# Attacks on SWIFT Banking System Benefit From Insider Knowledge

securingtomorrow.mcafee.com/mcafee-labs/attacks-swift-banking-system-benefit-insider-knowledge

May 20, 2016

By Christiaan Beek on May 20, 2016

In recent months, we've seen headlines about the compromise of a bank in Bangladesh from which cybercriminals attempted to steal US$951 million. The malware they used was able to manipulate and read unique messages from SWIFT (Society for Worldwide Interbank Financial Telecommunication), as well as adjust balances and send details to a remote control server. BAE Systems wrote a detailed analysis and concluded that the malware must be based on a framework of different modules that could be used for multiple targets.

This week SWIFT sent another warning without details about another bank, this time in Vietnam that was compromised. According to a bank spokesperson, they detected in a timely manner the fraudulent transfer of $1.13 million in December 2015. Because we know the attackers had some insight into the Bangladesh attack, McAfee assumed the attackers also knew something beforehand about the Vietnamese bank. We investigated possible malware indicators for the latter attack.

Files used for the investigation:

- MD5: 0b9bf941e2539eaa34756a9e2c0d5343
- MD5: 909e1b840909522fe6ba3d4dfd197d93

We focused our analysis primarily on the first sample. The file's compile timestamp is 2015-12-04 02:04:23. The first submission of the file from Vietnam was on December 22, 2015.

In the case of the Vietnamese bank, the file used for the attack is a fake version of the popular PDF reader Foxit. The malware installs itself in the original Foxit installation directory and renames the original file to FoxltReader.exe.

Once the user starts using the fake reader, the malware executes and writes to a log file in the temp directory C:\\Windows\temp\\WRTU\ldksetup.tmp. Analyzing this file, we see the log data is XOR encoded using the value 0x47.

```
push    1                   ; int
push    eax                 ; int
push    offset aCWindowsTempWr ; "c:\\windows\\temp\\WRTU\\ldksetup.tmp"
call    sub_401000
```

As in the case of the Bangladeshi bank, the malware uses the configuration file Lmutilps32.dat, which can also be found in C:\\Windows\\temp\WRTU\. This file is also XOR encoded, with the value 0x7C4D5978.

Was this malware part of a targeted attack? Yes, absolutely. As in the malware used against the Bangladeshi bank, we found the SWIFT code for the target in multiple places in the malware:


The code TPBVVNVX is the SWIFT code for the Tienphong Commercial Joint Stock Bank, in Hanoi.

We also noticed that there were more SWIFT codes in the code:

```
data:0048F410 [redacted]sgsgxxx    db '[redacted]SGSGXXX',0      ; DATA XREF: sub_45AB30+26↑o
data:0048F410                                                    ; .data:0048F40C↑o
data:0048F41C [redacted]au3mxxx    db '[redacted]AU3MXXX',0      ; DATA XREF: .data:0048F408↑o
data:0048F428 [redacted]jpjtxxx    db '[redacted]JPJTXXX',0      ; DATA XREF: .data:0048F404↑o
data:0048F434 [redacted]jpjtxxx    db '[redacted]JPJTXXX',0      ; DATA XREF: .data:0048F400↑o
data:0048F440 [redacted]krsexxx    db '[redacted]KRSEXXX',0      ; DATA XREF: .data:0048F3FC↑o
data:0048F44C [redacted]vnvnxxx    db '[redacted]VNVNXXX',0      ; DATA XREF: .data:0048F3F8↑o
data:0048F458 [redacted]itmmxxx    db '[redacted]ITMMXXX',0      ; DATA XREF: .data:0048F3F4↑o
data:0048F464 [redacted]us33xxx    db '[redacted]US33XXX',0      ; DATA XREF: .data:off_48F3F0↑o
data:0048F470 ; char aI64d[]
data:0048F470 aI64d                db '%I64d',0                  ; DATA XREF: sub_45A500+91↑o
data:0048F476                      align 4
data:0048F478 ; char a_2i64d[]
data:0048F478 a_2i64d              db '%.2I64d',0                ; DATA XREF: sub_45A500+6F↑o
```

These banks are based in Australia, Singapore, Japan, Korea, Vietnam, Italy, and the United States. We wondered why the actors would put this particular list in the malware. Further analyzing the working of the malware, we discovered an interesting part in the code concerning ”Executing the real Foxit reader” and the next section in the code states “PDFmodulation success. ...” This hints of the manipulation of PDF files.

```
data:00465480 ; char aFoxit_readerEx[]
data:00465480 aFoxit_readerEx db '[FOXIT_READER] : Executing real foxit reader with CommandLine = %'
data:00465480                                          ; DATA XREF: _main+203↑o
data:00465480                 db 's.',0
data:004654C4 ; char aS[]
data:004654C4 aS              db ' "%s"',0             ; DATA XREF: _main+1D1↑o
data:004654CA                 align 4
data:004654CC ; char aSTS[]
data:004654CC aSTS            db '"%s" /t "%s"',0      ; DATA XREF: _main+19B↑o
data:004654D9                 align 4
data:004654DC ; char aFoxit_readerPd[]
data:004654DC aFoxit_readerPd db '[FOXIT_READER] : PDFModulation success, so real foxit reader will'
data:004654DC                                          ; DATA XREF: _main+17D↑o
data:004654DC                 db ' be executed.',0
data:0046552B                 align 4
```

In the code, we found that the malware uses the original driver fpdsdk.dll from the Foxit SDK to execute the transformation of the files.

We discovered functionality in the code that converts PDF files to XML files, which are stored in the folder C:\Documents and Settings\Test\Local Settings\Temp\. The filenames start with XXX or RSP followed by a value between 0-F and finish with the extension .tmp.

Let's return to our list of SWIFT codes of other banks. The malware reads the SWIFT messages and checks if the sender of the message is one of the listed banks. Once it finds these messages, it reads their information:

```
data:0048F484 aStatementLine  db ': Statement Line',0 ; DATA XREF: sub_45A760+5C↑o
data:0048F495                 align 4
data:0048F498 ; char aClosingBalance[]
data:0048F498 aClosingBalance db 'Closing Balance (Booked Funds)',0
data:0048F498                                         ; DATA XREF: sub_45A880+5D↑o
data:0048F4B7                 align 4
data:0048F4B8 aPos_temp       db 'POS_TEMP',0         ; DATA XREF: sub_45A880+36↑o
data:0048F4B8                                         ; sub_45A880+109↑o ...
data:0048F4C1                 align 4
data:0048F4C4 ; char asc_48F4C4[]
data:0048F4C4 asc_48F4C4      db '----------------------',0 ; DATA XREF: sub_45A9C0+D1↑o
data:0048F4C4                                         ; sub_45B630+1A0↑o
data:0048F4DA                 align 4
data:0048F4DC ; char aOpeningBalance[]
data:0048F4DC aOpeningBalance db ' Opening Balance',0 ; DATA XREF: sub_45AAE0:loc_45AAFB↑o
data:0048F4ED                 align 10h
data:0048F4F0 ; char aSender[]
data:0048F4F0 aSender         db 'Sender',0           ; DATA XREF: sub_45AB80:loc_45ABA0↑o
data:0048F4F0                                         ; sub_45B630+234↑o
data:0048F4F7                 align 4
data:0048F4F8 ; char aS_5[]
data:0048F4F8 aS_5            db '#%s#',0             ; DATA XREF: sub_45AC00+117↑o
data:0048F4F8                                         ; sub_45AC00+27D↑o ...
```

The malware can manipulate these messages: deleting transactions, transaction history, and system logs, and prevent the printing of the fraudulent transactions:

```
0x42DC    delete from graft..graft_history where file_name = '%s'
0x4314    [LOG_CLEAR] : get file_id for %s failed.
0x4340    [LOG_CLEAR] : clearing error_log success.
0x436C    [LOG_CLEAR] : clear_log - deleting error_log failed.
0x43A4    delete from graft..error_log where file_id = '%s'
0x43D8    [LOG_CLEAR] : file_id = %s
0x43F4    select file_id from graft..graft_history where file_name = '%s'
0x4434    [LOG_CLEAR] : clearing "%s" in %s
0x4458    %s\%s_%d%.2d%.2d.txt
0x4470    TPBVVNVX_TPBVVNVX_PrnOut
0x448C    TPBVVNVX_TPBVVNVX_PrnIn
0x44A4    TPBVVNVX_TPBVVNVX_PrintedOut
0x44C4    TPBVVNVX_TPBVVNVX_PrintedIn
```

As in the Bangladeshi attack, we found some typos:

- Bangladesh: "fandation" instead of "foundation" and "alreay" instead of "already"
- Vietnam: "FilleOut" instead of "FileOut"


Does this analysis tell us anything about the actors? It might, but these details form a weak indicator. How easy is it to misspell some words on purpose to mislead investigators?

**Conclusion**

In both attacks we can see that the attackers have done their reconnaissance properly and may have used an insider to get the details they needed to prepare their attacks. In the Bangladeshi case, for example, the malware samples are tuned to the environment and how the banking system operates, including the supported software, databases, and printer. In the Vietnamese case, the malware is also tuned to fit the environment. The attackers knew that the bank used Foxit and replaced it with a fake version. The attackers have a very good understanding of the SWIFT messaging system and how to manipulate the system to prevent the detection of their fraudulent attempts of transferring the money. The malware in each attack was compiled just before the attack happened.

Although both attacks were discovered at some point during the attempts to transfer large amounts of money, the actors may well have executed a few test runs to check their operations before the real attacks.

The operation in Vietnam happened in December 2015 and was discovered after an investigation of the incident in February 2016 in Bangladesh. The Vietnamese attack was reported to the banking world in May 2016. Would logs still be available for an incident that happened about six months ago? Would the possible test runs be traceable? These are some of the many questions that arise. One lesson from both cases is that when a fraud alert is triggered by either an internal system or by transaction authorities, a thorough analysis— including an in-depth analysis of the malware—of the tactics and procedures used by the attackers is needed. In this case, investigators can share indicators such as MD5 sums, but because the attackers have customized their malware, sharing would be of little value. On the other hand, sharing the methods used by the attackers, the inner working of the malware, and its manipulation of the systems should teach us where to look and adapt our defenses.