APT Group Sends Spear Phishing Emails to Indian Government Officials

June 03, 2016 | by Yin Hong Chang, Sudeep Singh | Targeted Attack

Introduction
On May 18, 2016, FireEye Labs observed a suspected Pakistan-based APT group sending spear phishing emails to Indian government officials. This threat actor has been active for several years and conducting suspected intelligence collection operations against South Asian political and military targets.

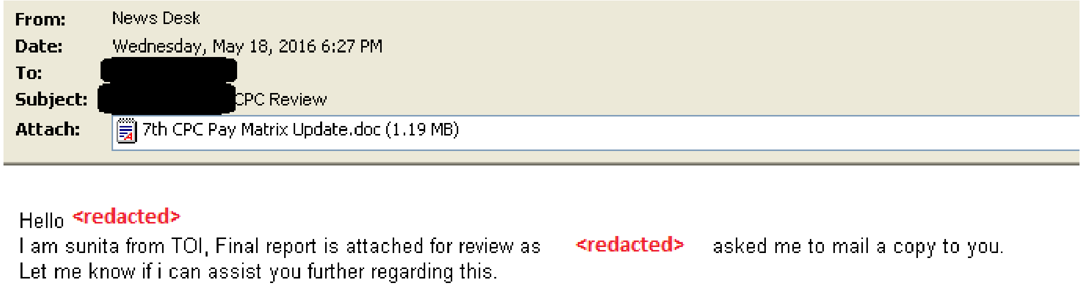
This group frequently uses a toolset that consists of a downloader and modular framework that uses plugins to enhance functionality, ranging from keystroke logging to targeting USB devices. We initially reported on this threat group and their UPDATESEE malware in our FireEye Intelligence Center in February 2016. Proofpoint also discussed the threat actors, whom they call Transparent Tribe, in a March blog post.

In this latest incident, the group registered a fake news domain, timesofindiaa[.]in, on May 18, 2016, and then used it to send spear phishing emails to Indian government officials on the same day. The emails referenced the Indian Governments 7th Central Pay Commission (CPC). These Commissions periodically review the pay structure for Indian government and military personnel, a topic that would be of interest to government employees.

Malware Delivery Method
In all emails sent to these government officials, the actor used the same attachment: a malicious Microsoft Word document that exploited the CVE-2012-0158 vulnerability to drop a malicious payload.

In previous incidents involving this threat actor, we observed them using malicious documents hosted on websites about the Indian Army, instead of sending these documents directly as an email attachment.

The email (Figure 1) pretends to be from an employee working at Times of India (TOI) and requests the recipient to open the attachment associated with the 7th Pay Commission. Only one of the recipient email addresses was publicly listed on a website, suggesting that the actor harvested the other non-public addressees through other means.



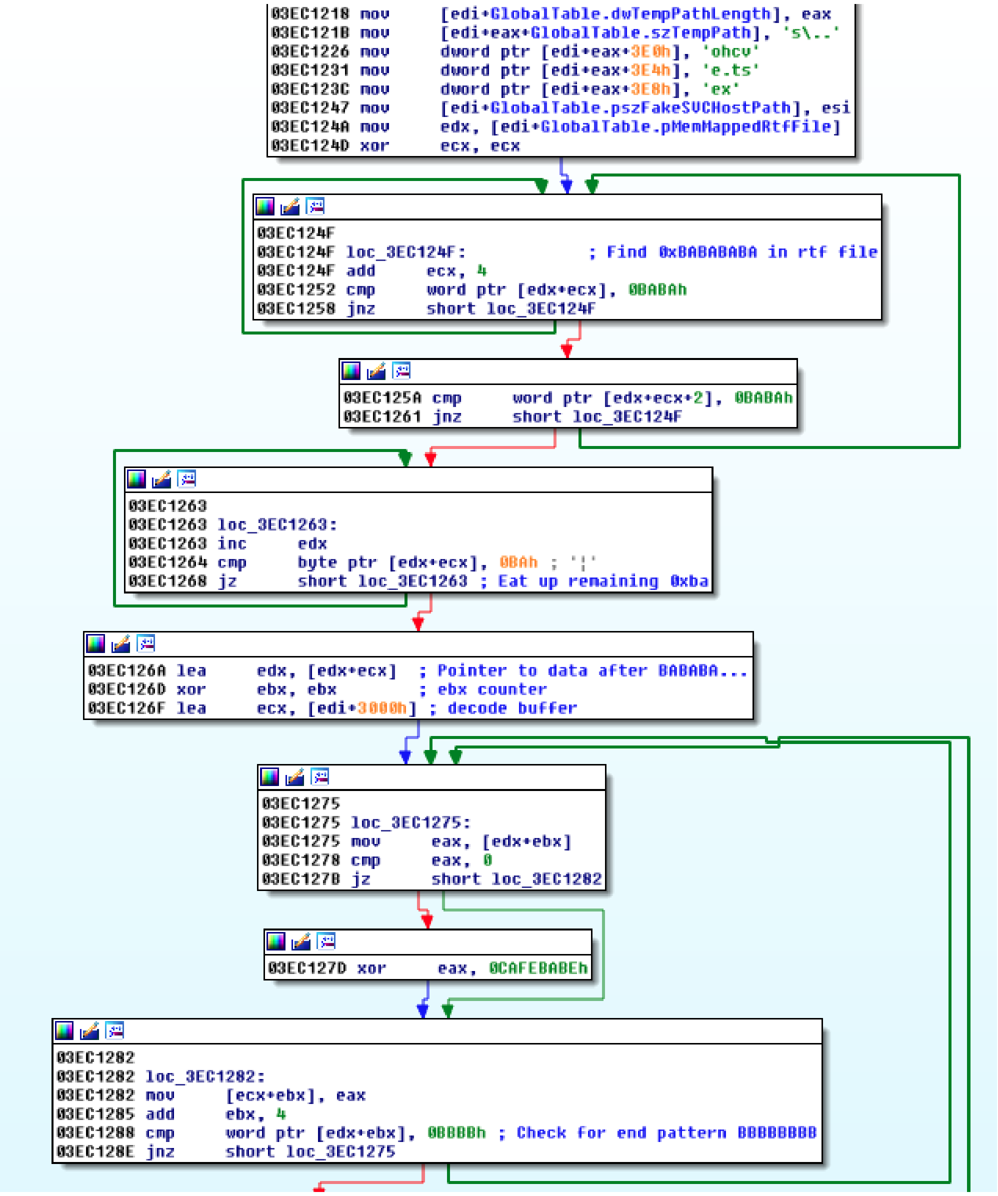
From: News Desk
Date: Wednesday, May 18, 2016 6:27 PM
To: [redacted]
Subject: [redacted] CPC Review
Attach: 7th CPC Pay Matrix Update.doc (1.19 MB)

Hello <redacted>
I am sunita from TOI, Final report is attached for review as <redacted> asked me to mail a copy to you.
Let me know if i can assist you further regarding this.

Figure 1: Contents of the Email

A review of the email header data from the spear phishing messages showed that the threat actors sent the emails using the same infrastructure they have used in the past.

Exploit Analysis
Despite being an older vulnerability, many threat actors continue to leverage CVE-2012-0158 to exploit Microsoft Word. This exploit file made use of the same shellcode that we have observed this actor use across a number of spear phishing incidents.



```
03EC1218 mov     [edi+GlobalTable.dwTempPathLength], eax
03EC121B mov     [edi+eax+GlobalTable.szTempPath], 's\..'
03EC1226 mov     dword ptr [edi+eax+3E0h], 'ohcv'
03EC1231 mov     dword ptr [edi+eax+3E4h], 'e.ts'
03EC123C mov     dword ptr [edi+eax+3E8h], 'ex'
03EC1247 mov     [edi+GlobalTable.pszFakeSVCHostPath], esi
03EC124A mov     edx, [edi+GlobalTable.pMemMappedRtfFile]
03EC124D xor     ecx, ecx
```

```
03EC124F
03EC124F loc_3EC124F:              ; Find 0xBABABABA in rtf file
03EC124F add      ecx, 4
03EC1252 cmp      word ptr [edx+ecx], 0BABAh
03EC1258 jnz      short loc_3EC124F
```

```
03EC125A cmp      word ptr [edx+ecx+2], 0BABAh
03EC1261 jnz      short loc_3EC124F
```

```
03EC1263
03EC1263 loc_3EC1263:
03EC1263 inc      edx
03EC1264 cmp      byte ptr [edx+ecx], 0BAh ; '¦'
03EC1268 jz       short loc_3EC1263 ; Eat up remaining 0xba
```

```
03EC126A lea      edx, [edx+ecx]   ; Pointer to data after BABABA...
03EC126D xor      ebx, ebx         ; ebx counter
03EC126F lea      ecx, [edi+3000h] ; decode buffer
```

```
03EC1275
03EC1275 loc_3EC1275:
03EC1275 mov      eax, [edx+ebx]
03EC1278 cmp      eax, 0
03EC127B jz       short loc_3EC1282
```

```
03EC127D xor      eax, 0CAFEBABEh
```

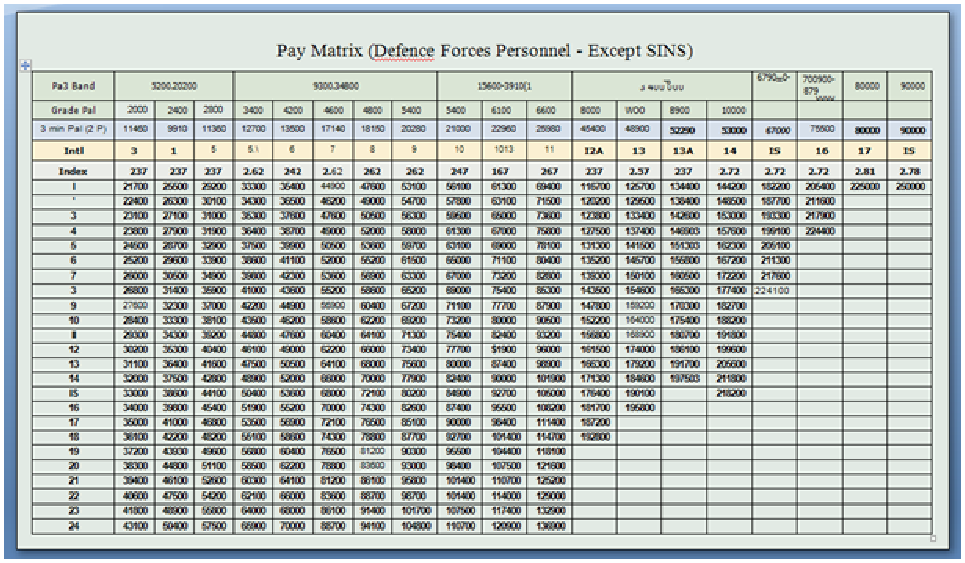
```
03EC1282
03EC1282 loc_3EC1282:
03EC1282 mov      [ecx+ebx], eax
03EC1285 add      ebx, 4
03EC1288 cmp      word ptr [edx+ebx], 0BBBBh ; Check for end pattern BBBBBBBB
03EC128E jnz      short loc_3EC1275
```

Figure 2: Exploit Shellcode used to Locate and Decode Payload

The shellcode (Figure 2) searches for and decodes the executable payload contained in memory between the beginning and ending file markers 0xBABABABA and 0xBBBBBBBB, respectively. After decoding is complete, the shellcode proceeds to save the executable payload into %temp%\svchost.exe and calls WinExec to execute the payload. After the payload is launched, the shellcode runs the following commands to prevent Microsoft Word from showing a recovery dialog:

```
cmd.exe /c reg delete
"HKCU\Software\Microsoft\Office\14.0\Word\Resiliency" /F
cmd.exe /c reg delete
"HKCU\Software\Microsoft\Office\12.0\Word\Resiliency" /F
```

Lastly, the shellcode overwrites the malicious file with a decoy document related to the Indian defense forces pay scale / matrix (Figure 3), displays it to the user and terminates the exploited instance of Microsoft Word.

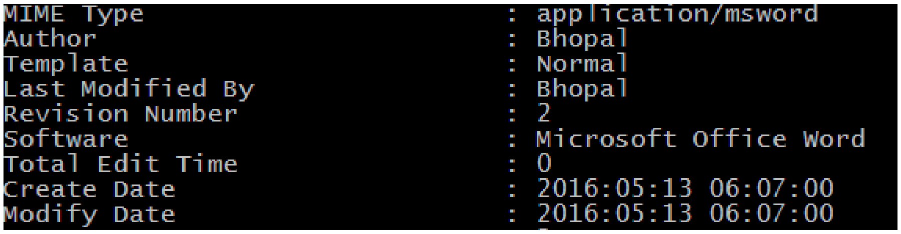


Pay Matrix (Defence Forces Personnel - Except SINS)

Pa3 Band	5200.20200			9300.34800					15600-3910(1			[illegible]				6790₌0-	700900-879	80000	90000
Grade Pal	2000	2400	2800	3400	4200	4600	4800	5400	5400	6100	6600	8000	WOO	8900	10000				
3 min Pal (2 P)	11460	9910	11360	12700	13500	17140	18150	20280	21000	22960	25980	45400	48900	52290	53000	67000	75500	80000	90000
Intl	3	1	5	5.\	6	7	8	9	10	1013	11	12A	13	13A	14	IS	16	17	IS
Index	237	237	237	2.62	242	2.62	262	262	247	167	267	237	2.57	237	2.72	2.72	2.72	2.81	2.78
I	21700	25500	29200	33300	35400	44900	47600	53100	56100	61300	69400	116700	125700	134400	144200	182200	205400	225000	250000
'	22400	26300	30100	34300	36500	46200	49000	54700	57800	63100	71500	120200	129500	138400	148500	187700	211600		
3	23100	27100	31000	35300	37600	47600	50500	56300	59600	65000	73600	123800	133400	142600	153000	193300	217900		
4	23800	27900	31900	36400	38700	49000	52000	58000	61300	67000	75800	127500	137400	146903	157600	199100	224400		
5	24500	28700	32900	37500	39900	50500	53600	59700	63100	69000	78100	131300	141500	151303	162300	205100			
6	25200	29600	33900	38600	41100	52000	55200	61500	65000	71100	80400	135200	145700	155800	167200	211300			
7	26000	30500	34900	39800	42300	53600	56900	63300	67000	73200	82800	139300	150100	160500	172200	217600			
3	26800	31400	35900	41000	43600	55200	58600	65200	69000	75400	85300	143500	154600	165300	177400	224100			
9	27600	32300	37000	42200	44900	56900	60400	67200	71100	77700	87900	147800	159200	170300	182700				
10	28400	33300	38100	43500	46200	58600	62200	69200	73200	80000	90500	152200	164000	175400	188200				
ll	29300	34300	39200	44800	47600	60400	64100	71300	75400	82400	93200	156800	168900	180700	191800				
12	30200	35300	40400	46100	49000	62200	66000	73400	77700	$1900	96000	161500	174000	186100	199600				
13	31100	36400	41600	47500	50500	64100	68000	75600	80000	87400	98900	166300	179200	191700	205600				
14	32000	37500	42800	48900	52000	66000	70000	77900	82400	90000	101900	171300	184600	197503	211800				
IS	33000	38600	44100	50400	53600	68000	72100	80200	84900	92700	105000	176400	190100		218200				
16	34000	39800	45400	51900	55200	70000	74300	82600	87400	95500	108200	181700	195800						
17	35000	41000	46800	53500	56900	72100	76500	85100	90000	98400	111400	187200							
18	36100	42200	48200	55100	58600	74300	78800	87700	92700	101400	114700	192800							
19	37200	43300	49600	56800	60400	76500	81200	90300	95500	104400	118100								
20	38300	44800	51100	58500	62200	78800	83600	93000	98400	107500	121600								
21	39400	46100	52600	60300	64100	81200	86100	95800	101400	110700	125200								
22	40600	47500	54200	62100	66000	83600	88700	98700	101400	114000	129000								
23	41800	48900	55800	64000	68000	86100	91400	101700	107500	117400	132900								
24	43100	50400	57500	65900	70000	88700	94100	104800	110700	120900	136800								

Figure 3: Decoy Document related to 7th Pay Commission

The decoy document's metadata (Figure 4) suggests that it was created fairly recently by the user Bhopal.



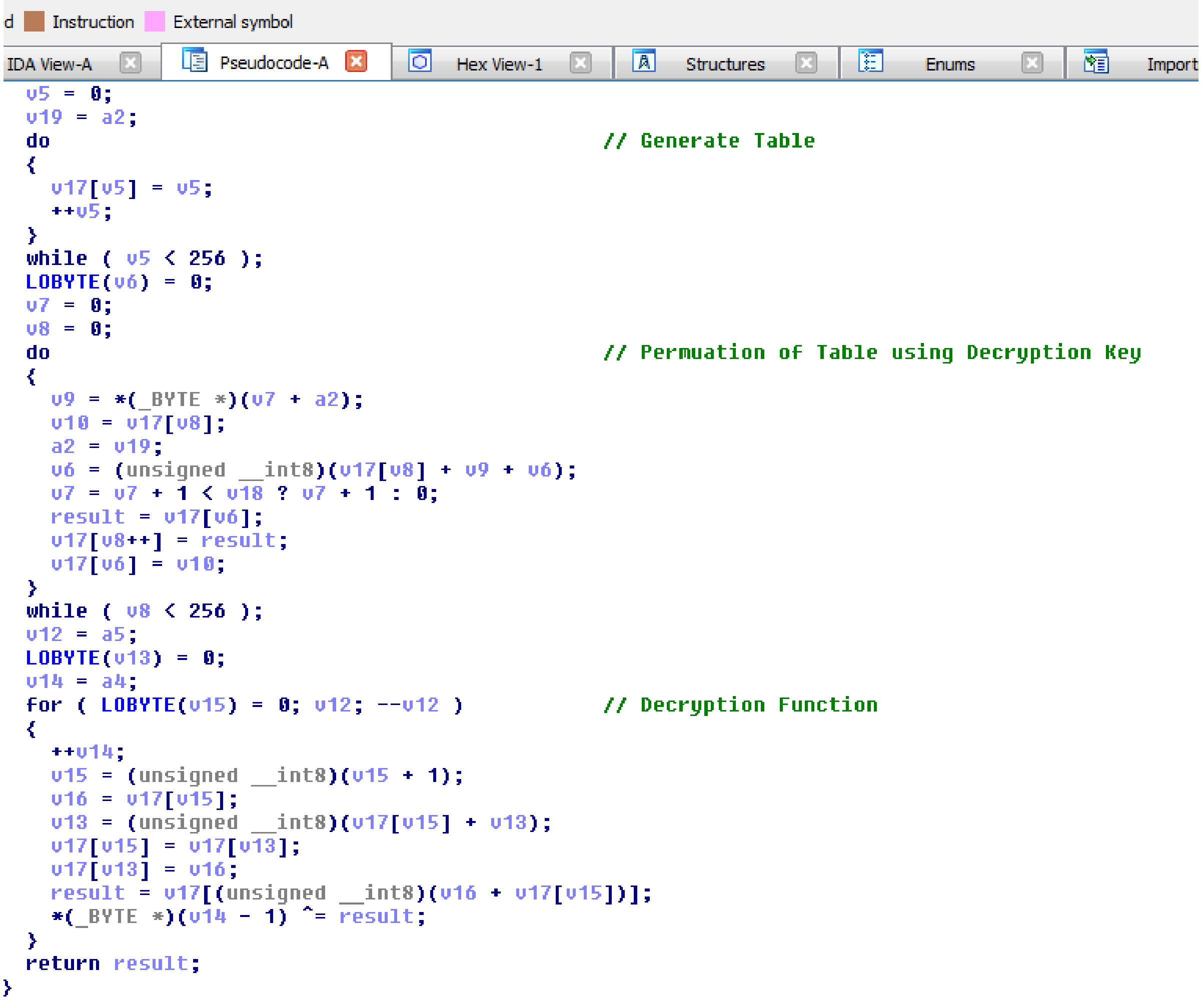
```
MIME Type                       : application/msword
Author                          : Bhopal
Template                        : Normal
Last Modified By                : Bhopal
Revision Number                 : 2
Software                        : Microsoft Office Word
Total Edit Time                 : 0
Create Date                     : 2016:05:13 06:07:00
Modify Date                     : 2016:05:13 06:07:00
```

Figure 4: Metadata of the Document

The payload is a backdoor that we call the Breach Remote Administration Tool (BreachRAT) written in C++. We had not previously observed this payload used by these threat actors. The malware name is derived from the hardcoded PDB path found in the RAT: C:\Work\Breach Remote Administration Tool\Release\Client.pdb. This RAT communicates with 5.189.145.248, a command and control (C2) IP address that this group has used previously with other malware, including DarkComet and NJRAT.

The following is a brief summary of the activities performed by the dropped payload:

1. Decrypts resource 1337 using a hard-coded 14-byte key "MjEh92jHaZZOl3". The encryption/decryption routine (refer to Figure 5) can be summarized as follows:



```
  v5 = 0;
  v19 = a2;
  do                                            // Generate Table
  {
    v17[v5] = v5;
    ++v5;
  }
  while ( v5 < 256 );
  LOBYTE(v6) = 0;
  v7 = 0;
  v8 = 0;
  do                                            // Permuation of Table using Decryption Key
  {
    v9 = *(_BYTE *)(v7 + a2);
    v10 = v17[v8];
    a2 = v19;
    v6 = (unsigned __int8)(v17[v8] + v9 + v6);
    v7 = v7 + 1 < v18 ? v7 + 1 : 0;
    result = v17[v6];
    v17[v8++] = result;
    v17[v6] = v10;
  }
  while ( v8 < 256 );
  v12 = a5;
  LOBYTE(v13) = 0;
  v14 = a4;
  for ( LOBYTE(v15) = 0; v12; --v12 )          // Decryption Function
  {
    ++v14;
    v15 = (unsigned __int8)(v15 + 1);
    v16 = v17[v15];
    v13 = (unsigned __int8)(v17[v15] + v13);
    v17[v15] = v17[v13];
    v17[v13] = v16;
    result = v17[(unsigned __int8)(v16 + v17[v15])];
    *(_BYTE *)(v14 - 1) ^= result;
  }
  return result;
}
```

Figure 5: Encryption/ Decryption Function

- Generate an array of integers from 0x00 to 0xff
- Scrambles the state of the table using the given key
- Encrypts or decrypts a string using the scrambled table from (b).
- A python script, which can be used for decrypting this resource, is provided in the appendix below.

2. The decrypted resource contains the C2 servers IP address as well as the mutex name.

3. If the mutex does not exist and a Windows Startup Registry key with name System Update does not exist, the malware performs its initialization routine by:

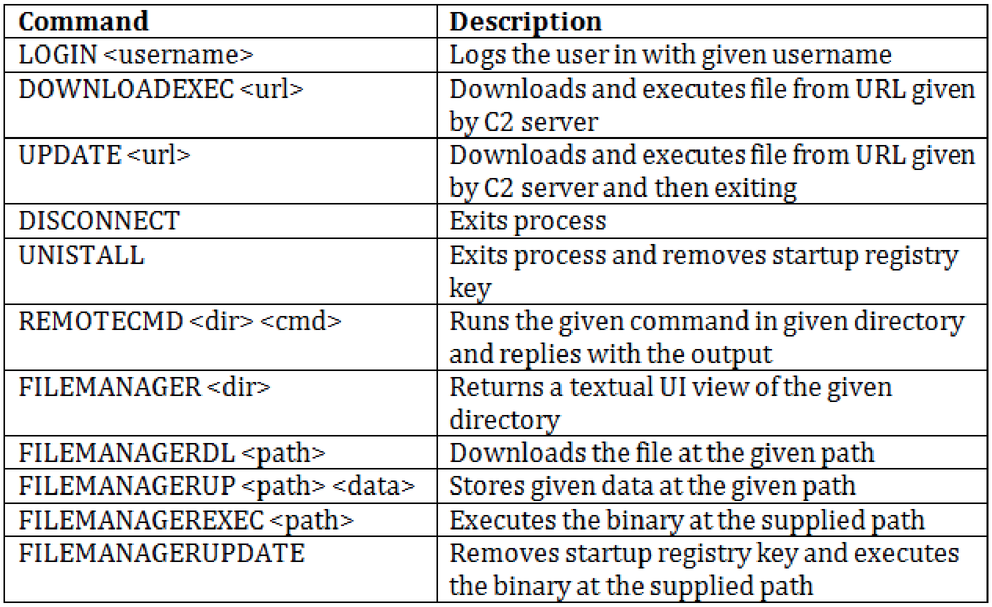
- Copying itself to the path %PROGRAMDATA%\svchost.exe
- Sets the Windows Startup Registry key with the name System Update which points to the above dropped payload.

4. The malware proceeds to connect to the C2 server at 5.189.145.248 at regular intervals through the use of TCP over port 10500. Once a successful connection is made, the malware tries to fetch a response from the server through its custom protocol.

5. Once data is received, the malware skips over the received bytes until the start byte 0x99 is found in the server response. The start byte is followed by a DWORD representing the size of the following data string.

6. The data string is encrypted with the above-mentioned encryption scheme with the hard-coded key AjN28AcMaNX.

7. The data string can contain various commands sent by the C2 server. These commands and their string arguments are expected to be in Unicode. The following commands are accepted by the malware:



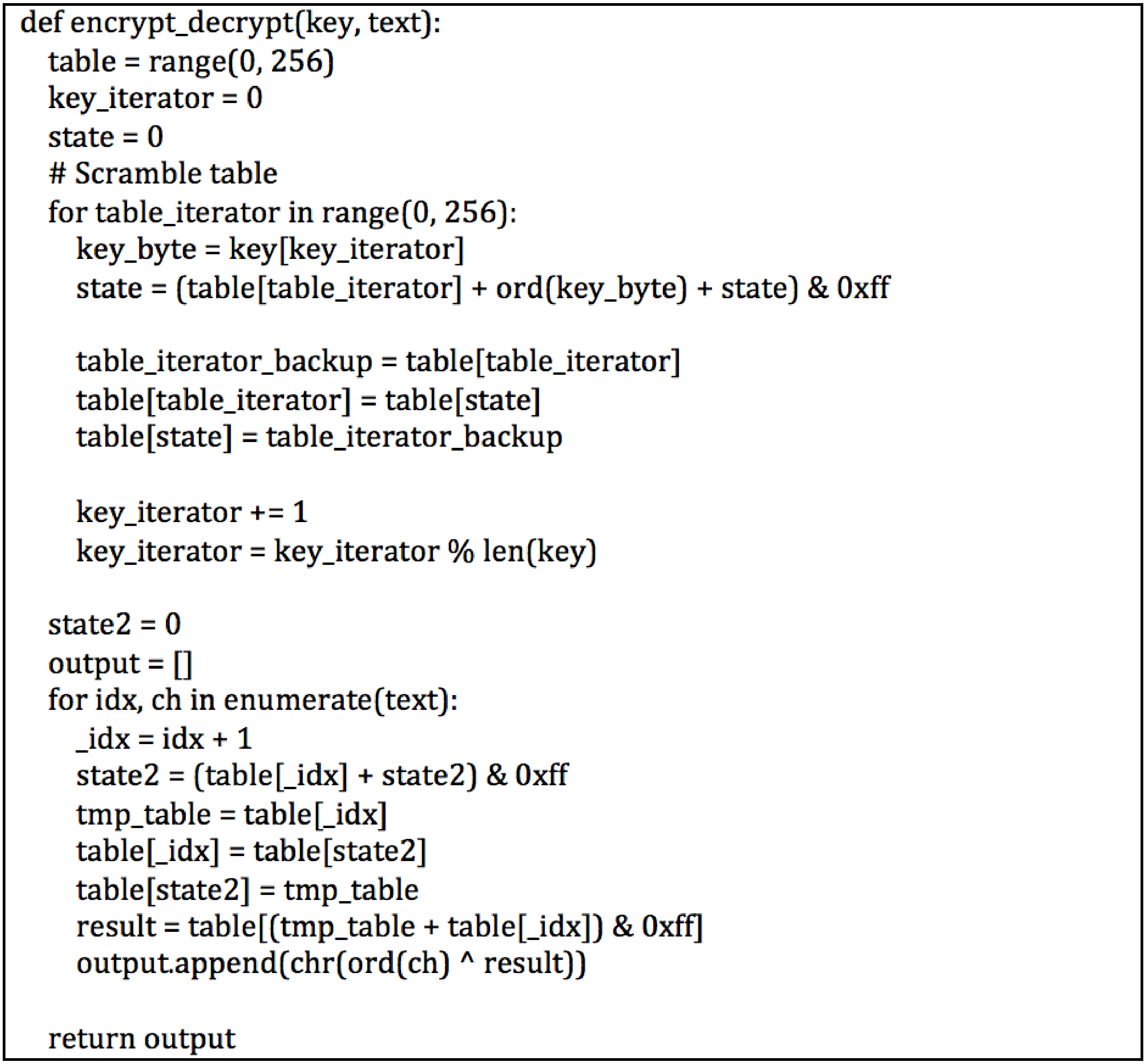
Command	Description
LOGIN <username>	Logs the user in with given username
DOWNLOADEXEC <url>	Downloads and executes file from URL given by C2 server
UPDATE <url>	Downloads and executes file from URL given by C2 server and then exiting
DISCONNECT	Exits process
UNISTALL	Exits process and removes startup registry key
REMOTECMD <dir> <cmd>	Runs the given command in given directory and replies with the output
FILEMANAGER <dir>	Returns a textual UI view of the given directory
FILEMANAGERDL <path>	Downloads the file at the given path
FILEMANAGERUP <path> <data>	Stores given data at the given path
FILEMANAGEREXEC <path>	Executes the binary at the supplied path
FILEMANAGERUPDATE	Removes startup registry key and executes the binary at the supplied path

Conclusion

As with previous spear-phishing attacks seen conducted by this group, topics related to Indian Government and Military Affairs are still being used as the lure theme in these attacks and we observed that this group is still actively expanding their toolkit. It comes as no surprise that cyber attacks against the Indian government continue, given the historically tense relations in the region.

Appendix

Encryption / Decryption algorithm translated into Python



```
def encrypt_decrypt(key, text):
    table = range(0, 256)
    key_iterator = 0
    state = 0
    # Scramble table
    for table_iterator in range(0, 256):
        key_byte = key[key_iterator]
        state = (table[table_iterator] + ord(key_byte) + state) & 0xff

        table_iterator_backup = table[table_iterator]
        table[table_iterator] = table[state]
        table[state] = table_iterator_backup

        key_iterator += 1
        key_iterator = key_iterator % len(key)

    state2 = 0
    output = []
    for idx, ch in enumerate(text):
        _idx = idx + 1
        state2 = (table[_idx] + state2) & 0xff
        tmp_table = table[_idx]
        table[_idx] = table[state2]
        table[state2] = tmp_table
        result = table[(tmp_table + table[_idx]) & 0xff]
        output.append(chr(ord(ch) ^ result))

    return output
```

This entry was posted on Fri Jun 03 01:30:00 EDT 2016 and filed under APT, Latest Blog Posts, Spear Phishing, Sudeep Singh, Targeted Attack and Yin Hong Chang.