Analysis Report for: 43.txt

Overall Functionality

The provided code is VBA (Visual Basic for Applications) code embedded within an Excel (.xls) file. It doesn't contain any executable code in the traditional sense (like a C program). Instead, it consists of event handlers that respond to user actions within a spreadsheet. Specifically, the code monitors cell selections on "Sheet1". If a cell in column 1 (rows 2 and above row 4) or column 8 (rows 4 and above) is selected, it beeps, moves the selection to the adjacent cell to the right, and displays a message box indicating that the selected cell is read-only. This suggests that the VBA code is designed to restrict user access to specific cells within the spreadsheet. The `xlm_macro` section shows metadata about the workbook's sheets.

Function Summaries

The code contains only one VBA subroutine:

* **`Worksheet_SelectionChange(ByVal Target As Range)`**: This is an event handler that automatically executes whenever the user changes the selected cell(s) on "Sheet1".
* **Parameters:** `Target` is a `Range` object representing the newly selected cell(s).
* **Return Value:** This subroutine has no explicit return value.

Control Flow

The `Worksheet_SelectionChange` subroutine's control flow is primarily based on nested `If` statements:

1. **Check Column:** It first checks if the selected cell (`Target`) is in column 1 (A) or column 8 (H).

2. **Check Row (Column 1):** If the cell is in column 1, it further checks the row number:
* If the row is 2, it executes the actions (beep, select adjacent cell, display message box).
* If the row is greater than 4, it also executes the same actions.

3. **Check Row (Column 8):** If the cell is in column 8, it checks if the row number is greater than 4. If true, it executes the same actions (beep, select adjacent cell, display message box).

4. **Actions:** The actions consist of:
* `Beep`: Plays a system beep sound.
* `Cells(Target.Row, Target.Column).Offset(0, 1).Select`: Selects the cell immediately to the right of the originally selected cell.
* `MsgBox ...`: Displays a message box with a predefined text indicating the cell is read-only.

Data Structures

The primary data structure used is the `Range` object provided by VBA, representing a selection of cells in the spreadsheet. There are no custom data structures defined within this code snippet.

Malware Family Suggestion

This VBA code is not malicious in itself. It simply restricts user access to certain cells within an Excel spreadsheet. However, the presence of this code within a larger context (the entire Excel file) needs to be considered. The code's behavior could be used as a simple obfuscation technique to hide malicious actions taking place in other parts of the Excel workbook or associated files. For example, there could be hidden macros triggered through other means that use the restricted cells to store or manipulate malicious data. A standalone, harmless VBA script is unusual.

The analysis by olevba reports "Suspicious|Hex Strings" suggesting that the broader Excel file might contain obfuscated code or data that warrants further investigation. Without further analysis of the complete Excel file and its associated components, it's impossible to definitively classify it as malware. However, the need for further investigation is flagged. The type of malware it *could* be part of is a macro-based malware, possibly a downloader or dropper of further malicious components.