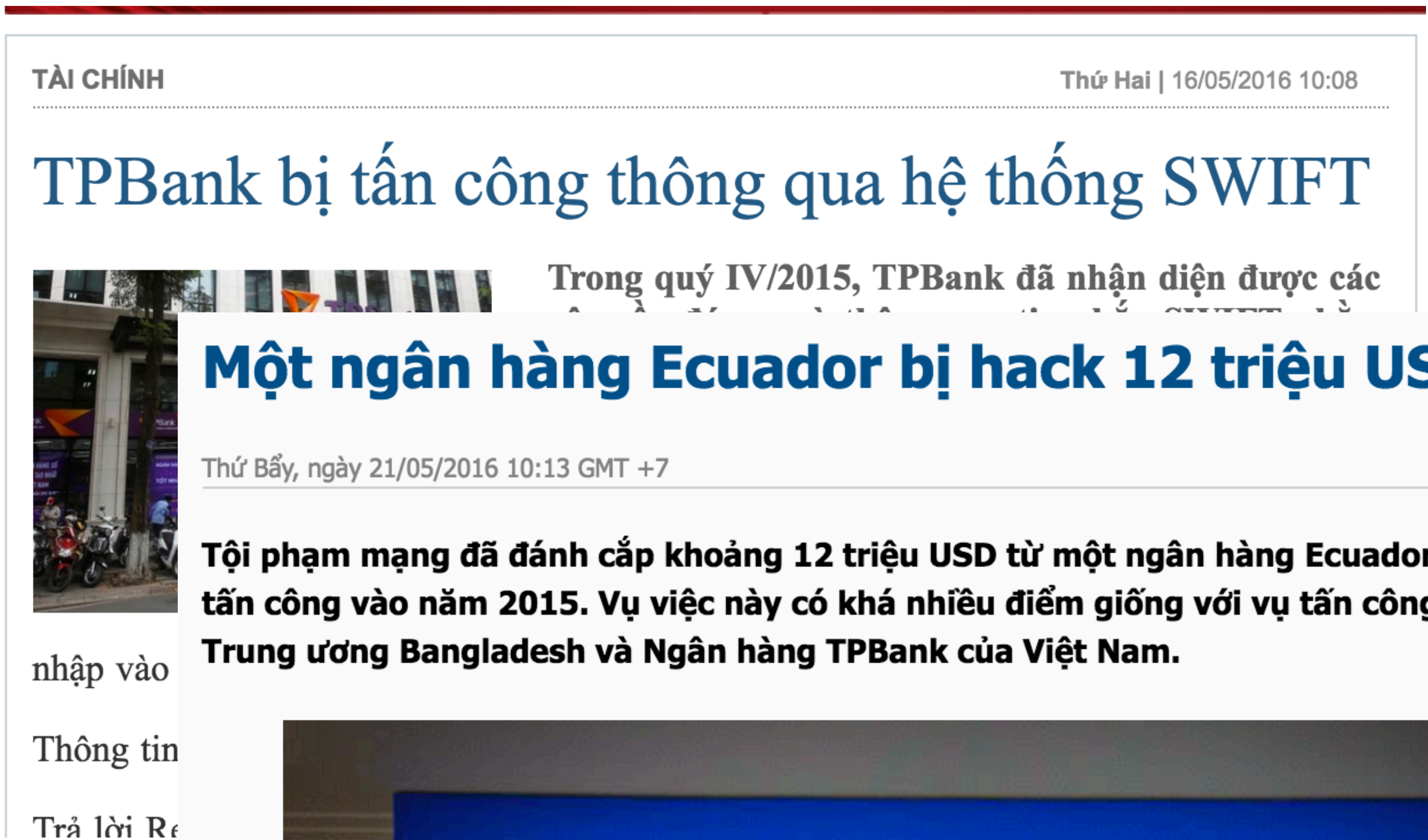
Thảo luận
Bank Swift [Attack]

#tradahacking



TÀI CHÍNH

Thứ Hai | 16/05/2016 10:08

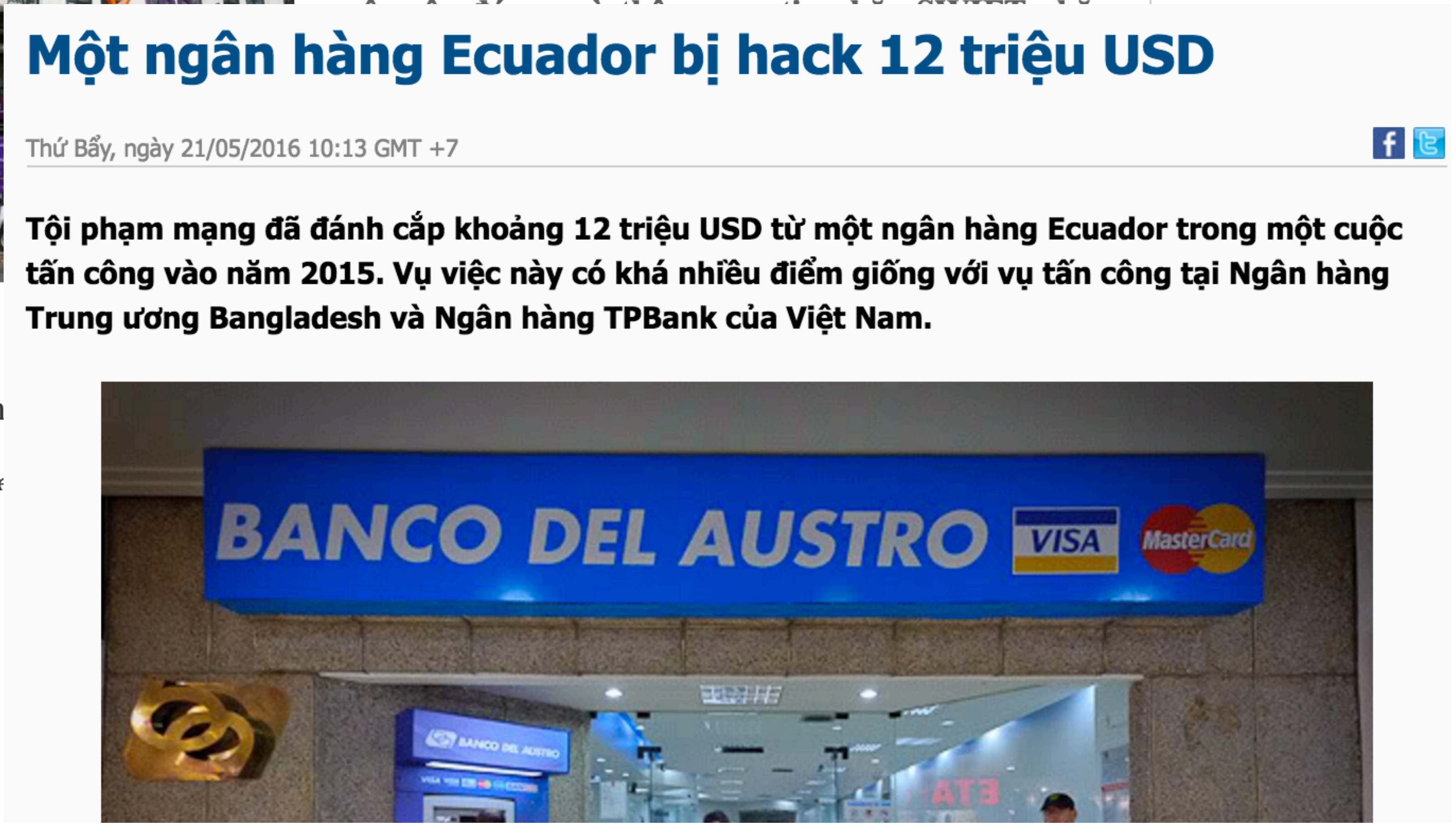
TPBank bị tấn công thông qua hệ thống SWIFT

Trong quý IV/2015, TPBank đã nhận diện được các

nhập vào

Thông tin

Trả lời Re



Một ngân hàng Ecuador bị hack 12 triệu USD

Thứ Bẩy, ngày 21/05/2016 10:13 GMT +7

Tội phạm mạng đã đánh cắp khoảng 12 triệu USD từ một ngân hàng Ecuador trong một cuộc tấn công vào năm 2015. Vụ việc này có khá nhiều điểm giống với vụ tấn công tại Ngân hàng Trung ương Bangladesh và Ngân hàng TPBank của Việt Nam.

www.wsj.com/articles/u-s-attorney-says-people-should-be-horrified-about-bangladesh-bank-h...

Subscribe | Sign In

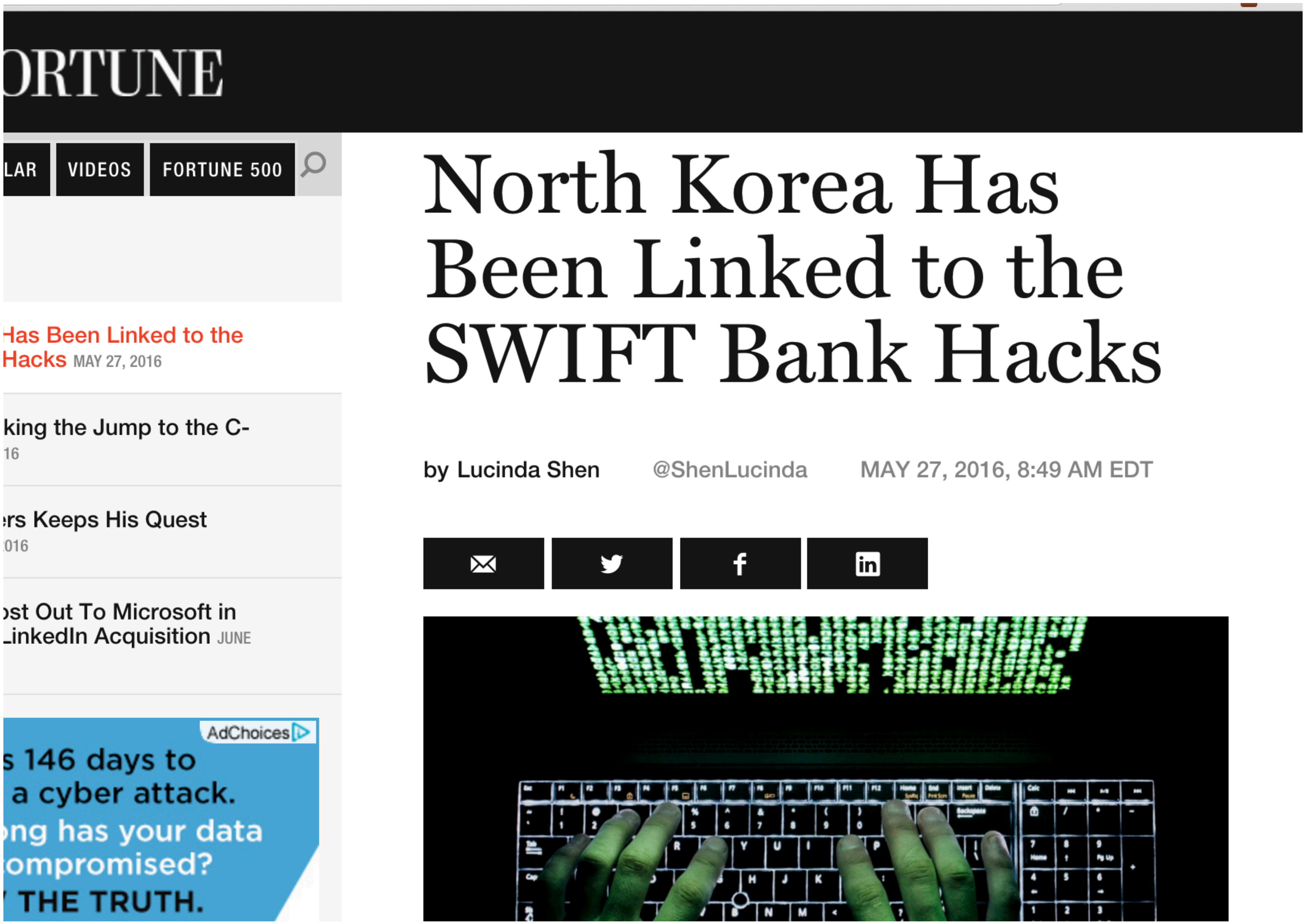
U.S. Attorney Says ‘People Should Be Horrified’ About Bangladesh Bank Hack

Preet Bharara’s office and FBI are investigating the suspected theft of nearly $1 billion by computer hackers

https://www.theguardian.com/technology/2016/may/27/swift-network-bank-theft-sony-pictures-ha...

Swift network bank thefts 'linked' to Sony Pictures hack

Unique code signatures shared between malware used in multiple bank attacks suggest involvement of hacking group named 'Lazarus', Symantec says



ORTUNE

LAR VIDEOS FORTUNE 500

Has Been Linked to the Hacks MAY 27, 2016

king the Jump to the C-
16

rs Keeps His Quest
016

ost Out To Microsoft in
LinkedIn Acquisition JUNE

AdChoices

s 146 days to
a cyber attack.
ng has your data
ompromised?
THE TRUTH.

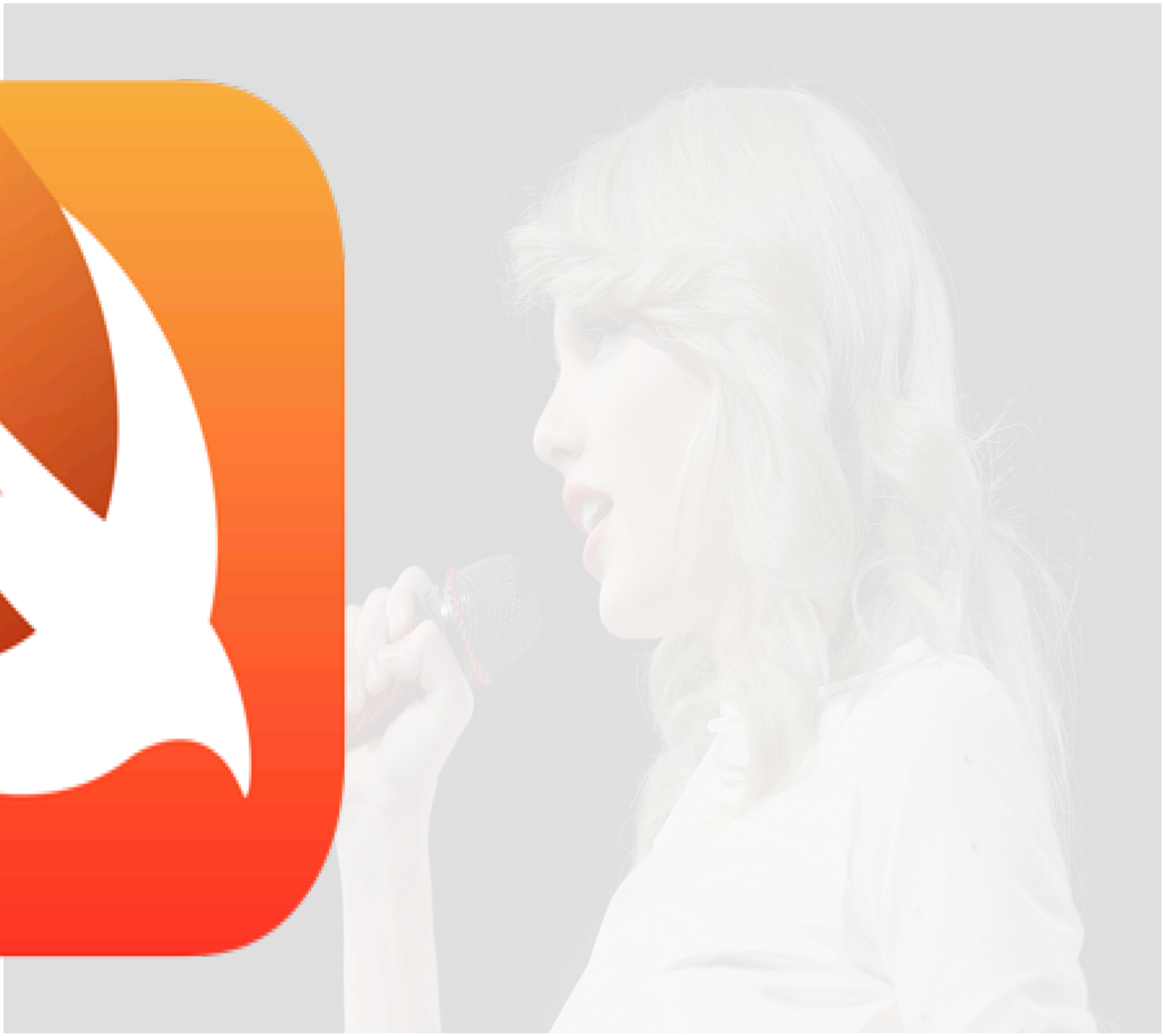
North Korea Has Been Linked to the SWIFT Bank Hacks

by Lucinda Shen @ShenLucinda MAY 27, 2016, 8:49 AM EDT

Thảo luận
Bank Swift [Attack]

#tradahacking

[1] SWIFT?



Society for Worldwide Interbank Financial Telecommunication

SWIFT

[1.2] BIC

- Business Identifier Codes (BICs, previously Bank Identifier Codes) ~ "SWIFT codes".
- ~ IP Addresses.
- DongA Bank: EACBVNVX
- Eximbank: EBVIVNVX, EBVIVN2X, EBVIVNVXDNG...
- Google: GOOGIE21, GOOGIE31, GOOGUS66...

[1.3] SWIFT Message Types

* MT103 - Cash Transfer

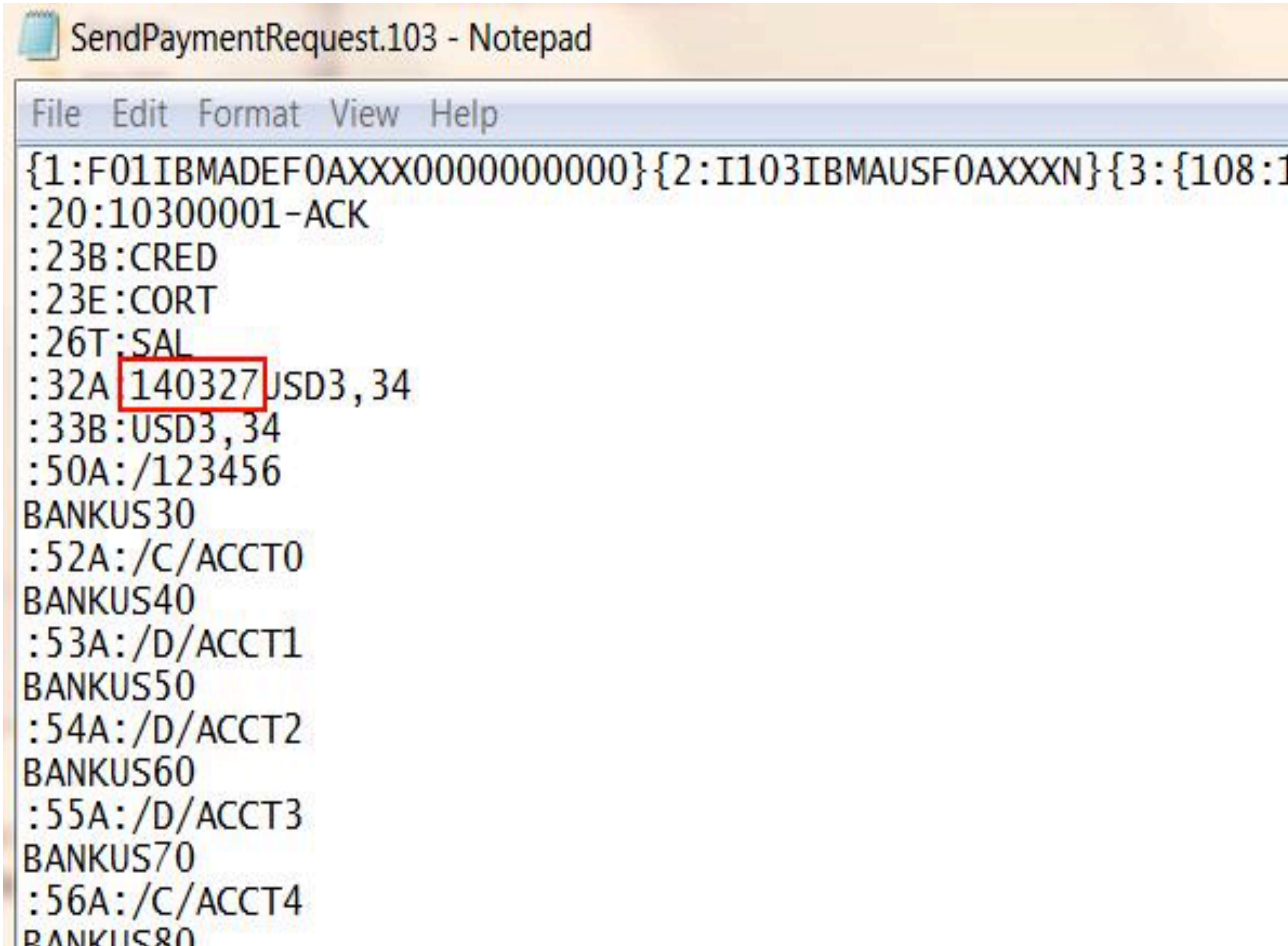
* :32A Value Date / Currency / Inter bank Settled

* :50A, F or K Ordering Customer (Payer)

* :59 or 59A Beneficiary

- en.wikipedia.org/wiki/MT103
- en.wikipedia.org/wiki/SWIFT_message_types

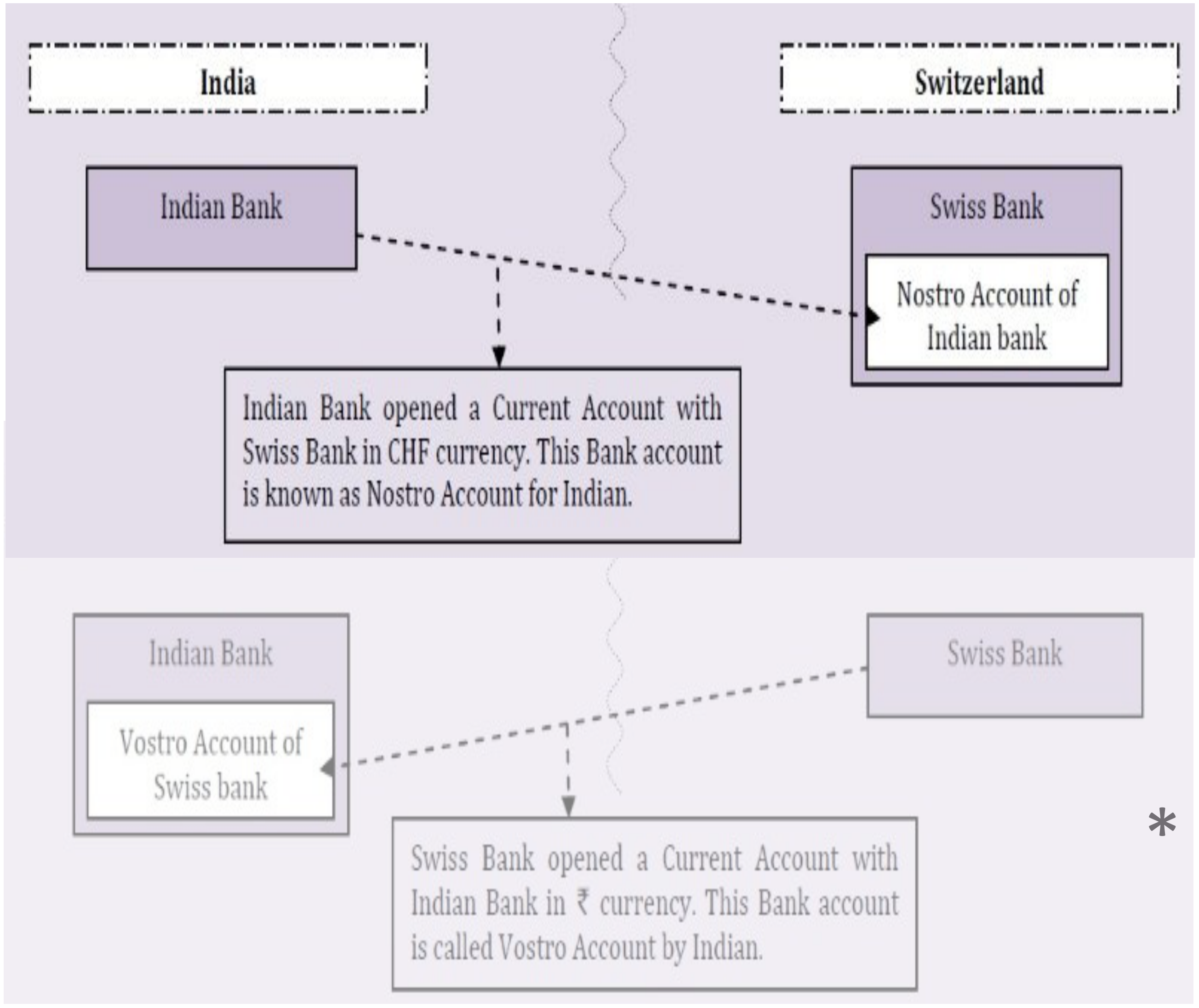
** Source: IBM, Wikipedia.*



```
{1:F01IBMADEF0AXXX0000000000}{2:I103IBMAUSF0AXXXN}{3:{108:1
:20:10300001-ACK
:23B:CRED
:23E:CORT
:26T:SAL
:32A:140327USD3,34
:33B:USD3,34
:50A:/123456
BANKUS30
:52A:/C/ACCT0
BANKUS40
:53A:/D/ACCT1
BANKUS50
:54A:/D/ACCT2
BANKUS60
:55A:/D/ACCT3
BANKUS70
:56A:/C/ACCT4
BANKUS80
```

[2] How SWIFT?

[2.1] Nostro accounts

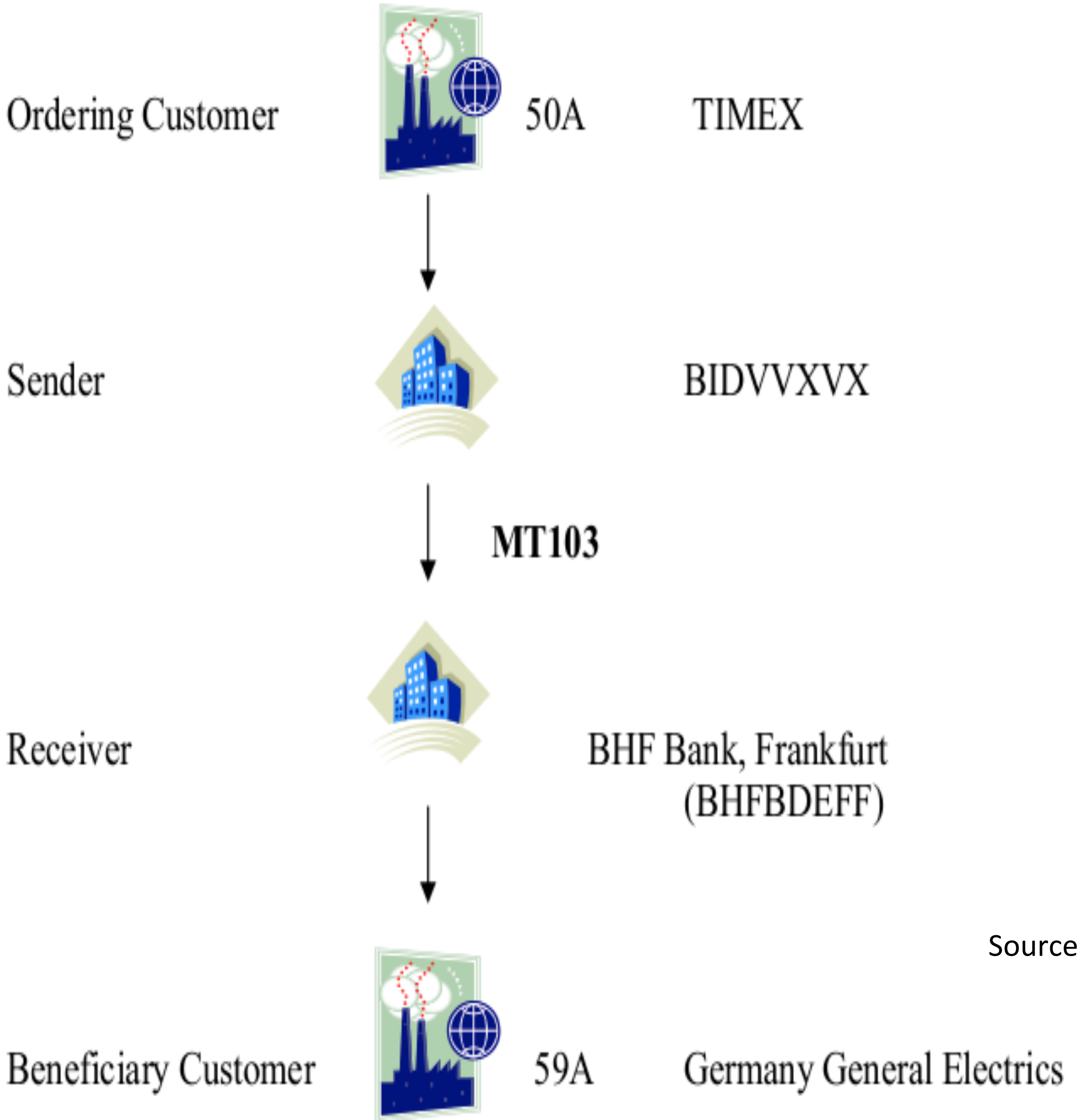

India
Switzerland
Indian Bank
Swiss Bank
Nostro Account of Indian bank
Indian Bank opened a Current Account with Swiss Bank in CHF currency. This Bank account is known as Nostro Account for Indian.
Indian Bank
Vostro Account of Swiss bank
Swiss Bank
Swiss Bank opened a Current Account with Indian Bank in ₹ currency. This Bank account is called Vostro Account by Indian.

* Nostro = ours

* Source: CA BHAVIN SHAH (Linkedin Pulse)

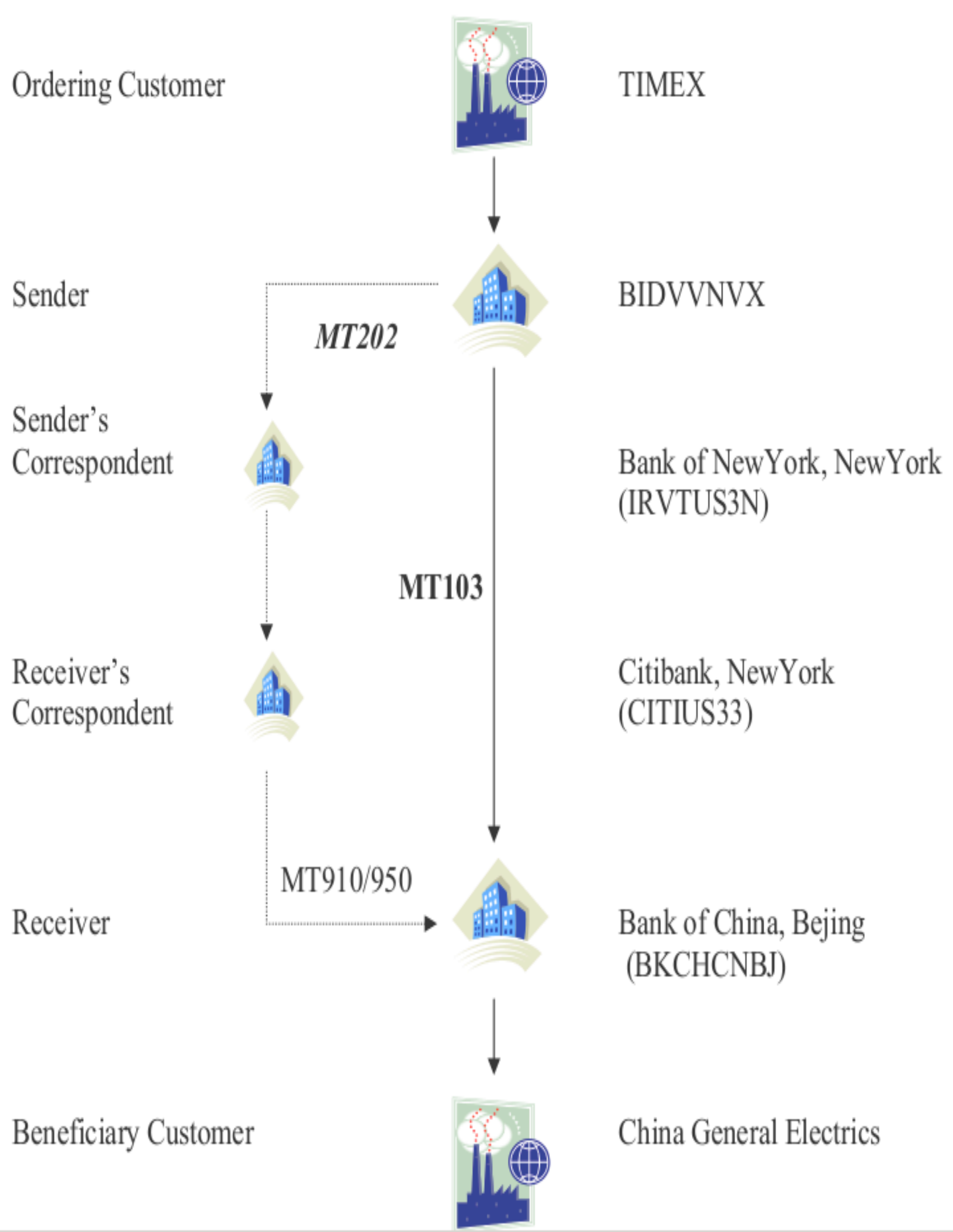
[2.2] How SWIFT Messaging?

Sơ đồ 3.1: Sơ đồ thanh toán sử dụng phương pháp trực tiếp



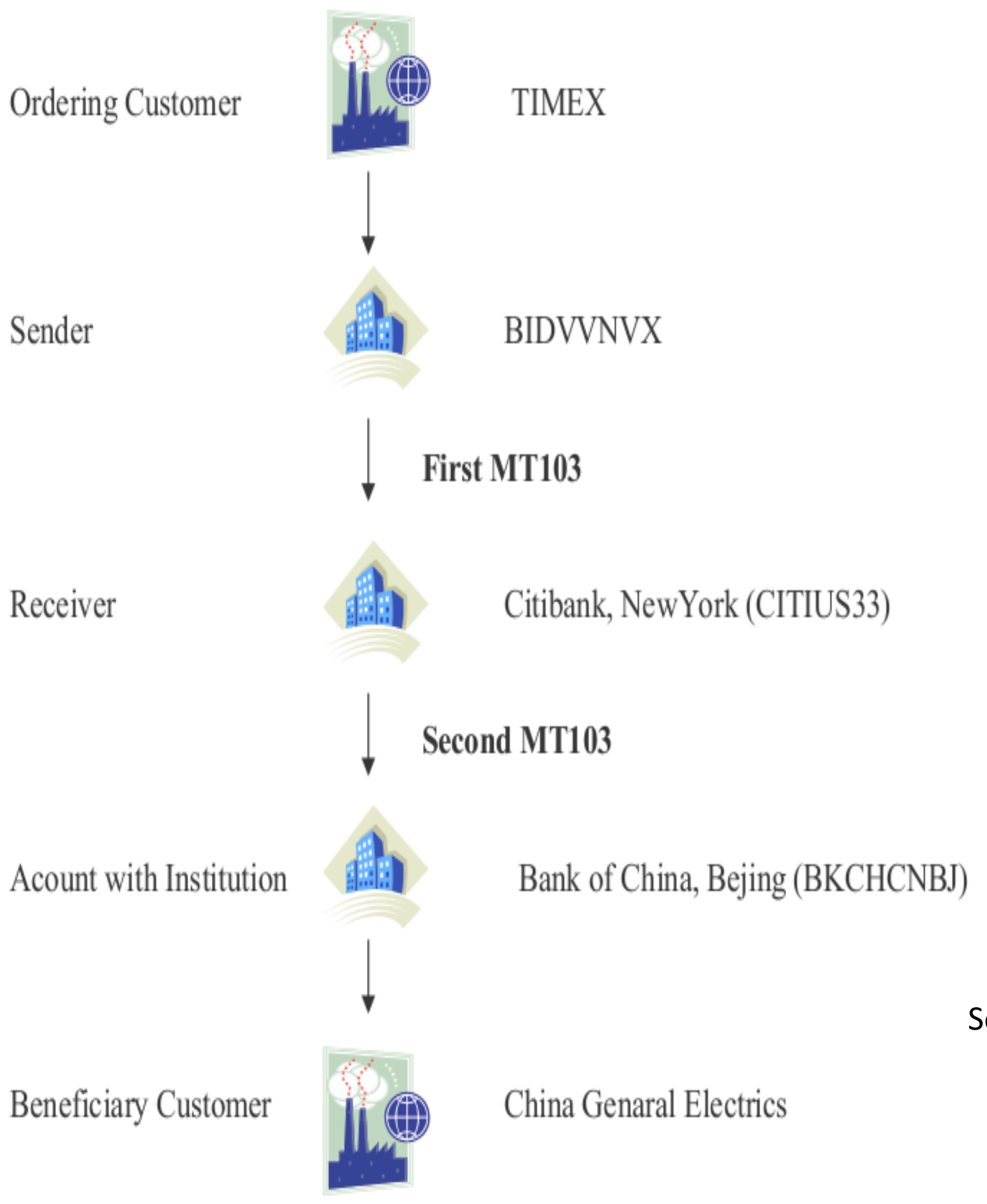
Source: HUI.edu.vn

Sơ đồ 3.2: Sơ đồ thanh toán sử dụng phương pháp gián tiếp



Source: HUI.edu.vn

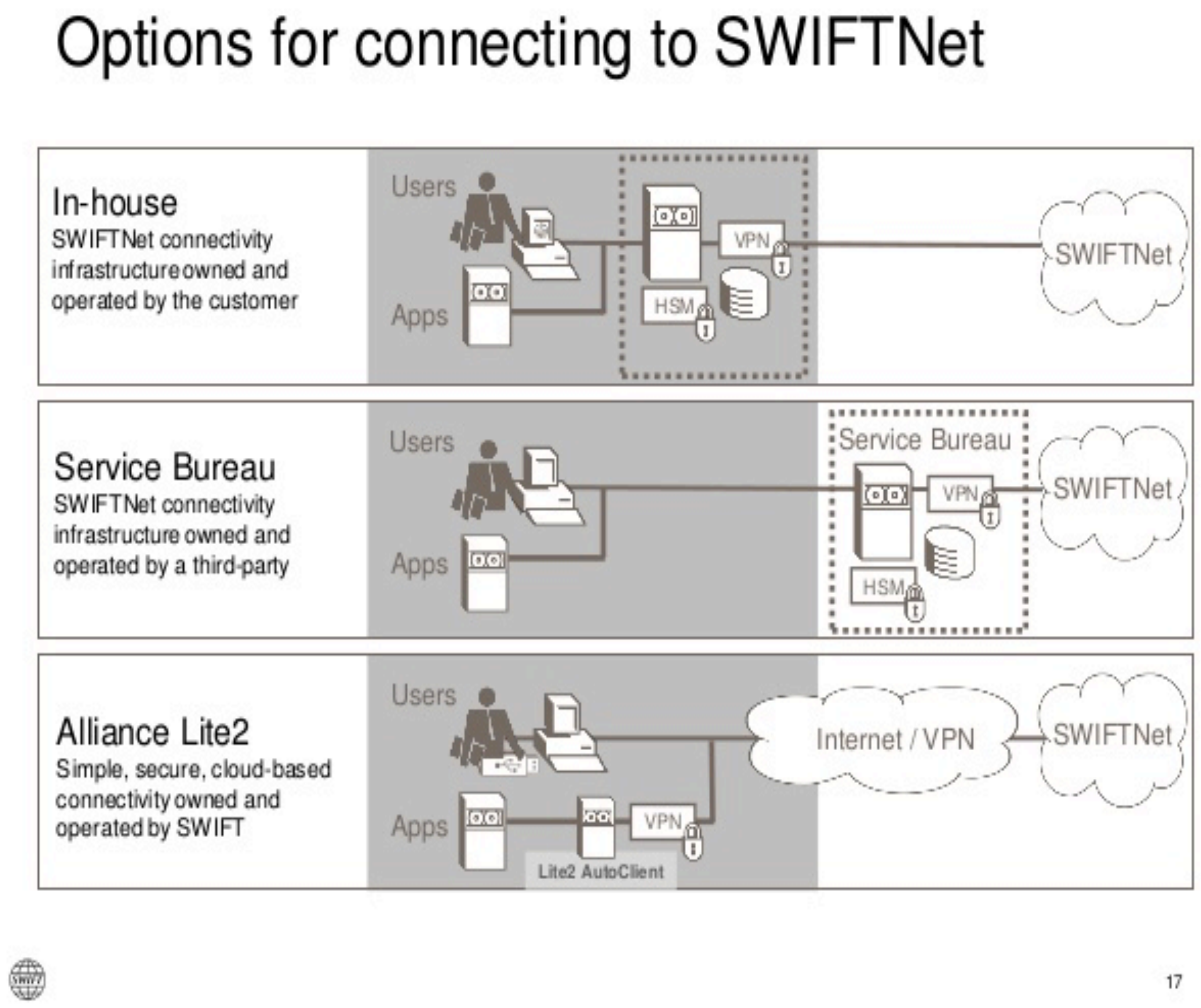
Sơ đồ 3.3: Sơ đồ thanh toán sử dụng phương pháp chuỗi



Source: HUI.edu.vn

[3] SWIFT Systems

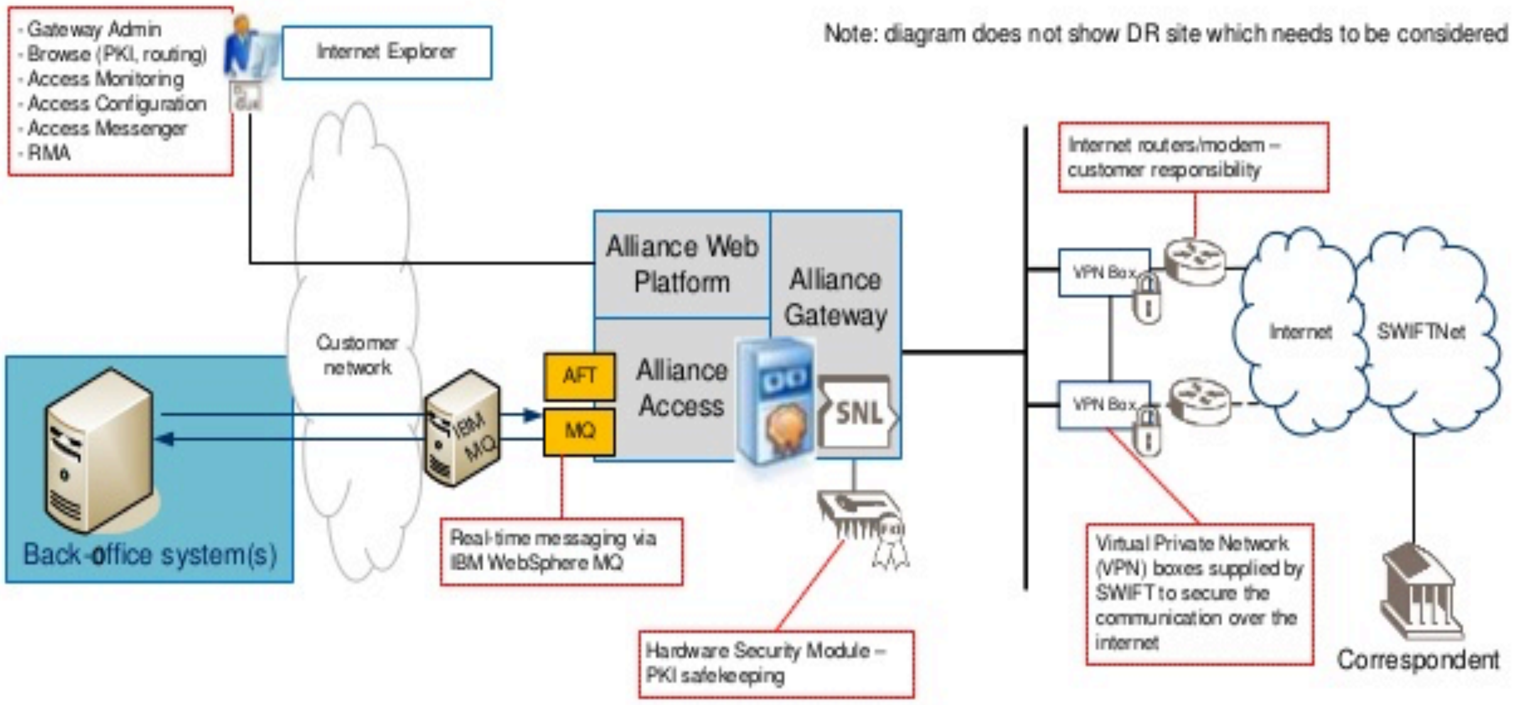
[3.1] Connect to SWIFT // SWIFT


Options for connecting to SWIFTNet
In-house
SWIFTNet connectivity infrastructure owned and operated by the customer
Users
Apps
VPN
HSM
SWIFTNet
Service Bureau
SWIFTNet connectivity infrastructure owned and operated by a third-party
Users
Apps
Service Bureau
VPN
HSM
SWIFTNet
Alliance Lite2
Simple, secure, cloud-based connectivity owned and operated by SWIFT
Users
Apps
Internet / VPN
SWIFTNet
VPN
Lite2 AutoClient
17

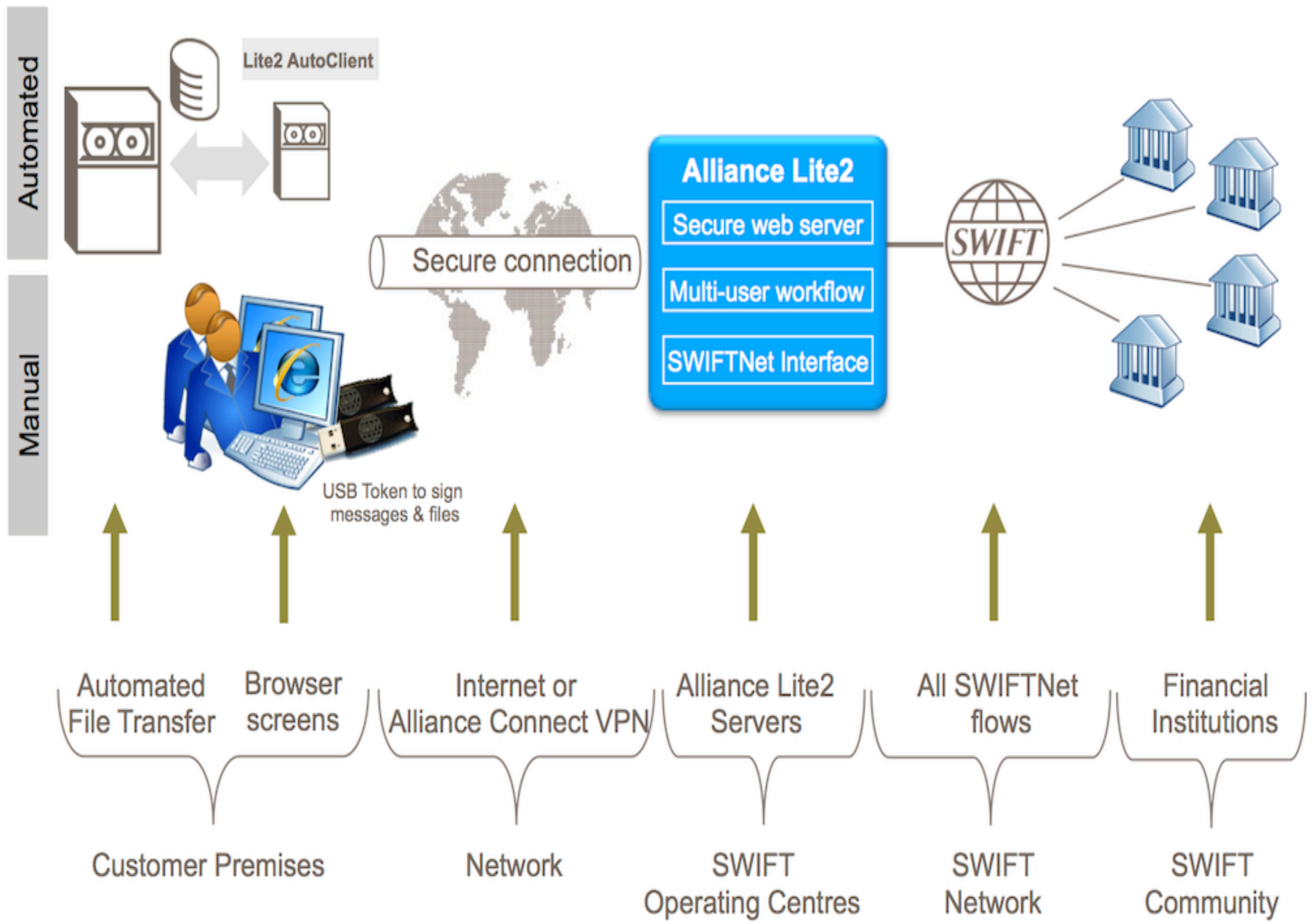
Service Bureau // Banks (via Internet Banking)

Source: SWIFT

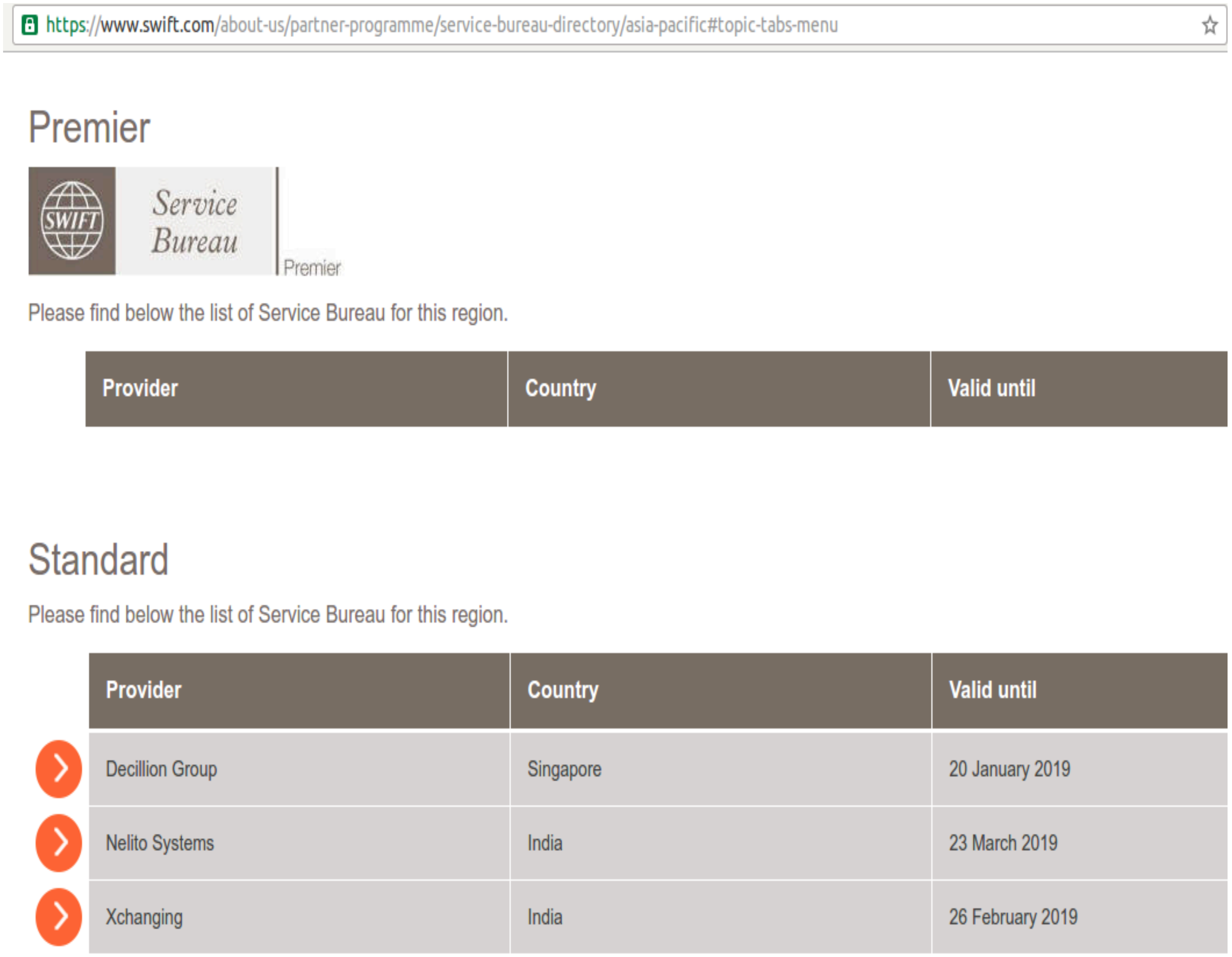
Private Infrastructure


- Gateway Admin
- Browse (PKI, routing)
- Access Monitoring
- Access Configuration
- Access Messenger
- RMA
Internet Explorer
Note: diagram does not show DR site which needs to be considered
Internet routers/modem – customer responsibility
Alliance Web Platform
Alliance Gateway
Alliance Access
AFT
MQ
IBM MQ
SNL
Customer network
VPN Box
VPN Box
Internet
SWIFTNet
Back-office system(s)
Real-time messaging via IBM WebSphere MQ
Hardware Security Module – PKI safekeeping
Virtual Private Network (VPN) boxes supplied by SWIFT to secure the communication over the internet
Correspondent

Source: SWIFT



Source: SWIFT



https://www.swift.com/about-us/partner-programme/service-bureau-directory/asia-pacific#topic-tabs-menu

Premier

Please find below the list of Service Bureau for this region.

Provider	Country	Valid until

Standard

Please find below the list of Service Bureau for this region.

Provider	Country	Valid until
Decillion Group	Singapore	20 January 2019
Nelito Systems	India	23 March 2019
Xchanging	India	26 February 2019

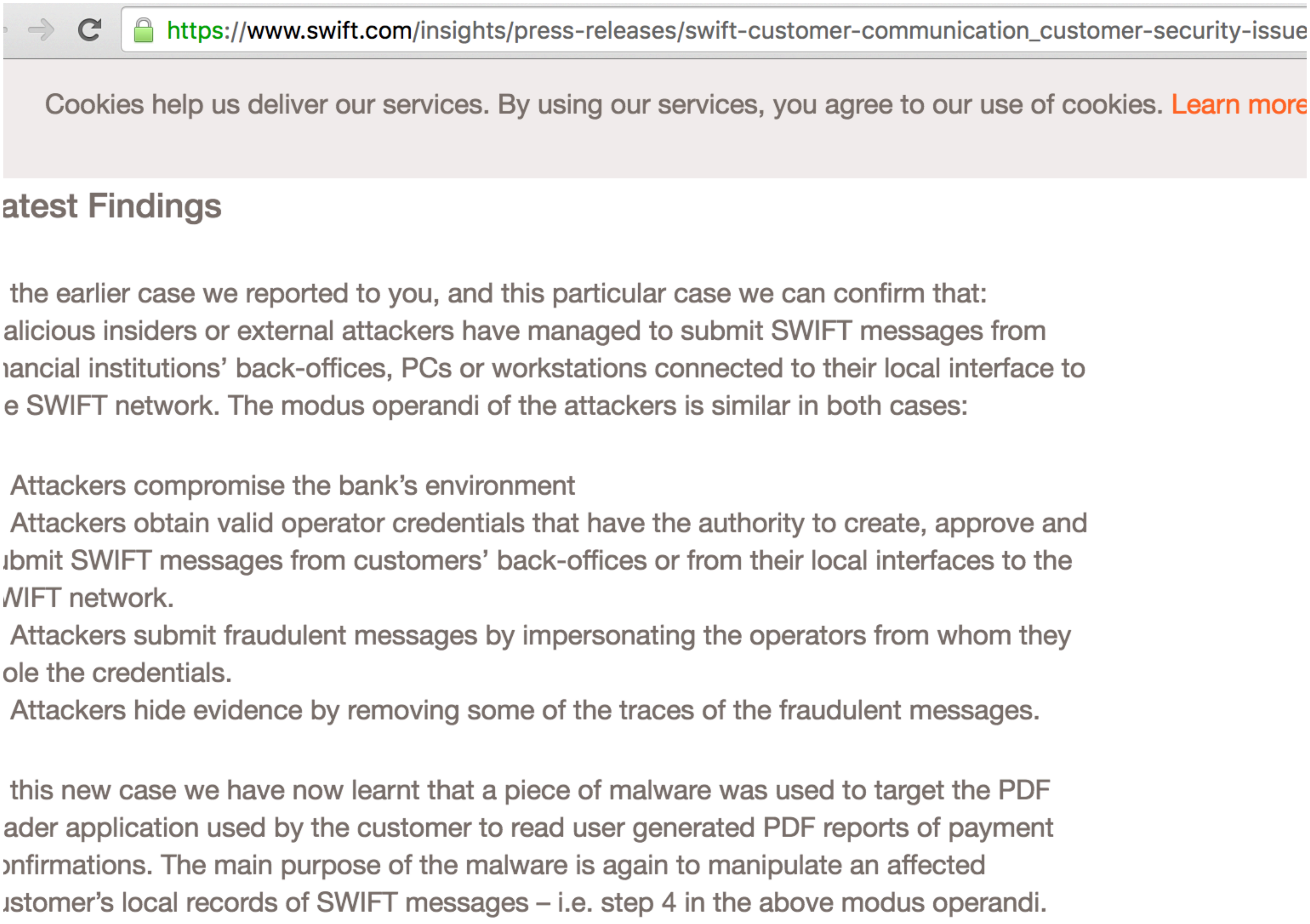
[3.2] SWIFT Alliance Access/Entry Accounts

1) LSO, RSO
2) Create, Verify & Authorize

x) HSM

Thảo luận
Bank Swift [Attack]

#tradahacking


https://www.swift.com/insights/press-releases/swift-customer-communication_customer-security-issue

Cookies help us deliver our services. By using our services, you agree to our use of cookies. Learn more

atest Findings

the earlier case we reported to you, and this particular case we can confirm that:
alicious insiders or external attackers have managed to submit SWIFT messages from
ıancial institutions' back-offices, PCs or workstations connected to their local interface to
e SWIFT network. The modus operandi of the attackers is similar in both cases:

Attackers compromise the bank's environment
Attackers obtain valid operator credentials that have the authority to create, approve and
ıbmit SWIFT messages from customers' back-offices or from their local interfaces to the
WIFT network.
Attackers submit fraudulent messages by impersonating the operators from whom they
ole the credentials.
Attackers hide evidence by removing some of the traces of the fraudulent messages.

this new case we have now learnt that a piece of malware was used to target the PDF
ader application used by the customer to read user generated PDF reports of payment
ɔnfirmations. The main purpose of the malware is again to manipulate an affected
ıstomer's local records of SWIFT messages – i.e. step 4 in the above modus operandi.

SWIFT.COM

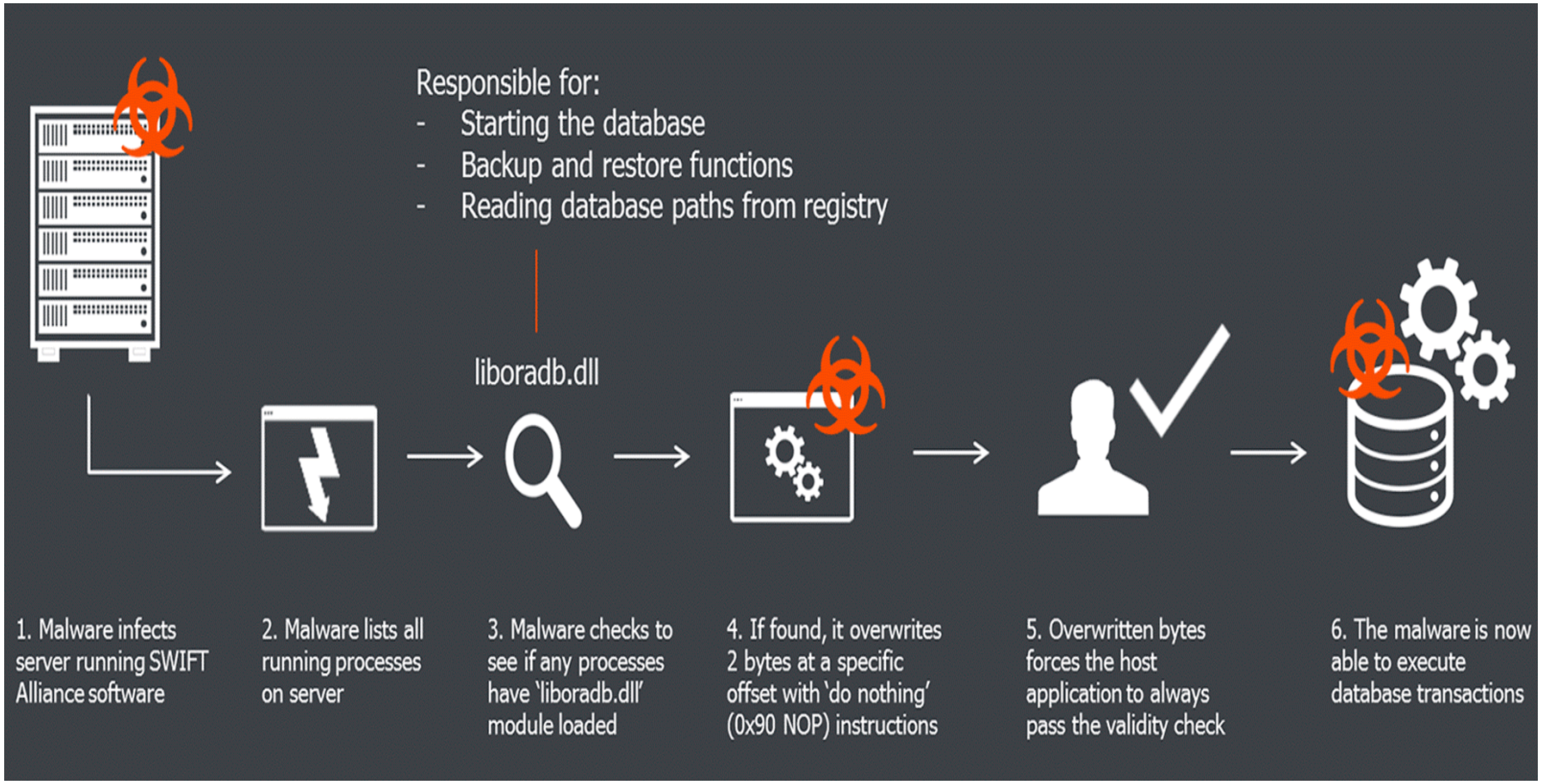
1. Attackers <u>compromise</u> the customer's environment

2. Attackers <u>obtain</u> valid <u>operator credentials</u> that have the authority to create, approve and submit SWIFT messages from customers' back-offices or from their local interfaces to the SWIFT network;

3. Attackers <u>submit</u> fraudulent <u>messages</u> by impersonating the operators from whom they stole the credentials;

4. Attackers <u>hide evidence</u> of the fraud by removing some of the traces of the fraudulent messages.

TWO BYTES TO $951M


Responsible for:
- Starting the database
- Backup and restore functions
- Reading database paths from registry
liboradb.dll
1. Malware infects server running SWIFT Alliance software
2. Malware lists all running processes on server
3. Malware checks to see if any processes have 'liboradb.dll' module loaded
4. If found, it overwrites 2 bytes at a specific offset with 'do nothing' (0x90 NOP) instructions
5. Overwritten bytes forces the host application to always pass the validity check
6. The malware is now able to execute database transactions

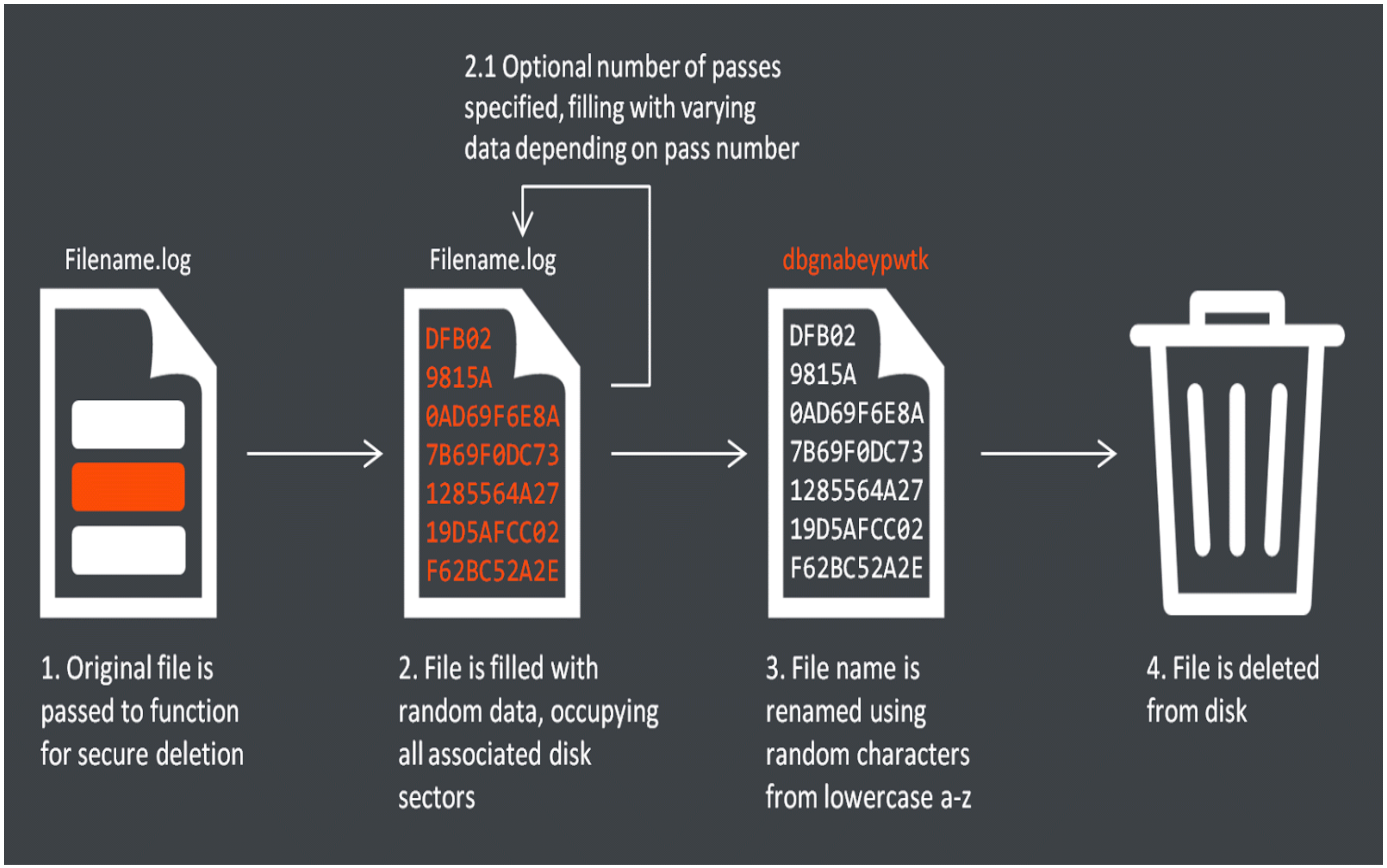
- Before patch

```
85 C0              test eax, eax ; some important check
75 04              jnz  failed   ; if failed, jump to 'failed' label below
33 c0              xor  eax, eax ; otherwise, set result to 0 (success)
eb 17              jmp  exit     ; and then exit
          failed:
B8 01 00 00 00     mov  eax, 1   ; set result to 1 (failure)
```

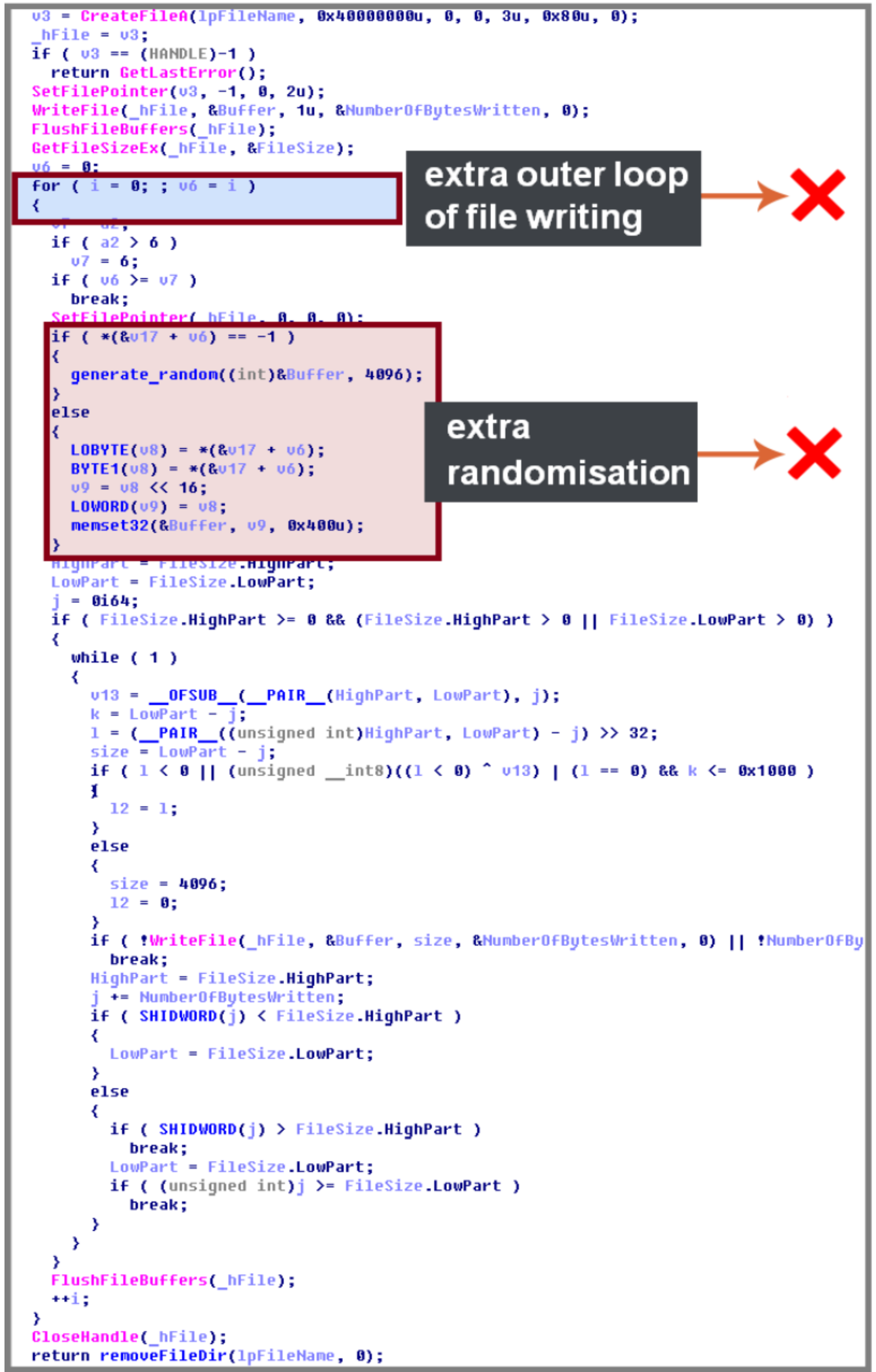
- After patch

```
85 C0            test eax, eax ; some important check
90               nop           ; 'do nothing' in place of 0x75
90               nop           ; 'do nothing' in place of 0x04
33 c0            xor  eax, eax ; always set result to 0 (success)
eb 17            jmp  exit     ; and then exit
          failed:
B8 01 00 00 00   mov  eax, 1   ; never reached: set result to 1 (fail)
```

'wipe-out' and 'file-delete' functions

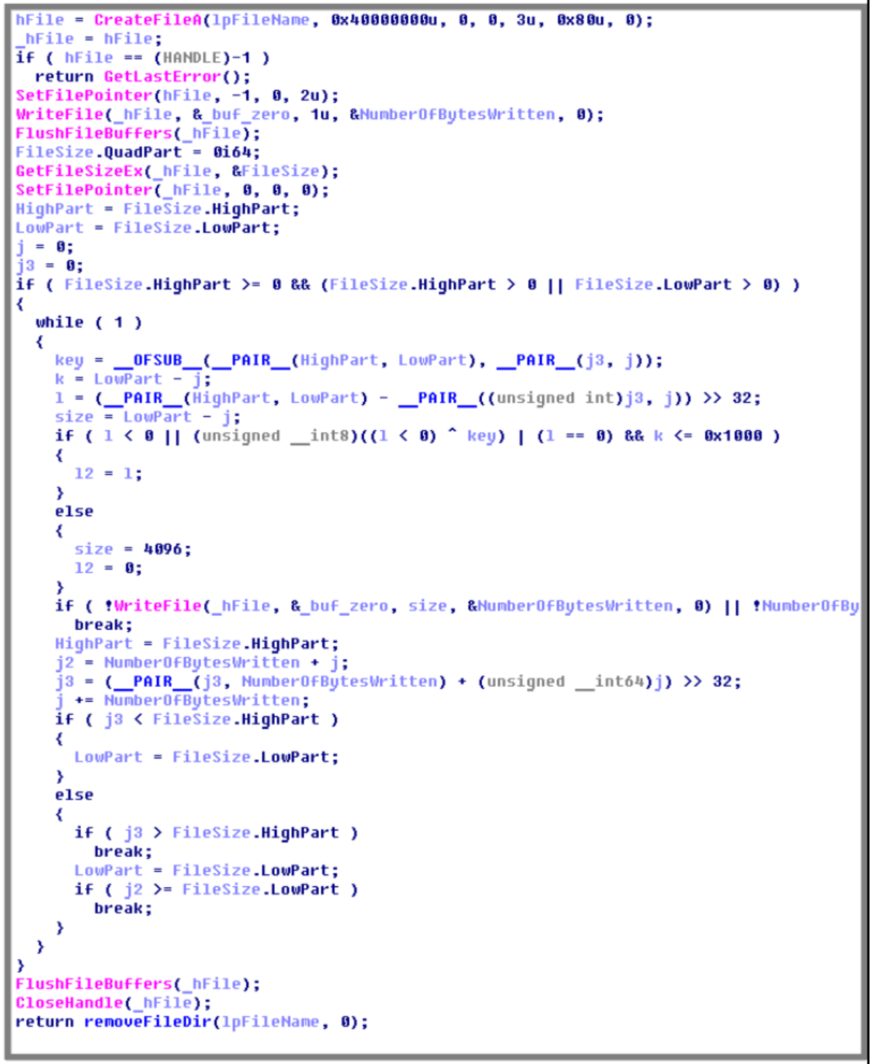

2.1 Optional number of passes specified, filling with varying data depending on pass number
Filename.log
Filename.log
dbgnabeypwtk
DFB02
9815A
0AD69F6E8A
7B69F0DC73
1285564A27
19D5AFCC02
F62BC52A2E
DFB02
9815A
0AD69F6E8A
7B69F0DC73
1285564A27
19D5AFCC02
F62BC52A2E
1. Original file is passed to function for secure deletion
2. File is filled with random data, occupying all associated disk sectors
3. File name is renamed using random characters from lowercase a-z
4. File is deleted from disk

Wipe-out function in the msoutc.exe bot (2014)



```
v3 = CreateFileA(lpFileName, 0x40000000u, 0, 0, 3u, 0x80u, 0);
_hFile = v3;
if ( v3 == (HANDLE)-1 )
  return GetLastError();
SetFilePointer(v3, -1, 0, 2u);
WriteFile(_hFile, &Buffer, 1u, &NumberOfBytesWritten, 0);
FlushFileBuffers(_hFile);
GetFileSizeEx(_hFile, &FileSize);
v6 = 0;
for ( i = 0; ; v6 = i )
{
  v7 = a2;
  if ( a2 > 6 )
    v7 = 6;
  if ( v6 >= v7 )
    break;
  SetFilePointer(_hFile, 0, 0, 0);
  if ( *(&v17 + v6) == -1 )
  {
    generate_random((int)&Buffer, 4096);
  }
  else
  {
    LOBYTE(v8) = *(&v17 + v6);
    BYTE1(v8) = *(&v17 + v6);
    v9 = v8 << 16;
    LOWORD(v9) = v8;
    memset32(&Buffer, v9, 0x400u);
  }
  HighPart = FileSize.HighPart;
  LowPart = FileSize.LowPart;
  j = 0i64;
  if ( FileSize.HighPart >= 0 && (FileSize.HighPart > 0 || FileSize.LowPart > 0) )
  {
    while ( 1 )
    {
      v13 = __OFSUB__(__PAIR__(HighPart, LowPart), j);
      k = LowPart - j;
      l = (__PAIR__((unsigned int)HighPart, LowPart) - j) >> 32;
      size = LowPart - j;
      if ( l < 0 || (unsigned __int8)((l < 0) ^ v13) | (l == 0) && k <= 0x1000 )
      {
        l2 = l;
      }
      else
      {
        size = 4096;
        l2 = 0;
      }
      if ( !WriteFile(_hFile, &Buffer, size, &NumberOfBytesWritten, 0) || !NumberOfBy
        break;
      HighPart = FileSize.HighPart;
      j += NumberOfBytesWritten;
      if ( SHIDWORD(j) < FileSize.HighPart )
      {
        LowPart = FileSize.LowPart;
      }
      else
      {
        if ( SHIDWORD(j) > FileSize.HighPart )
          break;
        LowPart = FileSize.LowPart;
        if ( (unsigned int)j >= FileSize.LowPart )
          break;
      }
    }
  }
  FlushFileBuffers(_hFile);
  ++i;
}
CloseHandle(_hFile);
return removeFileDir(lpFileName, 0);
```

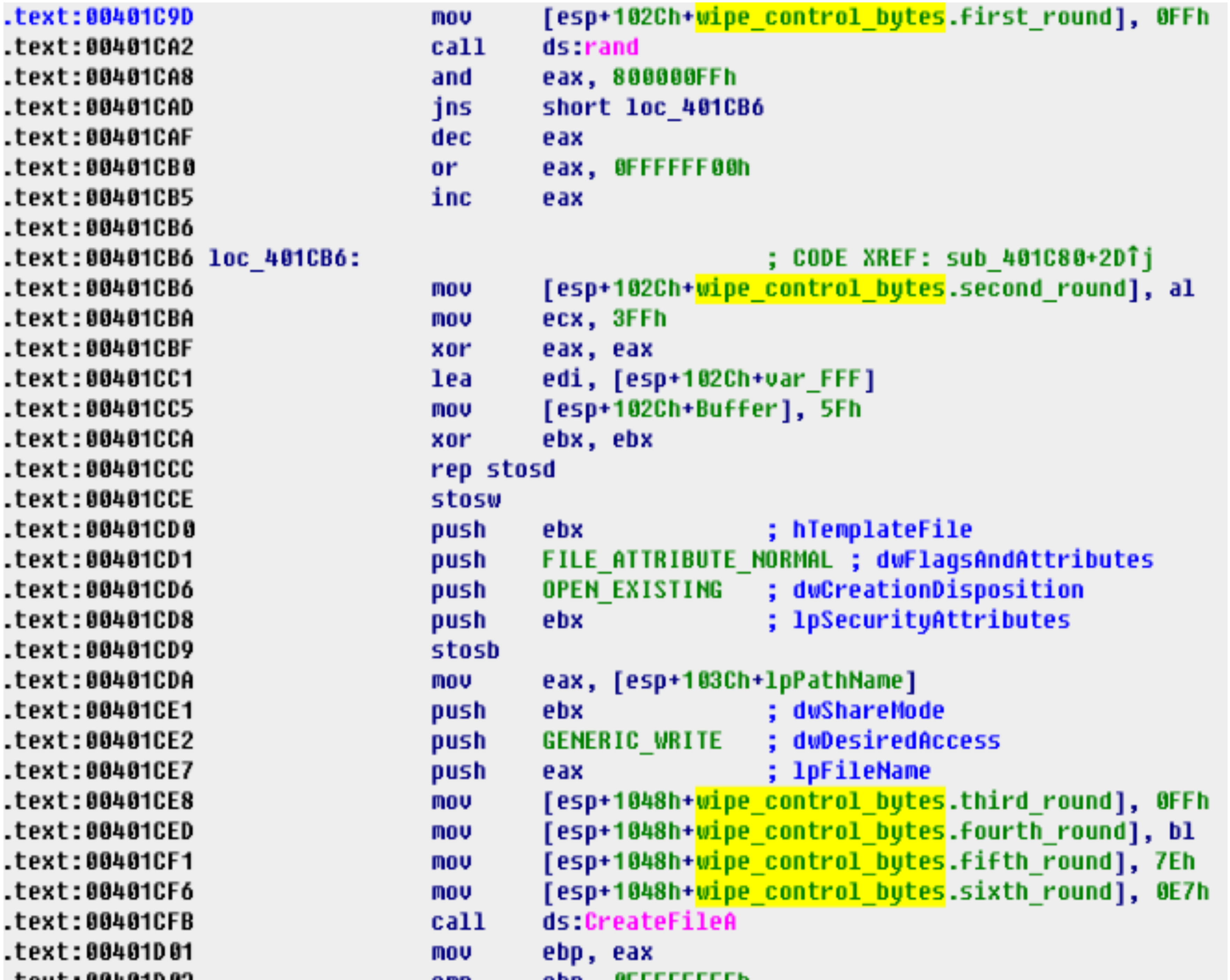
Wipe-out function in the Bangladesh case malware (2016)



```
hFile = CreateFileA(lpFileName, 0x40000000u, 0, 0, 3u, 0x80u, 0);
_hFile = hFile;
if ( hFile == (HANDLE)-1 )
  return GetLastError();
SetFilePointer(hFile, -1, 0, 2u);
WriteFile(_hFile, &_buf_zero, 1u, &NumberOfBytesWritten, 0);
FlushFileBuffers(_hFile);
FileSize.QuadPart = 0i64;
GetFileSizeEx(_hFile, &FileSize);
SetFilePointer(_hFile, 0, 0, 0);
HighPart = FileSize.HighPart;
LowPart = FileSize.LowPart;
j = 0;
j3 = 0;
if ( FileSize.HighPart >= 0 && (FileSize.HighPart > 0 || FileSize.LowPart > 0) )
{
  while ( 1 )
  {
    key = __OFSUB__(__PAIR__(HighPart, LowPart), __PAIR__(j3, j));
    k = LowPart - j;
    l = (__PAIR__(HighPart, LowPart) - __PAIR__((unsigned int)j3, j)) >> 32;
    size = LowPart - j;
    if ( l < 0 || (unsigned __int8)((l < 0) ^ key) | (l == 0) && k <= 0x1000 )
    {
      l2 = l;
    }
    else
    {
      size = 4096;
      l2 = 0;
    }
    if ( !WriteFile(_hFile, &_buf_zero, size, &NumberOfBytesWritten, 0) || !NumberOfBy
      break;
    HighPart = FileSize.HighPart;
    j2 = NumberOfBytesWritten + j;
    j3 = (__PAIR__(j3, NumberOfBytesWritten) + (unsigned __int64)j) >> 32;
    j += NumberOfBytesWritten;
    if ( j3 < FileSize.HighPart )
    {
      LowPart = FileSize.LowPart;
    }
    else
    {
      if ( j3 > FileSize.HighPart )
        break;
      LowPart = FileSize.LowPart;
      if ( j2 >= FileSize.LowPart )
        break;
    }
  }
}
FlushFileBuffers(_hFile);
CloseHandle(_hFile);
return removeFileDir(lpFileName, 0);
```

Link between Banswift & Lazarus

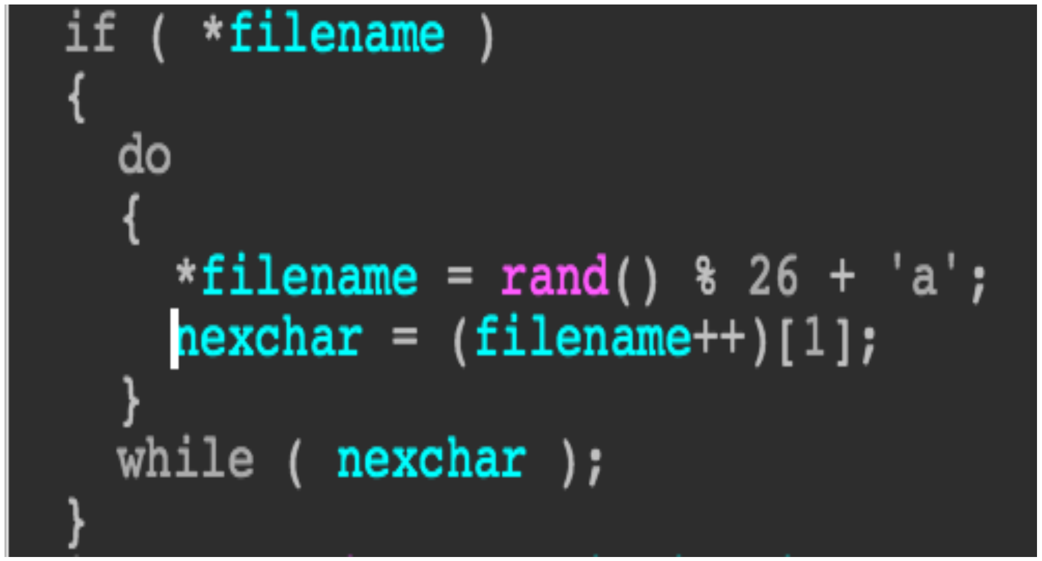
- Function takes two parameters: path of file to overwrite and number of iterations (max six)
- It will initially overwrite the last byte of the target file with 0x5F
- Six “control” bytes are supplied which dictate what bytes are used during the overwrite process



```
.text:00401C9D                 mov     [esp+102Ch+wipe_control_bytes.first_round], 0FFh
.text:00401CA2                 call    ds:rand
.text:00401CA8                 and     eax, 800000FFh
.text:00401CAD                 jns     short loc_401CB6
.text:00401CAF                 dec     eax
.text:00401CB0                 or      eax, 0FFFFFF00h
.text:00401CB5                 inc     eax
.text:00401CB6
.text:00401CB6 loc_401CB6:                             ; CODE XREF: sub_401C80+2D↑j
.text:00401CB6                 mov     [esp+102Ch+wipe_control_bytes.second_round], al
.text:00401CBA                 mov     ecx, 3FFh
.text:00401CBF                 xor     eax, eax
.text:00401CC1                 lea     edi, [esp+102Ch+var_FFF]
.text:00401CC5                 mov     [esp+102Ch+Buffer], 5Fh
.text:00401CCA                 xor     ebx, ebx
.text:00401CCC                 rep stosd
.text:00401CCE                 stosw
.text:00401CD0                 push    ebx             ; hTemplateFile
.text:00401CD1                 push    FILE_ATTRIBUTE_NORMAL ; dwFlagsAndAttributes
.text:00401CD6                 push    OPEN_EXISTING   ; dwCreationDisposition
.text:00401CD8                 push    ebx             ; lpSecurityAttributes
.text:00401CD9                 stosb
.text:00401CDA                 mov     eax, [esp+103Ch+lpPathName]
.text:00401CE1                 push    ebx             ; dwShareMode
.text:00401CE2                 push    GENERIC_WRITE   ; dwDesiredAccess
.text:00401CE7                 push    eax             ; lpFileName
.text:00401CE8                 mov     [esp+1048h+wipe_control_bytes.third_round], 0FFh
.text:00401CED                 mov     [esp+1048h+wipe_control_bytes.fourth_round], bl
.text:00401CF1                 mov     [esp+1048h+wipe_control_bytes.fifth_round], 7Eh
.text:00401CF6                 mov     [esp+1048h+wipe_control_bytes.sixth_round], 0E7h
.text:00401CFB                 call    ds:CreateFileA
.text:00401D01                 mov     ebp, eax
.text:00401D03                 cmp     ebp, 0FFFFFFFFh
```

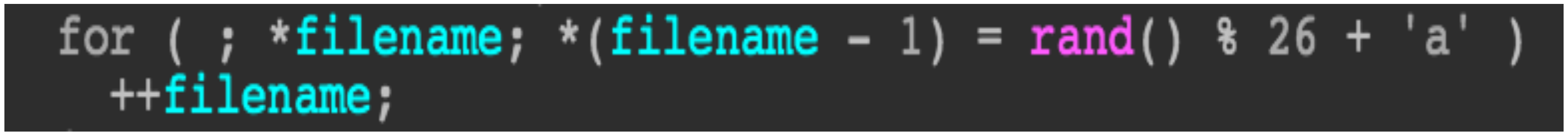
- Using same style random file rename in wipeout function

- Bankswift



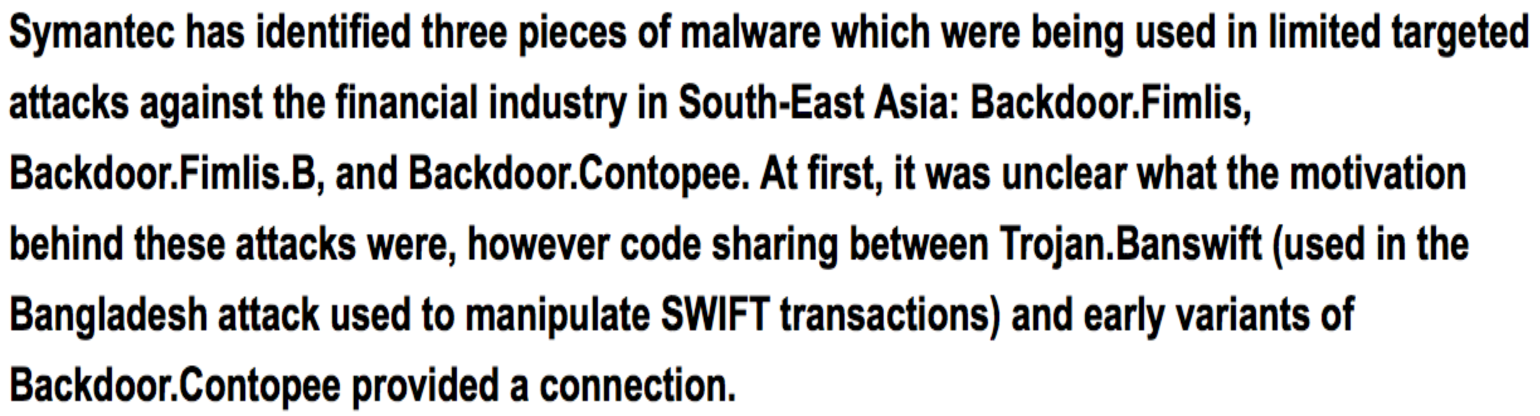
```
if ( *filename )
{
  do
  {
    *filename = rand() % 26 + 'a';
    nexchar = (filename++)[1];
  }
  while ( nexchar );
}
```

- Lazarus's tool (Backdoor.Contopee)



```
for ( ; *filename; *(filename - 1) = rand() % 26 + 'a' )
  ++filename;
```

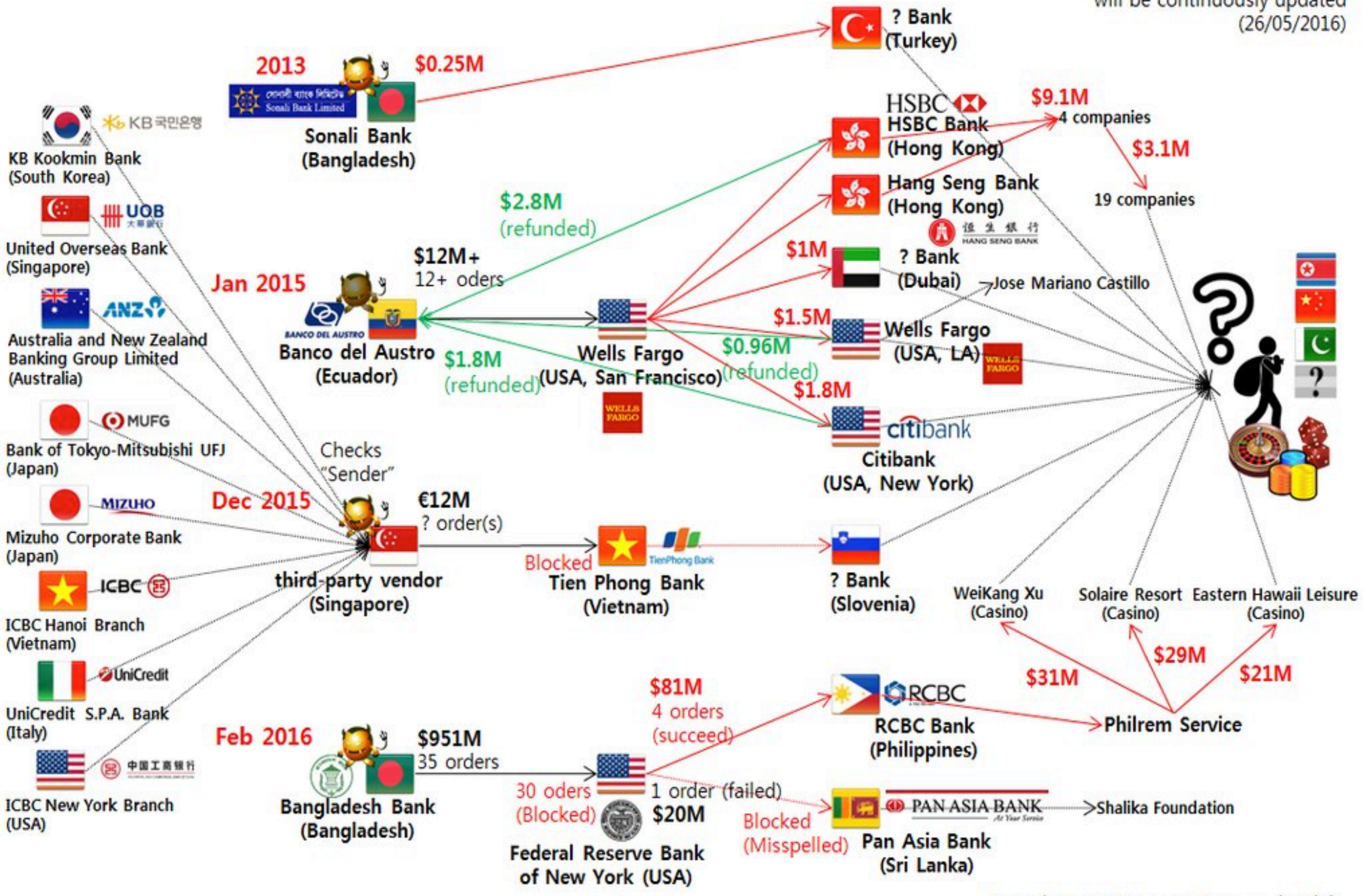
Backdoor.Contopee found with Banswift in Ecuadoran bank



Symantec has identified three pieces of malware which were being used in limited targeted attacks against the financial industry in South-East Asia: Backdoor.Fimlis, Backdoor.Fimlis.B, and Backdoor.Contopee. At first, it was unclear what the motivation behind these attacks were, however code sharing between Trojan.Banswift (used in the Bangladesh attack used to manipulate SWIFT transactions) and early variants of Backdoor.Contopee provided a connection.

Hacking the Worldwide Banking System (Using fraudulent SWIFT messages)

will be continuously updated
(26/05/2016)



From https://twitter.com/issuemakerslab

Thảo luận

Bank Swift [Attack]

#tradahacking