Analysis Report for: b.xls

Overall Functionality

The provided code is a VBA macro embedded within an Excel (.xls) file. The main functionality is centered around the `Auto_Close()` subroutine, which executes when the workbook is closed. This subroutine writes the current date and time to cell A1 of the first worksheet. The rest of the macros are empty. The `xlm_macro.txt` section appears to contain metadata related to the Excel workbook's sheets, not executable code. Therefore, this is not a full C code analysis, because the provided input is VBA code within an excel file.

Function Summaries

* **`Auto_Close()`**: This subroutine is automatically executed when the Excel workbook is closed. It takes no parameters and returns no value (it's a `Sub` routine, not a `Function`). Its purpose is to timestamp the workbook's closure by writing the current date and time to the top-left cell (A1) of the first worksheet.

Control Flow

* **`Auto_Close()`**: The control flow is straightforward and linear. It consists of a single statement: `Worksheets(1).Cells(1, 1).Value = Now()`. This statement directly accesses the first worksheet (`Worksheets(1)`), selects the cell at row 1, column 1 (`Cells(1, 1)`), and assigns the current date and time (`Now()`) to its value. There are no loops or conditional statements within this subroutine.

Data Structures

The code directly interacts with the Excel worksheet's cells as a two-dimensional array-like structure. `Worksheets(1).Cells(1, 1)` represents a specific cell within this structure, using row and column indices. No other explicit data structures are defined within the provided VBA code.

Malware Family Suggestion

Based on the observed functionality, this VBA macro is not inherently malicious. Its behavior is simple logging—recording the closing time of the Excel workbook. However, this seemingly benign action could be part of a larger attack. The timestamp could be used to correlate activity with other events or to create a trail for a more sophisticated malicious activity (e.g., data exfiltration at specific times). The fact that the other macros are empty might be a technique to obfuscate malicious code and make it more difficult to detect. Without further context or examination of the rest of the Excel file, it is difficult to definitively categorize it as belonging to a specific malware family. It could be a component of a larger attack, potentially a simple data collection tool, and further investigation into the overall Excel file is needed before a definitive classification. It does not display characteristics of common malware families like ransomware, trojans, or viruses. The lack of network communication or file system modification makes it less likely to be highly destructive. However, one should always treat suspicious VBA macros with caution.

Important Note: The analysis is limited by the provided snippet. A complete analysis requires examining the entire Excel file, including other potentially hidden or obfuscated code, and analyzing the file's behavior in a controlled environment (sandbox) to determine its full capabilities and potential harm.