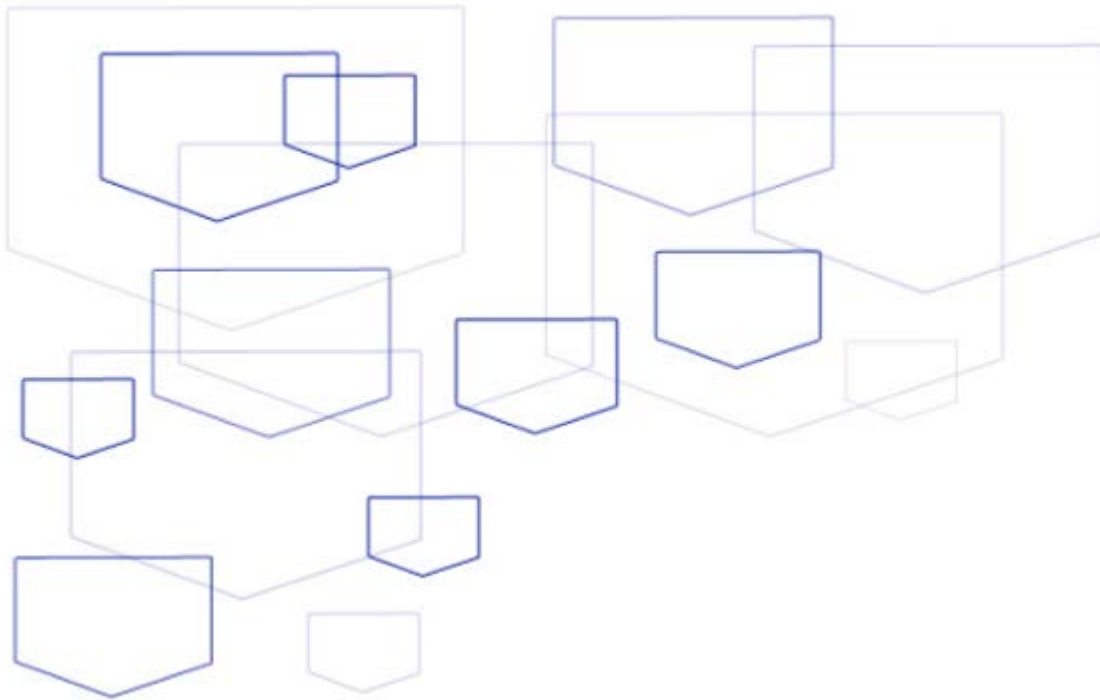


U.S. Bank E-Payment Service

Web Session Transfer



All of **us** serving you™

The information contained in this document is confidential and is not to be disclosed to any third party, nor disseminated, used, quoted or otherwise referred to for any purpose other than in connection with the internal evaluation of this product.

© 2015 U.S. Bank National Association



Contents

| | |
|--|-----------|
| Overview..... | 3 |
| Chapter 1: Web Session Transfer | 4 |
| 1.0 Session Transfer HTML | 4 |
| 1.1 Payment Session Transfer Example..... | 4 |
| 1.2 Payment Inquiry Session Transfer Example..... | 5 |
| 1.3 Session Data | 6 |
| Chapter 2: Session Transfer Encryption | 12 |
| 2.0 How to Obtain the Encryption Key | 12 |
| 2.1 Generate a New AES Encryption Key | 12 |
| 2.2 Implementation..... | 13 |
| 2.3 Payer Single Sign-On..... | 14 |
| 2.4 Real-Time Authorization and Registration | 15 |
| 2.5 Disallow Login | 15 |
| 2.6 Authorized Products..... | 15 |
| Chapter 3: Return Session Transfer Data | 16 |
| 3.0 Distribution | 16 |
| 3.1 File Creation/Processing..... | 16 |
| 3.2 Return Session Transfer Code Sample | 16 |
| 3.3 Return Session (Encrypted) Data | 18 |
| Appendix A: Test and Sample Data | 24 |
| Appendix B: Java Code Examples..... | 26 |
| Sample AES Encryption and Decryption class..... | 26 |
| Appendix C: .Net Example..... | 28 |

Deposit products offered by U.S. Bank National Association. Member FDIC.
©2015 U.S. Bank National Association. All rights reserved.

MMWR-69766 (06/15)

Overview

In order for payers to access the U.S. Bank E-Payment Service website, the biller must provide the appropriate data parameters to U.S. Bank via an Internet session transfer. Payers cannot bookmark any of the pages in the E-Payment Service site to return to at a later time or date. The payer must always be passed to E-Payment Service via the Session Transfer. The Session Transfer parameters are defined during the implementation process, with customized session transfer HTML provided to the biller during implementation of the biller's application. A Session Transfer must always include the Biller Group ID and Biller ID of the biller, and may include Product Code, Payer ID, Amount Due, Due Date or any number of product parameters, including payer contact information, payment method and payment type, depending upon the options selected during biller implementation.

Format

Session variables will be sent using the HTTP Post method, using standard SSL encryption (128-bit key). A two-way encryption algorithm may be used to encrypt all session transfer variables by the biller (using the biller's unique key) before being sent to E-Payment Service. Encrypting the session transfer is optional for all billers except those sending usernames and passwords (that is, using auto-sign on). For billers implementing auto-sign on, encryption is required. Likewise, enabling the Return Session Transfer also requires encryption.

Chapter 1: Web Session Transfer

1.0 Session Transfer HTML

Every time a payer clicks into E-Payment Service, the biller's site will send session transfer data to control how the E-Payment Service application behaves for this payer. A payer may arrive at E-Payment Service with the intention of:

- Making a payment
- Inquiring about a payment
- Managing accounts

1.1 Payment Session Transfer Example

The following is an example of the Session Transfer HTML a biller may use to pass a payer to E-Payment Service to make a payment:

```
<HTML>

<body>

<form action="https://epayment.epymtservice.com/epay.jhtml" method="post">

<input type="hidden" name="billerPayorId" value="12345">

<input type="hidden" name="productCode" value="ProductOne">

<input type="hidden" name="amountDue" value="nnn.nn">

<input type="hidden" name="dueDate" value="yyyy-mm-dd">

<input type="hidden" name="parameter" value="parameter value">

<input type="hidden" name="billerId" value="AAA">

<input type="hidden" name="billerGroupId" value="BBB">

<input type="hidden" name="disallowLogin" value="Y">

<input type="hidden" name="lastName" value="Payer">

<input type="hidden" name="firstName" value="Joe">

<input type="hidden" name="streetAddress1" value="5000 California Blvd.">

<input type="hidden" name="streetAddress2" value="Ste. 1000">

<input type="hidden" name="city" value="Chicago">

<input type="hidden" name="companyName" value="USBank">

<input type="hidden" name="stateRegion" value="IL">

<input type="hidden" name="zipPostalcode" value="12345">

<input type="hidden" name="countryCode" value="US">

<input type="hidden" name="emailAddress" value="joe.payer@aol.com">
```

```
<input type="hidden" name="phoneNumber" value="1234567890">
<input type="hidden" name="paymentMethod" value="Credit Card">
<input type="hidden" name="paymentType" value="Single">
<input type="submit" value="Make Payment">
</form>
</body>
</HTML>
```

1.2 Payment Inquiry Session Transfer Example

Below is an example of the session transfer HTML that can be used to pass a payer to E-Payment Service to inquire about a payment or manage their account:

```
<HTML>
<body>
<form action="https://epayment.epymtservice.com/epay.jhtml" method="post">
<input type="hidden" name="billerId" value="ABC">
<input type="hidden" name="billerGroupId" value="ABC">
<input type="hidden" name="firstName" value="XXXXXX">
<input type="hidden" name="lastName" value="XXXXXX">
<input type="hidden" name="streetAddress1" value="XXXXXX">
<input type="hidden" name="streetAddress2" value="XXXXXX">
<input type="hidden" name="city" value="XXXXXX">
<input type="hidden" name="companyName" value="XXXXXX">
<input type="hidden" name="stateRegion" value="XXXXXX">
<input type="hidden" name="zipPostalcode" value="XXXXXX">
<input type="hidden" name="countryCode" value="XXXXXX">
<input type="hidden" name="emailAddress" value="XXXXXX">
<input type="hidden" name="phoneNumber" value="XXXXXX">
<input type="hidden" name="paymentMethod" value="XXXXXX">
<input type="hidden" name="paymentType" value="XXXXXX">
<input type="submit" value="Payment Inquiry">
</form>
</body>
</HTML>
```

1.3 Session Data

Below are the data layouts of the Web Session Transfer for each field. These data layouts are also used for Encrypted Session Transfers.

| Key | Data Type | Length | Comments | Database Validations |
|---------------|-----------|--------|---|--|
| billerGroupId | Char | 25 | <ul style="list-style-type: none"> Required Field Identifies the biller group to the system | Must be a valid biller group/biller combination found in BILLER where: biller_group_id = BillerGroupID and biller_id = BillerID |
| billerId | Char | 25 | <ul style="list-style-type: none"> Required Field Identifies the biller to the system | Must be a valid biller group/biller combination found in BILLER where: biller_group_id = BillerGroupID and biller_id = BillerID |
| productCode | Char | 25 | <ul style="list-style-type: none"> Required if user is trying to make a payment for a specific product Identifies the product that the user is paying for | If the biller is trying to use the authorized products feature, the value must be 'AUTHORIZED_PRODUCT' |
| billerPayorId | Char | 25 | Required if user is trying to make a payment and if configured in Biller Setup | Used to authenticate the user for ACH payments. |
| amountDue | Decimal | 16, 2 | <ul style="list-style-type: none"> Required if user is trying to make a payment and if product defined it as required during Biller Setup If passed, this is the amount due for a given product code Must be all numeric May contain a decimal point, but may only contain 2 digits after the decimal point Not valid if the productCode equals 'AUTHORIZED_PRODUCT' | <ul style="list-style-type: none"> Field length cannot exceed 8 digits to the left of the decimal point. Is_amount_due_required must equal 'Y' for the biller group/ biller/ product code combination in the biller_product table. |
| dueDate | Date | 10 | <ul style="list-style-type: none"> Required if user is trying to make a payment and if product defined it as required during Biller Setup If passed, this is the due date for a given product code Must be in the format of 'YYYY-MM-DD' and be a valid date Not valid if the productCode | Is_due_date_provided must equal 'Y' for the biller group/ biller/ product code combination in the biller_product table. |

| Key | Data Type | Length | Comments | Database Validations |
|---------------|-----------|--------|---|--|
| | | | equals 'AUTHORIZED_PRODUCT' | |
| paymentType | N/A | N/A | <ul style="list-style-type: none"> For single payments the key value must be 'Single' For recurring payments, the key value must be 'Recur' | If this is included, it will be the only type available for the payer. |
| paymentMethod | N/A | N/A | <ul style="list-style-type: none"> For ACH payments, the payment type must be 'ACH' For credit card payments, the payment type must be 'Credit Card' | If this is included, it will be the only method available for the payer. |
| billerUserId | Char | 16 | Required if password is sent | Identifies the user for auto-sign on |
| password | Char | 30 | Required if billerUserId is sent | User password for auto-sign on |
| lastName | Char | 30 | <ul style="list-style-type: none"> Must have at least one letter Cannot be all spaces Characters allowed: <ul style="list-style-type: none"> Letters Dashes Spaces Periods Commas Parentheses | |
| firstName | Char | 30 | <ul style="list-style-type: none"> Must have at least one letter Cannot be all spaces Characters allowed: <ul style="list-style-type: none"> Letters Dashes Spaces Periods Commas | |

| Key | Data Type | Length | Comments | Database Validations |
|----------------|-----------|--------|---|---|
| | | | – Parentheses | |
| emailAddress | Char | 256 | <ul style="list-style-type: none"> • Only one '@'. No limits on '.' As long as there is one after the '@'. • Characters not allowed: { } < > & / \ ' ` = ^ , | If a value is present, it must have the minimum format of x@x.x where 'x' is at least one alphanumeric value, including '-', '_' or '.', + * \$ % ! ? ~. |
| streetAddress1 | Char | 50 | <ul style="list-style-type: none"> • Required if City, State, ZipPostalCode, and Country are provided • Must have at least one letter • Cannot equal a space • Alphanumeric, special and control characters are allowed | |
| streetAddress2 | Char | 50 | Alphanumeric, special and control characters are allowed | Can only contain a value if StreetAddress1, City, State, ZipPostalCode, and Country all contain values |
| city | Char | 30 | <ul style="list-style-type: none"> • Required unless "Use the Credit Card Billing Address Just Entered" is selected • If a value is provided: <ul style="list-style-type: none"> – Must have at least one letter – Cannot equal a space – Alphanumeric, special and control characters are allowed | If StreetAddress1, State, ZipPostalCode, and Country are provided, then this field is required. |
| stateRegion | Char | 2 | <ul style="list-style-type: none"> • This version is used when Biller Group does not accept International Addresses: • Required unless "Use the Credit Card Billing Address Just Entered" is selected • Required if City, StreetAddress1, and ZipPostalCode/ZipCode5 are provided • If a value is provided: <ul style="list-style-type: none"> – Must have at least one | <ul style="list-style-type: none"> • Must be a valid code in the STATE_CODE table. • Codes may be found in the U.S. Bank E-Payment Service Technical Specifications |

| Key | Data Type | Length | Comments | Database Validations |
|---------------|-----------|--------|--|---|
| | | | letter – Cannot equal a space | |
| stateRegion | Char | 40 | This version is used when Biller Group does accept International Addresses: <ul style="list-style-type: none"> • Required unless “Use the Credit Card Billing Address Just Entered” is selected • Required if City, StreetAddress1, ZipPostalCode/ZipCode5, and Country are provided • Allow only numbers, letters, hyphens, single spaces, apostrophes, commas, periods and parentheses | <ul style="list-style-type: none"> • If Country is US, must be a code in the STATE_CODE table. • Codes may be found in the U.S. Bank E-Payment Service Technical Specifications |
| zipPostalcode | Char | 12 | <ul style="list-style-type: none"> • If Country is US, must follow standard US zip format and be five digits only. • If Country is not US, then only allow numbers, letters, hyphens, single spaces and periods. Cannot be single space/all spaces. • If StreetAddress1, City, State and Country are provided then this field is required | N/A |
| countryCode | Char | 2 | If StreetAddress1, City, State and Zip 5/ZipPostalCode are provided then this field is required | <ul style="list-style-type: none"> • Must be a valid code in the COUNTRY table • Codes may be found in the U.S. Bank E-Payment Service Technical Specifications |
| companyName | Char | 30 | <ul style="list-style-type: none"> • If value is entered, must have at least one letter; cannot equal a space • Alphanumeric, special and control characters are allowed | N/A |

| Key | Data Type | Length | Comments | Database Validations |
|---|-----------|--------|--|---|
| phoneNumber | Char | 3,3,4 | <ul style="list-style-type: none"> Used when Biller Group does not accept International Addresses Numeric Only Phone number must be 10 digits Cannot be all spaces | * Character format: <ul style="list-style-type: none"> 3, 3, 4 <ul style="list-style-type: none"> Ex: 1234567890 Ex. 3334445555 area code, prefix, extension |
| phoneNumber | Char | 20 | <ul style="list-style-type: none"> Used when Biller Group does accept International Addresses Cannot be all spaces Must be 10 digits If CountryCode is "US" If CountryCode is not "US", Alphanumeric and special characters (numbers, hyphens, single spaces, parentheses and periods) are allowed. | |
| returnValue | Char | 100 | <ul style="list-style-type: none"> This field is optional This field is only passed if the return session transfer is enabled Returns payer back to the site specified in the Return Session | Encrypted Session Transfers only. |
| returnURL | Char | 256 | <ul style="list-style-type: none"> Biller's address URL for Return Session must be a valid URL This field is optional This field is only passed if the return session transfer is enabled Returns payer back to the site specified in the Return Session The protocol must be HTTPS | Encrypted Session Transfers only. Required for Return Session Transfer. |
| Product Parameter Name (Key name is the actual parameter name entered during biller setup) | Char | 26 | <ul style="list-style-type: none"> Required if product parameter specified as required (either passed via session transfer and/or pre-registration file) There can be 1-n of the following 2 Product Parameter fields as | Not valid if the productCode equals 'AUTHORIZED_PRODUCTS' |

| Key | Data Type | Length | Comments | Database Validations |
|--|-----------|--------|---|---|
| | | | determined from Biller Setup. | |
| Product Parameter Value (Key name is the actual parameter name entered during biller setup) | Char | 80 | <ul style="list-style-type: none"> Required if product parameter specified as required (either passed via session transfer and/or pre-registration file) There can be 1-n of the following two Product Parameter fields as determined from Biller Setup | Not valid if the productCode equals 'AUTHORIZED_PRODUCTS' |
| disallowLogin | Char | 1 | If equals 'Y', it identifies the user is unregistered and should skip the welcome screen. | Note: The 'Y' is case sensitive. |

Chapter 2: Session Transfer Encryption

The inbound Web Session Transfer can be passed to E-Payment Service either encrypted or unencrypted. E-Payment Service requires session transfer encryption if the biller is a) utilizing the “Auto-Login” functionality (see E-Payment Service User Guide for more detailed information on the Auto-Login component of E-Payment Service), or b) is enabled to receive a Return Session Transfer from E-Payment Service.

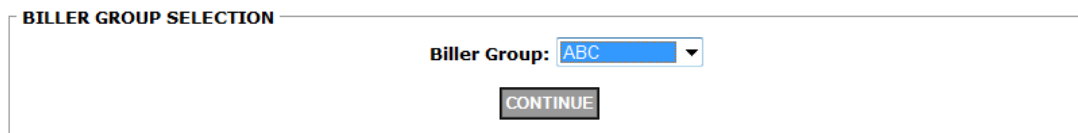
2.0 How to Obtain the Encryption Key

Security Officers may obtain the encryption key from the Main Menu of the E-Payment Service application by selecting “Manage Encryption Key” in the Manage Keys/Token section



Should you have more than one Biller ID, you will need to select the appropriate one from the following screen:

Biller Group Select

A screenshot of a web application form titled "BILLER GROUP SELECTION". The form contains a dropdown menu labeled "Biller Group:" with the value "ABC" selected. Below the dropdown is a "CONTINUE" button.

Once selected, any previously generated encryption keys will be displayed.

2.1 Generate a New AES Encryption Key

E-Payment Service uses AES encryption (Advanced Encryption Standard) for session transfers.

Note: Prior to implementing AES encryption, Blowfish encryption was used. The ability to generate a blowfish key is also available on the site. Blowfish encryption will eventually be phased out and we recommend that any new keys generated should be done utilizing the new and more secure AES process.

To obtain a new key, click **Generate AES Key**.

Manage Encryption Key

BILLER GROUP: ABC

BILLER: DEF

The encryption key is used to generate encrypted session transfers.

Blowfish is a legacy encryption algorithm. Please use AES for all new session transfers.

List of Existing Encryption Keys:

| Encryption Key | Algorithm Name | Initialization Vector | Key Bit Length | Action |
|---------------------------|----------------|-----------------------|----------------|--------|
| Nothing found to display. | | | | |

Your new encryption key will be displayed.

Manage Encryption Key

BILLER GROUP: ABC

BILLER: DEF

The encryption key is used to generate encrypted session transfers.

Blowfish is a legacy encryption algorithm. Please use AES for all new session transfers.

List of Existing Encryption Keys:

| Encryption Key | Algorithm Name | Initialization Vector | Key Bit Length | Action |
|--|----------------------|-----------------------|----------------|------------------------|
| +ZKU3HghMpt/HtVizD/2ogg79rieEDf33aepfxJWHQo= | AES/CBC/PKCS5Padding | tppf5T8rgP0H9ACY | 256 | Delete |

New key has been generated.
New key is: +ZKU3HghMpt/HtVizD/2ogg79rieEDf33aepfxJWHQo=

2.2 Implementation

The encrypted session transfer contains two unencrypted elements: billerGroupID and billerID, and one encrypted element: session. The process to create the encrypted element is described below. All other parameters are encrypted.

Use the AES key, along with the supplied Initialization Vector to perform the encryption. These values are available from the Administrative site using instructions above or can be provided by your U.S. Bank Technical Implementation Consultant. The encryption keys displayed on the admin site have been Base-64 encoded. Here are the other details of the AES scheme E-Payment Service utilizes:

- Algorithm: AES
- Cipher Encryption Mode: Cipher Block Chaining (CBC)
- Padding Scheme: PKCS5Padding
- Key Length: 256

To perform encryption:

1. Construct your session transfer string. This consists of the parameters you need to send to E-Payment Service from this list above. These should be pipe delimited (see Example 2 in **Appendix A** for

additional assistance):

```
productCode=Investments|amountDue=12.01|dueDate=2012-02-17
```

Note: Please refer back to your Biller Setup for required parameters for Session Transfer.

2. The keys displayed on the E-Payment administrative site are base-64 encoded. Decode your AES key using a Base-64 decoder.
3. Encrypt your session transfer string from Step 1 with decoded AES key and the supplied Initialization Vector.

Note: E-Payment service performs all encryption/decryption using an encryption mode of CBC (Cipher Block Chaining) and a padding scheme of PKCS5Padding. Consult your vendor documentation for how to perform AES encryption and decryption in your environment.

4. Encode the encrypted session transfer string created in Step 3 using a Base-64 encoder.
5. Once complete, create an HTML page like the one below. Place the value from Step 4 into the designated location. Clicking this button should successfully transfer you to E-Payment Service.

```
<HTML>
<body>
<form action="https://epayment.epymtservice.com/epay.jhtml" method="post">
<input type="hidden" name="session" value="value from Step 4 ">
<input type="hidden" name="billerId" value="DEF">
<input type="hidden" name="billerGroupId" value="ABC">
<input type="submit" value="Make Payment">
</form>
</body>
</HTML>
```

2.3 Payer Single Sign-On

For Billers that utilize a login process on their website, E-Payment Service provides a single sign-on option to make the user transfer to the U.S. Bank site seamless. In order to utilize single sign-on, Billers must first register their customers with E-Payment Service. This can be accomplished by utilizing the pre-registration file or by sending an API message. Through these, the Biller provides information about their customers that includes two pieces of data for authentication (User ID and Password) and basic demographic information.

Below are some important things to keep in mind when considering the Single Sign-On option;

- The data utilized for User ID must be unique across your customer base. Often Billers utilize the customers' account numbers for the User ID field.
- The password does not have to be unique across your customer base. Often Billers utilize information such as last four numbers of SSN, zip code or home phone number as the Password field.
- In most cases, the User ID and Password are used simply for authentication purposes and do not need to be values known by the customer.
- The User ID and Password provided to E-Payment Service via the pre-registration file are not required to be the User ID and Password the customer uses to login to the Biller site. In fact, if you

allow your customers to change the User ID or Password they use on your site, you will not want to use these values for E-Payment Service.

- The User ID should not be a value that can be passed to another customer in the future. For example, if you are accepting payments for utilities and the account number is tied to the property and not to the customer, the account number should not be used as the User ID.

2.4 Real-Time Authorization and Registration

Billers can choose whether or not to use the E-Payment Service real-time authorization and registration (RTAR) feature. This feature allows billers to authorize payers real-time when they attempt to log into the E-Payment Service customer payment site or the Integrated Voice Response system. Billers can send new user information to register the user or update existing information when payers log into E-Payment Service. For more information on this feature see *E-Payment Service Technical Specifications*.

2.5 Disallow Login

For Billers that do not utilize self-registration or pre-registration, E-Payment Service provides the option to skip the initial login/registration screen. To skip the initial login/registration screen, the Biller would simply include the following code in the session transfer.

Note: this feature should only be used for users transferring to E-Payment Service for the purpose of making a payment.

```
<input type="hidden" name="disallowLogin" value="Y">
```

2.6 Authorized Products

For Billers that utilize the pre-registration file, E-Payment Service provides the option to manage customers' eligibility to pay through the use of the "Authorized Product" record in the pre-registration file. In order for E-Payment Service to confirm eligibility, the product code in the session transfer must be set to AUTHORIZED_PRODUCT and not the actual product code setup for an application. For more information on this feature see *E-Payment Service Technical Specifications*.

```
<input type="hidden" name="productCode" value="AUTHORIZED_PRODUCT">
```

Chapter 3: Return Session Transfer Data

Billers may also elect to have payer and payment data returned to their website in the form of a Return Session Transfer after the payer has affected a transaction in E-Payment Service. The payer must select the <biller defined button name> on the E-Payment Service Payment Confirmation page in order to initiate this Session Transfer of data. This data will only reflect the last payment action taken by the payer and the data will be returned after a payer initiates one of the following actions:

- Makes a Payment (Single or Recurring Payment Type)
- Edits a Payment (Single or Recurring Payment Type)

In order for billers to receive this Return Session Transfer from E-Payment Service, the inbound Session Transfer must be encrypted (see section above entitled Session Transfer Encryption) using the encryption components provided by E-Payment Service, and a Return URL Address (HTTPS protocol only) must be passed in the inbound Session Transfer so that E-Payment Service knows exactly where to direct the payer after a transaction has been affected. If a return URL address is not provided in the inbound Web Session Transfer, then billers will not receive the Return Session Transfer. In addition, billers may also pass a unique Return Value to be returned in the Session Transfer along with payer and payment information to more easily identify the origin of the Return Session Transfer.

3.0 Distribution

The Return Session Transfer is sent to the URL address that is taken from the incoming session transfer. If there is no Return Session Transfer URL address, then the Biller is not set up to receive the return session or the Biller did not send a valid URL.

3.1 File Creation/Processing

When the payor selects the button on the Payment Confirmation page, the system will send the return session data as the Payor is redirected to the passed URL. The return session data will reflect only the last payment action by the Payor.

3.2 Return Session Transfer Code Sample

Use the instructions below to process a return session transfer from E-Payment Service. The return session transfer from E-Payment Service will contain an encrypted, Base-64 encoded string that will need to be processed by the receiving application.

1. Using a Base-64 Decoder, decode the encoded AES Key from the Administrative site.
2. Get the return session transfer data (Example 3 in Appendix).
3. Using a Base-64 Decoder, decode the string retrieved from Step 2.
4. Using the decoded key from step 1, decrypt the string you generated in Step 3

The raw string will contain E-Pay session parameters separated via "ampersand":

```
TransactionConfirmationID=ABCABC000000001&userID=user123&BillerProductCode=productABC&PaymentMethod=CC
```


Note: E-Payment service performs all encryption/decryption using an encryption mode of CBC (Cipher Block Chaining) and a padding scheme of PKCS5Padding. Consult your vendor documentation for how to perform AES encryption and decryption in your environment.

3.3 Return Session (Encrypted) Data

Below are the data layouts of the Return Session Transfer for each Payment Type:

Single Payment Types

| Single Payment Type | Comments | Required |
|---------------------------|---|----------|
| TransactionConfirmationID | Transaction Confirmation ID of the affected transaction | Yes |
| UserID | <ul style="list-style-type: none"> E-Payment Service UserID of the Payer Data field will be empty if Payer is not registered with the E-Payment Service | Yes |
| BillProductCode | Identifies the product code the Payer is making a payment for | Yes |
| PaymentMethod | Valid Values: <ul style="list-style-type: none"> ACH CC ATM | Yes |
| PaymentAmount | xxx.xx | Yes |
| ConvenienceFee | <ul style="list-style-type: none"> xxx.xx Data field will be empty if biller does not charge convenience fees on transactions | Yes |
| PaymentEffectiveDate | YYYYMMDD | Yes |
| AmountDue | <ul style="list-style-type: none"> xxx.xx Data field will be empty if biller does not provide E-Payment Service with an Amount Due for the Payer | Yes |
| DueDate | <ul style="list-style-type: none"> YYYYMMDD Data field will be empty if biller does not provide E-Payment Service with a Due Date for the Payer | Yes |
| TransactionMode | Valid Values: <ul style="list-style-type: none"> MakePayment EditPayment | Yes |
| ReplacesConfirmationID | Value present for EDIT only; otherwise data field will be empty | Yes |
| InitiationDateTime | YYYYMMDD:HHMMSS | Yes |

| Single Payment Type | Comments | Required |
|---------------------|--|------------------------------|
| CreditCardNumber | <ul style="list-style-type: none"> Only last four digits are provided in clear text Data field will be empty if Payment Method is ACH | Yes |
| CreditCardType | <ul style="list-style-type: none"> If PaymentType is "CC" If Credit Card is American Express – "AMEX" If Credit Card is Discover – "DISC" If Credit Card is Visa – "VISA" If Credit Card is Mastercard – "MC" If Card is an ATM debit card – "ATM Debit:" Data field will be empty if Payment Method is ACH | Yes |
| CreditCardExpDate | <ul style="list-style-type: none"> YYYYMM Data field will be empty if Payment Method is ACH | Yes if Payment Method = 'CC' |
| AccountNumber | <ul style="list-style-type: none"> Only last four digits are provided in clear text Data field will be empty if Payment Method is CC | Yes |
| ProductDescription | <ul style="list-style-type: none"> Identifies the product the Payer is making a payment for | Yes |
| BillerBusinessDate | <ul style="list-style-type: none"> YYYYMMDD Data field will be empty if Biller Business Date is not enabled for biller | Yes |
| IsPayorRegistered | <ul style="list-style-type: none"> Is Payer registered with the E-Payment Service? "Y" for yes, "N" for no | Yes |
| ReturnValue | Unique Value from the inbound session transfer | No |
| Email | Email address of Payer | No |
| [Parameter Name]* | <ul style="list-style-type: none"> Parameter name passed in from the biller on the inbound session transfer May be more than one | No |
| [Parameter Value]* | <ul style="list-style-type: none"> Parameter value Must match correct name | No |

| Single Payment Type | Comments | Required |
|--|----------|----------|
| * The '&' and '=' characters may be part of this data string | | |

Recurring Payment Types

| Recurring Payment Type | Comments | Required |
|---------------------------|--|----------|
| RecurringReferenceId | The Reference ID assigned by E-Payment Service to the Recurring Payment entry | Yes |
| UserID | The E-Payment Service User ID of the user affecting the transaction | Yes |
| BillProductCode | Identifies the product code the Payer is making a payment for | Yes |
| PaymentMethod | Valid Values: <ul style="list-style-type: none"> • ACH • CC • ATM | Yes |
| PaymentAmount | xxx.xx | Yes |
| ConvenienceFee | <ul style="list-style-type: none"> • xxx.xx • Data field will be empty if biller does not charge convenience fees on transactions | Yes |
| FirstScheduledPaymentDate | YYYYMMDD | Yes |
| AmountDue | <ul style="list-style-type: none"> • xxx.xx • Data field will be empty if biller does not provide E-Payment Service with an Amount Due for the Payer | Yes |
| OriginalDueDate | <ul style="list-style-type: none"> • YYYYMMDD • Data field will be empty if biller does not provide E-Payment Service with a Due Date for the Payer | Yes |
| TransactionMode | Valid Values: <ul style="list-style-type: none"> • StartRecurring • EditRecurring | Yes |

| Recurring Payment Type | Comments | Required |
|-------------------------|--|----------|
| Frequency | <ul style="list-style-type: none"> • Payer selected frequency of the Recurring Payment • Valid Values: <ul style="list-style-type: none"> – WEEKLY – BI_WEEKLY – TWICE_MONTHLY – MONTHLY – BI-MONTHLY – QUARTERLY – DUE_DATE – CUSTOM | Yes |
| DayOfMonthFirstPayment | <ul style="list-style-type: none"> • Data field will be populated if frequency of the Recurring Payment = TWICE_MONTHLY • Otherwise data field will be empty | Yes |
| DayOfMonthSecondPayment | <ul style="list-style-type: none"> • Data field will be populated if frequency of the Recurring Payment = TWICE_MONTHLY • Otherwise data field will be empty | Yes |
| NumberDaysBefore | <ul style="list-style-type: none"> • Data field will be populated if frequency of the Recurring Payment = DUE_DATE; field identifies the number of days before the due date that the payer has elected to pay. • Otherwise data field will be empty | Yes |
| Duration | Valid Values: <ul style="list-style-type: none"> • EndOnDate • NumberPayments • ContinueUntilCanceled | Yes |
| RemainingPayments | Data field will be populated if Duration of recurring Payment = NumberPayments; Otherwise data field will be empty | Yes |
| ExpirationDate | <ul style="list-style-type: none"> • Data field will be populated if Duration of recurring Payment = EndOnDate • YYYYMMDD | Yes |

| Recurring Payment Type | Comments | Required |
|------------------------|--|------------------------------|
| | <ul style="list-style-type: none"> Otherwise data field will be empty | |
| PaymentAmountType | <ul style="list-style-type: none"> Valid Values: Fixed Variable | Yes |
| PaymentUpdateDateTime | <ul style="list-style-type: none"> The date and time that the Recurring Payment was started, stopped or edited YYYYMMDD;HHMMSS | Yes |
| CreditCardNumber | <ul style="list-style-type: none"> Replace numbers with "*", only last four numbers in clear text; Empty if Payment Method is ACH | Yes |
| CreditCardType | <ul style="list-style-type: none"> IF PaymentType is "CC" If Credit Card is American Express – "AMEX" If Credit Card is Discover – "DISC" If Credit Card is Visa – "VISA" If Credit Card is Mastercard – "MC" If Card is an ATM Debit Card – "ATM Debit" Empty if Payment Method is ACH | Yes |
| CreditCardExpDate | <ul style="list-style-type: none"> Empty if Payment Method is ACH Yes if payment method is = "CC" Ignore if Payment Method = "ATM" YYYYMM | Yes if Payment Method = 'CC' |
| AccountNumber | <ul style="list-style-type: none"> Replace numbers with "*", only last 4 numbers in clear text Empty if Payment Method is CC | Yes |
| ProductDescription | Identifies the product the Payer is making a payment for | Yes |
| BillerBusinessDate | <ul style="list-style-type: none"> YYYYMMDD Empty if Biller Business Date | Yes |
| ReturnValue | From the incoming session | No |

| Recurring Payment Type | Comments | Required |
|--|--|----------|
| Email | Email address of Payer | No |
| [Parameter Name]* | <ul style="list-style-type: none"> • Parameter name passed in from the biller on the inbound session transfer • May be more than one | No |
| [Parameter Value]* | Parameter value. Must match correct name | No |
| * The '&' and '=' characters may be part of this data string | | |

Appendix A: Test and Sample Data

The test data below can be used to test the session transfer encryption process. Use this data to test that your application can correctly perform base-64 encoding/decoding, HTTP encoding and AES encryption. When using these examples use this AES key:

Test Encryption Key: TouCz6+LGk71Bpb2knTXUeShDI6+4R/96U1VwrQZLIo=

Test Initialization Vector: R2RpmUCDf9tH828l

Example 1: Base 64 encoding and HTTP encoding

Use the values below to test that your application can encode and decode data as required. This is the first step in implementing encrypted session transfer.

Raw String: EPayment Service

The same string as byte values: [69, 80, 97, 121, 109, 101, 110, 116, 32, 83, 101, 114, 118, 105, 99, 101]

Take either the string or the byte array above and encode it with a Base-64 encoder. You should receive the following result: RVBheW1lbnQgU2VydmljZQ==

Example 2: Session Transfer

Use the values below to test that your application can encrypt data using a sample AES key.

1. Start with the following raw string:

productCode=Investments|amountDue=12.01|dueDate=2012-02-17

2. Use the test encryption key above to encrypt that raw string and then encode the string above with Base-64 Encoder. After both steps you should see the value

TnKhpR0jT781SqYrQjvsgnBC2ff9JCab0xsd9i89gJ/LwMUY0a4zwB5Ban6oOXgY2t53w+KKdMPDowfTMiEWiQ==

Example 3: Return Session Transfer

The same steps are followed on the return session transfer, but in the reverse order. Your application will receive a string that needs to be decoded, then decrypted.

1. Start with the following HTTP Encoded, Base-64 encoded string.

DmrTxQhm2dwrKIWJaZF98YZfdX9TVOov1aifqt%2FmWv7f%2BPMvSWrPo6VdFKusMQyGx3mpG9NLGqQZPTKh8PA51DauKbYatR9UdGRuhAcnwOp7GjSSSlwK3GriI3teRY%2FR%2Bqw9QPQzth41LKHMqijaug%3D%3D

2. HTTP decode the string from Step 1 to receive:

DmrTxQhm2dwrKIWJaZF98YZfdX9TVOov1aifqt/mWv7f+PMvSWrPo6VdFKusMQyGx3mpG
9NLGqQZPTKh8PA51DauKbYatR9UdGRuhAcnwOp7GjSSSlwK3GriI3teRY/R+qw9QPQzth4
1LKHMQijaug==

3. Decode the string from Step 2 using your Base-64 decoder and then use the test encryption key above to decrypt the string and you should end up with the raw string:

TransactionConfirmationID=ABCABC000000001&userID=user123&BillerProductCode=produc
tABC&PaymentMethod=CC

Appendix B: Java Code Examples

The following code examples can be a reference for developing in a Java environment. If you are not working in a Java environment, use these examples as psuedocode in your development environment.

Sample AES Encryption and Decryption class

```
import java.security.spec.AlgorithmParameterSpec;
import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;
import org.apache.commons.codec.binary.Base64;

public class AESTestEncryptorDecryptor {
    private String encryptWithAES() {
        String encodedString = null;
        String sessionTransfer = "productCode=test|amountDue=12.01|dueDate=2012-02-17";
        try {
            byte[] arrayOfByte1 = Base64.decodeBase64("sampleEncryptionKey");
            SecretKeySpec keySpec = new SecretKeySpec(arrayOfByte1, "AES");
            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            AlgorithmParameterSpec algSpec = new IvParameterSpec(
                "initvector".getBytes());
            cipher.init(Cipher.ENCRYPT_MODE, keySpec, algSpec);
            byte[] encryptedData = cipher.doFinal(sessionTransfer.getBytes());
            encodedString = Base64.encodeBase64String(encryptedData);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return encodedString;
    }

    private String decryptWithAES() {
```

```
String str = null;
try {
    byte[] arrayOfByte1 = Base64.decodeBase64("sampleEncryptionKey");
    SecretKeySpec keySpec = new SecretKeySpec(arrayOfByte1, "AES");
    Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
    AlgorithmParameterSpec algSpec = new IvParameterSpec(
        "initvector".getBytes());
    cipher.init(Cipher.DECRYPT_MODE, keySpec, algSpec);
    byte[] text = Base64.decodeBase64("encryptedText");
    str = new String(cipher.doFinal(text));
} catch (Exception e) {
    e.printStackTrace();
}
return str;
}
}
```

Appendix C: .Net Example

The following code examples can be a reference for developing in a .Net environment. If you are not working in a .Net environment, use these examples as psuedocode in your development environment.

```
<% @ Language="C#" trace="true"%>
<% @ Import Namespace="System.Security.Cryptography" %>
<% @ Import Namespace="System.IO" %>

<script runat="server">
    void Page_Load(Object Sender, EventArgs E) {
        string a = "productCode=Investments|amountDue=12.01|dueDate=2012-02-17";
        string z = Encrypt(a);
        Trace.Warn("original=" + a);
        Trace.Warn("encrypted and encoded=" + z);

        //1. encrypt using AES
        //2. base 64 encode
        //3. urlencode
        //Algorithm: AES
        //Cipher Encryption Mode: Cipher Block Chaining (CBC)
        //Padding Scheme: PKCS5Padding
        //Key Length: 256
        //string TestEncryptionKey = "TouCz6+LGk71Bpb2knTXUeShDI6+4R/96U1VwrQZLIo=";
        //string TestInitializationVector = "R2RpmUCDf9tH828l";
    }

    private const string AesKey = @"TouCz6+LGk71Bpb2knTXUeShDI6+4R/96U1VwrQZLIo=";
    private const string AesIV = @"R2RpmUCDf9tH828l";

    public string Encrypt(string text)
```

```
{
    Trace.Warn("key=" + AesKey);
    Trace.Warn("IV=" + AesIV);

    AesCryptoServiceProvider aes = new AesCryptoServiceProvider();
    aes.BlockSize = 128;
    aes.KeySize = 256;
    aes.IV = Encoding.UTF8.GetBytes(AesIV);
    aes.Key = Convert.FromBase64String(AesKey);
    aes.Mode = CipherMode.CBC;
    aes.Padding = PaddingMode.PKCS7;

    // Convert string to byte array
    byte[] src = Encoding.UTF8.GetBytes(text);

    // encryption
    using (ICryptoTransform encrypt = aes.CreateEncryptor())
    {
        byte[] dest = encrypt.TransformFinalBlock(src, 0, src.Length);

        // Convert byte array to Base64 strings
        return Convert.ToBase64String(dest);
    }
}
</script>
```