Analysis Report for: da.txt

Decoded using latin-1...

Overall Functionality

This VBA code appears to be designed for managing and processing data within an Excel workbook, likely for financial or accounting purposes. It features several modules:

* **`EstaPasta_de_trabalho`**: Controls the visibility of scrollbars on different sheets ("Resumo" sheet has scrollbars turned off; others have them on). It also sets scroll areas for specific sheets.
* **`Plan1`**: Contains a single subroutine (`mostra`) that copies the string "Haislan" to the clipboard. This seems like a placeholder or a remnant of earlier functionality.
* **`Plan2`**: This module is the most complex, containing the `Worksheet_Change` event handler. This handler performs extensive data validation and manipulation on columns Q, R, and S, potentially tracking payments, partial payments, or adjustments, based on values in column P (likely representing the contract installment amount). It updates column 17 and 18 based on the changes and also adds data validation with input boxes.
* **`GerarCSV`**: Generates a CSV file from the data in `Plan2`, including a specified competency. It also handles file saving dialog and creates a folder, named "arquivos" under the path of the workbook, copying files within this folder to a network location. This folder is subsequently deleted.
* **`Funcoes`**: A module containing several utility functions:
* `copiaDadosAreaDeTransferencia`: Copies a string to the clipboard.
* `test1`: Attempts to check if a specific folder is open in Windows Explorer and opens it if not.
* `CheckFolderOpen`: Checks if a folder is already open in Windows Explorer.
* `CheckPath`: Checks if a given path points to a valid file or directory.
* `OpenFolder`: Opens a given folder in Windows Explorer.
* `changeFileName`: Presents a dialog box to the user allowing them to change the file name before saving.
* `VerificaVersaoMacro`: Checks for updates to a macro file on a network share and offers to download and replace the current macro if an update is found. This includes saving the current workbook before the update to keep a backup.
* `UserInfo`: Retrieves user information from the system environment variables.
* `CopiarArquivoServidor`: Copies files from a specified folder to a network share, categorizing files based on their names (starting with "I" for input and "O" for output).
* `IncluiAcessoRotina`: Logs access information to a file, then copies files to a network share.
* `OpenExcelDoc`: Opens an Excel file, given a path.
* **`xlm_macro.txt`**: Contains minimal information, just sheet information.

Function Summaries

Function Name	Purpose	Parameters	Return Value
`copiaDadosAreaDeTransferencia`	Copies a string to the clipboard.	`valor` (String)	None
`CheckFolderOpen`	Checks if a folder is open in Explorer and optionally closes it.	`strFolder` (String), `closeFolder` (Boolean)	Boolean (True if open, False otherwise)
`CheckPath`	Checks if a path is a valid file or directory.	`path` (String)	"I" (Invalid), "F" (File), "D" (Directory)
`changeFileName`	Prompts the user to change the file name before saving.	`strPathFileName` (String, ByRef), `strTitleDialog`, `strFilterFile` (Optional Strings)	String (new file name or "")
`UserInfo`	Retrieves user information from environment variables.	`Prop` (String, optional, defaults to "USERNAME")	String (user info or error message)
`CopiarArquivoServidor`	Copies files from a local folder to a network share.	`pathMacroArquivo` (String)	None
`OpenExcelDoc`	Opens a specified Excel file.	`pathFile` (String)	None

Control Flow

The most complex function is `Worksheet_Change` in the `Plan2` module. Its control flow is intricate due to numerous nested `If` statements and `Select Case` blocks. It processes cell changes in columns Q, R, and S (row > 1), triggering different actions depending on the cell's column and value. This function uses heavy input validation, making changes to cells values conditional, and showing dialog boxes, conditional formatting and input boxes to the user.

The `GerarCSV` function involves a loop to iterate through rows of data in `Plan2`, building a CSV string for each row. It also includes error handling for file operations.

The `VerificaVersaoMacro` function shows a complex control flow related to version checking. It retrieves version info from a file on a network share and decides whether to download and replace the current workbook based on comparison of versions and timestamps.

Data Structures

The primary data structure is the Excel worksheet itself. The code manipulates data within this structure, using cells and ranges as the fundamental data units. The `OldValues` Collection in `Plan2` temporarily stores the previous values of cells, enabling comparison before and after changes. Arrays are also used briefly to split strings.

Malware Family Suggestion

While the code itself doesn't directly exhibit malicious behavior like network communication or file deletion without user interaction, the sophistication, obfuscation techniques (commented-out code), extensive data manipulation, and the function to check and update itself from a network share are suspicious. The presence of functions that interact with the file system and registry make it a potential target for malicious modification or to serve as a dropper to install a bigger malware package.

It does not directly fit a known malware family but displays characteristics consistent with a potentially malicious macro. Further analysis (e.g., static and dynamic analysis of the compiled .exe) would be needed to definitively classify it. The potential for malicious intent is high due to its capabilities to manipulate data, files, and folders. It's particularly worrying that the macro checks for and downloads updates from an external source, and it performs numerous system calls (Environ, Shell, CreateObject, GetObject, etc.), which could be exploited. The extensive network operations (`\\df7014sr001`, etc.) suggest a potential for exfiltration of data or other harmful actions. The presence of `On Local Error Resume Next` statements could mask errors and make it hard to track malicious actions.

Therefore, I would classify this as a **potentially malicious VBA macro**, requiring further investigation before being considered safe. It shows characteristics often associated with **malicious tools for financial fraud or data exfiltration**.