Global PAYplus

Security

Business Guide

Product Version: 4.6.8
Catalog ID: GPP4.6-00-B45-05-201709

Copyright

Disclaimer

Finastra does not guarantee that any information contained herein is and will remain accurate or that use of the information will ensure correct and faultless operation of the relevant software, services or equipment. This document contains information proprietary to Finastra. Finastra does not undertake mathematical research but only applies mathematical models recognized within the financial industry. Finastra does not guarantee the intrinsic theoretical validity of the calculation models used.
Finastra, its agents, and employees shall not be held liable to or through any user for any loss or damage whatsoever resulting from reliance on the information contained herein or related thereto. The information contained in this document and the general guidance of Finastra staff does not take the place of qualified compliance personnel or legal counsel within your institution.

FINASTRA CANNOT RENDER LEGAL, ACCOUNTING OR OTHER PROFESSIONAL SERVICES TO YOUR INSTITUTION. THE INFORMATION CONTAINED HEREIN IS GENERAL IN NATURE AND DOES NOT CONSTITUTE LEGAL ADVICE OR A LEGAL OPINION. CONSULT YOUR LEGAL COUNSEL FOR LEGAL ADVICE SPECIFIC TO YOUR SITUATION OR CIRCUMSTANCES OR TO ANSWER ANY LEGAL QUESTIONS.

This document is not intended as a substitute for formal education in the regulatory requirements of banking, banking operations, lending, lending operations, or other topics generally applicable to financial institutions. Your financial institution is solely responsible for configuring and using the software or services in a way that meets policies, practices, and laws applicable to your institution, including, without limitation: (1) options and selections made on prompts; (2) entries in the software program; (3) program setup; and (4) documents produced by the software or services. It is the obligation of the customer to ensure that responsible decisions are taken when using Finastra products. Information in this document is subject to change without notice and does not represent a commitment on the part of Finastra.

Feedback

Do you have comments about our guides and online help? Please address any comments and questions to your local Finastra representative.
Need more information? Read more about our products at http://www.finastra.com or contact your local Finastra office at http://www.finastra.com/contact.

Version Control

Version	Date	Summary of Changes
1.0	Aug 2015	Document created
2.0	Nov 2015	Updated for rebranding
3.0	Mar 2016	Added new section, Entitlements
4.0	Jun 2016	Added additional details for WebSphere configuration
4.1	Oct 2016	Added additional configuration details to the WebSphere Configuration to Support SSO section
5.0	Dec 2016	Added new sections for LDAP and PCI DSS

Table of Contents

1 Introduction

Note: This Business Guide has not yet been certified for GPP V4.6; therefore, there may be inaccuracies in this document that may require amendments in the future. For more information, please contact your Finastra Project Manager.

Global PAYplus (GPP) uses integrated security and access control procedures and a mechanism, which are designed to guarantee the integrity and confidentiality of application data and limit access to the application to authorized users.

At this stage this Business Guide includes only information related to Single Sign On (SSO).

1.1 Target Audience

This business guide is intended for system administrators, IT personnel, and GPP users who need to know about security within GPP.

2 Single Sign On (SSO)

2.1 Overview

Single Sign On (SSO) is a user authentication process that permits a user to enter one name and password to access multiple applications. It allows users to be authenticated only once with no further authentications required when accessing any applications protected in the same security federation.

GPP supports the use of SSO authentication using a combination of GPP system parameters, WebSphere configuration, and the use of a GPP URL designed for use with SSO.

When using the SSO mode, two approaches are supported:

- The external authentication system supplies both a valid GPP user ID and a valid GPP role (which is defined as the user’s role in GPP). The user is allowed to login to GPP using SSO.
- An external authentication system supplies the user ID and role information which are not found in GPP. In this case, GPP determines if the unknown user is allowed to login to GPP according to the server configuration. When the unknown user is allowed to login to GPP, GPP creates the user based on the predefined user and entitlement templates to login to GPP using SSO.

SSO basics:

- User ID and User role are provided to GPP in the http header fields, or the principal object is provided in the request from an external authentication system.
- When an unknown user logs in to GPP, the new user is created with the office predefined in the Entitlement profile.
- Unknown user creation is supported when the financial institution creates an Entitlement profile. For more information, see Entitlements.
- Financial institution creates a default Entitlement profile. In this case an unknown user is created using the default Entitlement profile.
- Financial institution is responsible to create and maintain the GPP User and Entitlement profiles.

2.2 Processing

When GPP uses SSO it needs to be configured for SSO login mode. GPP can be configured to accept both User ID’s that exist and do not exist in the GPP database. For more information, see Server Configuration.

When the user navigates to the login page, GPP receives the authentication information from the http header fields, or the principal object from an authentication system configured as a reverse proxy to GPP.

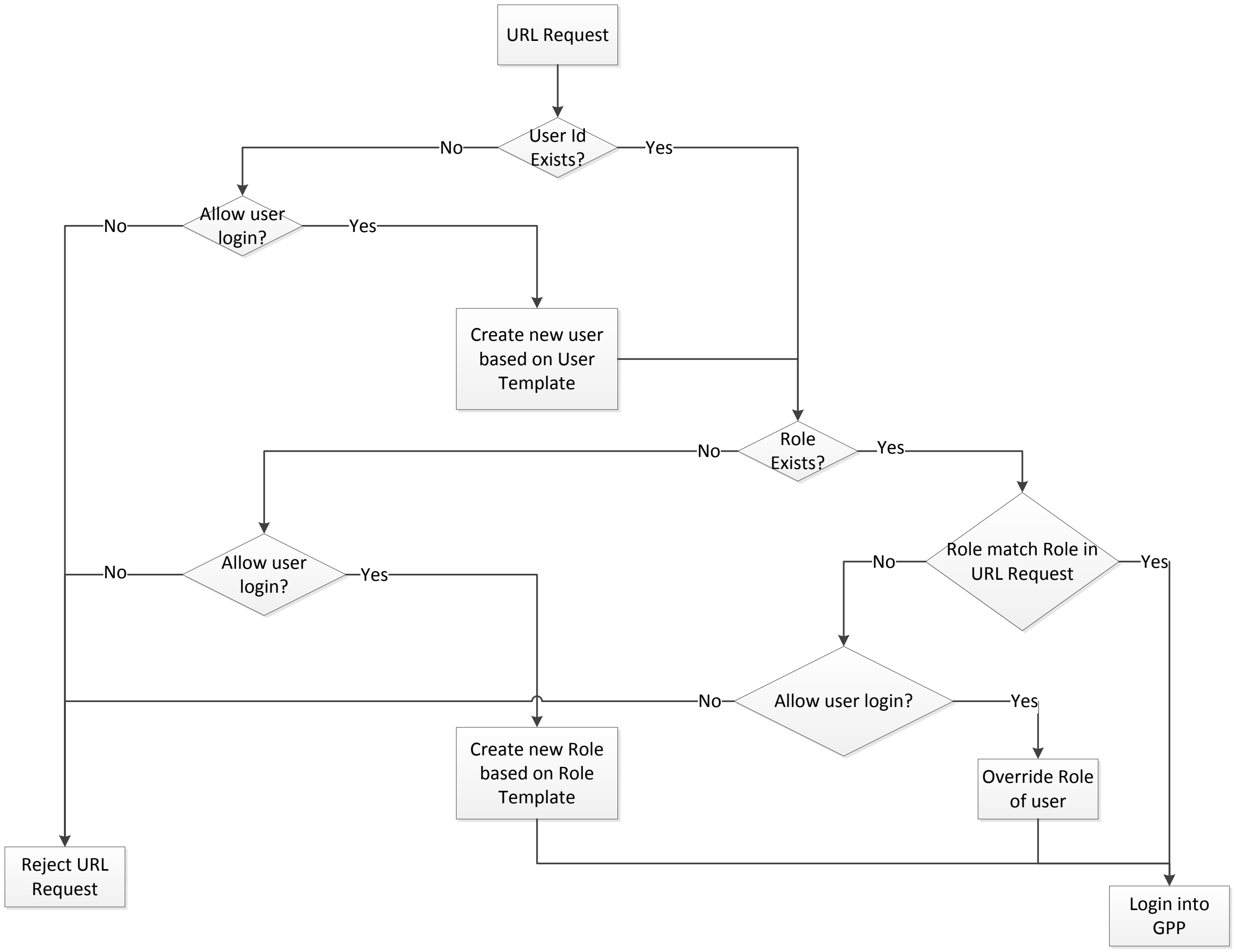
Note: A valid GPP User ID requires a unique case-sensitive User Name. The user must exist with active status (AC) and is not logged in.

GPP uses two methods to identify the User ID. The FI (financial institution) determines which method is used for SSO.

- By Java Authentication and Authorization Service (JAAS) principal object. The user name is obtained from the principal object and GPP checks that the user has the required existing GPP entitlements. For more information, see GPP Online Help, Entitlement Classes.
- HTTP request headers. GPP receives the user name and the user role as values of the HTTP request headers. GPP determines whether the User ID exists in the GPP database.

Note: For both methods the workflow is the same.

2.2.1 SSO Workflow



2.2.2 URL Request

When a user accesses the path to the GPP environment, a request on behalf of the user is sent to the financial institution’s server.

Communication errors, if they occur, are handled by the financial institution. If the URL is not available then an error defined by the financial institution displays.

2.2.3 User ID Exists

When the URL is opened, the GPP Login page is displayed and the user is prompted to enter their User ID and Password.

GPP determines whether the user exists in the GPP database.

- If the user is found, GPP continues with the process and checks if the role exists.
- If the user is not found, GPP checks the Server Configuration option and determines whether the unknown user is allowed to login to GPP.
  - When the user is allowed:
    - GPP creates a new user based on the User template
    - GPP continues with the process and checks if the role exists
  - When the user is not allowed:
    - GPP rejects URL request and the user is not allowed to login to GPP
    - GPP prompts a failed login message

2.2.4 Role Exists

GPP determines whether the user role exists in the GPP database.

- When the user role is found, GPP continues with the process and matches the user role with the user entitlement in the GPP database.
- When the user role is not found, GPP checks the Server Configuration option and determines whether the unknown role is allowed to login to GPP.
  - When a role is allowed:
    - GPP creates a new role based on the Role template
    - GPP authorizes the user login
  - When a role is not allowed:
    - GPP rejects URL request and the user is not allowed to login to GPP
    - GPP prompts a failed login message

2.2.5 Match Role in URL Request

GPP determines whether the user role matches the user entitlement in the GPP database.

- When the user role matches the user entitlement, GPP authorizes the user login.
- When the user role does not match the user entitlement, GPP checks the Server Configuration option and determines whether the user is allowed to login to GPP.
  - When a user is allowed:
    - GPP updates the user record with a new role
    - GPP authorizes the user login
  - When a user is not allowed:
    - GPP rejects URL request and the user is not allowed to login to GPP
    - GPP prompts a failed login message

2.2.6 Login to GPP

When a user is authorized to login to GPP, the main GPP page is displayed.

2.2.7 LDAP Authentication Over SSO

In addition to the SSO process, GPP also supports LDAP Authentication Over SSO. A financial institution can use SSO with LDAP, in order to authenticate the User.

When GPP is configured for the LDAP Authentication Over SSO mode, the user navigates to the SSO login page to enter GPP. In this mode, the SSO login page expects authentication information (user name and password) to be supplied in the http header fields from LDAP.

The server then logs on to the LDAP server on behalf of the user by providing the LDAP server with the user's ID and password. If successful, the security server then proceeds with any authorization procedure and/or lets the user proceed to GPP.

2.3 System Configuration and Business Setup

2.3.1 Business Setup

2.3.1.1 System Parameters

System Parameter	Description
WEBLOGMODE	Specifies the WEB login mode according to which authentication is done. Possible options: • INTEGRATED - User authentication is done via GPP; all users are managed through the USERS table. In this mode, User profiles can be created and edited. • SSO - User authentication is done via an outside application, (e.g. WebSEAL), where the USERS table is used only for storing the users that use the application in order to display the list of logged in users in the Users profile. In this mode, User profiles cannot be created and edited. Note: If the value is not one of the above values, GPP assumes the value to be INTEGRATED.
IV_USER	Specifies the name of the mandatory header field that contains the user ID when operating in SSO mode. This header field must be present in the login request, and its associated value must be a valid GPP user.
IV_CREDS	Specifies the name of the optional header field that contains the user’s GPP role. • If this field is not present, GPP determines the user’s role. • If this header field is present, its value must be a valid GPP role (an entitlement name, in GPP terminology). Use of this optional header field is useful to control privileges from the authentication server.
SSO_USER_PROVISIONING	Specifies whether user authorization by an external SSO service and not defined in the internal GPP repository can be created. • Yes - Unknown user request is created using predefined default template and the user logs in to GPP. • No - Unknown user request is rejected. Note: If the value is not one of the above values, GPP assumes it is No.

For example:

- User defined in GPP with a user ID FCIB002
- This user's role (column U_ENT_NAME in the user's table) is defined as BB1ALL
- The IV_USER system parameter is set to the value http_uid
- The IV_CREDS system parameter is set to the value iv_creds.
- Other roles are defined for different users, one of which is ALL

For example, a request arrives at the SSO login page with the http_uid field set with the value FCIB002, and the iv_creds field set to ALL. This user is given a session with the role ALL.

Some authentication systems such as SiteMinder, require another system parameter to be set so that GPP can allow it to end the user's session at logout. In SSO mode, GPP uses the LOGOUT_FORWARD_REQUEST system parameter to forward the logout request to the authentication system after the GPP application logout is complete. The following is an example of a setting with this value, which is specific to SiteMinder:

Update syst_par set parm_value = 'https://ssointrad.dev.ipc.us.bank.com/SSOI/request?request_type=un_logoff&SSOURL=http://somehost/index.html'

Where param_name = 'LOGOUT_FORWARD_REQUEST';

This example, which is specific to SiteMinder, starts with the SM logout URL, and includes several parts. The last part, SSOURL=http://somehost/index.html, specifies the URL where the user's browser will come to rest after the authentication system logout is complete.

Other authentication systems require a different format for the 'LOGOUT_FORWARD_REQUEST' value.

2.3.1.2 Profiles

These are the details of the setup in GPP profiles that are related to SSO. For more information, see GPP Online Help.

- System Users Profile: Create a user with the **User ID** name TEMPLATE
- Roles Profile: Create a role with the **User Entitlement Profile** name TEMPLATE

2.3.2 Server Configuration

If the user does not exist in the database, GPP determines whether the unknown user is allowed to login to GPP, based on the server configuration.

To allow a user to use SSO, when they do not exist in the GPP database:

1. Set the WEBLOGMODE system parameter with the value SSO.

   The user authentication is done via an outside application, where the Users table is used only for storing the users that use the application in order to display the list of logged in users in the Users profile. In this mode, User profiles cannot be created and edited.

2. Set the IV_USER system parameter to the value of the HTTP request header that contains the User name.

   Note: This step is not relevant if using the JAAS method. For more information, see Processing.

3. Set the IV_CREDS system parameter to the value of the HTTP request header that contains the User role.

   Note: This step is not relevant if using the JAAS method. For more information, see Processing.

4. Set the SSO_USER_PROVISIONING system parameter to Yes. The default value is No.

- Yes - Unknown user request is created using predefined default template and the user logs in to GPP.

2.3.3 WebSphere Configuration to Support SSO

When the authentication system uses an agent to protect the site, the agent setup takes care of intercepting the login requests and sending them to the authentication system. When the authentication system does not include an agent, than WebSphere configuration is required to intercept the login requests and send them to the authentication system.

For more details, see the IBM Knowledge Center page:

http://www.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/csec_SPNEGO_explain.html?cp=SSAW57_8.5.5

Note: When configuring IBM WebSphere, verify that the **Enable application security** checkbox is selected in the **Security** menu, **Global Security**, under the **Application security**.

After setting the security configuration using the Deployment Manager console, edit these files (inside the GPP deployment ear file):

- ibm-application-bnd.xml
- web.xml

These files must be edited according to the customer list of roles in the Authentication server and mapped to a GPP roles (1 -> 1).

The files must be part of the delivered version and updated for every change in the GPP roles or the customer Authentication server roles list.

2.3.3.1 Websphere Security for Authentication to GPP

When deploying the GPP ear file on the WebSphere application server, the Websphere 'security constrain' option can be used in order to ensure that the http access to GPP passes the security authentication managed by WebSphere.

To enable and use this approach, configuration is required on both the application XML files and on the WebSphere console.

web.xml – Update the web.xml file, which exists in the GPPFrontEnd.war/WEB-INF to include a security-constraint tag.

```
<security-constraint>
<display-name>Complete application lockdown</display-name>
<web-resource-collection>
        <web-resource-name>Whole site</web-resource-name>
        <description></description>
        <url-pattern>/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
</web-resource-collection>
<auth-constraint>
        <role-name>*</role-name>
</auth-constraint>
</security-constraint>
```

In addition, a list of all GPP roles that exist in the GPP DB, must exist as a list of entries in the security-role tags.

```
<security-role id="SUPER1">
        <role-name>SUPER1</role-name>
</security-role>
<security-role id="SUPER2">
        <role-name>SUPER2</role-name>
        </security-role>
```

The security-role tag needs to be inserted to the application.xml of the META_INF directory of the ear file.

3 Entitlements

3.1 Amount Limit Level

GPP provides an Amounts Limits Entitlement Class in the User Entitlements Profile. User Entitlement Profiles define a collection of profiles for the purpose of assigning user accessibility, permissions and approvals.

When creating a new user entitlement, the Amount Limits class enables the user to define currency (CCY) amount restrictions per payment for specific system statuses in an office. Settings are defined in the Amount Limits profile.

The Amount Limits profile provides columns Office, List of statuses, Limit Amount, and Limit Ccy. All statuses that are configured as Limit enrolled'are available in the Amount Limits profile.

The Limit enrolled statuses may include:

- Received
- Release
- Repair
- Verify

Additional statuses may be configured as Limit enrolled with database configuration. For each office, the user can define one or more message statuses, including but not limited to Create, Verify, Repair, and/or Release, Limit Amount, and Limit Ccy.

GPP supports validation of user limits by applying amount limits to each message status. These limits restrict the user's ability to action a message based on the message status. GPP compares the payment amount (in limit base currency) with the user limit amount (also in limit base currency) set for the message status. If the limit base currency amount exceeds the limit amount, the actions buttons available on the screen are limited to Print and Exit.

Limits do not restrict a user's ability to view messages, but work in conjunction with other user entitlements. For example, a user with Repair status cannot view or act on a message that is in Repair status, even though the message is within the user's amount limit. Amount limit restrictions apply to all message statuses, including Create.

If a newly created payment fails validation an error message displays, and the payment cannot be created. For more information about entitlement classes, see the GPP Online Help.

4 Lightweight Directory Access Protocol (LDAP)

LDAP is an application protocol used over an IP network to manage and access the distributed directory information service.

When logging in to GPP and LDAP is enabled, the user is authenticated by performing a remote compare operation of the user's password against the password attribute in the directory entry for the Distinguished Name (DN). The login mode is defined by the WEBLOGMODE system parameter, which controls whether to use Integrated or LDAP mode (default is Integrated mode). The selected mode has no effect on payment processing. When the financial institution defines more than one LDAP domain, the user can select the required domain from the Login page.

GPP manages LDAP authentication as follows:

- Users must exist in the GPP database, in addition to LDAP.
- User roles are defined in GPP and are taken from the GPP database.
- GPP does not check the state of the password (change password, suspended, locked) and changing passwords is not done in GPP.
- The financial institution may configure more than one LDAP server and the user is required to select the domain from the GPP login page. When no domain is selected an error is displayed.
- When a user is already logged in and attempts to login again an error message is displayed indicating that the user already logged in.

GPP supports the following for LDAP:

- LDAP host setup
- LDAP port setup
- LDAP user search base DN
- LDAP user search scope
- LDAP Proxy using DN

5 Payment Card Industry Data Security Standard (PCI DSS)

5.1 Overview

PCI DSS is a proprietary information security standard for organizations that handle branded credit cards from the major card schemes including Visa, MasterCard, American Express, Discover, and JCB. GPP only supports Private label cards, which are part of a major card scheme.

The PCI Standard is mandated by the card brands and administered by the Payment Card Industry Security Standards Council. It increases controls around cardholder data to reduce credit card fraud via its exposure (validation of compliance is performed annually). In the event of a security breach, any compromised entity which was not PCI DSS compliant at the time of breach is subject to additional card scheme penalties, such as fines.

GPP support for PCI DSS depends on the definition of the PCI DSS SUPPORT system parameter.

5.2 PCI DSS Use Cases

5.2.1 GPP Tokenization

In this mode, tokenization is only done internally in GPP and does not cover payments that are sent or received from other financial institutions. This means that the payments can only be received and sent with a clear PAN.

Use Case Name	GPP Tokenization
Actors	Bank Channels, Banks core systems, GPP Payment Engine, Clearing/SWIFT Gateway
Description	PAN tokenization, when required, is provided by a service in GPP, during payment processing and interface calls. GPP tokenization service receives a PAN as an input.

Use Case Name	GPP Tokenization
	PAN may be received either in the clear or tokenized. GPP tokenization service returns the tokenized/de-tokenized PAN.
Trigger	GPP tokenization service is triggered in these points in the flow: • During payment processing, for parties derivation. • During interface calls, depending on the financial institution's systems PCI DSS compliance.
Pre-conditions	• PCI DSS SUPPORT system parameter is GPP Tokenization. • Payment is received or initiated in GPP and contains a PAN. • PAN may be received either in the clear or tokenized.
Post-conditions	PAN is tokenized or de-tokenized based on GPP internal tokenization algorithm.
Definition of Done	GPP tokenization service successfully tokenized or de-tokenized the received PAN.
Positive flow	Payment is received or initiated in GPP and contains a PAN. PAN may be received either in the clear or tokenized. GPP tokenization service is triggered in these points in the flow: • During payment processing, for parties derivation. • During interface calls, depending on bank's systems PCI DSS compliance. • During Formatting out, depending on MOP PCI DSS compliance. GPP tokenization service returns the tokenized or e-tokenized PAN. For more information, see
Negative flows	Tokenization Failure

5.2.1.1 Payment Received with Clear PAN

GPP receives a payment and uses the PAN during the payment process.

1. GPP receives or initiates a payment that contains a PAN. The PAN is received in one of the pre-defined list of payment fields which allow PAN.
2. GPP parses the payment.
3. GPP triggers a call to the tokenization service in order to tokenize the PAN. Once it has been tokenized the PAN is stored in the GPP database.
4. During payment processing, for parties derivation, GPP uses the tokenized PAN to derive the relevant account.
5. In the GPP user interface, the PAN is displayed in a masked form, displaying only the last four characters (for example, XXXX-XXXX-XXXX-1234).
6. During interface calls, depending on the financial institution's system PCI DSS compliance (as indicated in interface type configuration), GPP triggers the tokenization service in order to de-tokenize the PAN before generating the requested interface.
7. For outgoing payments, during formatting out, GPP checks the destination MOP PCI DSS compliance (as indicated in the interface type configuration).
    a. When the MOP is PCI DSS Compliant, GPP sends the payment with a clear PAN.
    b. When the MOP is not PCI DSS Compliant, GPP triggers the tokenization service in order to de-tokenize the PAN before sending out the payment.

5.2.2 External Token Vault

Use Case Name	External Token Vault
Actors	Bank Channels, Banks core systems, GPP Payment Engine, Clearing/SWIFT Gateway, External Token Vault
Description	PAN tokenization provided by an external service (token vault), during payment processing and interface calls, when required. External token vault call is called from GPP, providing a PAN as an input. PAN is sent either in the clear or tokenized. External token vault returns the tokenized or de-tokenized PAN.
Trigger	External token vault call triggers in the below points in the flow: • During payment processing, for parties derivation. • During interface calls, depending on the financial institution's systems PCI DSS compliance. • During Formatting out, depending on MOP PCI DSS compliance.
Pre-conditions	Payment received or initiated in GPP and contains a PAN. PAN is received either in the clear or tokenized.
Post-conditions	PCI DSS SUPPORT is External Token Vault. PAN is tokenized or de-tokenized by an external token vault.
Definition of Done	External token vault successfully tokenized or de-tokenized the received PAN.
Positive flow	Payment received or initiated in GPP and contains a PAN. PAN is received either in the clear or tokenized. External token vault call is triggered in the below points in the flow: • During payment processing, for parties derivation. • During interface calls, depending on financial institution's systems PCI DSS compliance • During Formatting out, depending on MOP PCI DSS compliance. External token vault will return the tokenized or de-tokenized PAN.
Negative flows	Tokenization Failure

5.2.2.1 Payment Received with Tokenized PAN

1. GPP receives or initiates a payment that contains a tokenized PAN.
2. PAN is received in one of the pre-defined list of payment fields which allow PAN.
3. GPP parses and stores the payment in the GPP database. The PAN is stored as tokenized.
4. During payment processing, for parties derivation, GPP uses the tokenized PAN to derive the relevant account.
5. In the GPP user interface, the PAN is displayed in a masked form, displaying only the last four characters in the clear (for example, XXXX-XXXX-XXXX-1234).
6. During interface calls, depending on the financial institution's systems PCI DSS compliance (as indicated in interface type configuration), GPP triggers the tokenization service in order to de-tokenize the PAN before generating the requested interface.
7. For outgoing payments, during formatting out, GPP checks the destination MOP PCI DSS compliance (as indicated in interface type configuration).

   a. When the MOP is PCI DSS Compliant, GPP will send the payment with the tokenized PAN.
   b. When the MOP is not PCI DSS Compliant, GPP will trigger the external tokenization service in order to de-tokenize the PAN before sending out the payment.

5.2.2.2 Payment Received with Clear PAN

1. GPP receives or initiates a payment that contains a tokenized PAN. The PAN is received in one of the pre-defined list of payment fields which allow PAN.
2. GPP parses the payment.
3. GPP triggers a call to the tokenization service in order to tokenize the PAN. Once it has been tokenized the PAN is stored in the GPP database.
4. During payment processing, for parties derivation, GPP uses the tokenized PAN to derive the relevant account (the stored PAN in GPP is only stored as tokenized).
5. In message UI, GPP displays the PAN in a masked form, displaying only the last four characters in the clear (for example, XXXX-XXXX-XXXX-1234).
6. During interface calls, depending on the financial institution’s systems PCI DSS compliance (as indicated in interface type configuration), GPP triggers the external tokenization service in order to de-tokenize the PAN before generating the requested interface.
7. For outgoing payments, during formatting out, GPP checks the destination MOP PCI DSS compliance (as indicated in interface type configuration).
   a. When the MOP is PCI DSS Compliant, GPP sends the payment with the tokenized PAN.
   b. When the MOP is not PCI DSS Compliant, GPP triggers the external tokenization service in order to de-tokenize the PAN before sending out the payment.

5.2.3 Payment Manual Create/Repair

Use Case Name	Payment Manual Create/Repair
Actors	GPP User, Banks core systems, GPP Payment Engine, Clearing/SWIFT Gateway, External Token Vault
Description	Authorized user manually creates or repairs a payment in GPP. User input a clear PAN in GPP message user interface. GPP triggers the tokenization service in order to tokenize the PAN before storing it and processing the payment.
Trigger	Authorized user manually creates or repairs a payment in GPP. User input a clear PAN in GPP message user interface.
Pre-conditions	User input a clear PAN in GPP message user interface.
Post-conditions	PAN is tokenized or de-tokenized by tokenization service.
Definition of Done	Tokenization service successfully tokenized or de-tokenized the received PAN. PAN is displayed as masked in GPP user interface and stored tokenized.
Positive flow	Authorized user manually creates or repairs a payment in GPP. User inputs a clear PAN in GPP message user interface. Upon submit, GPP triggers the tokenization service in order to tokenize the PAN before storing it and processing the payment. In the Verify queue, the PAN is displayed as masked, and no re-key verification is required for the PAN.

5.2.4 Tokenization Failure

Use Case Name	Tokenization Failure
Actors	GPP Payment Engine, GPP tokenization service
Description	GPP tokenization service is triggered to tokenize or de-tokenize a PAN. A failure occurs during tokenization/de-tokenization. GPP tokenization service returns an error.
Trigger	GPP tokenization service is triggered in the below points in the flow: • During payment processing, for parties derivation. • During interface calls, depending on the financial institution's systems PCI DSS compliance.
Pre-conditions	PCI DSS SUPPORT <> No Payment received or initiated in GPP and contains a PAN. PAN may be received either in the clear or tokenized.
Post-conditions	GPP tokenization service returns an error.

5.2.4.1 Tokenization Service Triggered During Payment Processing

1. GPP receives or initiates a payment that contains a tokenized PAN. The PAN is received in one of the pre-defined list of payment fields which allow PAN.
2. GPP parses the payment.
3. GPP triggers a call to the tokenization service in order to tokenize the PAN. Once it has been tokenized the PAN is stored in the GPP database.
4. A failure occurs in tokenization service (PAN could not be tokenized or technical error etc.)
5. The tokenization service returns an error response.
6. Payment is routed to the Repair PCI queue.
7. In the Repair PCI queue, users may:

   Retry tokenization service, by submitting the payment back to the flow; after submit, GPP will trigger the tokenization service again.

   a. Amend the payment details (for example, input a new PAN, remove the PAN and replace with account number) and submit. After submit, GPP triggers the tokenization service again.
   b. Cancel or Reject the payment.

5.2.4.2 Tokenization Service Triggered During Interface Call

1. During interface calls, depending on the financial institution systems PCI DSS compliance (as indicated in interface type configuration), GPP triggers the tokenization service in order to de-tokenize the PAN before generating the interface request.

   Or

   For outgoing payments, during formatting out, GPP checks the destination MOP PCI DSS compliance (as indicated in interface type configuration).

   a. When the MOP is PCI DSS Compliant, GPP sends the payment with the tokenized PAN.
   b. When the MOP is not PCI DSS Compliant, GPP triggers the tokenization service in order to de-tokenize the PAN before sending out the payment.

2. A failure occurs in tokenization service (PAN could not be tokenized or technical error etc.)
3. The tokenization service returns an error response.
4. Payment is routed to Repair PCI queue.
5. In the Repair PCI queue, users may:
   a. Retry tokenization service, by submitting the payment back to the flow. After submit, GPP triggers the tokenization service again.
   b. Amend the payment details (for example, input a new PAN, remove the PAN and replace with account number) and submit. After submit, GPP triggers the tokenization service again.
   c. Cancel or Reject the payment.

5.3 Processing

GPP processes the payment for PCI DSS taking into consideration the following:

- PAN
  - GPP can receive the primary account number (PAN - credit account number) as encrypted (tokenized) or as a full number.
  - When a user manually inputs a PAN, then GPP triggers a call to the tokenization service in order to retrieve a token.
  - GPP can derive the debit or credit party out of the tokenized PAN (using either internal or external tokenization).
  - GPP does not store the full PAN in static data (including MINF).
  - In the GPP user interface, the PAN is masked (for example, XXXX-XXXX-XXXX-1234), and the full PAN is not displayed (including audit, error log, original XML, reporting database). The only case when the full PAN is displayed is in the Create or Repair status (based on user authorization).
  - Based on the financial institution's system compliance with PCI DSS, GPP may need to send either an encrypted or decrypted PAN to external systems (for example, Accounting Interface, Balance Inquiry Interface).
  - When a PAN is uploaded to GPP using an upload task, or web service, GPP triggers a call to the tokenization service in order to retrieve a token.
  - When the tokenization service fails, GPP will not store the PAN and an error is generated when saving the profile.
  - PANs are displayed as tokenized in GPP reporting DB (Golden Gate).

- Credit card numbers
  - When credit card numbers are stored in GPP, the Credit Cards tab in the Accounts profile, holds the account's credit card numbers.
  - When credit card numbers are not stored in GPP, then they are retrieved by an account lookup call during payment processing.
- PCI DSS SUPPORT system parameter. GPP supports PCI DSS, using the following methods, depending on the definition in the PCI DSS SUPPORT system parameter:
  - External Token Vault: Tokenization performed outside of GPP. GPP provides an interface to the external token vault which is used during payment processing and interface calls.
  - GPP Tokenization: Tokenization performed in GPP, during payment processing and interface calls.
  - GPP Token Vault: Tokenization performed in GPP, and GPP is serving as an external token vault to other systems.

For more details of how GPP processes the payment for PCI DSS, see PCI DSS Use Cases.

5.4 System Configuration and Business Setup

5.4.1 Business Profiles

5.4.1.1 Accounts Profile, Credit Cards Tab

A new tab was added to the GPP Accounts profile, which is called Credit Card Numbers. This tab holds the account's credit card numbers, only when they are stored in GPP. When credit card numbers are not stored in GPP than they are retrieved by an account lookup call during payment processing.

5.4.1.2 Interface Profile

The PCI DSS Compliant attribute in the Interface Types profile indicates whether the interface (either an interface to an external system or an interface to a clearing) is PCI DSS compliant.

Based on this attribute, GPP knows whether the interface request or payment should be sent with a tokenized or de-tokenized PAN.

5.4.2 System Parameters

To enable PCI DSS support, the system parameter PCI_DSS_SUPPORT needs to be defined accordingly: The PCI DSS functionality is optional and can be configured accordingly, using the PCI_DSS_SUPPORT system parameter. This system parameter indicates the manner in which GPP should apply the PCI DSS support.

- No - GPP does not apply PCI DSS support.
- External Token Vault - GPP applies the external token vault tokenization.
- GPP Tokenization - GPP applies the internal tokenization. GPP tokenizes or de-tokenizes when deriving the account details, and when sending interface requests to financial institutions systems that are not PCI DSS compliant. This option includes the case where GPP serves as an external Token Vault

6 LAU (Local Authentication)

6.1 Overview

Local authentication is used to authenticate the origin and ensure integrity of each message.

LAU is signature based. The authentication and integrity of the message is achieved by calculating and verifying the signature. GPP supports SHA-2(256) algorithm in combination with a symmetric key that is used to calculate the signature of a message. The algorithm output gets converted to base64 encoding.

These features are based on the clearing scheme, which are being used and currently GPP only supports NPP, SWAA-FT.

Note: The symmetric key is shared by relevant entities.

- For outgoing messages - GPP calculates the signature and based on configuration sends it
- For incoming messages - GPP recalculates the signature and compares it to the signature received in the message
- On successful calculation or verification GPP continues the flow, otherwise park the payment to technical exception queue

6.2 Processing

6.2.1 Key Management

- A standalone utility is provided to manage keys
- DevOps user to run the utility and input the key
- The inputted key gets encrypted and stored in the file location
- GPP reads the file and decrypts the key while startup
- Decrypted key then used in algorithm

Key expiry

This will be used in overlap period.

6.2.1.1 DevOps Utility Setup

The folder structure contains:

- Log4j.properties (file contains the logger information)
- LAUEncryptionConfig.properties (file contains the Encryption details)
- GenerateSecurityKey.class (this is the class file to run)
- log4j-1.2.17.jar (Jar file for logger information)

6.3 System Configuration and Business Setup

6.3.1 Security Feature Configuration

Security features are configured at the interface level. A master switch is provided to turn on/off the security feature and the sequence of security operations are configurable.

Interface type properties are configured at the interface level.

Note: Security features are supported for MQ protocol only.

Any failure in security operation will send the payment to technical exception queue.

6.3.1.1 Interface Types Properties for Security Operation

- SECURITY_CURNT_KEY_APNDER:-

  Value of this property is used to identify whether the signature for incoming or outgoing is calculated using CURRENT key.

  In case of NPP implementation, the value of this field is A. if GPP calculates signature using CURRENT key, then the value of this field is appended to the signature and then sent to the clearing.

  For example, if signature calculated is OA0Greg9kATsG0dFpjdNEw== then GPP sends the message with signature as AOA0Greg9kATsG0dFpjdNEw==.

- SECURITY_CURR_DCRPT_KEY:-

  This property is used only while caching the object of type InterfaceSecurityProperties.

  Note: Value of this field should not be altered. GPP uses this property internally

- SECURITY_DS_CUR_KEY_FILEPATH :-

Defines the path where the *current security key for digital signature* resides. GPP loads the key from this path during server start up or cache refresh and use it for calculating the digital signature.

- SECURITY_DS_OVR_KEY_FILEPATH :-

  This property defines the path where the overlap security key for digital signature resides. GPP loads the key from this path during server start up or cache refresh and use it for calculating the digital signature.

- SECURITY_ED_ALGO_NM :-

  This property defines the Algorithm name to be used for Encryption/Decryption of the Message.

- SECURITY_ED_CUR_KEY :-

  This property is used only while caching the object of type InterfaceSecurityProperties.

Note: Value of this field should not be altered. GPP uses this property internally

- SECURITY_ED_CUR_KEY_FILE_PATH :-

  This property defines the path where the current security key for Encryption/Decryption resides. GPP loads the key from this path during server start up or cache refresh and use it for encrypting/decrypting the message.

- SECURITY_ED_OVERLAPED_KEY :-

  This property is used only while caching the object of type InterfaceSecurityProperties.

Note: Value of this field should not be altered. GPP uses this property internally.

- SECURITY_ED_OVR_KEY_FILE_PATH :-

  This property defines the path where the overlap security key for Encryption/Decryption resides. GPP loads the key from this path during server start up or cache refresh and use it for encrypting/decrypting the message.

- SECURITY_IN_OPERATION_SEQ :-

  This property defines the sequence of security operation for incoming message. Defined security operation for incoming message are Signature verifications (DSV) and Decryption (D).

  For example, if the signature of incoming message is calculated on encrypted message then, GPP should first validate the digital signature and then decrypt the message. For this case the value of this property should be DSV,D(comma separated).

  Other values that it can hold are:

  - D,DSV
  - DSV
  - D

- SECURITY_MODE:-

  This property defines the mode or scheme for which this security operation is enabled.

  For example, NPP (for NPP) , SAA_FINMT (SWIFT Fin MT)

- SECURITY_OUT_OPERATION_SEQ:-

  This property defines the sequence of security operation for outgoing message. Defined security operation for outgoing message are Signature calculation (DSC) and Encryption (E)

For example, if the requirement is to configure GPP such that Signature is calculated (DSC) and then the encryption (E) of message should be done. Then for the sequence value for this case should be DSC,E (comma separated).

Other values that it can hold are: -

- E
- DSC
- E,DSC

- SECURITY_OVRLP_DCRPT_KEY:-

  This property is used only while caching the object of type InterfaceSecurityProperties.

  Note: Value of this field should not be altered. GPP uses this property internally.

- SECURITY_OVR_KEY_APNDER:-

  Value of this property is used to identify whether the signature for incoming or outgoing is calculated using OVERLAP key.

  In case of NPP implementation, the value of this field is B. if GPP calculates signature using OVERLAP key, then the value of this field is appended to the signature and then sent to the clearing.

  For example, if signature calculated is OA0Greg9kATsG0dFpjdNEw== then GPP sends the message with signature as BOA0Greg9kATsG0dFpjdNEw==.

- SECURITY_SIGN_ALGO_NM:-

  This property defines the Algorithm name to be used for calculating the digital signature of Message.

- SECURITY_SIGN_CHK_ENABLED:-

  This property acts as a master switch to turns ON or OFF the entire security operation for the particular interface. To enable security operation functionality, the value of this property should be TRUE and to disable it, the value of the property should be FALSE.

- SECURITY_SIGN_MQ_PROP_NM:-

  This property defines the name of the property in which, the calculated signature is send or receive from or to GPP.

  For example, for NPP, the value of this property is set in the MQ property PAG_MAC. So while sending or receiving the payments from or to GPP, the value of calculated signature appears in the value of this property, which is PAG_LMAC as follows:

  PAG_LMAC= AOA0Greg9kATsG0dFpjdNEw==

Notes:

- All above properties should be defined at interface level
- Value for properties which are not relevant and not internal to GPP should be set to (null)

6.3.1.2 Details of the Configuration File

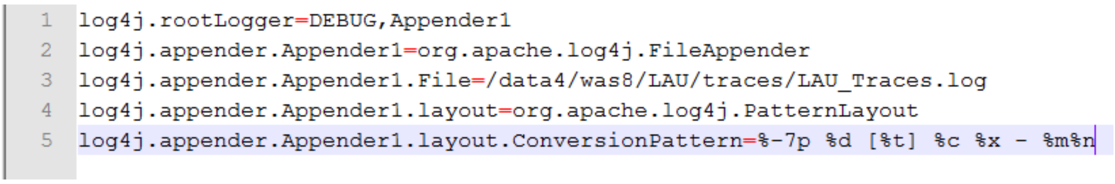
6.3.1.2.1 Prerequisites

Any path defined in the **log4j.properties** property should be created manually on the server before running the utility.

There are two user configuration files in the structure:

1. log4j.properties

This property file contains the configuration details of the logs.



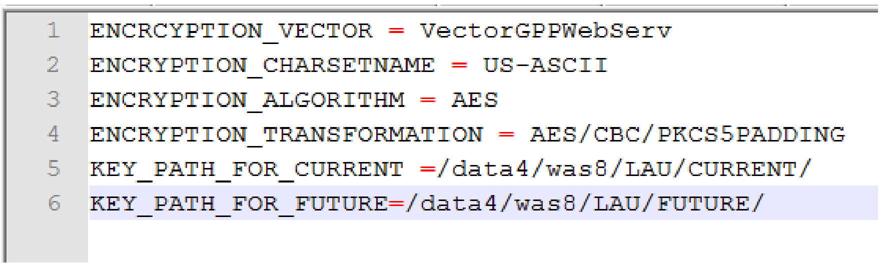
```
log4j.rootLogger=DEBUG,Appender1
log4j.appender.Appender1=org.apache.log4j.FileAppender
log4j.appender.Appender1.File=/data4/was8/LAU/traces/LAU_Traces.log
log4j.appender.Appender1.layout=org.apache.log4j.PatternLayout
log4j.appender.Appender1.layout.ConversionPattern=%-7p %d [%t] %c %x - %m%n
```

2. Log4j.appender.Appender1.File

This property allows users to generate the log file in the user defined path. The path is configurable and can be changed accordingly.

Notes:

- The path to be mentioned in the Log4j.appender.Appender1.File must be created. In addition, the path should contain the folder structure /LAU/traces. For example, /home/was8/…../LAU/traces.
- The configured path to this property should be accessible to GPP.



```
ENCRCYPTION_VECTOR = VectorGPPWebServ
ENCRYPTION_CHARSETNAME = US-ASCII
ENCRYPTION_ALGORITHM = AES
ENCRYPTION_TRANSFORMATION = AES/CBC/PKCS5PADDING
KEY_PATH_FOR_CURRENT =/data4/was8/LAU/CURRENT/
KEY_PATH_FOR_FUTURE=/data4/was8/LAU/FUTURE/
```

KEY_PATH_FOR_CURRENT

This property is used to locate the path where the current key is to be generated. By default, it is generated inside the **/data4/was8/LAU/CURRENT/** folder. The path is user configurable and users can change this accordingly. This path should be created manually before running the utility.

KEY_PATH_FOR_FUTURE

This property is used to locate the path where the future key is to be generated. By default, it will get generated inside the /data4/was8/LAU/FUTURE/ folder. The path is user configurable and can be changed accordingly. This path should be created manually before running the utility.

Note: Only KEY_PATH_FOR_CURRENT and KEY_PATH_FOR_FUTURE properties should be changed. Other properties should not be changed.

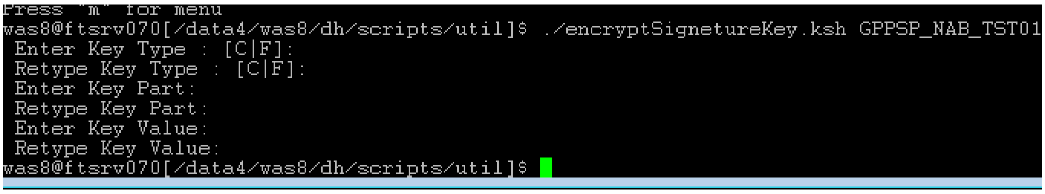
- The path to be configured in the KEY_PATH_FOR_CURRENT and KEY_PATH_FOR_FUTURE should be created manually.
- The path should contain the folder structure /LAU/ CURRENT/ and /LAU/ FUTURE/

  For example, /home/was8/…../LAU/CURRENT/, /home/was8/…../LAU/FUTURE/

- The configured path to this property should be accessible to GPP.

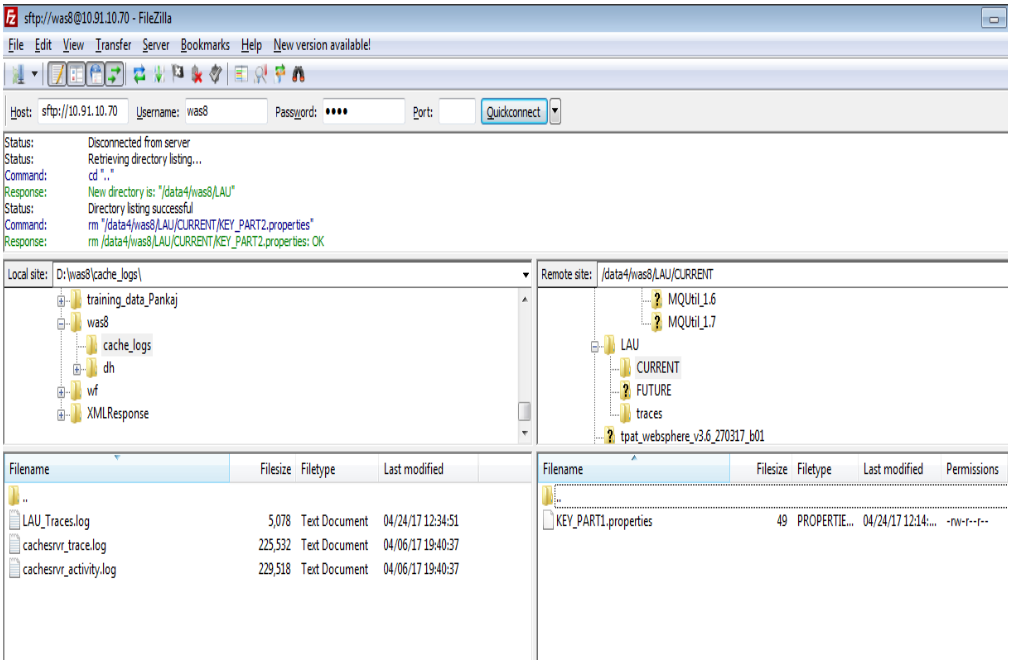
How to Run the Utility

Key Value: Actual key value



```
Press "m" for menu
was8@ftsrv070[/data4/was8/dh/scripts/util]$ ./encryptSignetureKey.ksh GPPSP_NAB_TST01
 Enter Key Type : [C|F]:
 Retype Key Type : [C|F]:
 Enter Key Part:
 Retype Key Part:
 Enter Key Value:
 Retype Key Value:
was8@ftsrv070[/data4/was8/dh/scripts/util]$
```

Verify that the file was generated in the configured path.



Host: sftp://10.91.10.70 Username: was8 Password: •••• Port:

```
Status:      Disconnected from server
Status:      Retrieving directory listing...
Command:     cd ".."
Response:    New directory is: "/data4/was8/LAU"
Status:      Directory listing successful
Command:     rm "/data4/was8/LAU/CURRENT/KEY_PART2.properties"
Response:    rm /data4/was8/LAU/CURRENT/KEY_PART2.properties: OK
```

Local site: D:\was8\cache_logs\

Filename	Filesize	Filetype	Last modified
..			
LAU_Traces.log	5,078	Text Document	04/24/17 12:34:51
cachesrvr_trace.log	225,532	Text Document	04/06/17 19:40:37
cachesrvr_activity.log	229,518	Text Document	04/06/17 19:40:37

Remote site: /data4/was8/LAU/CURRENT

Filename	Filesize	Filetype	Last modified	Permissions
..				
KEY_PART1.properties	49	PROPERTIE...	04/24/17 12:14:...	-rw-r--r--

7 Message Encryption/Decryption

In addition to the local authentication (LAU), GPP supports the encryption/decryption of the payload of the MQ messages for outgoing and incoming messages.

The encryption is done using the AES-128 algorithm in combination with a symmetric key.

Note: These features are based on which clearing scheme is being used and currently GPP only supports NPP.

For System Configuration and Business Setup see, System Configuration and Business Setup.

Appendix A: Glossary

This table provides definitions for terms used in this document.

Term	Description
CCY	Currency
JAAS	Java Authentication and Authorization Service
GPP	Global PAYplus
LDAP	Lightweight Directory Access Protocol
SOA	Service-Oriented Architecture
SSO	Single Sign On
UID	Unique ID. Each GPP profile is identified by a unique identification string.
URL	Uniform Resource Locator
Tokenization	Tokenization, when applied to data security, is the process of substituting a sensitive data element with a non-sensitive equivalent, referred to as a token that has no extrinsic or exploitable meaning or value.
PAN	Primary Account Number: is the 14, 15 or 16 digit number that appears on the primary account holder’s credit card.