**Bank Payment**

|  |  |
| --- | --- |
| IMG_256 | **Author : Roopa R**  **Creation Date : 27.09.2021** |

**Table of Contents**

1. **Bank Payment 3**

**1.1 Template 3**

**1.2 Template Types 3**

1. **Etext and XSLT 3**

**2.1 Introduction of eTEXT 3**

**2.2 Types of Etext Template 3**

**2.3 Fixed Position based attributes 3**

**2.4 Delimiter based attributes 3**

**2.5 Electronic Fund Transfer 3**

**2.6 Electronic Data Interchange 4**

**2.6.1 Structure of EFT and EDT 4**

**2.7 XSL Template 4**

**2.8 Requirements 4**

**2.8.1 XML 5**

**2.8.2 How to generate XML? 5**

**2.8.2.1 System Generated XML 5**

**2.8.2.2 Custom XML 5**

**2.8.3 Where do we get the Mapping sheet? 5**

**3. Payment Types 5**

**3.1 Outbound Transaction 5**

**3.2 Inbound Transaction 5**

**4. Transmission Configuration Setup 6**

**4.1 Create Transmission Configuration 6**

**5. Payment Process profile 6**

**5.1 Create/Update Payment Process Profile (PPP) 6**

**6. Keys 7**

**6.1 PGP key 7**

**6.2 Signing Key 7**

**6.3 How to upload and generate the PGP & Signing keys?(Outbound) 8**

**6.3.1 PGP Public Encryption Key 8**

**6.3.2 PGP Private Signing Key 8**

**6.4 How to upload and generate the PGP & Signing keys? (Inbound) 9**

**6.4.1 PGP Private Decryption Key 9**

**6.4.2 PGP Public Signature Verification Key 9**

**6.5 Steps to Upload Bank Given Public Key File 9**

**6.6 Steps to Download Fusion Generated Public Key File 10**

**6.7 Alternate way to create PGP Key (Without oracle) 10**

**7. Extensions 11**

**7.1 Extension When Download 11**

**7.2 Extension When Upload to UCM 11**

**7.3 Key length 11**

**8. Algorithm 11**

**8.1 RSA 11**

**8.2 CIPHER 11**

**8.2.1 AES-128 11**

**8.2.2 BLOWFISH 12**

**8.3 HASHING 12**

**8.3.1 SHA - Secure Hashing Algorithm 12**

**8.4 Compression 12**

1. **Overall Flow 12**
2. **Bank Payment**

* A company have multiple suppliers and it leads to pay multiple payment.
* Individual payment is not possible in this scenario.
* So, we decided to make the **Payment automation**.
* In this case, we are using Etext & XSL Template.
* In Bank Payment, we are going to use two types of Templates namely **EText and XSL**.
  1. **Template**

Template design involves the construction of a report layout in a template file and is dependent upon what the core Oracle BI Publisher engines accept for processing.

The nature of the data plays a role in the selection of a template.

* 1. **Template Types**
* EText
* XSL Templates

1. **EText and XSLT**

**2.1 Introduction of eTEXT**

* An eText template is an RTF - based template.
* It is used to generate text output for Electronic Funds Transfer (EFT) and Electronic Data Interchange (EDI).
* That output text file can be transmitted to a bank or other customer.

**2.2 Types of Etext Template**

* Fixed-position based (EFT templates)
* Delimiter-based (EDI templates)

**2.3 Fixed-position based Attributes:**

Position,Length,Format,Pad,Data,Comments

**2.4 Delimiter-based Attributes:**

Maximum Length,Format,Data,Comments

**2.5 Electronic Fund Transfer(EFT)**

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

**Refer the sample EFT Sample Output**

**2.6 Electronic Data Interchange(EDI)**

* Exchanging business documents(such as PO,invoice)/business informations between companies.
* Similar to EFT but it involves only in transmission of payment to banks and Transmitted as flat text file.

**Refer the sample EDT Sample Output**

**2.6.1 Structure of EFT & EDT**

* Files in these formats are transmitted as flat files, rather than printed on paper.
* The length of a record is often several hundred characters and therefore difficult to layout on standard size paper.To accommodate the record length, the EFT and EDI templates are designed using tables.
* Each record is represented by a table.
* Each row in a table corresponds to a field in a record.
* The columns of the table specify the position, length, and value of the field.
* Levels are used in etext. Once the level is opened,that corresponding level will be ended.

**2.7 XSL Template**

* Using XSL Template we can generate the XML according to the bank requirement.
* It is capable to generate the output in three format **HTML, XML, TEXT**.
* Based on the tags, we create an XSL Template and save as **.xsl format.**

**Sample XSLT Sample Output**

**2.8 Requirements**

* For Payment automation, Company moves to Bank,they create an **FSD(Spec)**
* and send it to the company.
* They prepared an **XML and mapping sheet**, based on these etext template will be created and also flat file will be generated.
* Using that flat file bank automates the payment to the suppliers.

|  |  |
| --- | --- |
| **Etext** | **XSL** |
| FSD | FSD |
| XML | XML |
| Mapping sheet | Mapping sheet |

**2.8.1 XML**

* Xpath stands for XML path language.
* Xpath can be used to navigate through elements and attributes in an XML document.
* 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.

**2.8.2 How to generate XML ?**

There are two ways to generate the XML. One is **system generated XML**. Another one is **Customized XML**.

**2.8.2.1 System Generated XML**

* Functional give the payment number to the company.
* Using that payment number company fetch the corresponding datas and also put this below query into the Datamodel.

**Select \* From IBY\_TRXN\_DOCUMENTS**

* Copy the Document column and also check the Document type**(Note:Pick document type=100 data)**.
* Paste into notepad then save as **.xml** Format.

**2.8.2.2 Custom XML**

When we are not able to generate the flat file using system generated XML. Then we move to **custom XML**. Here, we define the levels and structure of the XML. Custom Xml is inserted through Integration or Bursting(Using WCC Delivery channel) to the oracle.

**2.8.3 Where do we get the Mapping sheet?**

Mapping sheet contains the path,field,length,field names etc. Ask the mapping sheet from the bank. Otherwise, company will create the mapping sheet based on the FSD with the help of Functional.

1. **Payment Types**

There are two types of payments

* Outbound Transaction
* Inbound Transaction

**3.1 Outbound Transaction( FUNDS DISBURSEMENTS )**

When the Bank makes the payment to the supplier, the bank contains the **Payment Details(**like the supplier names,and payment details etc.).

**3.2 Inbound Transaction ( FUNDS CAPTURE )**

When the Payment is complete, the Bank sends the **Bank Statement** (Amount debited and credited from company’s account)to the company.

1. **Transmission Configuration Setup**

To make a transmission between Oracle HCM Cloud and your own servers through Oracle WebCenter Content server , the Transmission Configuration must be setup.

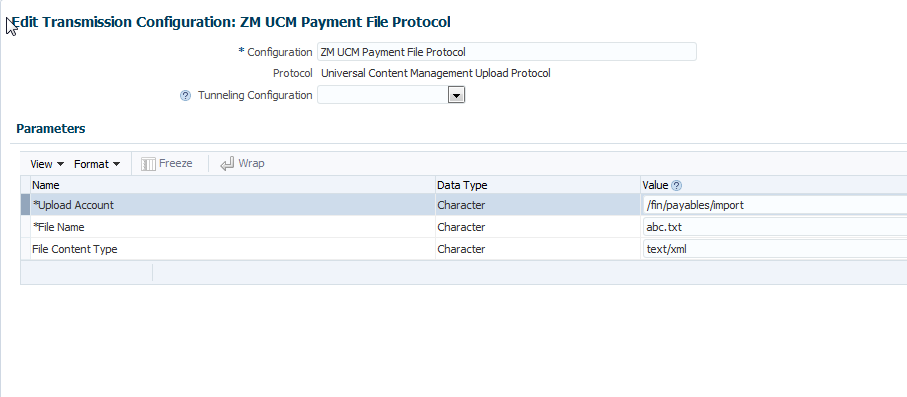
* After that, a connection made between the two for transmission.

**4.1 Create Transmission Configuration**

**Navigation:** Setup and Maintenance--->Manage Transmission Configurations

Create a new transmission configuration by selecting Universal Content Management Upload Protocol in select protocol LOV and click create. Enter following details:

* Upload Account: /fin/payables/import
* FileName:  Specify a static file name (abc.txt)
* File Content Type: text/xml



1. **Payment Process profile**

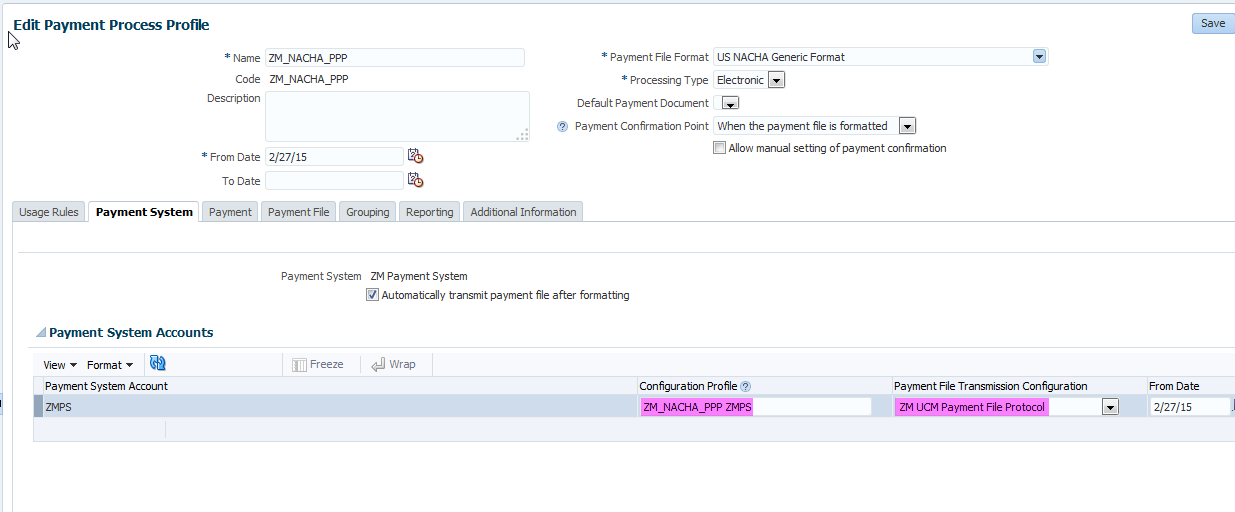
A Payment Process Profile (PPP) is a payment attribute assigned to documents payable (selected invoices & memos in a Payment Process Request), which specifies how Oracle Payments performs processing. PPPs are comprised of several types of payment processing information, including specifications for formatting and transmission.

**5.1 Create/Update Payment Process Profile (PPP)**

**Navigation:** Setup and Maintenance--->Manage Payment Process Profile

Create a new PPP or update an existing one to add payment system and transmission details. Perform the following setups related to transmission:

* Under Payment System Tab, select and add the payment system created earlier. This will add a record for the payment system account associated with this payment system.
* Under Payment File Transmission Configuration, select the transmission configuration we created in previous step.
* If you wish to automatically trigger the payment file transmission after formatting phase, check the option "Automatically transmit payment file after formatting"



At this point, we can submit the payment process profile to select installments, process payments and create formatted payment file. This payment file is then transmitted to destination specified in the transmission configuration.

Payment files saved under UCM server can be accessed using Fusion Applications File Import and Export functionality.

1. **Keys**

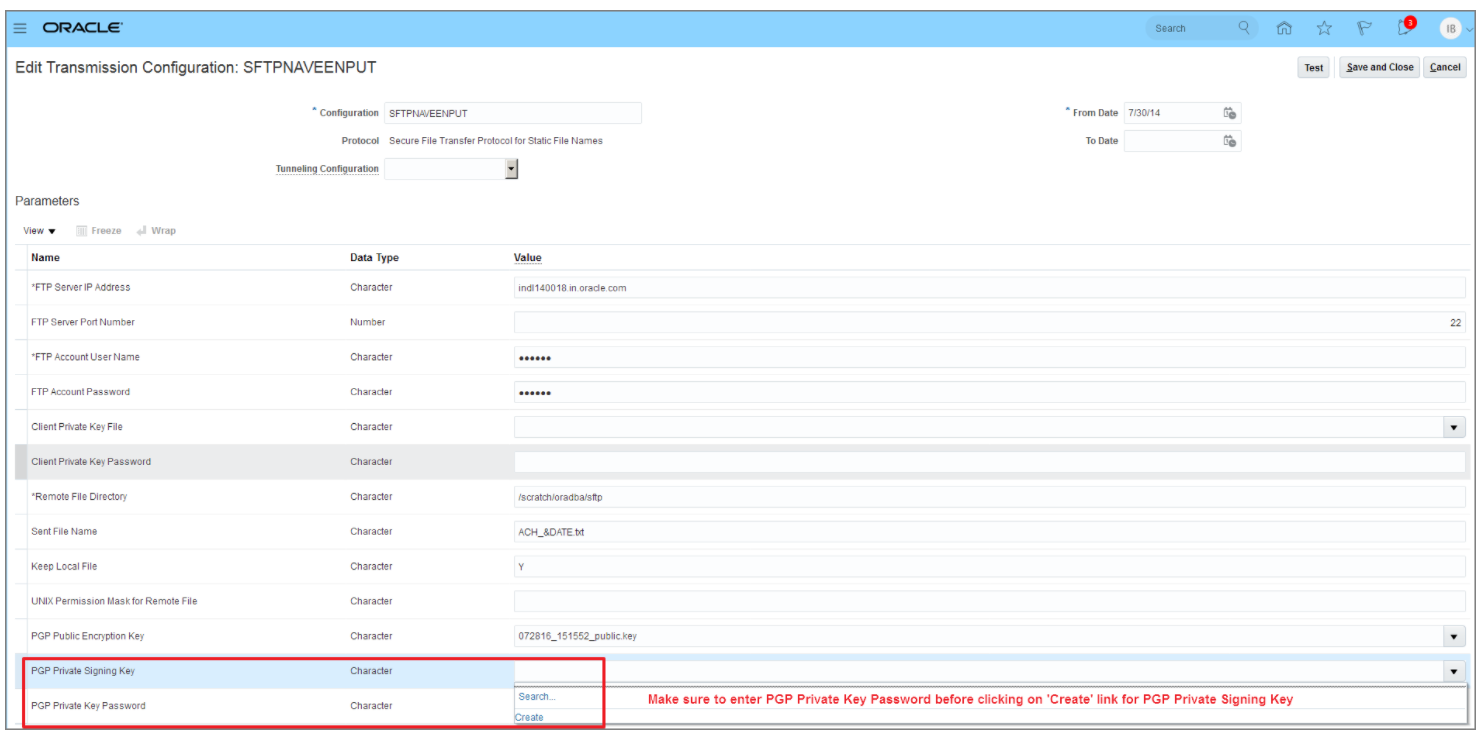
**6.1 PGP Key**

* For secure transmission : PGP(Pretty Good Privacy) keys are used.
* PGP keys are generated to encrypt and decrypt the files in the specific format.
* Receiver creates the PGP Keys (Private & Public Key).

**6.2 Signing Key**

* Sender creates the Signing Key (Private & Public Key).
* **Integrity :** Detects whether that file or message has been modified.
* **Authentication :** Verify cryptographically the person who signed a given message.
* **Non - Repudiation :** Prevent re-claim

### **6.3 How to upload and generate the PGP & Signing keys? (Outbound)**



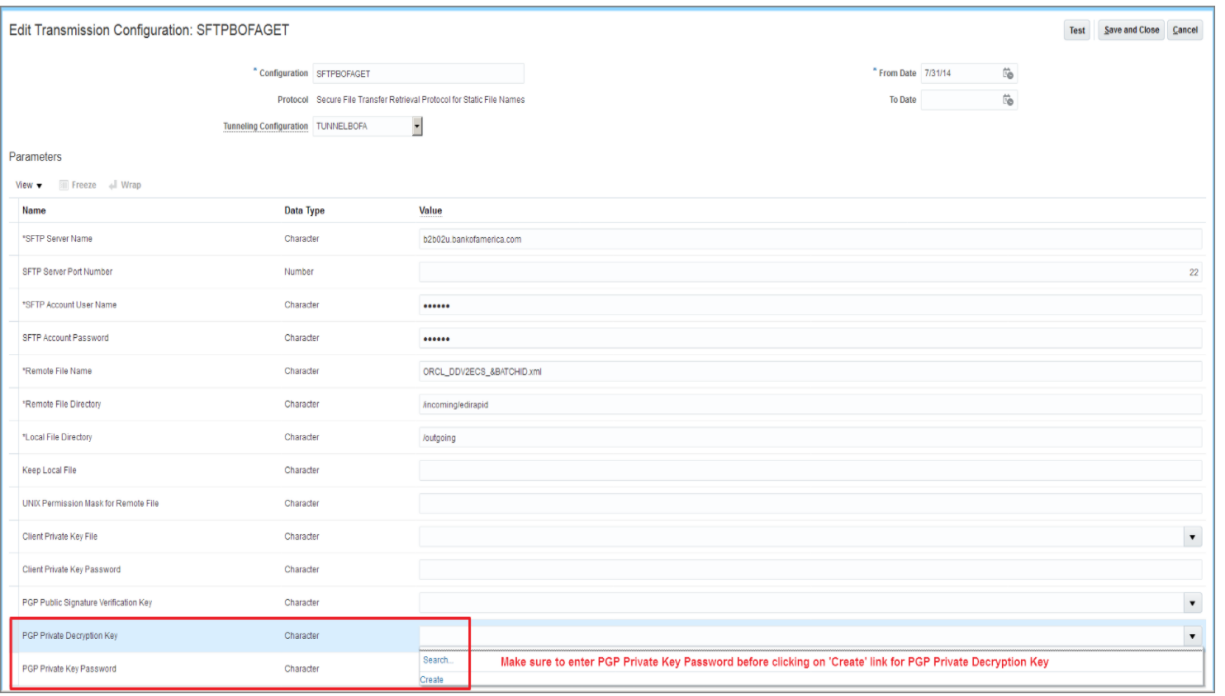
### **6.3.1 PGP Public Encryption Key**

* Bank given key which will be used for encrypting the outbound payment file.
* Once you upload the bank provided encryption public key file via UCM, you can select the same here.
* Once you select the file, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'.

### **6.3.2 PGP Private Signing Key**

* Key generated by you to digitally sign the outbound payment file.
* You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration.
* It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank.
* The bank will use your public key file to verify the digital signature for the payment files which you will be transmitting to bank.

### **6.4 How to upload and generate the PGP & Signing keys? (Inbound)**



### **6.4.1 PGP Private Decryption Key**

* Key generated by you to decrypt the inbound encrypted file.
* You can select the **Create** option from the LOV which will automatically generate the key and link the private key with your transmission configuration.
* It will also generate a public key file which you can download from UCM (File Import and Export) and share with your bank.
* The bank will use your public key file to encrypt the acknowledgement / bank statement file.

### 6.4.2 PGP Public Signature Verification Key

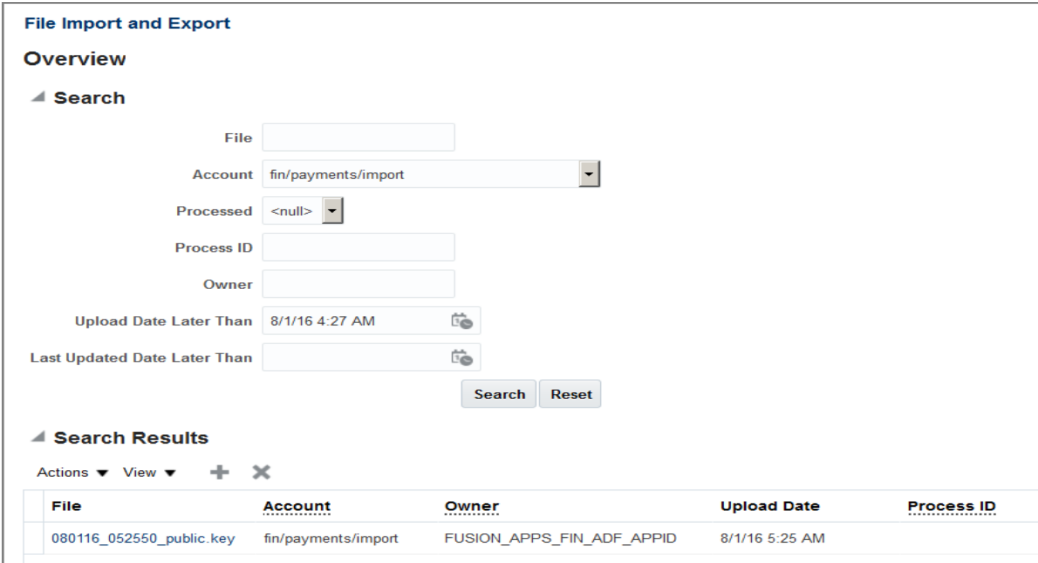
* Bank given key which will be used for validating the digital signature of inbound acknowledgement file or bank statement.
* Once you upload the bank provided signature verification public key via UCM, you can select the same here.
* Once you select the file here, it will be imported automatically and you do not need to manually submit the 'Import Security Credential Job'.

### **6.5 Steps to Upload Bank Given Public Key File**

* Rename the bank-provided key file by including ‘\_public.key’ as a suffix. Also, make sure the key file name doesn’t have any special characters other than an underscore (\_).
* Navigate to File Import and Export.
* Upload the bank given key file in account ‘fin/payments/import’.
* Navigate to your desired transmission configuration in the transmission configuration page and select the uploaded key file from LOV in the related parameter. The key name in the LOV will be the same which you uploaded in UCM.
* Once You select the key and save the configuration, the key will be automatically imported in Fusion.

### **6.6 Steps to Download Fusion Generated Public Key File**

* Select the **Create** option in the transmission configuration for the key related parameter.
* Navigate to File Import and Export.
* Search the generated key file using account ‘fin/payments/import’.
* Download the key file which will have similar name as private key file generated and attached in the transmission configuration.



**6.7 Alternate way to create PGP Key (Without oracle)**

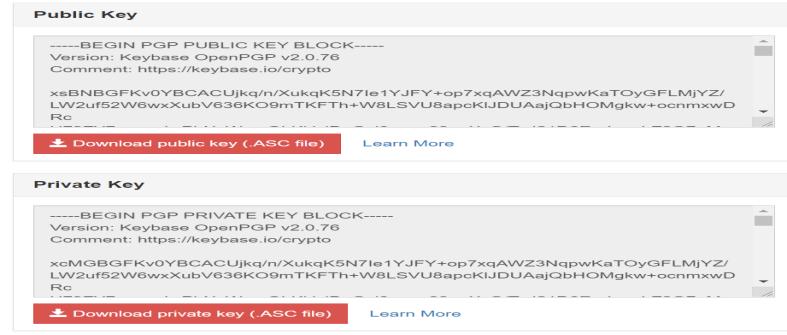
**Step1:**

Choose the algorithm, key size, expiry duration and give the passphrase then generate the keys.

|  |  |
| --- | --- |
|  |  |

**Step2:**

Downloads the private and public key





1. **Extensions**

**7.1 Extension When Download:**

* fusion-key\_pub.asc
* fusion-key\_priv.asc

**7.2 Extension When Upload to UCM:**

* fusion-key\_public.key
* fusion-key\_secret.key

**7.3 Key length** - 1024 or 2048

**8. Algorithms**

**8.1 RSA**

* Rivest-Shamir-Adleman
* Public key cryptosystem widely used for secure data transmission.

**8.2 CIPHER**

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

**Cipher Family:** AES-128, Blowfish

**8.2.1 AES-128**

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

**8.2.2 BLOWFISH**

* Blowfish encryption with a 64-bit block size and a variable-length key size from 32 bits to 128 bits.

**8.3 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.

**Hash:** SHA-1, SHA-256, SHA-224, SHA-512, MD5, SHA-384

**8.3.1 SHA - Secure Hashing Algorithm**

* Cryptographic hash function used to produce a hash value from the input file or message.

**8.4 Compression:**

Compression algorithms are normally used to reduce the size of a file without removing information.

Family - bzip2, zlib, .zip

**9. Overall Flow of Outbound & Inbound**

|  |  |
| --- | --- |
| **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. |