Supplement File 1: Computer code used in email analysis

(A) Code for Parsing Microsoft Outlook ".pst" files

This was accomplished using modified open-source Python code published as "mbox2csv" on GitHub (https://github.com/kiwiandroiddev/mbox2csv/blob/master/scripts/mbox2csv.py).

```
#### BEGIN MBOX2CSV CODE

#!/usr/bin/env python

#Converts an email archive from mbox to csv format with columns outlined in Methods, in single quotes in "writerow" below

#Column names may vary locally and can be determined by inspecting the mbox file output by readpst

#Dates are in ISO 8601 format (e.g. 2015-08-07T18:30:27Z)

##

# Usage:

#for MBOX files converted from readpst this code is invoked as:

#mbox2csv MBOX_FILE [CSV_FILE]

import sys, mailbox, csv

import dateutil.parser as parser

def print_progress (pct_progress):

    sys.stdout.write('\r[{0}] {1}%'.format('#'*(pct_progress/10), pct_progress))

    sys.stdout.flush()

# Recurses down a message payload tree until a string is found

def get_final_payload (msg):

    if isinstance (msg, basestring):

        return msg

    if isinstance (msg.get_payload(), basestring):

        return msg.get_payload()

    return get_final_payload (msg.get_payload()[0])

mbox_file = sys.argv[1]

output_file = sys.argv[2] if len (sys.argv) >2 else 'output.csv'

print 'Reading mbox file.'

messages = mailbox.mbox (mbox_file)

writer = csv.writer (open (output_file, "wb"))

    writer.writerow(['subject', 'from', 'reply-to', 'X-Forefront-Antispam-Report', 'list-owner', 'dkim-signature', 'received', 'date', 'return rcpt', 'disposition rcpt', 'message'])

n = len (messages)

print 'Writing messages.'

for (i, msg) in enumerate (messages):

    body = get_final_payload (msg)

    # convert any date format to ISO!
```

```
    date = parser.parse (msg['date'])
    iso_date = date.isoformat()

    writer.writerow([msg['subject'], msg['from'], msg['reply-to'], msg['x-forefront-antispam-report'], msg['list-owner'],
    msg['dkim-signature'], msg['received'], msg['return-receipt-to'], msg['disposition-notification-to'], iso_date, body])

    # update progress bar on every 10th message for speed
    if i % 10==0:
    pct_complete = int (round (i/float (n) *100.0))
    print_progress (pct_complete)
####END MBOX2CSV CODE
```

(B) Visual Basic code embedded in Microsoft Excel sheets:

```
####BEGIN REGEXEXTRACT VISUAL BASIC CODE
Function RegexExtract (ByVal text As String, _
        ByVal extract_what As String, _
        Optional separator As String = "," ") As String

Dim allMatches As Object
Dim RE As Object
Set RE = CreateObject("vbscript.regexp")
Dim i As Long, j As Long
Dim result As String
RE.Pattern = extract_what
RE.Global = True
Set allMatches = RE.Execute (text)

For i = 0 To allMatches.Count - 1
    For j = 0 To allMatches.Item (i).submatches.Count - 1
    result = result and (separator and allMatches.Item (i).submatches.Item (j))
    Next
Next

'If Len (result) <>0 Then
'    result = 1
'End If

If Len (result) <>0 Then
    result = Right$(result, Len (result) - Len (separator))
End If

RegexExtract = result

End Function
####END REGEXEXTRACT CODE
```

Additional code was needed to automatically indicate whether case-insensitive keywords such as “Conference” or “conference” were present in emails to streamline, in an unbiased manner, the analysis of a large number of emails. The above code was modified and renamed “RegexYN” (for yes/no) to search for a regular expression and return a numeric “1” or “0” according to whether that expression was found in the message body.

To illustrate the example above, searching for “conference” where the first letter is case invariant and the message body is in cell K3 of an Excel sheet called “Data” would be done as follows:

```
=RegexYN (Data!K3,”(.*)(C | c) onference(.*)”)

####BEGIN REGEXYN VISUAL BASIC CODE

Function RegexYN (ByVal text As String, _
        ByVal extract_what As String, _
        Optional separator As String = “, “) As String

Dim allMatches As Object

Dim RE As Object

Set RE = CreateObject(“vbscript.regexp”)

Dim i As Long, j As Long

Dim result As String

RE.Pattern = extract_what

RE.Global = True

Set allMatches = RE.Execute (text)

For i = 0 To allMatches.Count - 1
    For j = 0 To allMatches.Item (i).submatches.Count - 1
        result = result and (separator and allMatches.Item (i).submatches.Item (j))
    Next
Next

If Len (result) <>0 Then
    result = 1
End If

‘If Len (result) <>0 Then
‘    result = Right$(result, Len (result) - Len (separator))
‘End If

RegexYN = result

End Function

####END REGEXYN CODE
```