**E-text (Electronic Text)**

**Introduction**

* E-text template is RTF based template.
* Generate text output for electronic fund transfer and electronic data interchange.
* Input - XML and output - text file then transmitted to bank or other customer.
* Payment automation

**Electronic Fund Transfer(EFT)**

* Transmission of financial data and payments to bank in specific fixed position format flat file(text).

**Electronic Data Interchange(EDI)**

* Similar to EFT but it involves only in transmission of payment to banks and exchanging business document(such as PO,invoice) between companies.
* Transmitted as flat text file.
* It contains several hundred character record length so difficult to layout in standard size paper.
* In order to accommodate, EFT and EDI templates are designed using tables.
* These formats requires special handling for XML file input(global level).

**Structure of e-text templates**

**Two types**

1. Fixed position based(EFT)
2. Delimiter based(EDI)

* Template consists of series of tables.
* Tables defines - layout, setup commands and data field definitions.
* Required data description columns vary for these two template but commands and functions are similar.

**X-path**

Xpath can be used to navigate through elements and attributes in an XML document.

* Xpath stands for XML path language
* Xpath uses "path like" syntax to identify and navigate nodes in an XML document
* Xpath contains over 200 built-in functions
* Xpath is a major element in the XSLT standard

**Xpath functions**

There are functions for

* String values
* Numeric values
* Boolean
* Date and time comparison
* Node manipulation
* Sequence manipulation and much more.

**X-path nodes**

In Xpath, there are seven kinds of nodes:

* Element
* Attribute
* Text
* Name space
* Processing-instruction
* Comment
* Document nodes.

**PAYMENTS**

**INBOUND (FUNDS CAPTURE) TRANSACTION**

+ Money inward from other party.

+ From - Sales revenue / Refunds from suppliers / Financing transactions / Amounts awarded as a result of legal proceedings.

+ In the funds capture flow, the external payment system can be one of two kinds: gateway or processor.

For gateway-model payment systems, every transaction is on-line and involves real-time communication

with the payment system. Processor model payment systems support real-time communication for authorizations

and batch operation for settlement processing.

**OUTBOUND (FUNDS DISBURSEMENTS) TRANSACTION**

+ Money outward to other party.

+ To - Supplier / Partners / Employers.

+ In the disbursements flow, there is no concept of real-time transaction processing.

Payment processing is performed in batch mode only (usually by FTPing a payment file to the payment system).

**PAYMENT PROCESS PROFILE**

A payment process profile is a payment attribute assigned to documents payable, which specifies handling of the documents payable, payments, and payment files by Oracle Fusion Payments. Payment process profiles include several types of information, such as specifications for payment file formatting and transmission.

**POSITIVE PAY**

A Flat-File contains information about current negotiable and voided payments. It is electronically transmits to the bank.

**FUND CAPTURE**

* Source product creates a transaction, which is used to store payment-related attributes.
* Source product requests authorization from Oracle Payments for the transaction.
* Source product passes the transaction and authorization information to Oracle Receivables.
* Oracle Receivables initiates settlement process.
* Oracle Payments settles the transaction.

**Transmission Configuration Setup**

To make a transaction between Oracle HCM Cloud and your own servers through Oracle WebCenter Content server , the Transmission Configuration must be setup. After that, a connection made between the two for transmission.

For secure transmission : PGP keys are used.This PGP-based encryption support is available for secure file transfer using HCM Data Loader and HCM Extracts.

PGP keys are generated to encrypt and decrypt the files in the specific format.

**OUTBOUND**

**Bank** creates **PGP** keys --->  **Bank** sends **public** key to **Company** ---> ***Company*** *creates signing keys --->* ***Company*** *sends the public signing key to* ***Bank*** *--->* ***Company*** *sign the file using* ***private*** *signing key* ---> **Company** encrypts file using the **public** key ---> **Company** sends the encrypted file to **Bank**  ---> **Bank** decrypts the file using **private** key ---> ***Bank*** *verify the file using* ***public*** *signing key.*

**INBOUND**

**Company** creates **PGP** keys --->  **Company** sends **public** key to **Bank** ---> ***Bank*** *creates signing keys --->* ***Bank*** *sends the public signing key to* ***Company*** *--->* ***Bank*** *sign the file using* ***private*** *signing key* ---> **Bank** encrypts file using the **public** key ---> **Bank** sends the encrypted file to **Company**  ---> **Company** decrypts the file using **private** key ---> ***Company*** *verify the file using* ***public*** *signing key.*

**Outbound Transaction Flow**

- Bank creates both private and public keys in the Payment in Oracle application and export the public key to the system.

- Send the public key to the company.

- Company encrypt the file using the public key sent by the Bank.

- Send the encrypted file to the Bank.

- Bank decrypt the encrypted file using the private key made the transaction.

**Inbound Transaction Flow**

- Company creates both private and public keys in the Payment in Oracle application and export the public key to the system.

- Send the public key to the Bank.

- Bank encrypt the Bank statement using the public key sent by the Company.

- Send the encrypted file to the Company.

- The Statement is now verified by the financial, whether the transaction in financial are matched to the bank statement.

**Signing**

Outbound files are signed using the HCM Cloud private key. You verify these files using the HCM Cloud public key.

Inbound files are signed using your private key. The data-loading process verifies inbound files using your public key.

**Algorithm**

**RSA**

Rivest-Shamir-Adleman

Public key cryptosystem widely used for secure data transmission.

Cipher: AES-128, Blowfish, CAST5, 3DES

Compression: bzip2, zlib, .zip, uncompressed

Hash: SHA-1, SHA-256, SHA-224, SHA-512, MD5, SHA-384, RIPEMD-160

**CIPHER**

A cipher is the algorithm used to encrypt and decrypt data. Generally speaking, the more bits that a cipher uses during encryption, the stronger or more secure the encryption is.

**AES-128**

AES128 uses the AES-128 cipher, which has a key size of 128 bits.

**BLOWFISH**

Blowfish encryption with a 64-bit block size and a variable-length key size from 32 bits to 128 bits. Use BLOWFISH only for backward compatibility with earlier Oracle GoldenGate versions.

**HASHING**

It is a process to convert information to a shorter fixed value known as the key that is used to represent the original information.

**SHA - Secure Hashing Algorithm**

Cryptographic hash function used to produce a hash value from the input file or message. Family - SHA-1, SHA-256, SHA-224, SHA-512 etc.,.

**ENCRYPTION**

It is the process to encode data securely such that only the authorized user who knows the key or password is able to retrieve the original data for everyone else it is just garbage.

**COMPRESSION**

Reduces the size of the files.

* **Lossy**
* **Lossless**

**Key length** - 1024 or 2048

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Encryption Mode** | **Encryption Key** | **Decryption Key** | **Signing Key** | **Verification Key** |
| Outbound PGP Signed | customer-key\_pub | customer-key\_priv | fusion-key\_priv | fusion-key\_pub |
| Outbound PGP Unsigned | customer-key\_pub | customer-key\_priv | N/A | N/A |
| Inbound PGP Signed | fusion-key\_pub | fusion-key\_priv | customer-key\_priv | customer-key\_pub |
| Inbound PGP Unsigned | fusion-key\_pub | fusion-key\_priv | N/A | N/A |

**Process**

Write programs to send your encrypted files to Oracle WebCenter Content, using the Oracle WebCenter Content Web Services.

**Extensions**

Standard extension - > \*.asc

Any filename extension are supported.

**Naming Convention**

**While Create**

fusion-key\_pub.asc

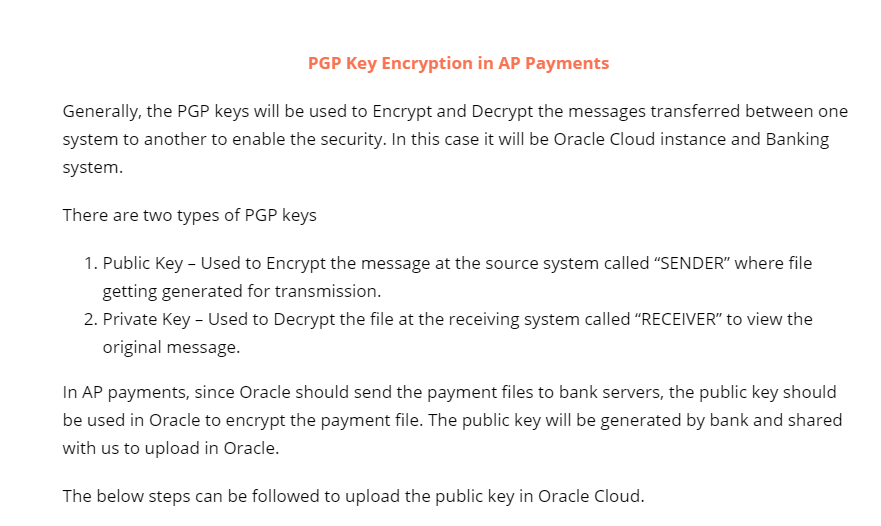
fusion-key\_priv.asc

**While upload to oracle**

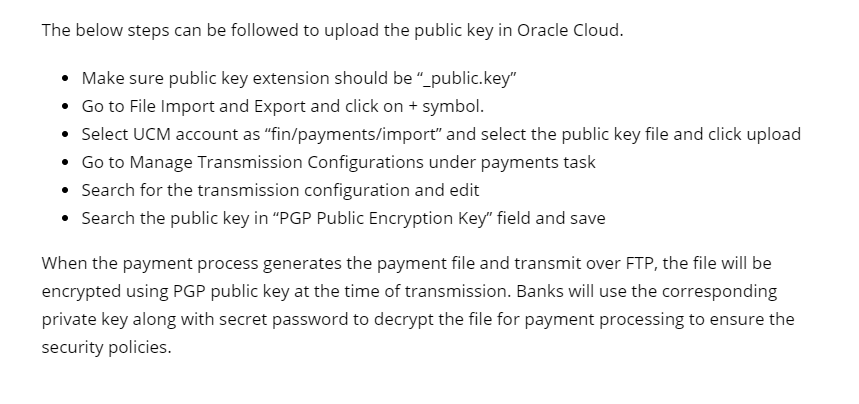
\_public.key

\_secret.key

**PGP Encryption**



**Upload public key to Oracle Cloud**



**SECURITY OF PAYMENT DATA**

**Channel Security**

By using secured transmission protocol such as SFTP, HTTPS etc.

**Payload Security**

By securing the payment file via payment file encryption and digital signature