ATOM
an NTT DATA Company

ATOM
an NTT DATA Company

Atom Technologies Ltd.

www.atomtech.in

PENNYDROP

Process Note - API

Internet Payment Gateway

Atom Technologies Ltd.

www.atomtech.in

Contents

Document Information

Document Attributes	Information
ID	ProcessNote_PennyDrop _V1
Owner	Product Management & Audit
Product Author	Dharam Badheka

Revision Chart

This chart contains a history of this document's revisions.

Version	Primary Author	Description of Version	Date Completed	Reviewed By
1.0	Dharam	Process Note V1	07/05/2019	

Introduction

“***Security Is not product, but a process***”- Bruce Schneier

It’s always said that man in the middle attack is cause of hack or lack of awareness is a loss. Vulnerability leads to fraud, but approach of safe request is only to be taken care of to avoid this event, NO. It’s not only technology, that leads to aforesaid uncertain events, sometime a false manual work can have an obscure consequence.

PennyDrop

Not only product, but a step toward safe fund flow to a distinguished merchant, but a mechanism of validation executed after manual verification of Merchant’s Bank Account, used for settlement. Currently process of manual verification of Bank details is not guarantee to that same account is active or dormant, beholder is same as per cheque or not, does it differ from merchant’s said info or not. To overcome said assumption, and to boost up the verification, Penny Drop validation is an introduced. It’s an API, request sourced by **Merchant**, processed by **Atom an NTT DATA Company**, validated by **Issuing Bank**

API Specs

PROTOCOL	HTTPS
PAYLOAD	Plain String
REQUEST METHOD TYPE	POST

Request

```
{
"payInstrument":
{"headDetails":
{ "version":"OTSv1.1",
"payMode":"SL",
"channel":"ECOMM",
"api":"SALE",
"platform":"WEB",
"stage":1
},
"merchDetails":
{"merchId":mid,
"password":"",
"merchTxnId":"",
"mccCode":,
"merchType":"M",
"merchTxnDate":""
},
"payDetails":
{"prodDetails":
[{"prodName": "Pennydrop","prodAmount": 0.00 }],
"amount":0.00,
"surchargeAmount":0.00,
"totalAmount":0.00,
"custAccNo":"",
"custAccIfsc":"",
"custEmail":"",
"clientCode":"",
"txnCurrency":"INR",
"remarks":"",
"signature" :
"11a6d7ccf4fb140d15bb108bbaf1e84f35b8df1f45789e8f9a66875e
f7447e556a505fe123e37345708950c18c600465bbf020fd148c756
09bf3c5477a185243"
},
"payModeSpecificData":
{ "subChannel":[ "PD" ]
},
"custDetails":
{
"custFirstName":"",
"custLastName":"",
"custEmail":"",
"custMobile":"",
"billingInfo":
{
"custAddr1":"",
"custAddr2":"",
"custCountry":"",
"custCity":"",
"custState":"",
"custZipCode" : ""
}
}
}
}
```

Request Specification

File shared separately

Response

Decrypted Response:

{
"payInstrument":
{"merchDetails":
{"merchId":,
"userId":null,
"merchTxnId":"",
"mccCode":"",
"merchTxnDate":""},
"payDetails":{"atomTxnId":11000001178156,
"prodDetails":[{"prodName":"Pennydrop","prodAmount":0.00}],
"amount":0.00,
"surchargeAmount":0.00,
"totalAmount":0.00,
"custAccNo":"",
"custAccIfsc":"",
"clientCode":"",
"txnCurrency":"",
"remarks":"",
"signature":"3d026b3c74906bb749e2cdde08c4262b5dd2bc1ec1ce679e09d9987d81150d86e81532455c7ec956beb5cb4fbaa1816f771826523fcfc34c1fa8c24d64f1f87b",
"txnInitDate":"2020-09-29 15:41:22",
"txnCompleteDate":"2020-09-29 15:41:25"
},

"payModeSpecificData":
{
"subChannel":["PD"],
"bankDetails":
{"otsBankId":,
"bankTxnId":"",
"authId":"NA",
"otsBankName":"",
"cardType":null,
"cardMaskNumber":null,
"qrString":null}},
"extras":null,
"custDetails":
{"custFirstName":"",
"custLastName":null,
"custEmail":null,
"custMobile":null,
"billingInfo":null
},
"responseDetails":
{
"statusCode":"OTS0000",
"message":"SUCCESS",
"description":"TRANSACTION IS SUCCESSFUL."
}
}

}

Response Specification

File shared separately

Report

Report in merchant console will be comprising of below details of Penny Drop:

A. **Discriminator**: PD
B. **Description**: Description received from Bank
C. **Beneficiary Name**: Name of the Beneficiary received in response
D. **IMPS Status Code**: Received from Bank

AES

UAT Test Server Details:

URL: https://caller.atomtech.in/ots/payment/txn

UAT Details

Merchant Id: 8952

Txn Password: Test@123

Request Enc Key: A4476C2062FFA58980DC8F79EB6A799E

Response Enc Key: 75AEF0FA1B94B3C10D4F5B268F757F11

Merchant Id: 8952

Txn Password: Test@123

Request Enc Key: A4476C2062FFA58980DC8F79EB6A799E

Response Enc Key: 75AEF0FA1B94B3C10D4F5B268F757F11

Account Number: 123456789012

IFSC: DLXB0000092

MCC: 5999

Merchant Type: R

Production Test Server Details:

URL: https://payment1.atomtech.in/ots/payment/txn

Production Details

Merchant Id: Production MID

Txn Password: Production Password

Request Enc Key : Production Key

Response Enc Key : Production Key

Parameter Name	**Mandatory**	**Data Type & Max Length**	**Sample Value**	**Content/ Remarks**
encResKey	Mandatory	String(32)	75AEF0FA1B94B3C10D4F5B268F757F11	Encryption Details
encReqKey	Mandatory	String(32)	A4476C2062FFA58980DC8F79EB6A799E	

Decrypted Request

```
{
"payInstrument" :
{
"headDetails" :
{
"version" : "OTSv1.1",
"payMode" : "SL",
"channel" : "ECOMM",
"api" : "SALE",
"platform" : "WEB",
"stage" : 1
},
"merchDetails" :
{
"merchId" : MID,
"password" : "Password",
"merchTxnId" : "Penny123",
```

"mccCode" : 5862,

"merchType" : "M",

"merchTxnDate" : "2020-09-29 15:42:00"

},

"payDetails" :

{

"prodDetails" : [{"prodName": "Pennydrop","prodAmount": 1.00}],

"amount" : 1.00,

"surchargeAmount" : 0.00,

"totalAmount" : 1.00,

"custAccNo" : "Bank Account",

"custAccIfsc" : "IFSC",

"custEmail":"mail@mail.in",

"clientCode" : "1008",

"txnCurrency" : "INR",

"remarks" : "OTSPenny",

"signature" :
"11a6d7ccf4fb140d15bb108bbaf1e84f35b8df1f45789e8f9a66875ef7447e556a505fe123e37345708950c
18c600465bbf020fd148c75609bf3c5477a185243"

},

"payModeSpecificData"

:

{

"subChannel" : ["PD"]

```
},

"custDetails" : {

    "custFirstName" : "D",

    "custLastName" : "B",

    "custEmail" : "mail@mail.in",

    "custMobile" : "",

    "billingInfo" :

          {

    "custAddr1" : "Mumbai",

    "custAddr2" : "Mumbai",

    "custCountry" :"India",

    "custCity" : "Mumbai",

    "custState" :"Maharashtra",

          "custZipCode" : "400031"

    }

  }


}

}
```

Encrypted Request

https://payment1.atomtech.in/ots/payment/txn?merchId=107031&encData=6B221F09353F0EA08523B
C800E66EC011109712E2DE83B3F223BD1B6A7A4BFE420E0018D7F7C061074748B7E6FC38324D8CFECE
319C464E13E4FFC4A25D6AECE81B7BEA08053EEC094710566EA65467F2AF9EE43730B047C934E8EC519

726B7BC5243ACF56C89609337921B68FD0383BAD3A478C594A7130554E0BE79956461735FCBF5DEA16
0F76478DA48140E362FBAA5AD3486326617D32A41FAC42F3A81A6D010000D99B544E35E11636BCF4A
20710E5830BDDA089733C539626EB0795A390FF6B09F17ADC1A12F2F3E80EBA1878A621CB25DFEA569
C627A0DF1BE1FD19A231C40418AB2623CB09C28098984E4817F3FF1B88572025BC0B9C666BF1225AB9
31B21FBD4A600E5C032EED5F25D9AAA2AD20D8968D971C46AC9BBBB259E2B0924BA4BAF745B48F235
5071DFE16044BE9D53362AB7F7642943589FA45787B44F79BA0E7C6FED8137F0108B3A0B1966EAB562
6C02D8F086E173B1D711A04DBAD77858A12D60C74ABF496562E3EBECFFB9344EAF51C69B351950504C
9E32BACF92F2F1D6D3FD09FD0BFF8C86EAC375A9A8D1705A3A5B86A3E259379D0C5A70DC81C4704D5
E49F99EE208A2509DCC68290BAB74D884C38BAAFE1015AD81401FD029CE5C7500694BFC1C5EB73B969
4C56563CDF838F3D40190FF602519F571088540266D50ED594C7DE9614937235123C9F81657281D6765
872AB9FA63155C5A1FEB34C41593A1F83FF880E493863D5B2BE836D5ABD49550B5F79538F774091F63
B2F7D55D5FE3A0FA5CC371EA936BEB21D73199F77C11B020B0BA1E173799BCF1E3250DCFCF689BC9DC
8BDF2FE0143ABB512C11D8804290124582B4315D04FBFD2A6E9E4244BAD94BDEF40B086E8434475A3E
C138AF459249A07E2B0805F6EF5242367D8EDF0EB608F6B58FC5B1D2AE28DA4767966D60CDA1CDE623
A36585B36A1983D3FE4BB293B333586962556940E4F2B677181BD2EB6A87E4369CAD600C5244C25609
4CAF89C9D22F3CE80D4EF165F3F7DEABA7F50E7E123E3C54734AC7EDCAAB1187B210C220935733D4BB
CF418D89A599173929567B2C162582963E038CFF22CDD15AA17F2EB44117FE32E188666C876D506B41
8A6A64284099DF01C1446DC5DD9444AF8F078A3E6B56FDB04976ECD330D53B12BA165BC9A73B67FC2
A4AE3597FA398715B94916734ECCC1313FE1CC56803ED723B895EA68A0CBC316D2455A80CFD4120279
712932E624DFCC9B6D55A0F2841C2C4805349EFCBCF65AE9A1E7A4ABE83A7E05B3100CDD2E05EC097F
386655AB65939210072F0A01DFC41CE51EE03873955C25E3B02363310230AFF244E102AFACB0F289C50
890B7A58B4A8FB5DD5C8819913046DA92E2F177BE4975F4E2C40DA601EC5A41C908A53700FC282EA9F
02F3F180D8C750332AB95ADE11A7BF25F4A037D7EB40F7EE3CDD34EA093D54FF7984F560D59570CE8A
712AFEBD5F13B5523BD79A1F59DA29D03243056E09FE2F6BDE98A3B048A3D40AAACA816D174FFEC51
6ABB285E9618A15ACF733A84A42A04F4CFE4C8E4C8BC33B4E9C879255287C8D93DB183D72876A3BB1
1CFCCF7971876A788D7C4F55ED20F3E144D7D91CC70FADD82D1360D782A1C0C0AE3A95AF231375FA0
F6AB310C0FB35F7ED77D1EF3FB3ACCF813562758C8B2C3470BE0E0D11F21DE0BD5ACEB3B316BFACA9F
C7785499C5AF92677B012D4421E4EAEC9F9D9

Encrypted Response

6B221F09353F0EA08523BC800E66EC011109712E2DE83B3F223BD1B6A7A4BFE420E0018D7F7C0610747
48B7E6FC38324D8CFECE319C464E13E4FFC4A25D6AECE81B7BEA08053EEC094710566EA65467F2AF9EE
43730B047C934E8EC519726B7BC5243ACF56C89609337921B68FD0383BAD3A478C594A7130554E0BE7
9956461735FCBF5DEA160F76478DA48140E362FBAA5AD3486326617D32A41FAC42F3A81A6D010000D
99B544E35E11636BCF4A20710E5830BDDA089733C539626EB0795A390FF6B09F17ADC1A12F2F3E80EB
A1878A621CB25DFEA569C627A0DF1BE1FD19A231C40418AB2623CB09C28098984E4817F3FF1B885720

25BC0B9C666BF1225AB931B21FBD4A600E5C032EED5F25D9AAA2AD20D8968D971C46AC9BBBB259E2B
0924BA4BAF745B48F2355071DFE16044BE9D53362AB7F7642943589FA45787B44F79BA0E7C6FED8137
F0108B3A0B1966EAB5626C02D8F086E173B1D711A04DBAD77858A12D60C74ABF496562E3EBECFFB93
44EAF51C69B351950504C9E32BACF92F2F1D6D3FD09FD0BFF8C86EAC375A9A8D1705A3A5B86A3E2593
79D0C5A70DC81C4704D5E49F99EE208A2509DCC68290BAB74D884C38BAAFE1015AD81401FD029CE5C
7500694BFC1C5EB73B9694C56563CDF838F3D40190FF602519F571088540266D50ED594C7DE96149372
35123C9F81657281D6765872AB9FA63155C5A1FEB34C41593A1F83FF880E493863D5B2BE836D5ABD49
550B5F79538F774091F63B2F7D55D5FE3A0FA5CC371EA936BEB21D73199F77C11B020B0BA1E173799B
CF1E3250DCFCF689BC9DC8BDF2FE0143ABB512C11D8804290124582B4315D04FBFD2A6E9E4244BAD9
4BDEF40B086E8434475A3EC138AF459249A07E2B0805F6EF5242367D8EDF0EB608F6B58FC5B1D2AE28
DA4767966D60CDA1CDE623A36585B36A1983D3FE4BB293B333586962556940E4F2B677181BD2EB6A8
7E4369CAD600C5244C256094CAF89C9D22F3CE80D4EF165F3F7DEABA7F50E7E123E3C54734AC7EDCAA
B1187B210C220935733D4BBCF418D89A599173929567B2C162582963E038CFF22CDD15AA17F2EB4411
7FE32E188666C876D506B418A6A64284099DF01C1446DC5DD9444AF8F078A3E6B56FDB04976ECD330
D53B12BA165BC9A73B67FC2A4AE3597FA398715B94916734ECCC1313FE1CC56803ED723B895EA68A0C
BC316D2455A80CFD4120279712932E624DFCC9B6D55A0F2841C2C4805349EFCBCF65AE9A1E7A4ABE83
A7E05B3100CDD2E05EC097F386655AB65939210072F0A01DFC41CE51EE03873955C25E3B0236331023
0AFF244E102AFACB0F289C50890B7A58B4A8FB5DD5C8819913046DA92E2F177BE4975F4E2C40DA601E
C5A41C908A53700FC282EA9F02F3F180D8C750332AB95ADE11A7BF25F4A037D7EB40F7EE3CDD34EA09
3D54FF7984F560D59570CE8A712AFEBD5F13B5523BD79A1F59DA29D03243056E09FE2F6BDE98A3B048
A3D40AAACA816D174FFEC516ABB285E9618A15ACF733A84A42A04F4CFE4C8E4C8BC33B4E9C8792552
87C8D93DB183D72876A3BB11CFCCF7971876A788D7C4F55ED20F3E144D7D91CC70FADD82D1360D782
A1C0C0AE3A95AF231375FA0F6AB310C0FB35F7ED77D1EF3FB3ACCF813562758C8B2C3470BE0E0D11F2
1DE0BD5ACEB3B316BFACA9FC7785499C5AF92677B012D4421E4EAEC9F9D9

Decrypted Response

```
{

"payInstrument":

{"merchDetails":

{"merchId":mid,

"userId":null,

"merchTxnId":"Penny123",
```

"mccCode":"123",

"merchTxnDate":"2020-09-29 15:42:00"},

"payDetails":{"atomTxnId":11000001128156,

"prodDetails":[{"prodName":"Pennydrop","prodAmount":0.00}],

"amount":0.00,

"surchargeAmount":0.00,

"totalAmount":0.00,

"custAccNo":"Account Number",

"custAccIfsc":"IFSC",

"clientCode":"",

"txnCurrency":"INR",

"remarks":"",

"signature":"3d026b3c74906bb749e2cdde08c4262b5dd2bc1ec1ce679e09d9987d81150d86e81532455c7ec956beb5cb4fbaa1816f771826523fcfc34c1fa8c24d64f1f87b",

"txnInitDate":"2020-09-29 15:41:22",

"txnCompleteDate":"2020-09-29 15:41:25"

},

"payModeSpecificData":

{

"subChannel":["PD"],

"bankDetails":

{"otsBankId":3,

"bankTxnId":"027315184127",

"authId":"NA",

"otsBankName":"Bank",

"cardType":null,

"cardMaskNumber":null,

"qrString":null}},

"extras":null,

"custDetails":

{"custFirstName":"",

"custLastName":null,

"custEmail":null,

"custMobile":null,

"billingInfo":null

},

"responseDetails":

{

"statusCode":"OTS0000",

"message":"SUCCESS",

"description":"TRANSACTION IS SUCCESSFUL."

}

}

}

Class File [Logic]

```
import java.util.logging.Logger;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.SecretKeySpec;

public class AtomEncryption
{
  static Logger log = Logger.getLogger(AtomEncryption.class.getName());

  private static int pswdIterations = 65536;
  private static int keySize = 256;
  private static final byte[] ivBytes = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 };

  public static String encrypt(String plainText, String key)
  {
    try
    {
      byte[] saltBytes = key.getBytes("UTF-8");

      SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA512");
      PBEKeySpec spec = new PBEKeySpec(key.toCharArray(), saltBytes, pswdIterations, keySize);

      SecretKey secretKey = factory.generateSecret(spec);
      SecretKeySpec secret = new SecretKeySpec(secretKey.getEncoded(), "AES");

      IvParameterSpec localIvParameterSpec = new IvParameterSpec(ivBytes);
      Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
      cipher.init(1, secret, localIvParameterSpec);

      byte[] encryptedTextBytes = cipher.doFinal(plainText.getBytes("UTF-8"));

      return byteToHex(encryptedTextBytes);
    }
    catch (Exception e) {
      log.info("Exception while encrypting data:" + e.toString());
    }

    return null;
```

```
}

public static String decrypt(String encryptedText, String key)
{
  try
  {
    byte[] saltBytes = key.getBytes("UTF-8");
    byte[] encryptedTextBytes = hex2ByteArray(encryptedText);

    SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA512");
    PBEKeySpec spec = new PBEKeySpec(key.toCharArray(), saltBytes, pswdIterations, keySize);

    SecretKey secretKey = factory.generateSecret(spec);
    SecretKeySpec secret = new SecretKeySpec(secretKey.getEncoded(), "AES");

    IvParameterSpec localIvParameterSpec = new IvParameterSpec(ivBytes);
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    cipher.init(2, secret, localIvParameterSpec);

    byte[] decryptedTextBytes = (byte[])null;
    decryptedTextBytes = cipher.doFinal(encryptedTextBytes);

    return new String(decryptedTextBytes);
  }
  catch (Exception e) {
    log.info("Exception while decrypting data:" + e.toString());
  }

  return null;
}

private static String byteToHex(byte[] byData) {
  StringBuffer sb = new StringBuffer(byData.length * 2);

  for (int i = 0; i < byData.length; ++i) {
    int v = byData[i] & 0xFF;
    if (v < 16)
      sb.append('0');
    sb.append(Integer.toHexString(v));
    }

    return sb.toString().toUpperCase();
  }

  private static byte[] hex2ByteArray(String sHexData) {
    byte[] rawData = new byte[sHexData.length() / 2];
    for (int i = 0; i < rawData.length; ++i) {
      int index = i * 2;
      int v = Integer.parseInt(sHexData.substring(index, index + 2), 16);
      rawData[i] = (byte)v;
    }

    return rawData;
  }

  public static void main(String[] args) {

            try {

                        String encryptedData = AtomEncryption.encrypt("1235", "ASWKLSLLFS4sd4g4gsdg");
                        System.out.println("encryptedData : "+encryptedData);
            } catch (Exception e) {
                        // TODO: handle exception
            }

}

}
```

Signature

- Signature generation sequence [merchantId + TxnPasword + MerchantTxnID + Payment Mode + Total Amount + currency + stage]
- The UAT request and response hash key are:

MerchId	reqHashKey	respHashKey
MID	Key	Key

Parameter Name	Mandatory	Data Type & Max Length	Sample Value	Content/ Remarks
reqHashKey	Mandatory	String(20)	KEY1234567234	Need to be save per MID on boarded in MW in Merchant Table
respHashKey	Mandatory	String(20)	KERESPY1234567234	Need to be save per MID on boarded in MW in Merchant Table

Class File [Logic]

```
import java.io.PrintStream;
import java.io.UnsupportedEncodingException;
import java.security.InvalidKeyException;
import java.security.Key;
import java.security.NoSuchAlgorithmException;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public class AtomSignature
{
  public static String generateSignature(String hashKey, String[] param)
  {
    String resp = null;

    StringBuilder sb = new StringBuilder();
    for (String s : param) {
      sb.append(s);
    }

    try
    {
      System.out.println("String =" + sb.toString());
      resp = byteToHexString(encodeWithHMACSHA2(sb.toString(), hashKey));
    }
    catch (Exception e)
    {
      System.out.println("Unable to encocd value with key :" + hashKey + " and input :" + sb.toString());
      e.printStackTrace();
    }
    return resp;
  }
  private static byte[] encodeWithHMACSHA2(String text, String keyString)
    throws NoSuchAlgorithmException, InvalidKeyException, UnsupportedEncodingException
  {
    Key sk = new SecretKeySpec(keyString.getBytes("UTF-8"), "HMACSHA512");
    Mac mac = Mac.getInstance(sk.getAlgorithm());
    mac.init(sk);
    byte[] hmac = mac.doFinal(text.getBytes("UTF-8"));

    return hmac;
  }
public static String byteToHexString(byte byData[])
        {
                StringBuilder sb = new StringBuilder(byData.length * 2);

                for(int i = 0; i < byData.length; i++)
                {
                        int v = byData[i] & 0xff;
                        if(v < 16)
                                sb.append('0');
                        sb.append(Integer.toHexString(v));
                }
                return sb.toString();
        }
```

Does and Don't

Does

1. Provide appropriate Account Details to avoid rejections
2. Refer transaction report in merchant console for below:

   A. Transaction Status
   B. Transaction Description

Don’t

1. Make use of same Merchant Txn Id [Except reinitiating for failed transaction]
2. Change MCC or Product Name. It’ll always be static
3. Initiate the transaction for Future Date.