Cre: HuyLQ

1. Tóm tắt hiện tượng:

- Không thể xem project VBA trong tab VBA của excel do project được bật password. Vì vậy để xóa được code macro của virus chúng ta phải sử dụng nhiều cách để mở được project đó nhằm xử lí code VBA macro.

* Cách để đặt mật khẩu: <https://blog.hocexcel.online/huong-dan-cach-dat-mat-khau-cho-macro-trong-excel-2013-2016.html>

-Tuy nhiên phương thức mật khẩu này chỉ xảy ra với vbaProject còn file excel và office nói chung sẽ có các phương thức bảo vệ khác.

2. Trước khi đi vào cách để crack password chúng ta sẽ tìm hiểu xem vbaProject bảo vệ dự án của nó như thế nào.

- Ở đây tôi sẽ tập trung vào 3 trường (CMG, DPB, GC) và các bạn có thể tìm hiểu thêm tại: <https://docs.microsoft.com/enus/openspecs/office_file_formats/ms-ovba/d88cb9d8-a475-423d-b370-cc0caaf78628>

(\*) Để xem các trường này bạn cần có một trình đọc mã Hex( link tải tôi sẽ để ở cuối bài).

- CMG: trường này sẽ thông báo cho chúng ta quyền truy cập vào dự án.

Xem thêm thông tin tại: <https://docs.microsoft.com/en-us/openspecs/office_file_formats/ms-ovba/55e770e2-e1a4-4d1c-a8a4-dcfca27d6663>

- Tiếp theo là DPB: trường này là trường mã hóa mật khẩu bằng hàm băm riêng.

Xem thêm thông tin tại: <https://docs.microsoft.com/en-us/openspecs/office_file_formats/ms-ovba/79685426-30fe-43cd-9cbf-7f161c3de7d8>

-Và cuối cùng là trường GC: Trường này sẽ thông báo cho chúng ta dự án có thể xem (hiển thị) hay không.

Xem thêm tại: <https://docs.microsoft.com/en-us/openspecs/office_file_formats/ms-ovba/690c96e8-e862-497f-bb7d-5eacf4dc742a>

3. Các cách crack password:

- Cách 1: Swap Password with Hex Editor:

B1: Tạo các tập tin Excel mới (XLS)

B2: Sau đó, đặt mật khẩu cho VBA Project mới của bạn (Alt + F11)

B3: Lưu tệp đó và nhấn Thoát

B4: Tiếp theo, mở một tệp (VBA Project-xlsm) mà bạn đã tạo gần đây bằng Hex Editor.

Sao chép các dòng dưới đây:

CMG =….

DPB =…

GC =…

B5: Sử dụng Hex Editor, mở tệp VBA Project mà bạn muốn phá mật khẩu, sau đó bạn thay thế các trường (CMG=, DPB=, GC=) trong VBA Project bạn muốn phá mật khẩu bằng các trường bạn sao chép trong project VBA vừa tạo

* Mục đích của việc này nhằm thế chỗ chuỗi mật khẩu đã mã hóa của file bị crack thành chuỗi mật khẩu mã hóa của project VBA bạn vừa tạo.

B6: Sau khi sửa xong, lưu file bạn vừa sửa và mở Project cần crack và xem thành quả.

(\*) Ngoài Hex Editor chúng ta đều có thể sử dụng các trình đọc Hex khác như CFF (Common File Format Explorer, HxD Editor, … )

* Cách 2: Sử dụng WinZip/Winrar

B1: Thay đổi các phần mở rộng tập tin của .xlsm thành .zip (F2). Sau đó mở

tệp vừa sửa với WinRar( Zip) và tìm thư mục .xl

B2: Bây giờ, bạn giải nén tệp vbaProject.bin và sau đó mở tệp đó trong Hex Editor

B3: Tìm kiếm và thay thế DPB bằng DPx và lưu tệp đó

* Mục đích của việc này nhằm làm cho chuỗi mã hóa mật khẩu bị lỗi, vì vậy khi mở project chúng ta chỉ cần vào tạo lại mật khẩu là có thể xem project mà không bị thất thoát dữ liệu.

B4: Trong WinRar (Zip), bạn có thể thay thế tệp vbaProject.bin trước đó bằng tệp vừa sửa.

B5: Sau đó thay đổi phần mở rộng tệp từ .ZIP thành .XLSM

B6: Sau đó bạn mở vbaProject bạn vừa sửa và đổi lại mật khẩu mới (như cách đặt mật khẩu bình thường) do trường mật khẩu cũ đã bị lỗi

B7: Nhập mật khẩu mới và xem thành quả của mình.

* Cách 3: Sử dụng lỗ hổng vbaProject
* Cách hoạt động của cách này:

1. VBE sẽ gọi một hàm hệ thống để tạo hộp thoại mật khẩu.

2. Nếu người dùng nhập đúng mật khẩu và nhấp OK, chức năng này trả về 1. Nếu người dùng nhập sai mật khẩu hoặc nhấp vào Hủy, chức năng này trả về 0.

3. Sau khi đóng hộp thoại, VBE kiểm tra giá trị trả về của hàm hệ thống nếu giá trị này là 1, VBE sẽ "nghĩ" rằng mật khẩu là đúng, do đó dự án VBA bị khóa sẽ được mở.

4. Đoạn mã dưới đây hoán đổi bộ nhớ của hàm gốc được sử dụng để hiển thị hộp thoại mật khẩu với một hàm do người dùng xác định sẽ luôn trả về 1 khi được gọi.

B1: Chuẩn bị một vbaProject mới và chèn đoạn mã dưới đây:

* Với Windows 32bit:

Option Explicit

Private Const PAGE\_EXECUTE\_READWRITE = &H40

Private Declare Sub MoveMemory Lib "kernel32" Alias "RtlMoveMemory" \_

(Destination As Long, Source As Long, ByVal Length As Long)

Private Declare Function VirtualProtect Lib "kernel32" (lpAddress As Long, \_

ByVal dwSize As Long, ByVal flNewProtect As Long, lpflOldProtect As Long) As Long

Private Declare Function GetModuleHandleA Lib "kernel32" (ByVal lpModuleName As String) As Long

Private Declare Function GetProcAddress Lib "kernel32" (ByVal hModule As Long, \_

ByVal lpProcName As String) As Long

Private Declare Function DialogBoxParam Lib "user32" Alias "DialogBoxParamA" (ByVal hInstance As Long, \_

ByVal pTemplateName As Long, ByVal hWndParent As Long, \_

ByVal lpDialogFunc As Long, ByVal dwInitParam As Long) As Integer

Dim HookBytes(0 To 5) As Byte

Dim OriginBytes(0 To 5) As Byte

Dim pFunc As Long

Dim Flag As Boolean

Private Function GetPtr(ByVal Value As Long) As Long

GetPtr = Value

End Function

Public Sub RecoverBytes()

If Flag Then MoveMemory ByVal pFunc, ByVal VarPtr(OriginBytes(0)), 6

End Sub

Public Function Hook() As Boolean

Dim TmpBytes(0 To 5) As Byte

Dim p As Long

Dim OriginProtect As Long

Hook = False

pFunc = GetProcAddress(GetModuleHandleA("user32.dll"), "DialogBoxParamA")

If VirtualProtect(ByVal pFunc, 6, PAGE\_EXECUTE\_READWRITE, OriginProtect) <> 0 Then

MoveMemory ByVal VarPtr(TmpBytes(0)), ByVal pFunc, 6

If TmpBytes(0) <> &H68 Then

MoveMemory ByVal VarPtr(OriginBytes(0)), ByVal pFunc, 6

p = GetPtr(AddressOf MyDialogBoxParam)

HookBytes(0) = &H68

MoveMemory ByVal VarPtr(HookBytes(1)), ByVal VarPtr(p), 4

HookBytes(5) = &HC3

MoveMemory ByVal pFunc, ByVal VarPtr(HookBytes(0)), 6

Flag = True

Hook = True

End If

End If

End Function

Private Function MyDialogBoxParam(ByVal hInstance As Long, \_

ByVal pTemplateName As Long, ByVal hWndParent As Long, \_

ByVal lpDialogFunc As Long, ByVal dwInitParam As Long) As Integer

If pTemplateName = 4070 Then

MyDialogBoxParam = 1

Else

RecoverBytes

MyDialogBoxParam = DialogBoxParam(hInstance, pTemplateName, \_

hWndParent, lpDialogFunc, dwInitParam)

Hook

End If

End Function

Sub unprotected()

If Hook Then

MsgBox "VBA Project is unprotected!", vbInformation, "\*\*\*\*\*"

End If

End Sub

* Với Windows 64bit:

Option Explicit

Private Const PAGE\_EXECUTE\_READWRITE = &H40

Private Declare PtrSafe Sub MoveMemory Lib "kernel32" Alias "RtlMoveMemory" \_

(Destination As LongPtr, Source As LongPtr, ByVal Length As LongPtr)

Private Declare PtrSafe Function VirtualProtect Lib "kernel32" (lpAddress As LongPtr, \_

ByVal dwSize As LongPtr, ByVal flNewProtect As LongPtr, lpflOldProtect As LongPtr) As LongPtr

Private Declare PtrSafe Function GetModuleHandleA Lib "kernel32" (ByVal lpModuleName As String) As LongPtr

Private Declare PtrSafe Function GetProcAddress Lib "kernel32" (ByVal hModule As LongPtr, \_

ByVal lpProcName As String) As LongPtr

Private Declare PtrSafe Function DialogBoxParam Lib "user32" Alias "DialogBoxParamA" (ByVal hInstance As LongPtr, \_

ByVal pTemplateName As LongPtr, ByVal hWndParent As LongPtr, \_

ByVal lpDialogFunc As LongPtr, ByVal dwInitParam As LongPtr) As Integer

Dim HookBytes(0 To 5) As Byte

Dim OriginBytes(0 To 5) As Byte

Dim pFunc As LongPtr

Dim Flag As Boolean

Private Function GetPtr(ByVal Value As LongPtr) As LongPtr

GetPtr = Value

End Function

Public Sub RecoverBytes()

If Flag Then MoveMemory ByVal pFunc, ByVal VarPtr(OriginBytes(0)), 6

End Sub

Public Function Hook() As Boolean

Dim TmpBytes(0 To 5) As Byte

Dim p As LongPtr

Dim OriginProtect As LongPtr

Hook = False

pFunc = GetProcAddress(GetModuleHandleA("user32.dll"), "DialogBoxParamA")

If VirtualProtect(ByVal pFunc, 6, PAGE\_EXECUTE\_READWRITE, OriginProtect) <> 0 Then

MoveMemory ByVal VarPtr(TmpBytes(0)), ByVal pFunc, 6

If TmpBytes(0) <> &H68 Then

MoveMemory ByVal VarPtr(OriginBytes(0)), ByVal pFunc, 6

p = GetPtr(AddressOf MyDialogBoxParam)

HookBytes(0) = &H68

MoveMemory ByVal VarPtr(HookBytes(1)), ByVal VarPtr(p), 4

HookBytes(5) = &HC3

MoveMemory ByVal pFunc, ByVal VarPtr(HookBytes(0)), 6

Flag = True

Hook = True

End If

End If

End Function

Private Function MyDialogBoxParam(ByVal hInstance As LongPtr, \_

ByVal pTemplateName As LongPtr, ByVal hWndParent As LongPtr, \_

ByVal lpDialogFunc As LongPtr, ByVal dwInitParam As LongPtr) As Integer

If pTemplateName = 4070 Then

MyDialogBoxParam = 1

Else

RecoverBytes

MyDialogBoxParam = DialogBoxParam(hInstance, pTemplateName, \_

hWndParent, lpDialogFunc, dwInitParam)

Hook

End If

End Function

Sub unprotected()

If Hook Then

MsgBox "VBA Project is unprotected!", vbInformation, "\*\*\*\*\*"

End If

End Sub

B2: Chạy module này, sau đó excel sẽ hiện hộp thoại “VBA Project is unprotected” và chúng ta quay trở lại file vba Project và xem thành quả.

Nguồn: [https://stackoverflow.com/questions/1026483/is-there-a-way-to-crack-the-password-on-an-excel-vba-project/31005696#31005696](https://stackoverflow.com/questions/1026483/is-there-a-way-to-crack-the-password-on-an-excel-vba-project/31005696%2331005696)

(\*) Ngoài ra chúng ta có thể sử dụng các tools hỗ trợ việc recovery hoặc crack password ở trên mạng

(\*) Link download trình Hex Editor:

+ CFF: <https://explorer-suite-iv.software.informer.com/8.0/>

+ HxD Editor: <https://hxd.en.softonic.com/>

-Hết-