Analysis Report for: d3.txt

Overall Functionality

The provided VBA code is part of a malicious Excel macro. The `URLPictureInsert` subroutine iterates through cells in column A (A4:A36) of an active worksheet. For each cell, it treats the cell's content as a filename, inserts a picture from that file into the worksheet, and then resizes and centers the picture relative to the adjacent cell in the next column. The "filenames" are likely to be URLs or paths leading to malicious content, making this a potential attack vector for downloading and executing malware. The `On Error Resume Next` statement is a clear indication of attempts to hide errors and continue execution even if files are not found or images cannot be inserted.

Function Summaries

* **`URLPictureInsert()`:** This is the main subroutine. It takes no parameters and returns no value (Subroutine). Its purpose is to insert images from files specified in cells A4:A36, resize them to fit adjacent cells, and position them centrally within those cells.

Control Flow

The `URLPictureInsert` subroutine's control flow is as follows:

1. **Initialization:** It declares variables for a Shape object (`Pshp`), a Range object (`xRg`), and a Long integer (`xCol`). It also sets error handling to ignore errors (`On Error Resume Next`) and disables screen updating for performance.

2. **Iteration:** It iterates through each cell (`cell`) in the range A4:A36 using a `For Each` loop.

3. **Image Insertion:** Inside the loop, it attempts to insert a picture using the cell's value (`filenam`) as the filename using `ActiveSheet.Pictures.Insert(filenam)`. It selects the inserted picture.

4. **Shape Handling:** It gets the first shape object from the selection. If no shape is found (`If Pshp Is Nothing Then GoTo lab`), it jumps to the cleanup label (`lab`).

5. **Resizing and Positioning:** If a shape is found, it calculates the column for the adjacent cell (`xCol`). It then resizes the picture to be at most 2/3 the width and height of the adjacent cell, ensuring the aspect ratio is not locked (`msoFalse`). Finally, it centers the picture within the adjacent cell using `.Top` and `.Left` properties.

6. **Cleanup:** The `lab:` label handles the case where an image insertion fails. It sets `Pshp` to `Nothing` to release the object.

7. **Loop Continuation:** The loop continues to the next cell.

8. **Finalization:** After the loop, it re-enables screen updating (`Application.ScreenUpdating = True`).

Data Structures

The code primarily uses simple data types:

* **`Shape` object (`Pshp`):** Represents a picture inserted into the worksheet.
* **`Range` object (`Rng`, `xRg`):** Represents a range of cells on the worksheet.
* **`Long` integer (`xCol`):** Represents the column number.
* **`String` (`filenam`):** Stores the filename (potentially a URL).

Malware Family Suggestion

Given the functionality, the code strongly suggests a **downloader** or **dropper** type of malware. The macro downloads files (images) from locations specified in the worksheet. These "images" could actually be executable files or other malicious content. The macro's purpose is not merely to insert images; instead, it uses image insertion as a cover for downloading and potentially executing harmful payloads. The use of error handling (`On Error Resume Next`) is a common technique to hide errors caused by failed downloads or execution attempts. This indicates a deliberate effort to make the malware more resilient to detection and analysis. The lack of error reporting would make debugging of the malware extremely challenging for a user.