULL.

* **`sub_4098F3()`**: This function likely closes a handle (`hObject`) if it's not -1 or -2.

Control Flow

The core control flow is relatively simple: `main` calls `sub_401000`, which attempts registry manipulation and error logging. `sub_407503` exhibits a more complex control flow involving conditional checks based on the presence of dynamically loaded functions and the results of API calls. The function uses multiple conditional branches to determine its behavior. Loops are limited to the `sub_4025B8` function which iterates over a character array.

Data Structures

The primary data structure is the set of global variables, many of which hold function pointers or seemingly arbitrary data, enhancing obfuscation. There are arrays (`dword_40BAF0`) and individual variables for various purposes (error codes, flags, handles etc). The structure is relatively flat and lacks complex nested structures.

Malware Family Suggestion

Based on the functionality (registry key creation, dynamic loading of system functions, file logging, and exception handling), this code strongly suggests a **rootkit or backdoor**. The obfuscation techniques (encoded pointers, weak symbols, function pointer use) are typical of malware designed to evade detection. The purpose of writing the "PanCar" registry key is suspicious and likely serves as a marker or configuration setting for the malware. Further investigation of the functions interacting with `USER32.DLL` would be needed to uncover additional malicious behavior.