Analysis Report for: BackdoorMalware (2).c

Overall Functionality

This C code implements a malicious program that likely belongs to the **Remote Access Trojan (RAT)** family. It establishes a network connection to a remote server, exfiltrates system information (computer name), and executes commands received from the server. The code uses several obfuscation techniques, including custom mathematical functions and seemingly random byte arrays, to hinder analysis. The program also attempts to delete itself after execution.

Function Summaries

* **`sub_401000`, `sub_401040`, `sub_401080`**: These functions appear to be custom bitwise manipulation routines used for obfuscation. They perform bit shifts and XOR operations on input integers, seemingly without a clear cryptographic purpose beyond confusing reverse engineering.

* **`sub_4010C0`**: This function transforms a byte array (8 bytes) into a three-integer array. This transformation is likely another obfuscation step.

* **`sub_401130`**: This function uses the three-integer array manipulated by `sub_4010C0` and applies the obfuscating functions (`sub_401000`, `sub_401040`, `sub_401080`) to them. It's a core part of the encryption/decryption process used in the code.

* **`sub_401190`**: A Boolean function that checks if at least two of its three input values have their high-order bits set. The result is used in `sub_401130`.

* **`sub_4011E0`, `sub_401220`**: These functions perform in-place XOR encryption/decryption of a buffer using the output of `sub_401130`. `sub_401220` simply calls `sub_4011E0`.

* **`WinMain`**: The main entry point of the program. It loads a string (likely a configuration string), then calls `sub_4012C0`.

* **`sub_401270`**: Sends data over a socket. It handles sending data in chunks to ensure complete transmission.

* **`sub_4012C0`**: Initializes events and calls `sub_401320`. It appears to manage thread synchronization.

* **`sub_401320`**: This is the core network communication and command execution function. It parses a configuration string, connects to a remote server, sends system information (computer name), receives commands, and executes them using `beginthread` to create new threads for command processing. This is the central malicious component.

* **`sub_401600`**: Creates two pipes for inter-process communication and returns a structure containing the pipe handles and a process handle.

* **`StartAddress`**: Creates two threads (`sub_401940`, `sub_401A70`) to handle communication and command execution and manages the process and pipe cleanup.

* **`sub_401860`**: Creates a process (`cmd.exe`) with the input and output redirected to the pipes created by `sub_401600`.

* **`sub_401940`**: A thread function that reads data from a named pipe, decrypts it using `sub_4011E0`, and sends it to the remote server.

* **`sub_401A70`**: A thread function that receives data from the remote server, encrypts it using `sub_401220`, and writes it to a named pipe.

* **`sub_401B50`**: Attempts to delete the executable itself using `cmd.exe` and `del`.

* **`UserMathErrorFunction`**: A dummy function that returns 0; likely a placeholder or remnant from development.

Control Flow

The control flow is complex due to obfuscation, but the key paths are:

1. **Initialization**: `WinMain` -> `sub_4012C0` -> `sub_401320` sets up the network connection and waits for commands.

2. **Network Communication**: `sub_401320` establishes a socket connection based on parsed configuration data, sends the computer name to the remote server, and then enters a loop receiving and executing commands.

3. **Command Execution**: Received commands are passed to `StartAddress`, which spawns threads to handle communication and execution via pipes and `cmd.exe`.

Data Structures

* **Arrays:** Several arrays (`byte_403014`, `byte_4030A1`, `aSyCmd`, etc.) are used, primarily for obfuscation and storing command strings.

* **`SOCKET`**: The standard Windows socket structure for network communication.

* **Pipes:** The code extensively uses named pipes (`HANDLE`) for inter-process communication between the main thread and the threads

managing command execution.

Malware Family Suggestion

The functionality strongly indicates this is a **Remote Access Trojan (RAT)**. It connects to a remote server, receives commands, executes those commands on the infected system (using `cmd.exe`), and exfiltrates data. The self-deletion attempt is a common characteristic of malware designed to evade detection. The extensive use of obfuscation further reinforces its malicious intent.