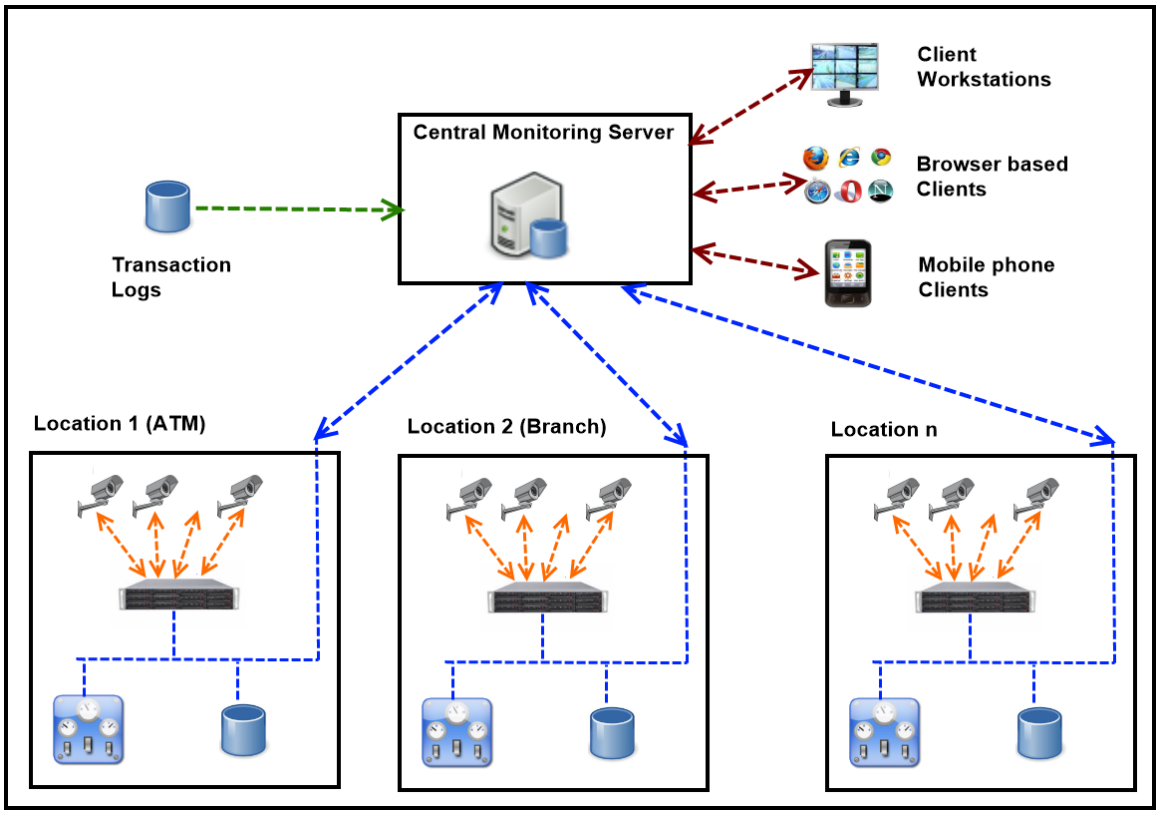
Security Management System
ATM POS Integration

Introduction

Security Management System supports integration with ATM / POS systems.



There are 3 types of integration modes available –

(a) Integration between ATM / POS device and Embedded VMS / NVR device (real-time messaging)
(b) Integration between transaction database manager system and Central Monitoring Server / CMS software (real-time messaging)
(c) Integration between transaction database manager system and Central Monitoring Server / CMS software (non real-time messaging)

This document describes these 3 integration modes

Integration between ATM / POS device and Embedded VMS / NVR device (real-time messaging)

Overall working

1. Embedded VMS / NVR has TCP server socket, listening for transaction record notifications from external systems (the external system in this case is ATM / POS device)
2. The ATM / POS device connects to the TCP server socket from Embedded VMS and sends a transaction record notification message in pre-defined XML format
3. The Embedded VMS processes the transaction record notification and sends back response indicating success / failure
4. If the transaction record notification message is valid, it is parsed by the Embedded VMS and the data is added to Embedded VMS database. The transaction record message is also forwarded to the CMS

TCP server socket

1. The TCP server socket is available at port number 5500
2. The TCP server socket is available at IP address displayed in the Embedded VMS GUI
3. The ATM / POS device connects to the TCP server socket whenever a transaction record notification is needed to be sent and closes it after sending the notification and receiving the response; for next transaction record notification, new connection is made with the TCP server socket.

Notification Format

The notification is in XML format, as follows –

```
<NData>
  <NRequestType>smsATMPOSNotificationRealTime</NRequestType>
  <NAuth>
    <UserName>ATMSource</UserName>
    <UserPassword>MTIzNA==</UserPassword>
  </NAuth>
  <NType>ATM_POS Transaction</NType>
  <DataStandard>
    <SMSModuleName>ATM01</SMSModuleName>
  </DataStandard>
  <DataCustom>
    <DParam>
```

```
      <ParamName>RequestID</ParamName>
      <ParamVal>A98265</ParamVal>
    </DParam>
    <DParam>
      <ParamName>CardNumber</ParamName>
      <ParamVal>9856728976AA</ParamVal>
    </DParam>
  </DataCustom>
</NData>
```

1. NData :: NRequestType – should have exact text ‘smsATMPOSNotificationRealTime’

2. NData:: NAuth :: UserName - has user name in plain text format

3. NData:: NAuth :: UserPassword - has base64 encoded user password

4. NData:: NType – should have exact text ‘ATM_POS Transaction’

5. NData:: DataStandard :: SMSModuleName – indicates the module ID

6. NData :: DataCustom – has multiple sub-nodes (DParam). One sub-node for every ‘parameter’. The list of parameters is pre-defined during integration analysis phase of every integration.

7. NData :: DataCustom :: DParam – has 2 sub-nodes –

   (a) ParamName – this is a string pre-defined during integration analysis phase of every integration
   (b) ParamVal – value associated with the parameter

Notes –

1. The list of parameters under ‘NData :: DataCustom’ needs to be defined, in the integration analysis phase

2. No software changes / customizations are required for every deployment, as the Video Management System software provides graphical user interface to specify these deployment specific parameters

Response Format

The response is in XML format, as follows –

```
<NResponse>
  <NRequestType>smsATMPOSNotificationRealTime</NRequestType>
  <ResponseStandard>
    <RCode>1</RCode>
    <RDescription>Success</RDescription>
  </ResponseStandard>
</NResponse>
```

1. NResponse :: NRequestType – indicates the type of request for which response is being sent back

2. NResponse :: ResponseStandard :: RCode – indicates the response code. It is a number.
   (a) 1 indicates success
   (b) 0 indicates unknown error
   (c) Negative values indicate specific errors
   (d) Positive values indicate warnings

3. NResponse:: ResponseStandard :: RDescription – indicates human readable string describing the success / failure of the notification processing

Integration between transaction database manager system and Central Monitoring Server / CMS software (real-time messaging)

Overall working

1. CMS has TCP server socket, listening for transaction record notifications from external systems (the external system in this case is the transaction database manager system)
2. The transaction database manager system connects to the TCP server socket from CMS and sends a transaction record notifications message in pre-defined XML format
3. The CMS processes the transaction record notifications and send back response indicating success / failure
4. If the transaction record notifications message is valid, it is parsed by the CMS and the data is added to CMS database.

TCP server socket

1. The TCP server socket is available at port number 5500
2. The TCP server socket is available at IP address displayed in the CMS GUI
3. The transaction database manager system connects to the TCP server socket whenever a transaction record notification is needed to be sent and closes it after sending the notification and receiving the response; for next transaction record notification, new connection is made with the TCP server socket

Notification Format

Notification format is same as the one for 'Integration between ATM / POS device and Embedded VMS / NVR device (real-time messaging)' integration mode.

Response Format

Response format is same as the one for 'Integration between ATM / POS device and Embedded VMS / NVR device (real-time messaging)' integration mode.

Integration between transaction database manager system and Central Monitoring Server / CMS software (non real-time messaging)

Overall working

1. The transaction database manager system writes the transaction logs to a CSV file. This CSV files has a pre-defined name and it is available in pre-defined directory path on the computer where CMS is running.
2. The frequency of CSV update is pre-defined
3. The CMS reads the available CSV file and processes the entries from it. For all valid entries, the data is added to CMS database.

CSV file format

1. The CSV file has one line for every transaction record
2. Each line / transaction record has multiple fields separated by 'comma' character
3. Names of available fields and their index (sequence) in the transaction log record need to be defined, in the integration analysis phase.
4. No software changes / customizations are required for every deployment, as the Video Management System software provides graphical user interface to specify these deployment specific parameters

CSV file name

1. Every CSV file created by transaction database manager system has unique name

2. The name format is YYYYMMDD_HHmmSS.csv, where
   YYYY indicates year, when the CSV file is created / copied
   MM indicates month, when the CSV file is created / copied
   DD indicates day, when the CSV file is created / copied
   HH indicates hour, when the CSV file is created / copied
   mm indicates minutes, when the CSV file is created / copied
   SS indicates seconds, when the CSV file is created / copied

CSV file management

1. The transaction database manager system should not 'update' the CSV file, once it is created. A CSV file should be opened for 'writing' only one time. Once the

file is created / copied, it should not be edited. For any new entries, new CSV file should be created

2. Better approach is to create the CSV file and copy it to the target directory path. Once the copy operation is completed, it should be edited / replaced by new file.
3. It is recommended to have entries in the CSV file, sorted by time stamp. Oldest records should be at the top and newest at the bottom.
4. The CMS processes the CSV file periodically and deletes the processed CSV file.