IBM Safer Payments 6.3

Implementation Guide

SC34-7719-00

This edition applies to IBM® Safer Payments release 6.3.x, Program Number 5725-Z82, and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About this publication

This publication describes how to install, configure, and operate IBM Safer Payments 6.3.1.*x*.

This Implementation Guide is valid for Safer Payments 6.3.1.*x*. See “Versioning Method” on page ix for more information about Safer Payments version numbers.

The most current version of the Implementation Guide is available on IBM Documentation.

Note: IBM Counter Fraud Management for Safer Payments has been renamed to "IBM Safer Payments" with version 6.0.

Who should use this publication

This publication is intended as a reference for system administrators and Safer Payments administrators who deploy Safer Payments in a PCI DSS compliant environment.

How to use this publication

The publication consists of the following sections:

- Chapter 2, “Installing and configuring Safer Payments,” on page 3 describes how to install, configure, and operate Safer Payments.
- Chapter 3, “Key management procedures for cryptographic keys,” on page 51 describes the cryptographic keys that are used by Safer Payments, how keys are generated, and how to enter and activate keys.

If you have feedback about this publication, send it to IBM Support:

https://www.ibm.com/support/entry/portal/support

Where to find more information

IBM Safer Payments resources

The IBM Safer Payments home page offers up-to-date information about related products and services, new functions, and other items of interest:

https://www.ibm.com/marketplace/payment-fraud-prevention

The IBM Support Portal is the central hub for support including Technotes:

https://www.ibm.com/support/entry/portal/support

Fix Central provides fixes and updates for IBM Safer Payments:

https://www.ibm.com/support/fixcentral

Versioning Method

IBM Safer Payments uses a release number scheme that is based on the PA-DSS versioning recommendations.

Versioning

The versioning of IBM Safer Payments consists of four numbers that are separated by periods, for example, 6.3.0.00. Each position in the version number has a meaning:

- The first number denotes the project generation.

- The second number denotes feature releases or High Impact changes per PA-DSS.
- The third number denotes Low Impact changes per PA-DSS.
- The fourth number denotes No Impact changes. It can be one or two digits. These can also be denoted with a wildcard ("*x*") For example, 6.4.0.*x*.

Changes in the fourth number denote changes that are not PCI relevant and therefore have no impact per the PA-DSS Program Guide.

PA-DSS change types

For information about PA-DSS change types, see the following sections in the PA-DSS Program Guide:

- High Impact

  See *Section 5.2.3.4 "High Impact Changes"*.
- Low Impact

  See *Section 5.2.3.3 "Low Impact Changes"*.
- No Impact

  See *Section 5.2.3.2 "No Impact Changes"*.
- Administrative Changes

  See *Section 5.2.3.1 "Administrative Changes"*.

Chapter 1. Introduction to PCI DSS

This section gives a brief overview of the Payment Card Industry Data Security Standard (PCI DSS).

The PCI DSS is defined by the PCI Security Standards Council (PCI SSC). It is a multifaceted security standard that includes requirements for security management, policies, procedures, network architecture, software design, and other critical protective measures. This comprehensive standard is intended to help organizations proactively protect customer account data.

In compliance with PCI DSS, a complementary standard framework for payment applications has been established, called the "Payment Application Data Security Standard (PA-DSS)".

The PCI DSS and PA-DSS documentation, which also describes the relationship between PCI DSS and PA-DSS, can be downloaded here:

https://www.pcisecuritystandards.org/document_library

PA-DSS certification

Safer Payments 6.3.1.x is certified against PA-DSS 3.2.

This document describes how to implement Safer Payments in a PCI DSS compliant environment.

For official certificate information, see the PCI website. In the **Company Name** search field, enter "IBM Safer Payments".

Applicability of PCI DSS and PA-DSS to Safer Payments

Safer Payments installations must be configured and operated in a way that ensures compliance to PCI DSS.

In payment card issuing the Primary Account Number (PAN) is the defining factor to prevent fraud in a successful Safer Payments operation.

PCI DSS compliance means considerable administrative work for Safer Payments licensees. An easy way to avoid these additional efforts is not to process and store any clear-text or encrypted PAN in Safer Payments. This can be achieved by hashing PAN numbers before they are sent to Safer Payments. Licensees following this path, can ignore the PCI DSS specifications regarding their Safer Payments installation. This is a fairly viable approach, unless you are a payment card issuer or its processor.

However, to use a partly hashed PAN throughout the whole Safer Payments installation is highly impracticable in card issuing fraud prevention. Safer Payments users need to see the clear-text PAN to retrieve additional information from other systems, talk to cardholders, analyze fraud patterns and trends, and so on. In addition, many fraud patterns can only be detected and stopped by including the PAN into Safer Payments decision models. Therefore, Safer Payments provides user access rights. For example, users with a legitimate need to work with the decrypted PAN can do so, whereas standard users see PANs only masked.

How to become PCI DSS compliant

If you have decided to process PANs, you must ensure compliance and PCI DSS applies to your Safer Payments installation.

To achieve compliance with version 3.2 of PA-DSS, you must properly configure your installation to meet the requirements. This document describes how to implement and operate Safer Payments in compliance with version 3.2 of PCI DSS.

As PCI DSS compliance cannot be achieved without extra contributions by the Safer Payments licensee, this document also highlights necessary compliance measures to be implemented by the licensee.

Chapter 2, “Installing and configuring Safer Payments,” on page 3 describes actions that are required for PCI DSS compliant Safer Payments installation and operation.

Chapter 3, “Key management procedures for cryptographic keys,” on page 51 provides deeper insight into Safer Payments key management procedures.

Frequently used terms

This topic lists terms that are used in PCI DSS, PA-DSS, and that are also commonly used throughout this publication.

Cardholder data

- Primary Account Number (PAN)
- Cardholder name
- Expiration date
- Service code

Sensitive authentication data

- Full magnetic stripe data or equivalent on a chip
- CAV2/CVC2/CVV2/CID
- PINs/PIN blocks

Note: Sensitive authentication data is not required by Safer Payments for full operational functionality. Supplying systems must be configured in a way that they do not send such data to Safer Payments. If you send such data to Safer Payments, your installation is not PCI DSS compliant.

Chapter 2. Installing and configuring Safer Payments

This section describes how to install and configure Safer Payments so that it meets all PCI DSS requirements. Further sections address software updates, ongoing operation of Safer Payments, and it's decommissioning.

Safer Payments provides a built-in PCI DSS compliance report that lists all relevant configuration settings that must be changed to achieve PCI DSS compliance. See “PCI DSS compliance report” on page 46 for details.

Preliminary considerations

Before you install and configure Safer Payments, your organization needs to define and implement certain operational processes, and periods.

Define and implement operational processes

To achieve PCI DSS compliance, it is not enough to configure Safer Payments as described here. You must also implement a set of operational processes within your organization for PCI DSS compliant operation.

Note: Therefore, it is important that you read the PCI DSS documentation and implement the operational processes that are described there.

https://www.pcisecuritystandards.org/document_library

Define a cryptoperiod

The cryptoperiod defines the lifetime of an encryption key. At the end of each cryptoperiod, keys must be replaced.

PCI DSS itself does not postulate a specific cryptoperiod. However, it is necessary that you as an organization define your own cryptoperiod. See “Enforce regular key changes ” on page 57 for details.

Define a retention period

Outdated cardholder data must be securely deleted. PCI DSS itself does not postulate when cardholder data becomes outdated. However, according to PCI DSS requirement 3.1 (aligns with “Requirement 2.1” on page 62) it is necessary that you as an organization define a retention period.

You can define different retention periods for different kind of data elements:

- A retention period for transaction data, according to your business requirements.
- A longer retention period for all other data, such as cases, or event logs.

Basically, you could also define the same retention period for both types of data. Retention requirements for cases or audit trails are typically longer than five years. However, usually there is no business need to retain transaction data for such extended periods, and memory consumption would be heavy given the typical transaction volumes.

Installation

This section describes how to install, uninstall, and update Safer Payments.

The instructions assume that you install Safer Payments as a cluster of multiple Safer Payments instances. If you want to install Safer Payments as stand-alone service, omit the installation steps for the other instances.

You must be logged in with an administrator account on your workstation to run some of the installation steps described.

Note: You do not need administrator privileges to run Safer Payments.

System requirements

This section lists the currently supported operating systems and further requirements.

Supported platforms

Safer Payments is tested for the following operating systems:

- Red Hat® Enterprise Linux® 7 (RHEL 7)
- Oracle Linux 7

User access is provided with all recent standard browsers. The following browsers are fully tested for compatibility:

- Microsoft Edge 79 or later
- Firefox 69 or later
- Google Chrome 64 or later

Apple Safari is partially tested for compatibility.

Screen resolution

The Safer Payments user interface is designed to work with a minimum resolution of 1280×768 pixel, thus WXGA (1280×768) screen resolution is the minimum for correct page display. For power users of Safer Payments, for example, fraud analysts, a dual monitor configuration with HDTV (1920x1080) screen resolution is recommended. With this resolution users can open multiple browser pages/tabs with the same Safer Payments session. Because of the high interactiveness of the Safer Payments user interface, the performance of the JavaScript engine is key to smooth operation of Safer Payments. The Google Chrome browser provides the best user interface performance for Safer Payments.

Installation overview

Installing Safer Payments requires the following steps.

1. If you have an existing Safer Payments installation, verify the version number you are running. Log in as root on the console and enter the following command:

   ```
   iris release
   ```

2. Read the Release Notes to find out the current major releases and fix packs that are available. Decide what major release or fix pack you need to install. Contact IBM Support if you have questions.
3. Download the installation image. For more information, see “Download the installation image ” on page 5.
4. Verify and unzip the installation zip file. For more information, see “Verify and unzip the installation zip file” on page 5.
5. Prepare JRE (optional). For more information, see “Prepare the Java Runtime Environment ” on page 6.
6. Run the installer. Follow the process that is right for your situation. For more information, see:
   - “Installing Safer Payments for the first time” on page 6
   - “Installing a major release” on page 8
   - “Installing a fix pack update” on page 10

Download the installation image

Before you begin

Read the Release Notes for the version that you are installing. The Release Notes explain where to get the installation image. Options are:

- Passport Advantage®

  Passport Advantage contains only major releases, not fix packs.
- Fix Central

  Fix Central contains only fix packs, not major releases. Every fix pack is a full installation, you do not need to install a major release and then install fix packs on top of it.

From IBM's perspective, a major release has zeros in the third and fourth positions of the version number. That means, for example, that 6.2.0.00 and 6.3.0.00 are major releases (available on Passport Advantage) and 6.2.1.2 and 6.3.0.03 are fix packs (available on Fix Central).

Downloading from Passport Advantage

If you are downloading the installation image from Passport Advantage:

1. Go to IBM Passport Advantage.
2. Sign in with your IBM ID and password. If you are not yet registered, follow the prompts to request access.
3. Click **Software download & media access**. All entitled products are listed on the Software Download and Media Access page. This page provides a guided process for selecting a platform and the product version that you want to download.
4. If you are licensed for more than one product, select the program offering that you want to download.
5. Click **Download finder** to see a list of available products. Follow the on-screen prompts to select and download Safer Payments.

   The download file is a tar file. The tar file name depends on the version.
6. Log in as root on the console.
7. Extract the tar file:

   ```
   tar -xf tarfilename.tar
   ```

   The result is an installation zip file named `SaferPayments_6_3_x_x.zip`.

Downloading from Fix Central

If you are installing the installation image from Fix Central:

1. Go to Fix Central.
2. Download the Safer Payments installation image. It is a zip file that is named:

   ```
   SaferPayments_6_3_x_x.zip
   ```

Verify and unzip the installation zip file

To verify that the installation zip file is not corrupted, you must check its integrity.

Use the preinstalled `sha256sum` tool to verify that the checksum for the zip file matches with the checksum provided in the Release Notes.

1. Go to the Safer Payments Release Notes for the release or fix pack you are installing. Get the checksum value.

2. Log in as root on the console.
3. Run the `sha256sum` checksum tool:

```
sha256sum SaferPayments_6_3_x_x.zip
```

4. Verify that the tool's output is the same as the checksum given in the Release Notes.
5. Extract the .zip file to a temporary directory:

```
unzip SaferPayments_6_3_x_x.zip
```

The following files are extracted from the zip file:

`SaferPayments.bin`
The installation image file.

`ibm_jre_8.0.2.10_linux_x64.vm`
The setup file to install LINUX IBM JRE 1.8 SR2 FP10 (64-bit). You need this file only if you don't have JVM installed.

For more information, see “Prepare the Java Runtime Environment ” on page 6.

`installer.properties`
The sample response file. You need this file only if you want to run the installer in silent mode.

Prepare the Java Runtime Environment

To run the installer, Java Runtime Environment must be installed on your system. If JRE is already installed, you can skip this step.

Important: The Linux IBM JRE is intended for use only with InstallAnywhere installers. Do not use it for any other purposes.

Root privileges are not needed to use Linux IBM JRE.

1. Locate the `ibm_jre_8.0.2.10_linux_x64.vm` file that is part of the download installation zip file. Use this file to install LINUX IBM JRE 1.8 SR2 FP10 (64-bit).
2. Log in on the console as root. Go to the directory where the installation zip file is located and run the following commands:

```
unzip ibm_jre_8.0.2.10_linux_x64.vm
tar xf vm.tar.Z
chmod +x jre/bin/java
chmod +x SaferPayments.bin
export PATH=`pwd`/jre/bin:$PATH
```

Note: You must set the **`export`** command with the same user as the install user.

Installing Safer Payments for the first time

This topic describes how to install Safer Payments if you don't have an existing Safer Payments installation.

If you already have an existing Safer Payments installation, do not follow these steps. Instead, see “Installing a major release” on page 8 or “Installing a fix pack update” on page 10.

Prerequisites

1. Follow the installation instructions up to running the installer. For more information, see “Installation overview ” on page 4.
2. Decide on the instance path. The instance path is the directory where you want to store the Safer Payments instance.
3. Decide whether to run the installer in normal or silent mode.
4. Create the following:

SPUser

SPUser is the user that runs the Safer Payments instance.

SPUserGroup

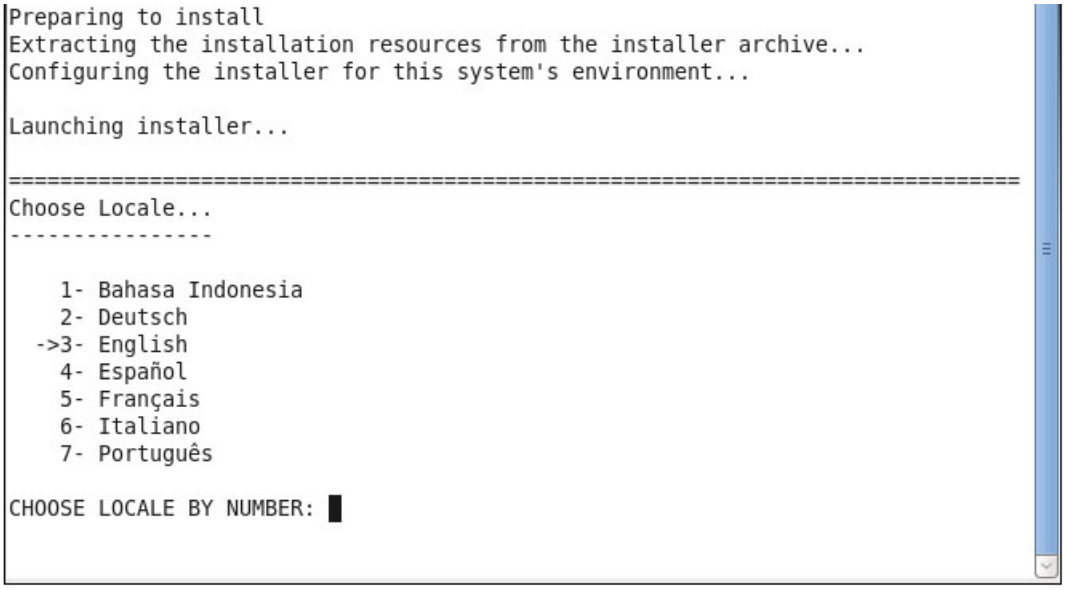
SPUser belongs to the SPUserGroup.

Run the installer (normal mode)

Log in as root on the console. Go to the directory where the installation file is located and run the following command:

```
sh ./SaferPayments.bin
```

The installer starts in console mode:



Log in as root on the console. Go to the directory where the installation file is located and run the following command:

Select a language, press the enter key, and follow the installation steps.

In the installer you can specify some special settings:

Change license language
Even if you selected a language other than English, the English license is shown by default. To display the license in the selected language press 5 when the English license is shown.

Choose install set

Typical
If you choose `Typical`, the configuration with the type "empty" is copied to the factory reset folder. The instance path must be created manually after installation.

Starter packs
If you choose `Starter packs`, one of several custom configurations can be selected. This custom configuration is then automatically copied to the instance path.

Starter packs settings

Configuration
This is the Safer Payments configuration that you want to start with in your instance path.

Path
The instance path. The configuration is copied to this path.

SPUser/SPuserGroup
The user and group that are created in “Prerequisites” on page 6 .

Note: The chosen install set and configuration don’t affect any security or PA-DSS relevant settings. All instructions in this and the following sections apply to all install sets and configurations.

The following folders are created:

/usr/bin
The Safer Payments and the Keygen binary files are installed in this folder.

/usr/lib64
The AES and SQL libraries of Safer Payments are located in this folder.

/installationPath/
The Safer Payments installation directory. The default is: `/opt/ibm/safer_payments/install`

/installationPath/inc
The JavaScript files of Safer Payments are located in this folder.

/installationPath/factory_reset
This folder contains an initial configuration to start Safer Payments for the first time. Never change the files in this folder and always use a copy with other user privileges for your initial configuration.

Run the installer (silent mode)

In silent mode, the installer has no user interaction and is run by using a response file that contains the values for various variables.

Safer Payments provides a sample response file that is called `installer.properties` with default values. To accept the license agreement, open the sample response file and set:

```
$LICENSE_ACCEPTED$=true
```

Make sure the response file and `SaferPayments.bin` are in the same directory.

Log in as root on the console. Go to the directory where the installation file is located and run the following command:

```
sh ./SaferPayments.bin -i silent
```

Complete post-requisites

Complete the post-requisites regardless of whether the installer ran in normal or silent mode. If you chose `Typical` you must run the following commands after installation:

```
cp -R /installationPath/factory_reset/* /instancePath
chown -R SPUser:SPUserGroup /instancePath
```

Where:

- */instancePath* is the instance path, as defined in “Prerequisites” on page 6.
- *SPUser* and *SPUserGroup* are the user and group, as defined in “Prerequisites” on page 6.

Installing a major release

A major release is indicated by a change of the first or second revision number position.

In a major release, file formats might be changed so that you cannot change back to an earlier release. Also, you might not be able to install such an update immediately, that is, on a running Safer Payments cluster that still fully runs during update. There might be specific instructions for the type of major release you plan.

Depending on your specific application needs, it might be advisable to contact IBM support for assistance with a major release.

Before you begin:

1. Go to the Release Notes for the major release and read the "Update information".
2. Follow the installation instructions up to running the installer. For more information, see “Installation overview ” on page 4.

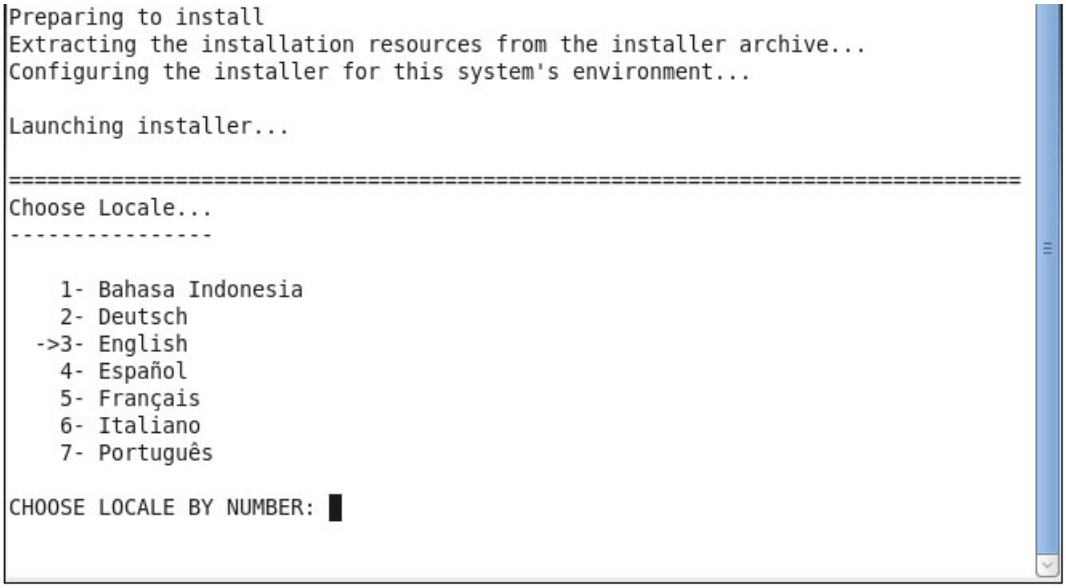
Suggested workflow to install a major release:

1. Shut down all Safer Payments instances in a cluster.
2. Back up at least one instance. Use the backup to restore other instances should you need to revert the update. For more information, see:

   https://www.ibm.com/support/pages/performing-backup-ibm-safer-payments
3. Log in as root on the console. Go to the directory where the installation file is located and run the following command:

   ```
   sh ./SaferPayments.bin
   ```

   The installer starts in console mode:

4. Select a language, press the enter key, and follow the installation steps.

   Note: Do not use `Starter packs` for major releases.
5. The installer continues and finishes.

   The following folders are updated:

   /usr/bin
   The Safer Payments and the Keygen binary files are installed in this folder.

   /usr/lib64
   The AES and SQL libraries of Safer Payments are located in this folder.

   /installationPath/
   The Safer Payments installation directory. The default is: `/opt/ibm/safer_payments/install`

   /installationPath/inc
   The JavaScript files of Safer Payments are located in this folder.

/installationPath/factory_reset
This folder contains an initial configuration to start Safer Payments for the first time. Never change the files in this folder and always use a copy with other user privileges for your initial configuration.

6. If you have a previous rpm installation, you must update all symbolic links (readme file and swidtag, license, inc folder) of all configurations after the installation. Navigate to the instance path, log in as SPUser on the console and run the following commands:

```
ln -f -s /installationPath/readme readme
ln -f -s /installationPath/swidtag swidtag
ln -f -s /installationPath/license license
ln -f -s /installationPath/inc inc
```

7. Read the Release Notes to understand if there are any additional files that require exchange.
8. Start all updated Safer Payments instances.
9. Verify the version number. Log in to Safer Payments and go to **Cluster** > **System Internals** > **General Information**.

Installing a fix pack update

A fix pack update is indicated by a change of the third or fourth revision number position.

A fix pack update can be run without downtime of the online message interfaces (MCI, MQ and Kafka). If possible, API activity should be reduced to case investigation during the update procedure until at least two instances have been updated to the new version. This is a precaution to prevent complications during model changes, go-lives, and simulation caused by instances running on different versions. If postponing model operations is not feasible, you may perform required model operations all the time on your own risk. Carefully check the Release Notes update information for further limitations.

Before you begin:

1. Go to the Release Notes for the fix pack and read the "Update information".
2. Follow the installation instructions up to running the installer. For more information, see “Installation overview ” on page 4.

Suggested workflow to install a fix pack update:

1. Disable the online message interfaces (MCI, MQI and KMI) and incoming FLI on instance (A) that will be updated first. Move API to another instance if API is active on instance (A)
2. Ensure that the outgoing FLI buffer of instance (A) to all other instances is empty.
3. If possible, minimize the API activity from that point on to case investigation on any remote instance during the update of instance (A).
4. Shut down instance (A).
5. Back up at least one instance. Use the backup to restore other instances should you need to revert the update. For more information, see:

   https://www.ibm.com/support/pages/performing-backup-ibm-safer-payments
6. Log in as root on the console. Go to the directory where the installation file is located and run the following command:

   ```
   sh ./SaferPayments.bin
   ```

   The installer starts in console mode. Select a language and follow the installation steps.
7. Start instance (A) and monitor logs during startup. Contact support if there are unexpected errors or warnings that have not been in the log before.
8. Clear browser cache before logging in to instance (A) (ctrl+shift+F5 on most browsers).
9. Activate FLI on instance (A) and wait for instance (A) to receive all buffered FLI messages.
10. Disable FLI on all instances except for (A).
11. Enable API and the online message interfaces on instance (A).

12. If the update was successful, continue to the next step and update the other instances. If something went wrong, back out the update. For more information, see “Backing out a fix pack update” on page 11.
13. Disable the online message interfaces on all other instances.
14. Shut down instance (B).
15. Perform an update on instance (B) as described in step 6.
16. Start instance (B).
17. Enable FLI on instance (B) and wait for synchronization.
18. Enable the online message interfaces on instance (B) after synchronization is complete.
19. If required, you may perform configuration and model changes on the updated instances again at this point.
20. Shut down, update, and start instance (C) and all other instances (X).
21. Enable FLI on instances (C) and (X).
22. Enable the online message interfaces on instances (C) and (X) after synchronization is complete.
23. Verify the version number. Log in to Safer Payments and go to **Cluster** > **System monitoring** > **System internals**.

Backing out a fix pack update

If you encounter problems installing a fix pack update, you can revert to the previous release.

Compare the fix pack versions. If the first three version numbers are the same and only the fourth number is different, the fix packs are probably exchangeable. This means you can switch freely back and forth between them. If they are not exchangeable, it is stated in the Release Notes.

Choose the procedure that is correct for your situation.

Standard backout procedure

Use this procedure if:

- A healthy instance is available for the older release before the update, and
- The fix packs are exchangeable.

1. Shutdown the instance that has failed.
2. Uninstall Safer Payments.
3. Install the previous version before the update.
4. Set `cfg/iris.iris` to `{\"iris\":{\"status\":\"New\"}}`
5. Restore the instance from a healthy instance that is using the previous version.
6. Contact support and provide information from logs and the cfg folder.

Back-out using backups or virtual machine snapshots

Use this procedure if:

- A healthy instance is not available for the older release before the update, or
- The fix packs are not exchangeable.

An administrator must restore the instance from a backup or snapshot that was taken of the instance before the update.

Uninstalling Safer Payments

This topic describes how to uninstall Safer Payments. Uninstalling Safer Payments uninstalls only installation files and folders, not configuration data.

Prerequisites

Decide whether to run the uninstaller in normal or silent mode.

Run the uninstaller (normal mode)

To uninstall, navigate to:

/installationPath/IBMSaferPayments_installation

Log in as root on the console and run the following command:

```
./uninstall_safer_payments
```

Run the uninstaller (silent mode)

In silent mode, the uninstaller has no user interaction and is run by using a response file that contains the values for various variables.

To run the silent uninstaller, navigate to:

/installationPath/IBMSaferPayments_installation

Log in as root on the console and run the following command:

```
./uninstall_safer_payments -i silent
```

Basic configuration

This section describes the basic configuration steps.

Check the ephemeral port range

Use ports outside your system's ephemeral port range.

To check the ephemeral port range, run the following command from the console:

```
sysctl -A | grep ip_local_port_range
```

Start the first Safer Payments instance

The browser-based Safer Payments user interface is used to configure a Safer Payments cluster. To access it, you must first start the first Safer Payments cluster instance.

1. To start the first Safer Payments cluster instance, run the following commands from the console on the server:

   ```
   su SPUser
   cd /instancePath/cfg
   iris id=i createinstances=n
   ```

   - */instancePath* is the path, where the instance configuration is stored.
   - *SPUser* is the user, which runs the Safer Payments instance.
   - *i* must be a unique ID of the instance you are currently installing. Preferably, start your first instance with 1. That is, if you set up three instances in total, use IDs 1, 2, and 3.

- *n* is the number of instances you want to create.

2. Check the system event log messages on the console window, and verify that they indicate a proper start of the Safer Payments cluster instance. That is, no warning (W), error (E), or fatal (F) type messages.

   Exception: The `status.iris` file does not exist yet and is being created during the first start. This creates an E 155 during the first start, followed by a message, that the file has been created. Therefore, this error message is expected.

3. Depending on the configuration of the server that you are installing on, you might have to configure the firewall open port, the API port for HTTP access of the browser. The default HTTP port of the Safer Payments first instance is "8001".

   Open a browser and enter:

   ```
   http://127.0.0.1:8001
   ```

4. The Safer Payments user interface login page is displayed.
5. Enter **`user`** as login and **12345678** as password. You are prompted to change the password of this account immediately.

   Note: To comply with PA-DSS requirement 3.1, you must create new personalized users for your configuration and disable the default configuration user.

6. Log in with one of your new users and continue the configuration.
7. The full Safer Payments user interface is displayed.
8. Click the **Cluster** tab.
9. The **Cluster Settings** section shows a table with one row for each Safer Payments instance.

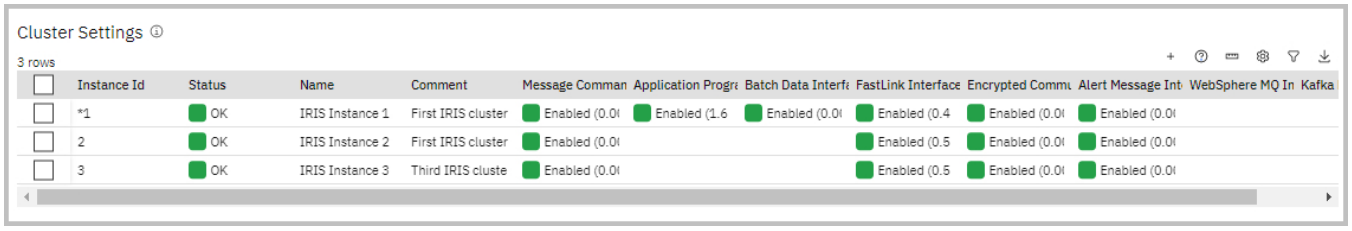


Figure 1. Cluster settings

10. Click anywhere in the row (except the checkbox) to open the configuration details of a Safer Payments instance.

Since Safer Payments was started without a previous configuration, it uses default settings for the number of cluster instances you specified with the `createinstances` command.

To use all Safer Payments interfaces, it might be required to open more ports in your firewall. By default Safer Payments uses the following ports:

- 8001 - Application Programming Interface
- 27921 - Fast Link Interface
- 27931 - Status Control Interface
- 27941 - Encrypted Communication Interface

Note: If you plan to use an MQ or Kafka server to deliver data to Safer Payments, you must set up your firewall appropriately.

Cluster instance configuration

You can now customize all cluster settings. This includes changing the IP addresses/ports, enabling SSL encryption as described in “Configure SSL encryption” on page 14, limiting IP address ranges, and changing local file storage locations as described in “Configure cardholder data storage locations” on page 21.

Make the appropriate settings for all cluster instances, not only the instance you are currently working on, even if the others are not set up physically yet.

Note: Changes to the local file storage are processed after a restart of a Safer Payments instance. Thus you can move the files while the instance is offline. All changes to the interfaces are processed immediately when the settings are saved.

Configure the Message Command Interface

The Message Command Interface (MCI) can have multiple endpoints, which can be configured independently. Each of them has a unique TCP port on the system.

Go to **Cluster** > **Interfaces** > **Inbound** to add new endpoints. You can configure business and performance settings. If you cannot view or edit the page, check how **Inbound Endpoint configuration** global privilege is defined for your user. For more information, see “Set user privileges” on page 40.

The endpoints are not open until they are added to any Safer Payments instances in **Cluster** > **Settings** > **Message Command Interface**. All security-related settings for MCI endpoints are defined there. For information about how to configure them for PCI DSS compliance, see “Configure SSL encryption” on page 14.

Configure SSL encryption

Note: TLS is the successor of SSL. Subsequently, the term SSL is used to refer to the secure communication technologies within Safer Payments. In the Safer Payments interfaces all equivalent elements are named SSL.

For PCI DSS compliance you must enable SSL encryption as follows:

- The API must be encrypted to securely transmit passwords.
- All MCI endpoints must be encrypted when cardholder data is sent over public networks.
- The ECI must be enabled for synchronization of encryption keys between cluster instances.
- SSL and early TLS are not considered strong cryptography. Payment applications must not use, or support the use of, SSL or early TLS. Therefore, TLS 1.0 and 1.1 must be disabled for API / MCI and ECI.

Note: The FLI / SCI do not support SSL/TLS encryption but instead encrypt attribute data based on the encryption settings defined in Safer Payments.

For each interface that uses SSL encryption, encrypted SSL certificate files must be provided. Safer Payments needs two files to support an encrypted connection. The server certificate and the private key in PEM format. The storage location of these files can be configured on the **SSL Settings** page. See “Create certificates with OpenSSL” on page 18 for details on how to create the required certificates.

1. On the Safer Payments user interface, click the **Cluster** tab.

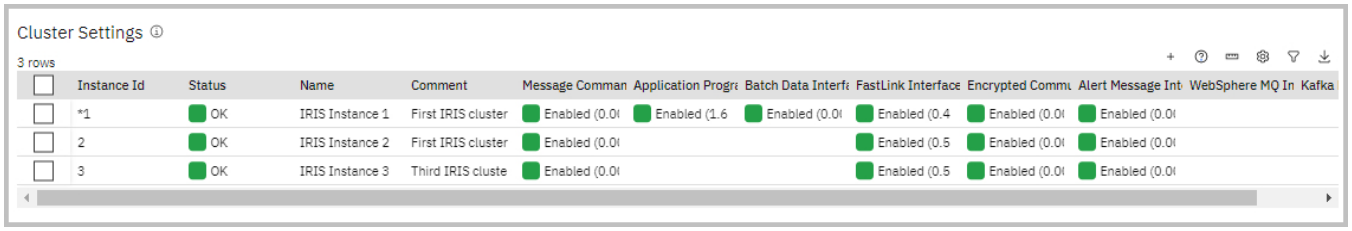


Figure 2. Cluster settings

2. Click the first instance of the **Cluster Settings** table.
3. Scroll down to the **Interfaces** section. Click the **Application Programming** tab.

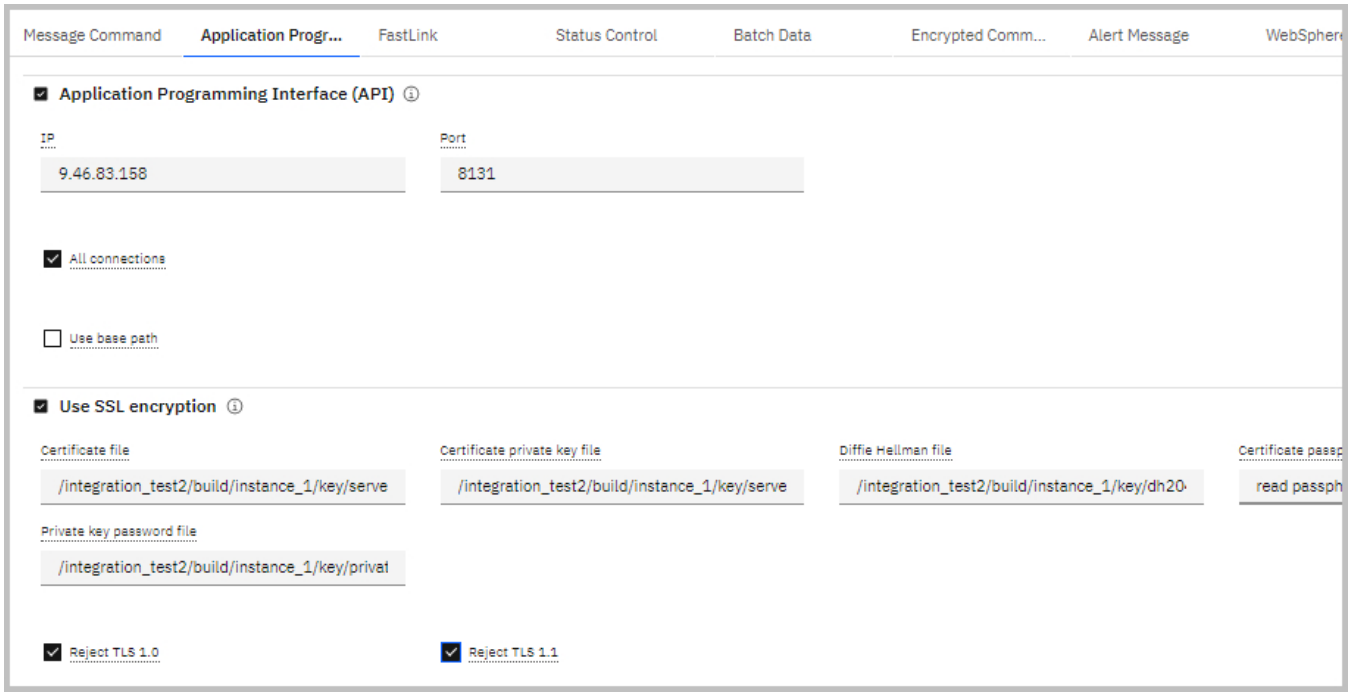


Figure 3. API - SSL settings

4. Select the **Application Programming Interface (API)**, **Reject TLS 1.0**, and **Reject TLS 1.1** checkboxes.

   Add the file paths for the **Certificate file**, **Certificate private key file**, and **Diffie Hellman file**.
5. Click the **Encrypted Communication** tab. Click the **Encrypted Communication Interface (ECI)**, **Reject TLS 1.0**, and **Reject TLS 1.1** checkboxes.

   Add the file paths for the **Certificate file**, **Certificate private key file**, and **Diffie Hellman file**.
6. Click the **Message Command** tab. Click the **Message Command Interface (MCI)**, **Reject TLS 1.0**, and **Reject TLS 1.1** checkboxes.

   Add the file paths for the **Certificate file**, **Certificate private key file**, and **Diffie Hellman file**.

   Repeat for each endpoint.
7. Repeat these steps for each instance.

Note:

- SSL settings are individual for each Safer Payments instance because different instances of Safer Payments running on different computers with different IP addresses require different certificates.
- Enabling SSL encryption and changing the settings becomes effective immediately.
- From now on, you are prompted to enter the certificate passphrase on the console during startup for each instance. See “Start and stop Safer Payments instances” on page 36 for details.

Sending cardholder data over public networks

If cardholder data is to be sent over public networks, Safer Payments must also validate the SSL certificates, and multi-factor authentication is enforced for the API.

If cardholder data is to be sent over public networks, Safer Payments must validate the SSL client certificates for MCI and ECI. Furthermore, API access needs to be secured by multi-factor authentication using either a third-party solution (for example, VPN) or by activating the validation of individual API client certificates as the second authentication factor.

Make sure that the “validate client certificate CN” option is enabled as well. This enforces individual certificates for each user, which must use the users login name as common name (CN).

Note: Before activating this option, make sure that you have created proper client certificates for at least the administrative Safer Payments users. The creation of client certificates for this purpose is covered by the section Create certificates with OpenSSL.

To change the API, MCI and ECI settings open the instance configuration for each cluster instance.

1. Click the **Cluster** tab.
2. Select a cluster instance.
3. Scroll down to the **Message Command Interface (MCI)** section and select the **Validate client certificate** checkbox for each endpoint.
4. Enter the correct path to the CA certificate file in **Client CA certificate file**.

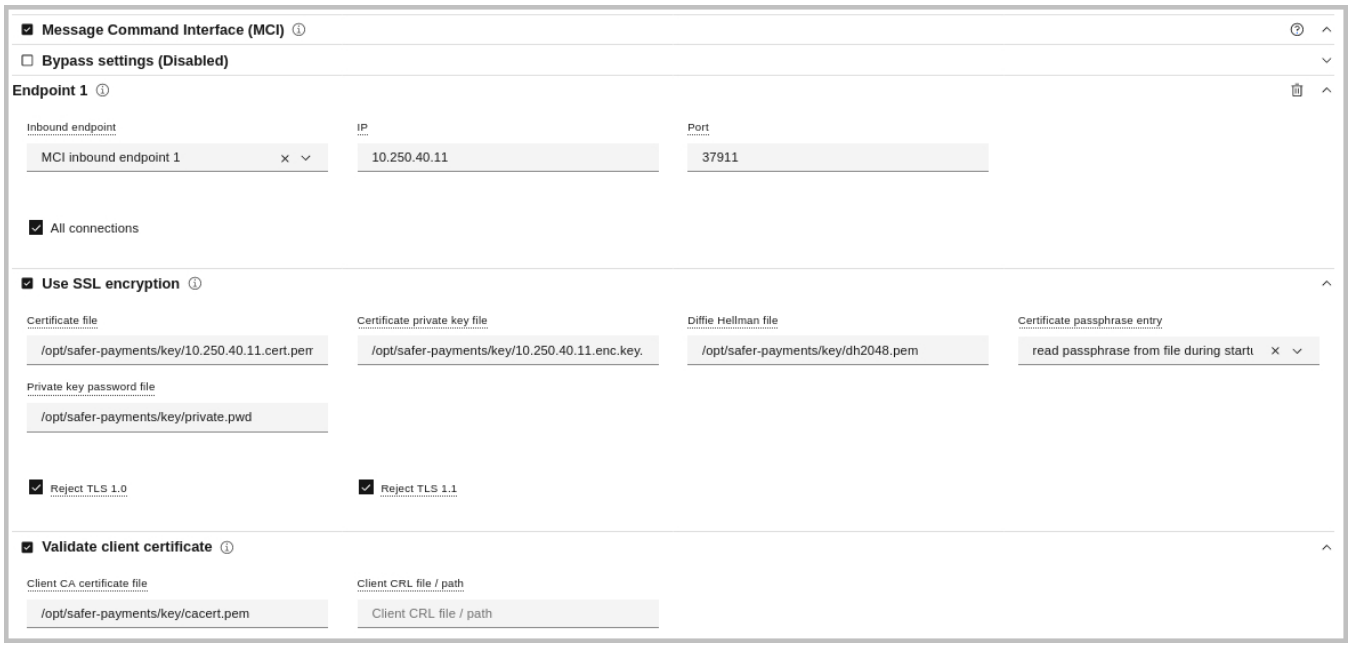


Figure 4. Message Command Interface (MCI) section

5. Click the **Application Programming** tab. Select the **Validate client certificate** checkbox.

   Note: You need a client CA certificate for each Safer Payments instance, and a corresponding certificate for each service consumer that is used to access Safer Payments.
6. Enter the correct path to the CA certificate file in **Client CA certificate file**.

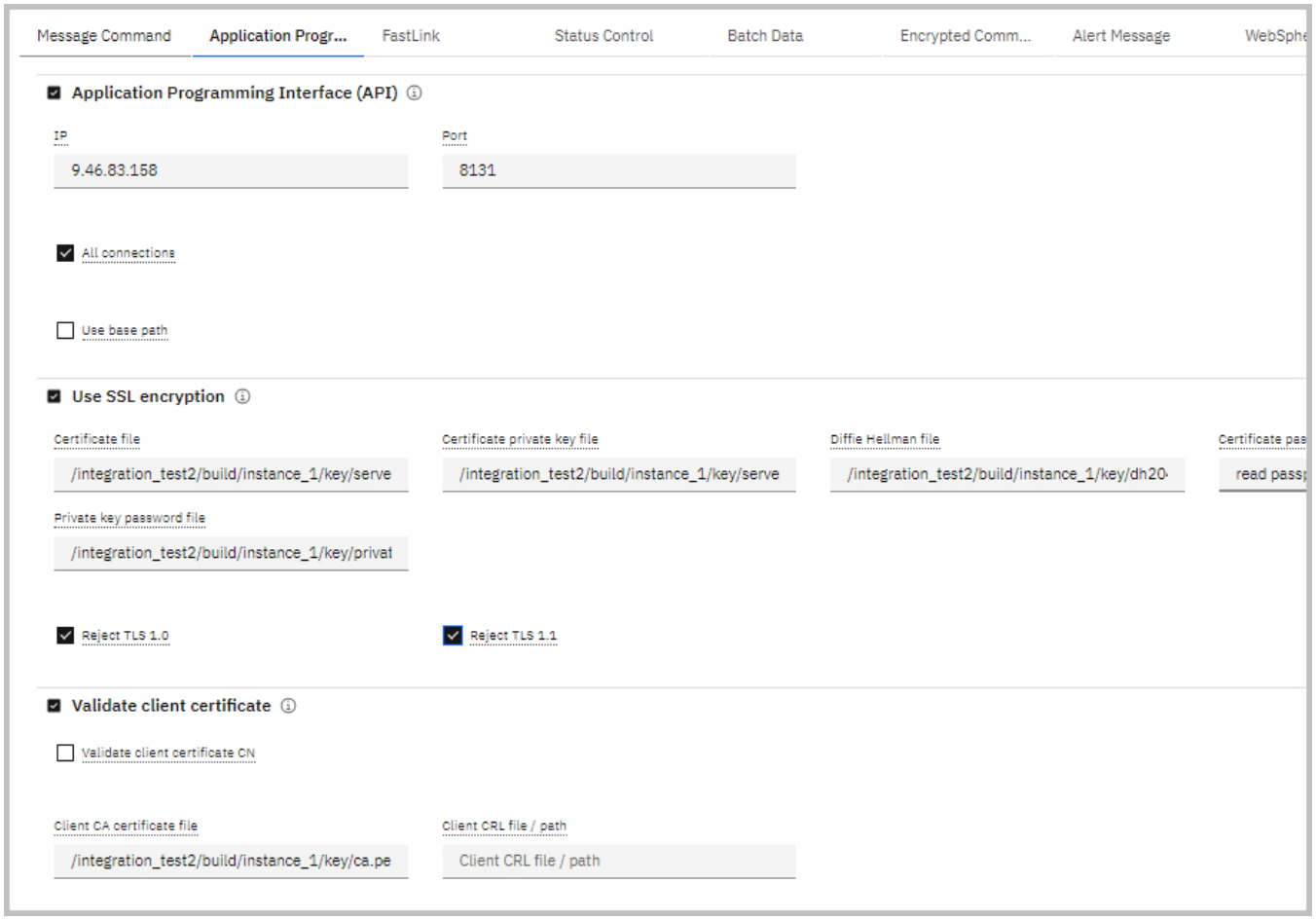


Figure 5. Application Programming Interface (API) section

7. Click the **Encrypted Communication** tab. Scroll down within the instance settings to the **Encrypted Communication Interface** section.

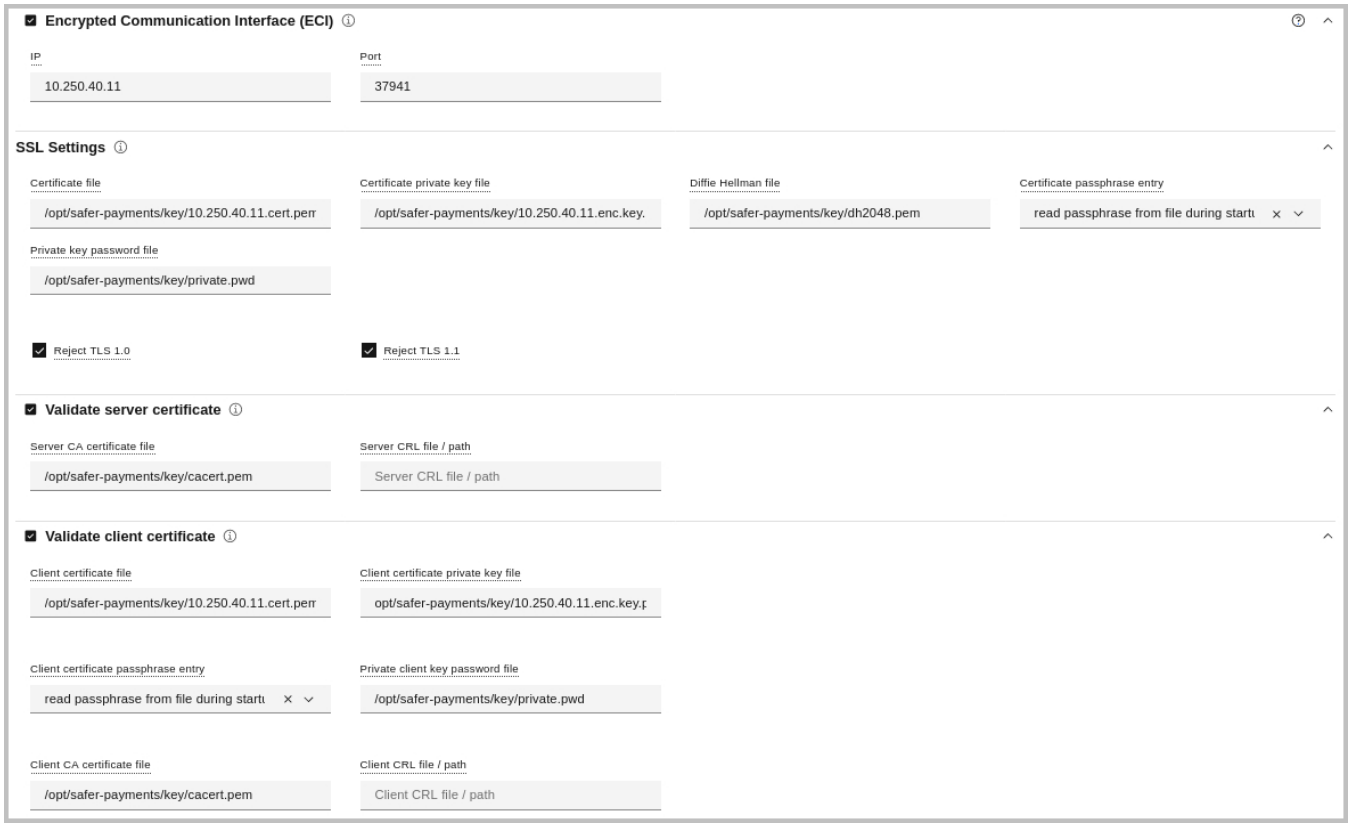


Figure 6. Encrypted Communication Interface (ECI) section

8. Select the **Validate server certificate** and **Validate client certificate** checkboxes.

   Note: You need both a server and client CA certificate for each Safer Payments instance, and a corresponding client certificate.

   The encrypted private key is usually stored within the client certificate file but can optionally be stored in a separate file. The **Client certificate private key file** entry points to the correct location.
9. Place the files in the `/key/` directory of the instance.
10. Optionally, **Server CRL file / path** and **Client CRL file / path** can be used to define certificate revocation lists.

Create certificates with OpenSSL

This topic shows the steps that you need to run to create certificates with OpenSSL.

Important: You must ask your security expert to review and run these steps. IBM takes no guarantee for security, as these steps might differ on different platforms.

The settings shown here are adapted to Safer Payments.

1. **Create config file**

   The `caconfig.cnf` file is the default config file for the certificate authority (CA). It has the following content:

```
#.................................
[ ca ]
default_ca = CA_default
[ CA_default ]
dir = .
certs = $dir/certs
crl_dir = $dir/crl
database = $dir/index.txt
new_certs_dir = $dir/newcerts
certificate = $dir/certs/cacert.pem
```

```
serial = $dir/serial
crl = $dir/crl/crl.pem
private_key = $dir/private/cakey.pem
#RANDFILE = $dir/private/.rand
x509_extensions = usr_cert
crl_extensions = crl_ext
default_days = 3650
#default_startdate = YYMMDDHHMMSSZ
#default_enddate = YYMMDDHHMMSSZ
default_crl_days = 183
#default_crl_hours = 24
default_md = sha256
preserve = no
#msie_hack
policy = policy_match

[ policy_match ]
countryName = match
#stateOrProvinceName = match
#localityName = match
organizationName = match
commonName = supplied
emailAddress = optional

[ req ]
default_bits = 4096 # Size of keys
default_keyfile = key.pem # name of generated keys
distinguished_name = req_distinguished_name
default_md = sha256 # message digest algorithm
attributes = req_attributes
x509_extensions = v3_ca
#input_password
#output_password
string_mask = nombstr # permitted characters
req_extensions = v3_req

[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = DE
countryName_min = 2
countryName_max = 2
#stateOrProvinceName = State or Province Name (full name)
#stateOrProvinceName_default = RLP
#localityName = Locality Name (city, district)
#localityName_default = Coblence
organizationName = Organization Name (company)
organizationName_default = IRIS
organizationalUnitName = Organizational Unit Name (department, division)
organizationalUnitName_default = Fraud Prevention
commonName = Common Name (hostname, IP, or user name)
commonName_max = 64
commonName_default = 192.168.1.1
emailAddress = Email Address
emailAddress_max = 40
emailAddress_default = support@iris.de

[ req_attributes ]
#challengePassword = A challenege password
#challengePassword_min = 4
#challengePassword_max = 20
#unstructuredName = An optional company name

[ usr_cert ]
basicConstraints= CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
#nsComment = ''OpenSSL Generated Certificate''
#nsCertType = client, email, objsign for ''everything including object signing''
subjectAltName=email:copy
issuerAltName=issuer:copy
#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl =
#nsRenewalUrl =
#nsCaPolicyUrl =
#nsSslServerName =

[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]
subjectKeyIdentifier = hash
```

```
authorityKeyIdentifier = keyid:always,issuer:always
basicConstraints = CA:TRUE
#keyUsage = cRLSign, keyCertSign
#nsCertType = sslCA, emailCA
#subjectAltName=email:copy
#issuerAltName=issuer:copy
#obj=DER:02:03

[ crl_ext ]
#issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
#.................................
```

2. **Create Diffie-Hellman files**

```
$ openssl dhparam -out dh2048.pem 2048
```

3. **Create CA**

```
$ mkdir ~/myca
$ cd ~/myca
$ mkdir private certs newcerts conf export crl
$ echo "01" > serial
$ touch index.txt
$ vim conf/caconfig.cnf (Step 1)
$ openssl req -new -x509 -extensions v3_ca -keyout private/cakey.pem -out
certs/cacert.pem -days 3650 -config conf/caconfig.cnf
   → PW: xxxxxxxx
```

Note: Create a strong password and distribute it only to entitled people.

4. **Create signed server certificate and private server key**

You need one certificate/key for each Safer Payments instance. Run the following commands once for each instance and replace SERVER_IP with the IP address or host name of the server.

```
$ openssl req -new -nodes -config conf/caconfig.cnf -out SERVER_IP.req.pem
-keyout private/SERVER_IP.key.pem
   → CN: SERVER_IP
$ openssl ca -config conf/caconfig.cnf -out newcerts/SERVER_IP.cert.pem
-infiles SERVER_IP.req.pem
```

5. **Create signed client certificate and private client key**

For MCI and ECI, you need at least one client certificate for each instance. Run the command twice per instance with unique file names and make sure that you enter unique common names when prompted.

If you want to use multi-factor authentication using API client validation, you might want to create one extra client certificate per user. For these, make sure that the common name matches the users login.

```
$ openssl req -new -nodes -out filename.req.pem -keyout private/filename.key.pem
-days 3650 -config conf/caconfig.cnf
(for MCI) -> CN: CLIENT_IP_OR_NAME
(for ECI) -> CN: IRIS_SERVER_NAME
(for Browser) -> CN: LOGIN
$ openssl ca -out newcerts/filename.cert.pem -days 3650 -config conf/caconfig.cnf
-infiles filename.req.pem
```

6. **Encrypt certificates**

To encrypt certificates for secure storage on the Safer Payment instances run the following command:

```
openssl rsa -des3 -in private/<filename>.key.pem -out private/<filename>.enc.key.pem
```

It is recommended to do this for both, client and server certificates.

7. **Create a certificate revocation list**

```
vim certs/ca.crl
vim crl.confg
```

Content of crl.confg

```
[ ca ]
default_ca      = CA_default                    # the default ca section

[ CA_default ]
dir             = ./                            # where everything is kept
database        = $dir/index.txt                # database index file.
certificate     = $dir/certs/cacert.pem         # the CA certificate
crl             = $dir/certs/ca.crl             # the current CRL
private_key     = $dir/private/cakey.pem        # the private key
default_crl_days = 183
```

```
$ openssl ca -config conf/caconfig.cnf -gencrl -out crl/crl.pem
```

8. **Configure client-side certificates in web browsers**

   Convert pem certificate to p12:

   ```
   openssl pkcs12 -export -out newcerts/filename.cert.p12 -inkey private/filename.key.pem
   -in newcerts/filename.cert.pem -certfile certs/cacert.pem
   ```

   Next import the client-side certificates in your browser.

Configure cardholder data storage locations

This section describes the requirements for cardholder data storage locations and how to configure them.

PA-DSS requirement 9 mandates that cardholder data must not be stored on a server that is connected to the internet.

To comply with this requirement, you must do one of the following:

- Disable access from the internet to the server that hosts the Safer Payments instances. Remote VPN access (PA-DSS requirement 10) is not considered as access from the internet, if the VPN tunnel does not end directly on a server that hosts the Safer Payments instances.
- Place the data storage directories on a separate server computer that is not connected to the internet and in a different network zone. If you want to use a storage area network (SAN) instead, additional measures might be needed to achieve PCI DSS compliance. Contact your local Qualified Security Assessor (QSA) for details.

Note:

- You must disable the **locate** commands for the separate server computer. See “Disable locate for Safer Payments folders” on page 26 for details.
- Changes to the file storage locations are processed after a restart of a Safer Payments instance. Thus you can move the files while the instance is offline.

Configuration steps

Safer Payments can store cardholder data in a number of locations. To identify and adjust these locations, complete the following the steps.

1. Go to the Safer Payments user interface.
2. Click the **Cluster** tab.
3. Select a cluster instance from the table.
4. Scroll down to the **Storage** section.
5. For each cluster instance, the following directory locations can contain encrypted cardholder data:

Table 1. Default cardholder storage locations		
Path name (default)	**PAN stored as**	**PAN contained in**
Archive (arc)	encrypted	archived cases
Configuration	encrypted	conditions
Disk data cache (DDC)	encrypted	attributes and indices
Email (eml)	masked, PANs are potentially also encrypted	notifications and case actions
FLI buffer (fli)	encrypted	FLI messages
Investigation (inv)	encrypted	cases
Log (log)	masked	log messages
User (usr)	encrypted	user preferences
Relational database interface (rdi)	masked	DML statements

6. You can now change the directory locations according to your configuration.

   Note: The locations are different for each Safer Payments instance. You must adjust the locations individually for each cluster instance.

Exporting data using external Python programs

Safer Payments can be configured to feed data to external Python programs, which in turn can store that data on the local or a remote machine. If sensitive data is involved, additional measures have to be taken to protect that stored data. See “Python code execution” on page 49 for details.

Data export jobs

Data export jobs allow you to export transaction data to a csv file. For example, to use it as training data for an external AI model. Because of this use case, data export jobs offer the option to export encrypted data like PANs as clear text, masked, or hashed. See Figure 7 on page 22 for an example of those settings.

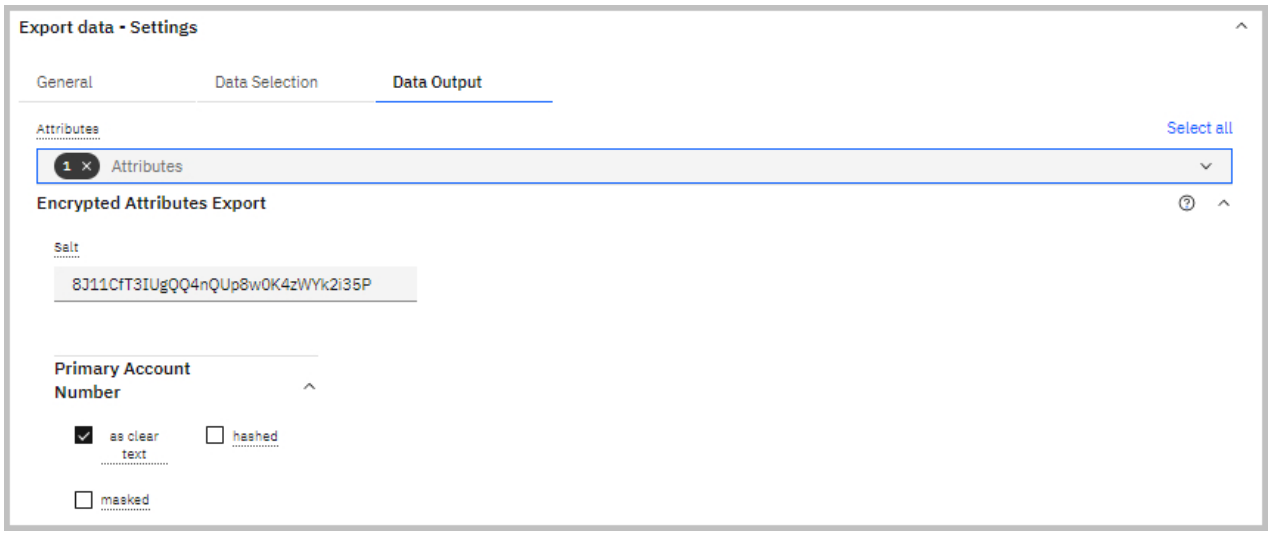


Figure 7. Data export options for encrypted attributes

The hashing algorithm used is SHA256. The job definition includes a salt that is added to the exported values before applying the hashing algorithm. This salt is usually randomly generated using the boost

library but can also be generated by a user. The salt is stored encrypted on disk and can only be viewed in the Safer Payments user interface if the user has the global privilege to change job definitions. If a user doesn't have the privilege, the salt is not delivered to the user interface and the field shows a random value with no meaning.

Whenever sensitive data is exported as clear text, you must make sure that the resulting export file is securely stored according to PCI DSS requirements 3.4.1, 3.5, 3.6, and all applicable subrequirements.

Attention: Users without the global privilege to view unmasked data inside of the application can still gain access to such data in clear text by accessing an exported file, if that file is not properly protected.

Whenever hashing is used together with masking, you must be aware that an attacker who gains access to the exported file and knowledge of the salt is able to reconstruct the plain text version of this data.

Simulation Query Data Export

The simulation query allows you to export transaction data to a .csv file. As with data export jobs, this data might be used to train an external AI model. For data that is encrypted within Safer Payments, it is possible to export the data as clear text, masked, or hashed.

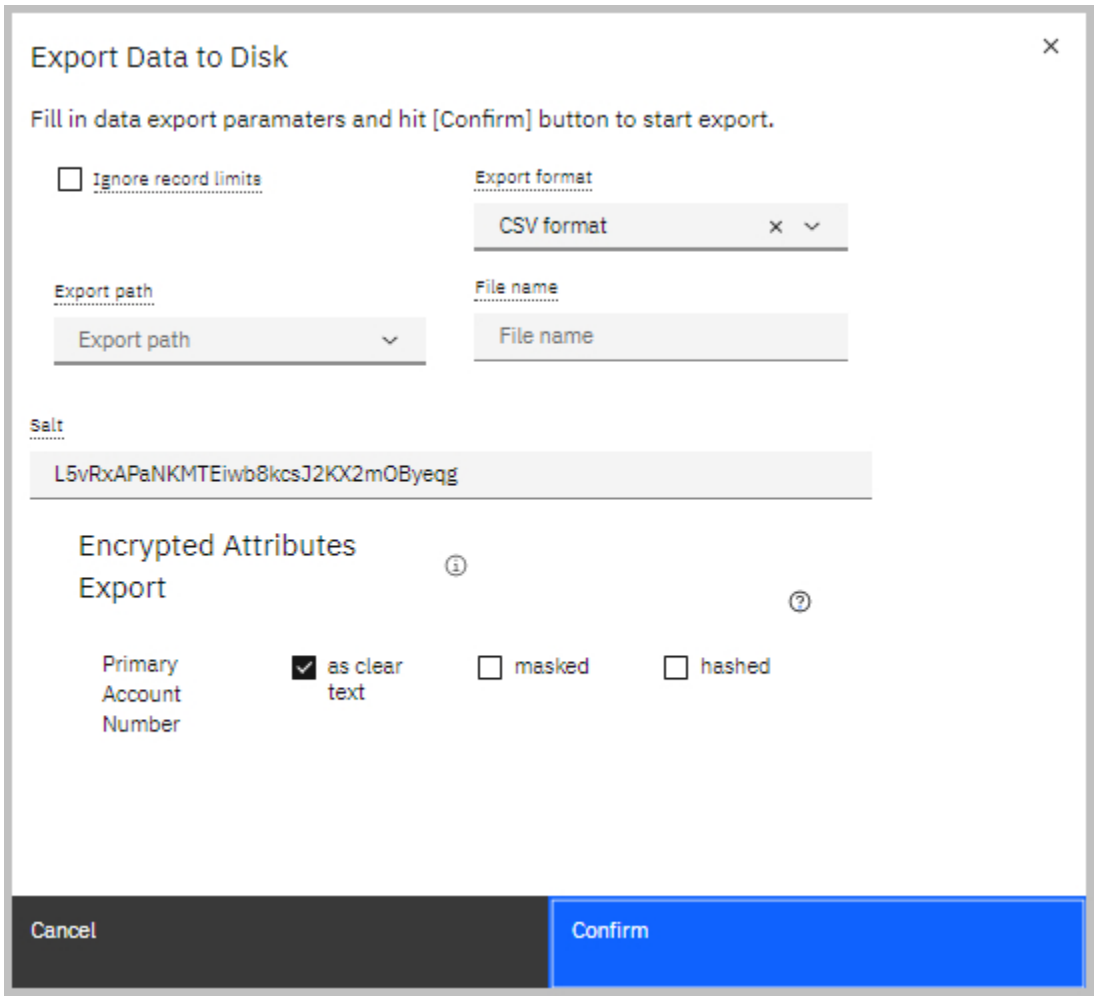


Figure 8. Simulation query data export options for encrypted attributes

The hashing algorithm that is used is SHA256, and as with data export jobs, the salt is usually generated randomly using the boost library but can also be specified by a user. The salt is stored encrypted on disk and can be accessed only within Safer Payments by users who have the privilege to see unmasked data. This can be set in the user account settings. Users without this privilege can only use reduced simulation query data export options.

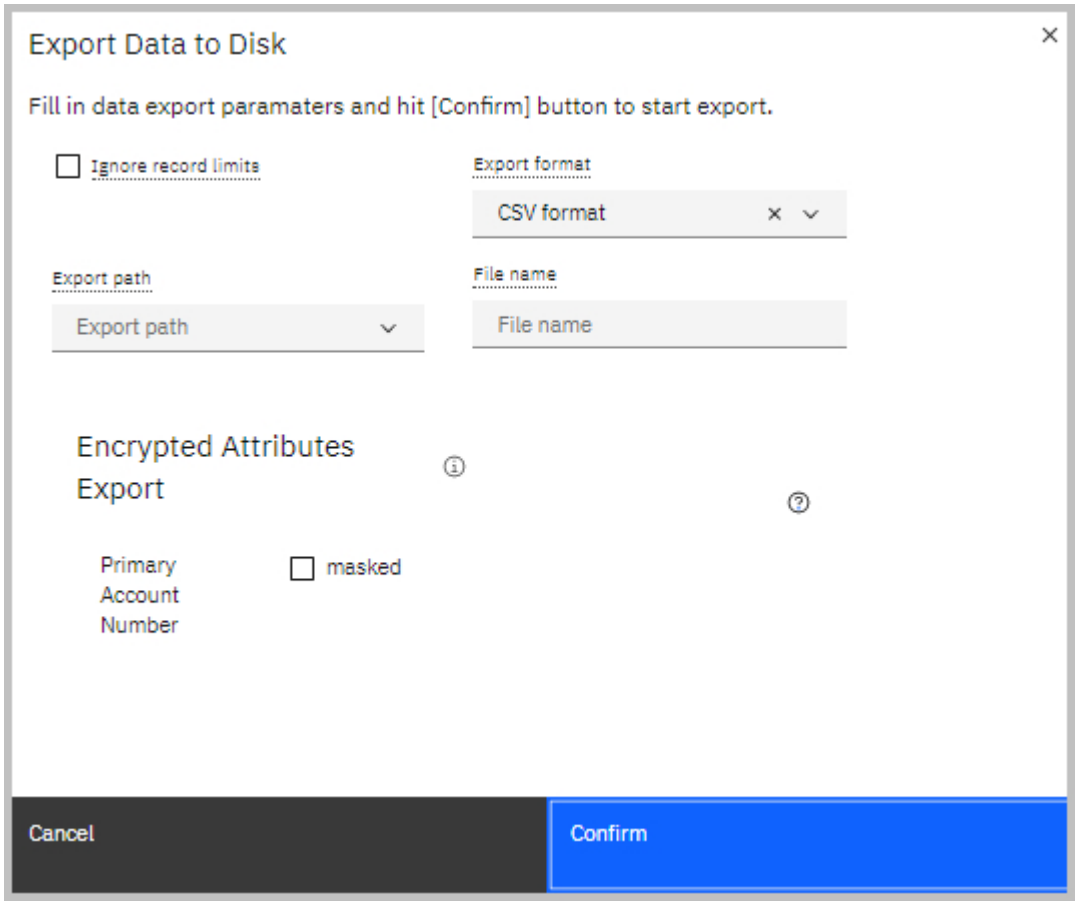


Figure 9. Reduced Simulation query data export options for users without the privilege to see unmasked data

Whenever sensitive data is exported as clear text, PCI DSS Requirements 3.4.1, 3.5, 3.6, and all applicable subrequirements mandate that the export file is securely stored.

Attention: Users without the global privilege to view unmasked data inside the application can still gain access to such data in clear text by accessing an exported file, if that file is not properly protected.

Whenever hashing is used together with masking, you must be aware that an attacker who gains access to the exported file and knowledge of the salt can reconstruct the plain text version of the data.

Copy settings to the other Safer Payments instances

When you have configured the initial cluster configuration as described in "Cluster instance configuration" on page 13, you must shut down the first Safer Payments instance.

1. Click the **Cluster** tab.
2. Click the checkbox for an entry.
3. Click the Shutdown (Shutdown) icon.
4. Copy the files `cluster.iris`, `settings.iris`, and every file that starts with `inbound_endpoint` from the `cfg` subdirectory of the first Safer Payments instance to the `cfg` subdirectories of all the other Safer Payments instances.
5. Delete the entire contents of the `fli` directory of the instance that you have used to create the files.
6. You must also assign SSL certificate files to each Safer Payments instance, as described in "Configure SSL encryption" on page 14.

7. When you have copied the configuration files to the other Safer Payments instances, you can now also start these instances as described in “Start and stop Safer Payments instances” on page 36.

Note: Do not send any cardholder data to Safer Payments yet, as configuration according to PCI DSS requirements is not complete.

Configure for operational use

This section describes how to configure Safer Payments for operational use.

Event logging

Users typically do not have access to the Safer Payments server. Safer Payments communicates its activities and status is generated through log messages.

In standard operations, these log messages are written to files where they can be viewed either directly using a text editor, by system tools, or Safer Payments itself.

Safer Payments contains a fully configurable event logging engine that supports three types of logging targets. The system and audit logs are Safer Payments logs. That is, Safer Payments has built-in viewer facilities to read these log messages.

System log
The system log informs about events relevant to technical operations of Safer Payments.

Audit log
The audit log traces relevant user activities.

Operating system logs
Operating system logs are sent to the operating system. In Linux/Unix operating systems such as RHEL, Safer Payments feeds operating system log messages to the local `syslogd` as "`IRIS_`*n*", where *n* is the ID of the Safer Payments instance, as defined by the command line parameter. Operating system logging is mandatory in PCI DSS compliant environments to facilitate centralized logging, and must be activated by selecting the "enable operating system logging" checkbox in the system configuration.

Make sure that all PCI DSS relevant log messages are forwarded to centralized logging as described in “Change log message settings” on page 39.

Note: If you use an IBM MQ or Kafka server to deliver data to Safer Payments, you must ensure that all relevant log messages are forwarded as well.

Swap disk configuration

Safer Payments is designed not to be swapped out by the operating system. However, the operating system itself determines whether Safer Payments memory is swapped to disk.

If Safer Payments memory is swapped out to disk, PAN data that is decrypted in RAM might temporarily be written to the swap file on disk. You must wipe all swap data securely after each new system restart or use an encrypted swap disk. You must also disable indexing of file contents.

For information about decreasing the swappiness of the system, see “Decrease swappiness” on page 29.

Wipe swap disk script

Note: Use this approach only, if swap disk encryption is not possible for certain reasons.

1. To find out the right path of your swap disk partition enter:

```
# fdisk -l#
cat /proc/swaps
```

2. If you have your swap partition name, write a small script, which runs on every startup using `sswap`, where */dev/sdaX* must be replaced by the path shown in the previous step.

```
# swapoff /dev/sdaX
# sswap -vll /dev/sdaX
# swapon /dev/sdaX
```

3. Add this code to a script. For example, to: `/usr/local/sbin/wipeSwap.sh`

```
chmod +x /usr/local/sbin/wipeSwap.sh
```

4. Add the script name `/usr/local/sbin/wipeSwap.sh` at the bottom of your init script `/etc/rc.local`.

Encrypt swap disk

With this preferred approach, you do not have to wipe out your swap on each system start.

1. Edit `/etc/fstab` to reflect the changes. Comment or delete previous swap entries before you add the new entry.

```
# vim /etc/fstab
/dev/mapper/swap none swap defaults 0 0
```

2. Create a `/etc/crypttab` file, and add the swap parameters.

```
# vim /etc/crypttab
swap /dev/volume /dev/urandom swap,cipher=aes-cbc-essiv:sha256
```

   Depending on your volume, group names, and layout, change the path to suit your needs. In most cases you only have to replace *volume* with the path you have commented or deleted in step 1. The encryption system then uses AES and SHA256 bit encryption during startup, with a random key. A new key is generated each time that the server is started.
3. Reboot the server to enable swap disk encryption.
4. Verify that swap disk encryption is enabled with the **lsblk** command.

```
# lsblk
```

Disable locate for Safer Payments folders

locate is a daemon, which creates a database with file contents. Make sure that either applies:

- **locate** is not installed.
- **locate** is disabled on the operating system.
- All Safer Payments folders are excluded from the **locate** search paths.

To change the **locate** search paths, edit the `/etc/updatedb.conf` file and add the Safer Payments folder to **PRUNEPATHS**.

Assuming PRUNEPATHS is set to:

```
PRUNEPATHS="/afs /media /net /sfs /tmp /udev /var/cache/ccache
/var/spool /var/tmp"
```

And you have installed Safer Payments to `/instancePath` as described in “Installing Safer Payments for the first time” on page 6.

Change PRUNEPATHS to:

```
PRUNEPATHS="/afs /media /net /sfs /tmp /udev /var/cache/ccache
/var/spool /var/tmp /instancePath"
```

Note:

- If you have adapted Safer Payments file storage locations during cluster configuration as described in “Configure cardholder data storage locations” on page 21, you must also add folders that are used outside `/instancePath` to PRUNEPATHS. Keep in mind that file locations can be configured differently per cluster instance.
- If any such folders are located on separate servers, PRUNEPATHS must be adjusted on those servers as well.

Miscellaneous system settings

The following extra configuration steps are necessary before you start Safer Payments.

1. Log in as root.
2. The default number of maximum open file descriptors on CentOS/RHEL systems is 1024 per process for normal users. To verify the limit that is valid for your system, run

   ```
   # cat /proc/sys/fs/file-max
   ```

3. Open the file `/etc/security/limits.conf`
4. If the recommended limit of 16384/32768 is valid for your system, add the lines

   ```
   SPUser hard nofile 32768
   SPUser soft nofile 16384
   ```

   Where *SPUser* is the name of the user account that you intend to run Safer Payments under.
5. To enable Safer Payments to run priority-based thread scheduling, you must also add

   ```
   SPUser - rtprio 20
   ```

6. Safer Payments starts numerous CPU threads for parallel processing of messages and simulations. To ensure, that the operating system can handle all threads, you must increase the number of maximum user processes. To do so also add the line

   ```
   SPUser - nproc 8192
   ```

7. Summary of necessary changes to `/etc/security/limits.conf`. In this example, the user name of the process running Safer Payments is *SPUser*.

   ```
   SPUser hard nofile 32768
   SPUser soft nofile 16384
   SPUser - rtprio 20
   SPUser - nproc 8192
   ```

8. Save `/etc/security/limits.conf` and reboot.

Firewall settings

Before you start Safer Payments, check your firewall settings to allow IP messaging between the Safer Payments cluster instances and other systems.

To change your local firewall settings, use **`firewall-cmd`** on RHEL7.

For general information on how to secure your operating system see:

- RHEL 7

  https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/index.html
- Oracle Linux 7

  https://docs.oracle.com/cd/E52668_01/E54670/E54670.pdf

Deferred writing and ultra-large memory configuration

Transparent Huge Pages (THP) is a Linux memory management system that reduces the overhead of Translation Lookaside Buffer (TLB) lookups on machines with large amounts of memory by using larger memory pages.

If you are using ultra-large main memory configurations and deferred writing (Safer Payments UI / Administration / System Configuration / Deferred Writing) on Linux operating systems, you might experience faster restarts and more stable latencies when disabling transparent huge pages. Transparent huge pages might block the memory for seconds, when it is defragmenting the RAM. In this time, it is not possible to make even small memory allocations.

Be careful with this setting. It can also result in slower overall message computation, when deferred writing is disabled.

To disable transparent huge pages temporarily, run the following command:

```
echo never > /sys/kernel/mm/transparent_hugepage/defrag
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

Next restart Safer Payments. To check, if transparent huge pages are disabled run the following command:

```
cat /sys/kernel/mm/transparent_hugepage/enabled
```

The output is:

```
always madvise [never]
```

In RHEL 7, you can disable transparent huge pages with the command **tuned**.

See https://access.redhat.com/solutions/1320153 for details.

You can query the current active profile with:

```
# tuned-adm active
Current active profile: latency-performance
```

To create a customized profile, create a new directory in the `/etc/tuned` directory with the wanted profile name.

```
# mkdir /etc/tuned/myprofile-nothp
```

Next, create a new `tuned.conf` file for *myprofile*-nothp, and insert the new tuning information.

```
# cat /etc/tuned/myprofile-nothp/tuned.conf
[main]
include= latency-performance
[vm]
transparent_hugepages=never
```

Next, make the script executable:

```
# chmod +x /etc/tuned/myprofile-nothp/tuned.conf
```

Next, enable *myprofile*:

```
# tuned-adm profile myprofile-nothp
```

This change immediately takes effect and persists a reboot.

Increase virtual memory map size

Linux restricts the maximum number of memory maps per process. The default is 65535 on RHEL 7.

This value might be enough for most Linux applications, but depending on the Safer Payments configuration and internal data allocation sizes, Safer Payments might actually need more memory maps. If the application has reached its maximum number of memory maps, a "cannot allocate memory" error message and other errors occur, even if enough free RAM is available in the system. To avoid this situation, run:

```
echo 1048576 > /proc/sys/vm/max_map_count
```

This command temporarily applies the new maximum number of virtual memory maps to 1048576. If you want to increase the value permanently, you must add this value as `vm.max_map_count=1048576` to the file `/etc/sysctl.conf` after server restart. To check if the configuration was applied correctly run:

```
cat /proc/sys/vm/max_map_count
```

If you have a very large configuration, you should monitor the current number of memory maps during high load. If the number of memory maps exceeds half of its maximum value, increase this value. To monitor the number of memory maps for your Safer Payments process, run:

```
cat /proc/IRIS_PID/maps | wc -l
```

Decrease swappiness

Safer Payments produces larger latencies, if the system uses swap memory.

Therefore, it is recommended to reduce the swappiness on a Linux system.

- To temporarily change the setting run

  ```
  sysctl -w vm.swappiness=1
  ```

- To permanently change the setting add

  ```
  vm.swappiness=1
  ```

  to `/etc/sysctl.conf`

Data encryption

This section describes how to enable data encryption, to be compliant with PA-DSS.

Activate data encryption - step 1

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **System** > **Configuration** from the left navigation pane.
3. Click the **System** tab. Scroll down to the **Encryption** section.

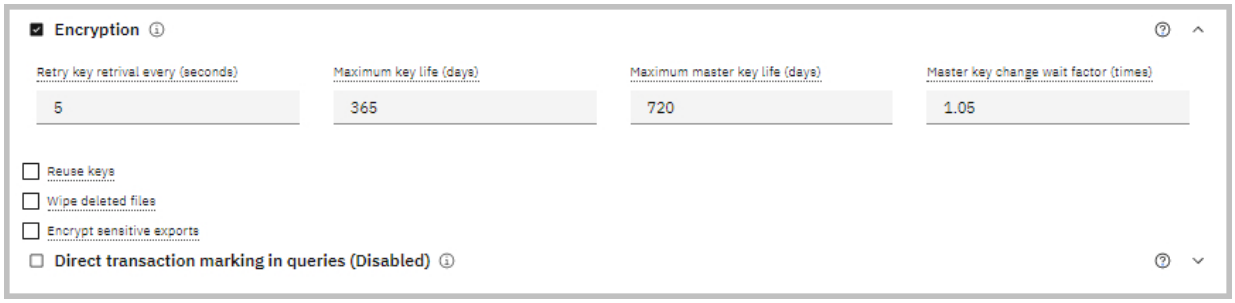


Figure 10. Encryption section

4. Clear the **Reuse keys** checkbox.
5. Select the **Wipe deleted files** and **Encrypt sensitive exports** checkboxes.

Encryption covers the actual production data and certain parts of the configuration where PANs are expected, for example, in conditions and audit trails. Other parts of the configuration are not encrypted. This implies that you must never store clear PAN in any name or comment field of Safer Payments.

The PA-DSS standard recommends defining a maximum cryptoperiod after which a key must be replaced with a new one. See “Enforce regular key changes ” on page 57 for details.

According to PA-DSS requirement 2.3, PANs must be rendered unreadable anywhere they are stored. Therefore, you must enable **Encrypt sensitive exports**.

Set up key entry and key management users

Before you proceed, you must set up user accounts in Safer Payments, which have sufficient privileges to manage data encryption.

According to PA-DSS requirement 2.5.6, two key holders are required. One must be granted the global privilege **left public key entry**, the other one the global privilege **right public key entry**.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **User management** > **Accounts** from the left navigation pane.
3. Click the + (New user account) icon to create a new user account.
4. The **New User Account** form is displayed.

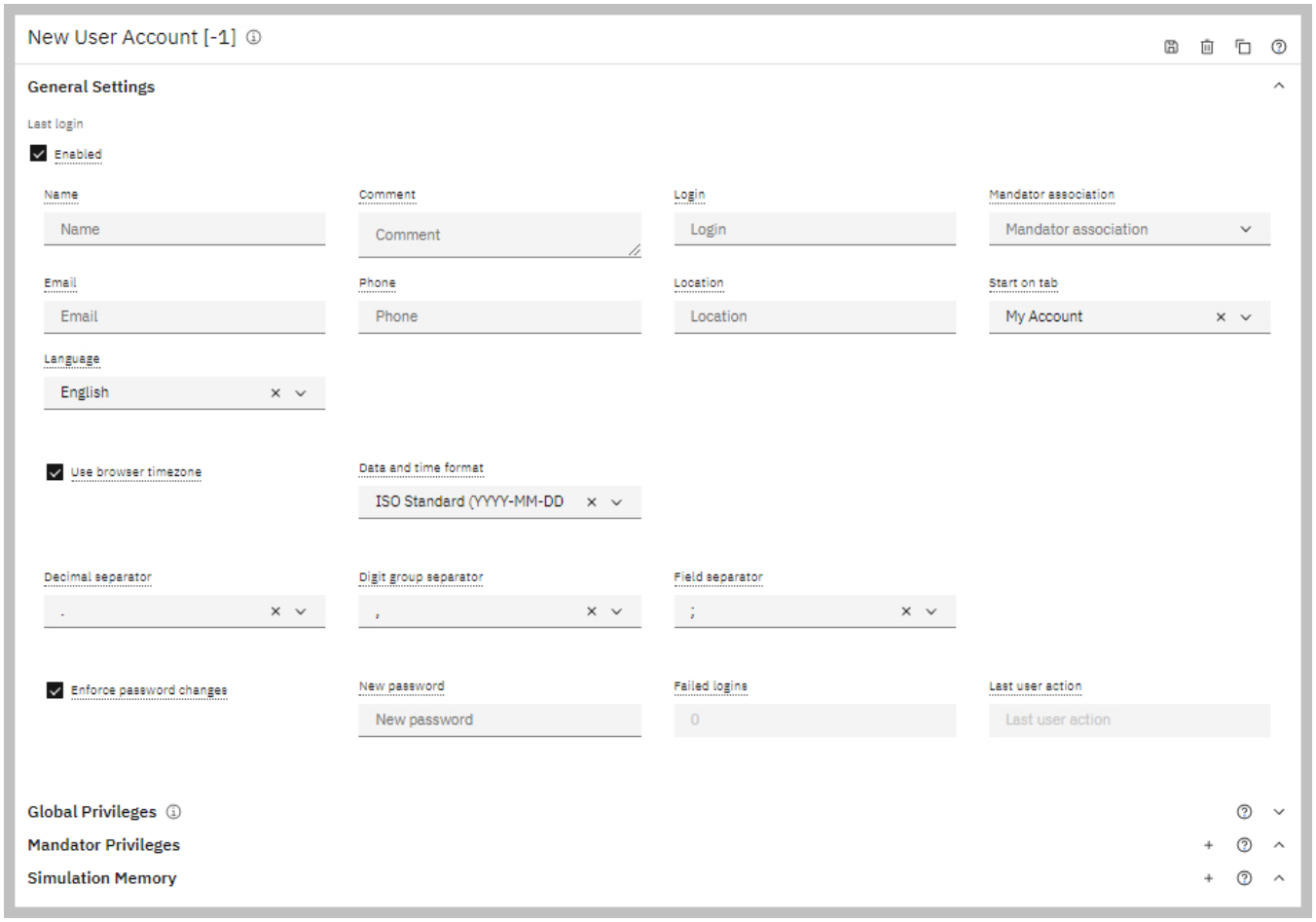


Figure 11. New User Account form

5. Select a mandator in **Mandator association**.

6. In the **Global Privileges** section, select **left public key entry** in the **Key entry** field.
7. Repeat the steps above and create a second user account. This time select the **right public key entry** in the **Key entry** field.

   Note: You cannot assign both left and right public key entry privileges to a single user.
8. There must be a user that is granted the privilege to **activate keys** in the **Key management** field. This privilege can be granted to one or both key holders, or any other user.
9. In the **Key management** field, assign the **change masterkey** privilege to a user. This is required for changing the master key as described in “Change the master key” on page 59.

   Important: The person who sets up the user accounts must make sure that the checkbox **enforce password changes** is selected.

Activate data encryption - step 2

Before you can use encryption in Safer Payments you must generate keys.

Key generation and distribution is described in “Key generation” on page 51.

After you have generated and distributed keys, you must

- Either restart all Safer Payments instances.
- Or

  1. Click the **Administration** tab.
  2. Select **Key management** > **Encryption keys** from the left navigation pane.
  3. Click the  (Reload private keys from disk) icon to reload private keys from disk.

Enable cardholder data encryption

In the next step, you must activate PA-DSS compliant encryption of cardholder data to be stored in Safer Payments. To comply with PA-DSS requirement 1 do not process sensitive authentication data.

If you intend to store the Primary Account Number (PAN) in Safer Payments, you must enable encryption for this data attribute in Safer Payments as defined in PA-DSS requirement 2.3.

Attribute names can be chosen freely in Safer Payments. However, in this documentation the attribute for the Primary Account Number is named "PAN".

1. Log on to Safer Payments with a user account as described in “Start the first Safer Payments instance” on page 12 that has been granted at least the following privileges:

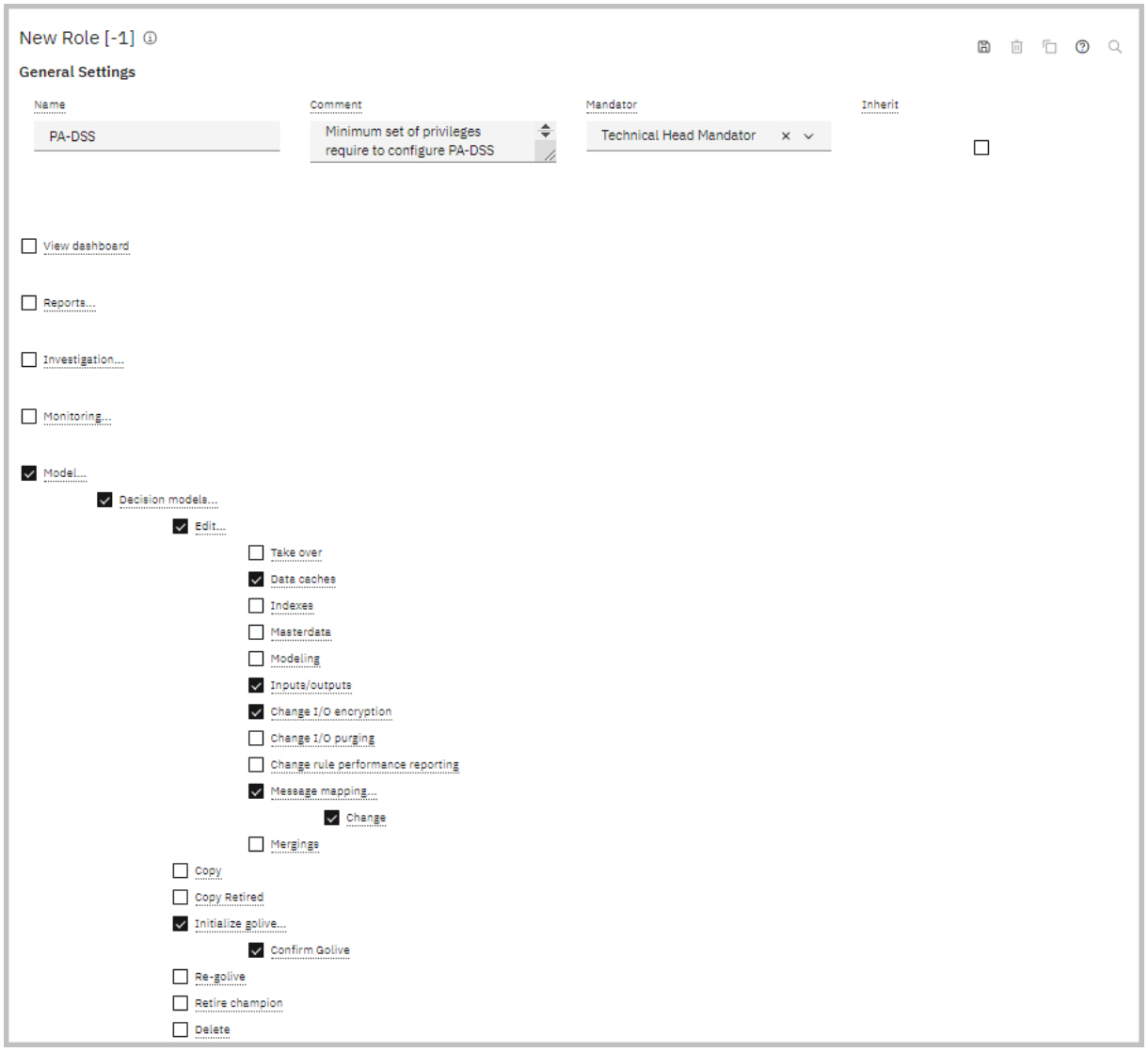


Figure 12. PA-DSS settings

Note: Refer to the online documentation for details of user access right administration.

2. On the Safer Payments user interface, click the **Model** tab.

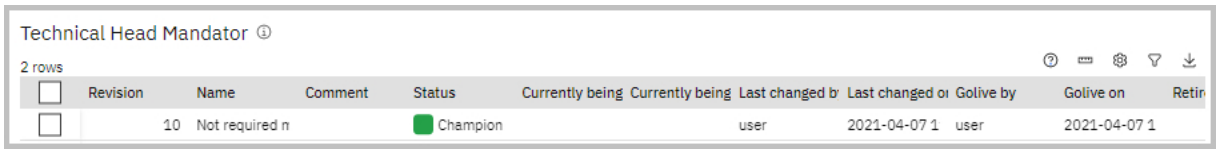


Figure 13. Model - Technical Head Mandator

3. Click the checkbox for the **Champion** entry.
4. Click the Copy (Copy) icon.
5. Click the newly created **Challenger** entry.
6. Select **Data model** > **Inputs** from the left navigation pane.

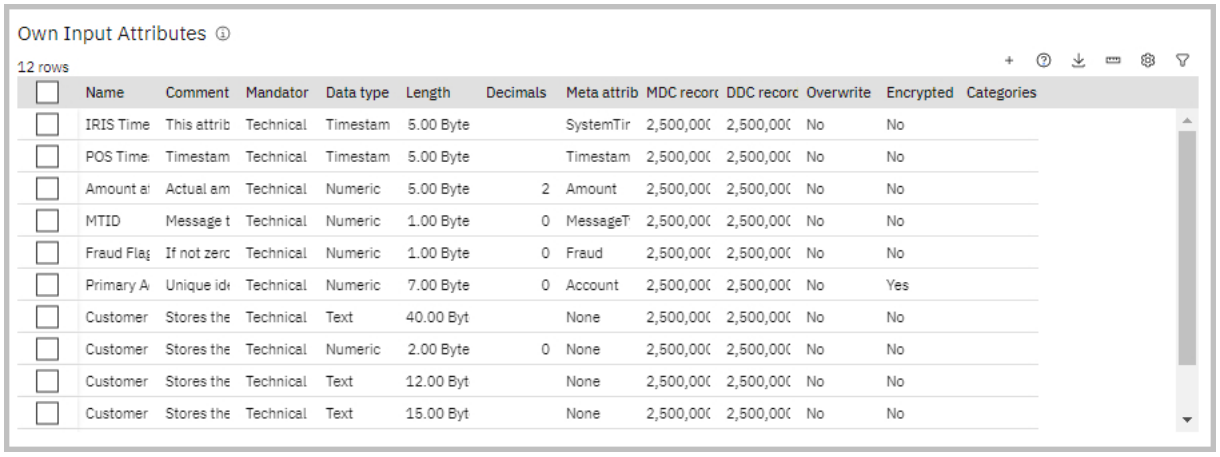


Figure 14. Own Input Attributes

7. Click the + (New input) icon to create a new attribute.
8. The **New Attribute** form opens.

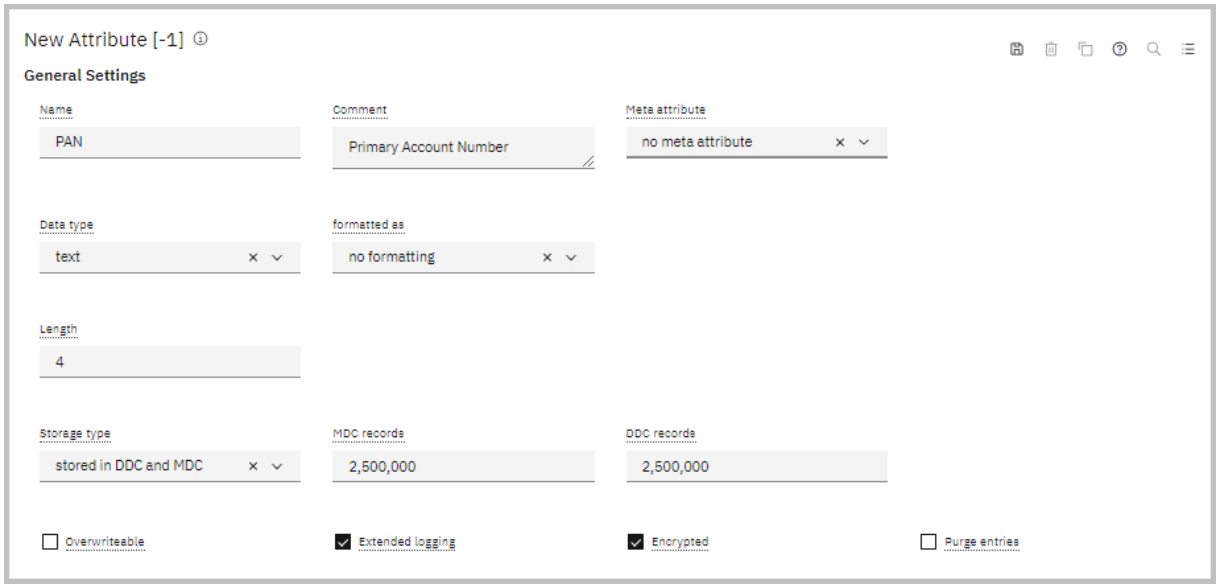


Figure 15. New attribute form

Note: To be PA-DSS compliant you must now enable encryption for the PAN attribute. For all other sections not relevant for PA-DSS refer to the online documentation.

9. Enter **PAN** in the Name field and select the checkbox in the **Encrypted** field.
10. Click the (Save) icon to save the new attribute.
11. You can now define other attributes that are required for your specific Safer Payments application. To comply with PA-DSS make sure that no sensitive authentication data is defined.
12. Next, select **Review overview** > **General** from the left navigation pane.

Figure 16. Activate changed revision

13. Click the ▷ (Golive) icon. The decision model is now being activated and all data that is stored in the PAN attribute is encrypted.

Key life expiration

PA-DSS requirement 2.5.4 mandates regular key changes. Therefore, Safer Payments automatically shuts down, if the maximum key life of a key is reached. You can implement Status Alarm Indicators, that alert you of keys that are expiring soon. See “Enforce regular key changes ” on page 57 for details on key changes and setting up SAIs.

Miscellaneous Safer Payments configuration settings

To meet various PA-DSS requirements, further configuration settings must be changed.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Locate the **User Account** section.

Figure 17. User account settings

3. Select the **Password must contain lower case**, **Password must contain upper case**, and **Password must contain digits** checkboxes.
4. Click the **Interfaces** tab and scroll down to the **Application Programming Interface** section.

Figure 18. API settings

5. Select the **Cross-site request forgery protection** checkbox. Make sure that **Session timeout (seconds)** is set to a value of 900 seconds or less.

   Note: If you want to use tested default and secure HTTP headers, ensure that **Use custom HTTP headers** is disabled.

6. Scroll up to locate the **Message Tracing** section under **Message Command Interface**.

Figure 19. Message Tracing settings

7. Select `Disabled` from the **Dump message data** drop-down list.

   Note: Dump message data needs to be disabled to comply with PA-DSS requirement 2.3.

8. Scroll down to locate the **IBM MQ Interface** section.

Figure 20. IBM MQ Interface settings

9. Select `Disabled` from the **Dump message data** drop-down list.

   Note: Dump message data needs to be disabled to comply with PA-DSS requirement 2.3.

10. Scroll down to the **Kafka Interface** section.
11. Select `Disabled` from the **Dump message data** drop-down list.

    Note: Dump message data needs to be disabled to comply with PA-DSS requirement 2.3.

12. Click the **Misc** tab and scroll down to the **Miscellaneous** section.
13. Verify that **SSL cipher list** has the following entries:

```
ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
```

    Note: This list might be outdated because new security leaks have been discovered in the meantime. The OpenSSL website provides regular security advisories, including information about potential security leaks.

    http://www.openssl.org/

14. If you are using the relational database interface, you must enable masking values of encrypted attributes.
15. Click the **Administration** tab and select **Mandators** > **Settings**. Select a mandator.
16. Scroll down to the **Relational Database Interface** section.

Figure 21. Relational Database Interface settings

17. Select the **Mask encrypted values** checkbox.

    Note: Repeat for all mandators.

18. Click the **Cluster** tab and select **Interfaces** > **Outgoing channel configurations** from the left navigation pane.

19. For every outgoing channel configuration, select the **Mask values** checkbox unless you have a business need not to do so.

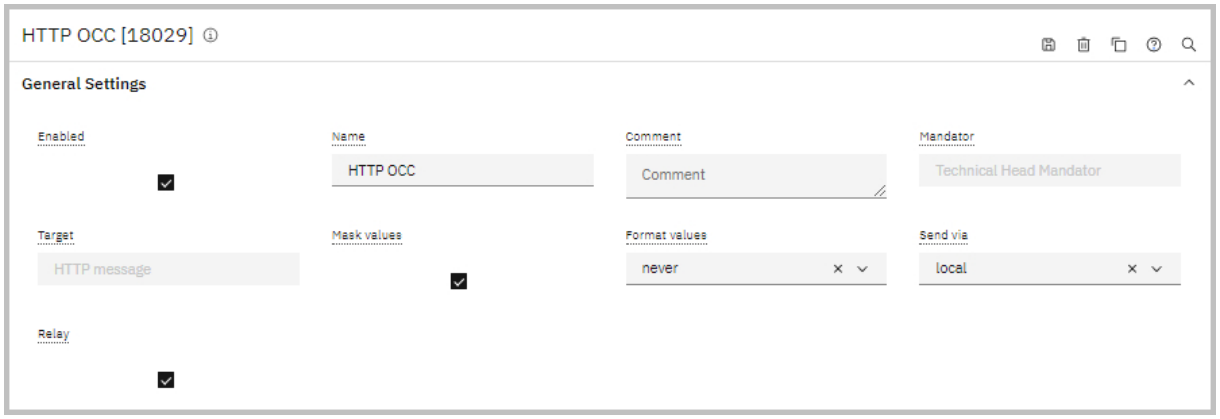


Figure 22. Outgoing channel configuration settings

Note: If you disable the **Mask values** option make sure that only notifications, case actions, and external queries with a business need are associated with that outgoing channel configuration. If you store unmasked data received via an outgoing channel configuration, you must make sure to protect it according to PCI DSS requirements 3.4.1, 3.5, 3.6 and all applicable subrequirements.

Implementing malware scanning on files

It is a best practice to implement a third-party antivirus tool to scan case attachment files for malware when they are uploaded.

Users can upload various types of files as case attachments, including malicious files. IBM Safer Payments does not validate case attachment files before they are uploaded. It also does not interpret or execute the uploaded files. Scanning the uploaded files for malware is outside the scope of IBM Safer Payments .

To mitigate the risk and protect the server where IBM Safer Payments is installed, implement a third-party antivirus scanning tool on the operating system level.

Case attachments are stored in the `inv/inv[mandator UID]/` folder. The file name format is `investigation_attachment_[timestamp]_[case UID]_[attachment number].iris`. The UIDs and attachment numbers in the file name are padded with leading zeros.

Do not run the scanning tool on any other IBM Safer Payments folders or files. It can negatively affect the operation of IBM Safer Payments .

Operation of Safer Payments

This section describes various tasks that need to be run during regular operation of Safer Payments.

Start and stop Safer Payments instances

This topic describes how to start and stop Safer Payments during regular operation.

Start a Safer Payments instance

Open a console window on the server and run:

```
iris console id=i
```

from */instancePath*/`cfg`.

- The `console` parameter activates event log message output on the console window.

- */instancePath* is the path where you want the instance configuration to be stored.
- The parameter *i* is the ID of the instance that you want to start.

  Note: Both home directory and instance ID have been defined in “Start the first Safer Payments instance” on page 12.

Safer Payments now starts up. You must enter the SSL certificate password when you are prompted.

Each Safer Payments instance now attempts to fetch the password for the encryption keys from its sister instances. If no other instances are running yet, the encryption keys must also be entered before full operation can start. See “Activate keys” on page 55 for details on key entry and activation.

Stop a Safer Payments instance

To stop Safer Payments, you can use a SIGTERM command. Safer Payments catches SIGTERM signals and performs a clean shutdown, similar to the API shutdown command.

Open a console window on the server and run:

```
killall iris
```

To immediately stop a Safer Payments process, you can use the SIGKILL signal. Only use the SIGKILL signal, if Safer Payments does not properly shut down after a SIGTERM command:

Open a console window on the server and run:

```
killall -9 iris
```

Securely delete outdated index entries

For certain functions, Safer Payments requires indexing certain data attributes.

If you require an index on the PAN attribute, old index attributes must be securely deleted after the retention period defined in “Preliminary considerations” on page 3.

Safer Payments can be configured to securely delete outdated index entries automatically, by enabling the outdated entries setting.

1. On the Safer Payments user interface, click the **Model** tab.
2. Click the checkbox for the **Champion** entry.
3. Click the  (Copy) icon.
4. Click the newly created **Challenger** entry.
5. Click **Data model** > **Indexes** on the left navigation pane.
6. From the **Own Indexes** section, click the **PAN** entry.

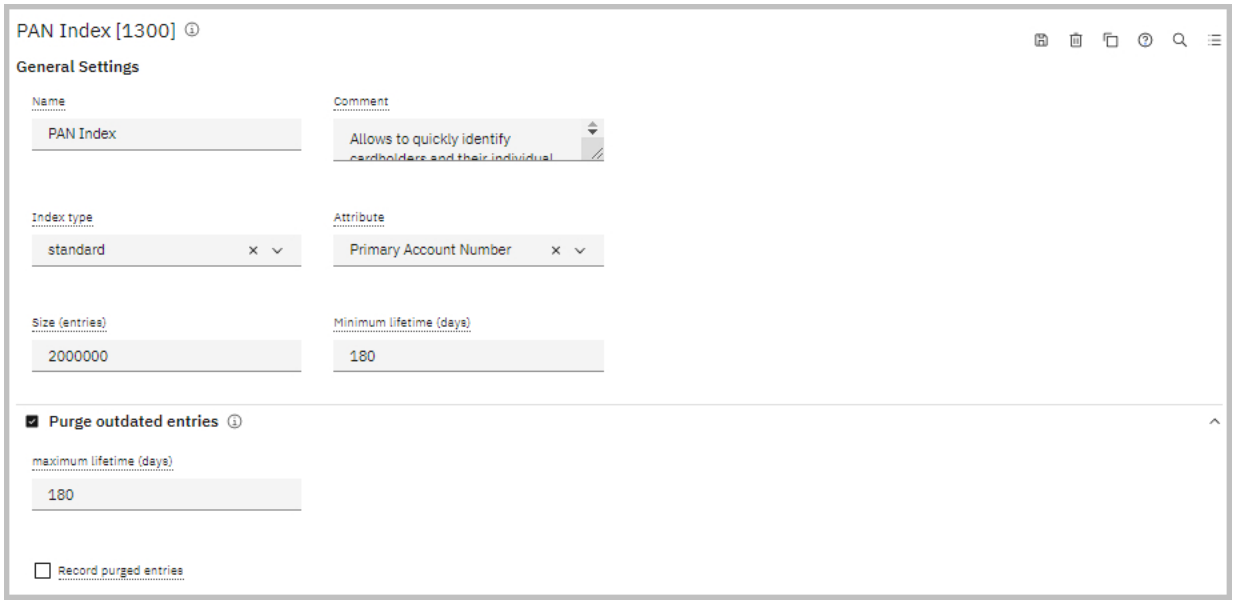


Figure 23. PAN Index section

7. Select the **Purge outdated entries** checkbox and enter a maximum lifetime.

Case archiving

Cardholder data must be securely deleted after the retention period defined in "Preliminary considerations" on page 3. If you enable case investigation, you must configure cases to be archived in Safer Payments no later than the end of the retention period.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **System** > **Configuration** from the left navigation pane.
3. Click the **Investigation & queries** tab. Click the **Case Investigation** checkbox.

Figure 24. Case investigation settings section

4. In the **Archive cases after (days)** field, you must enter a value that is equal or smaller to the retention period that you defined.

According to PA-DSS requirement 2.3, PANs must be rendered unreadable anywhere they are stored. Case attachments are stored decrypted. You must implement proper operational procedures to ensure that no PANs are stored in a case attachment or you must disable case attachments.

Change log message settings

To change the log message settings, you must perform the following steps:

1. On the Safer Payments user interface, click the **Cluster** tab.
2. Select **System monitoring** > **Event log messages** from the left navigation pane.

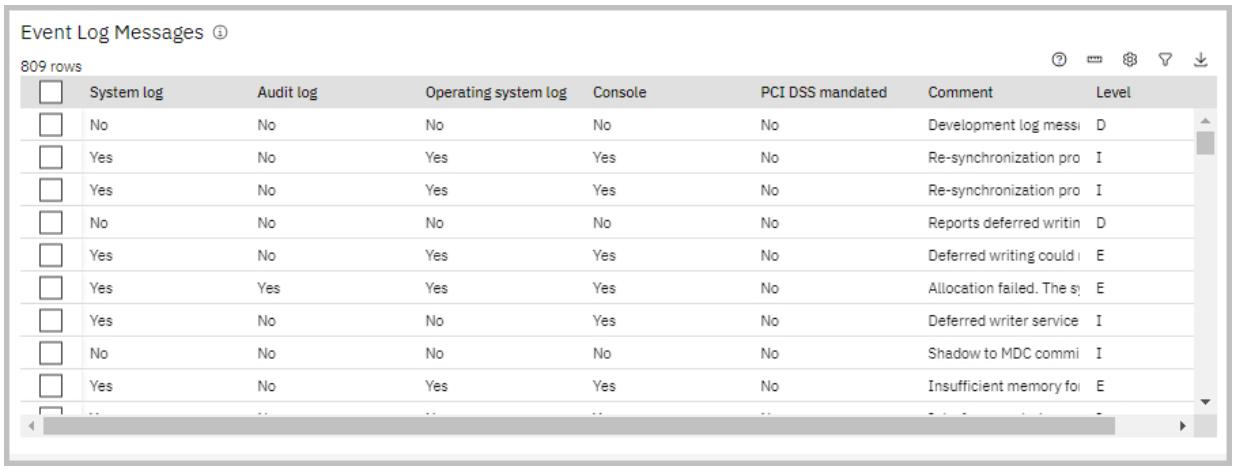


Figure 25. Event Log Message settings

3. Click a log row, for example 9, to adjust the settings. This must be done for each log message individually.

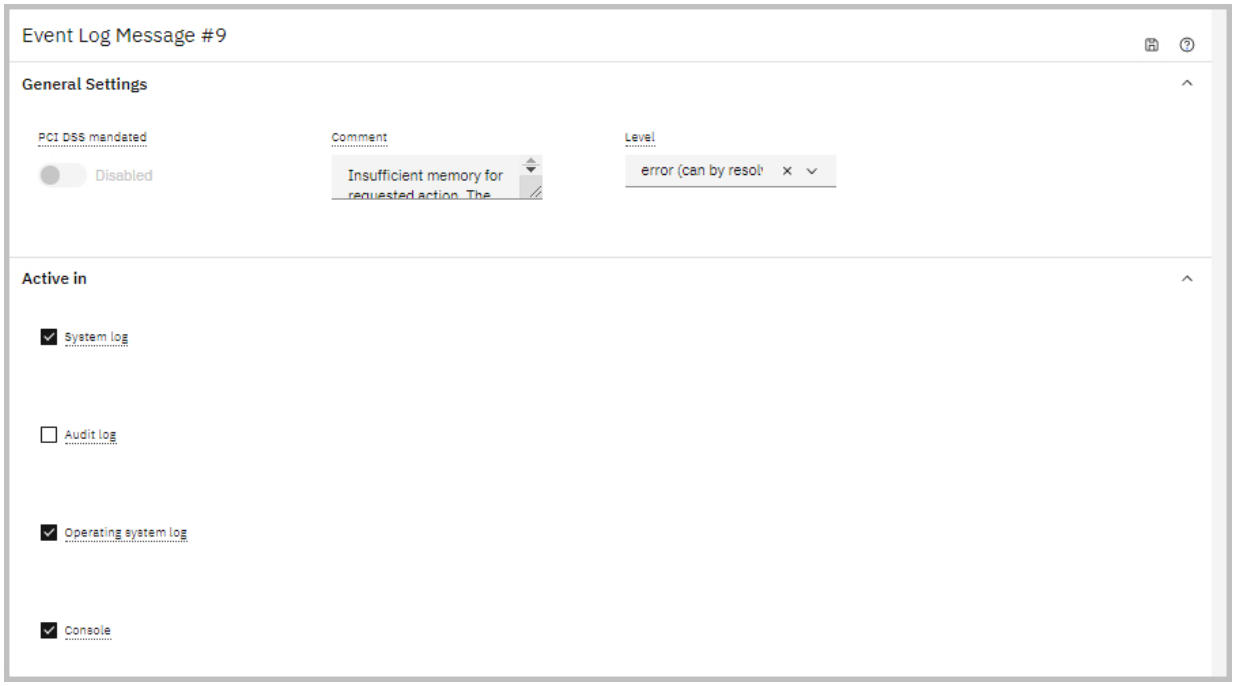


Figure 26. Event log message 9

4. Make your changes and click the (Save) icon.

Note: Your central log server must collect all relevant log messages from the system log. You must implement an operational process within your organization to collect the relevant logs from the operating systems logs.

For PCI DSS compliance, a minimum set of log messages must be forwarded to centralized logging. To do so the **Operating system log** checkbox must be selected for each message. The following log messages are mandatory for PCI DSS compliance:

14, 15, 16, 17, 38, 41, 70, 71, 90, 91, 92, 93, 100, 129, 131, 157, 194, 195, 196, 197, 198, 199, 200, 201, 209, 210, 211, 212, 213, 229, 251, 322, 324, 325, 364, 365, 366, 367, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400®, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 417, 418, 419, 420, 425, 426, 427, 428, 429, 430, 434, 435, 436, 437, 448, 449, 456, 457, 460, 461, 462, 463, 468, 471, 490, 508, 509, 510, 511, 517, 518, 519, 520, 523, 524, 530, 531, 535, 536, 537, 541, 542, 543, 565, 573, 575, 577, 578, 580, 581, 585, 586, 587, 591, 594, 595, 598, 599, 819, 834

These log messages are set to the correct values by default after the initial installation of Safer Payments. To enable additional log messages for centralized logging you must select the **Operating system log** checkbox for each corresponding message.

Note: If you disable these log messages your are not PCI DSS compliant.

Archiving and backup

Safer Payments does not automatically remove archived cases, log messages, and so on, from the file system.

This avoids the loss of the data before it is being archived for auditing reasons, according to the licensee's requirements.

However, PCI DSS requires purging cardholder data after the customer-defined retention period. Therefore, the licensee must implement a backup process for those files, and ensure that the archived files are purged before the end of the retention period.

"Configure cardholder data storage locations" on page 21 lists the directory locations, which might contain encrypted cardholder data, and to which such a backup/purging process applies.

Note: Backups must also be handled according to PCI DSS requirements.

Set user privileges

According to PCI DSS, the PAN must be displayed masked only, unless there is a legitimate business need to see the full PAN.

Full PAN visibility is controlled by the **Mask level** global privilege for each user.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **User management** > **Accounts** from the left navigation pane. Select a user.
3. Scroll down to the **Global Privileges** section.

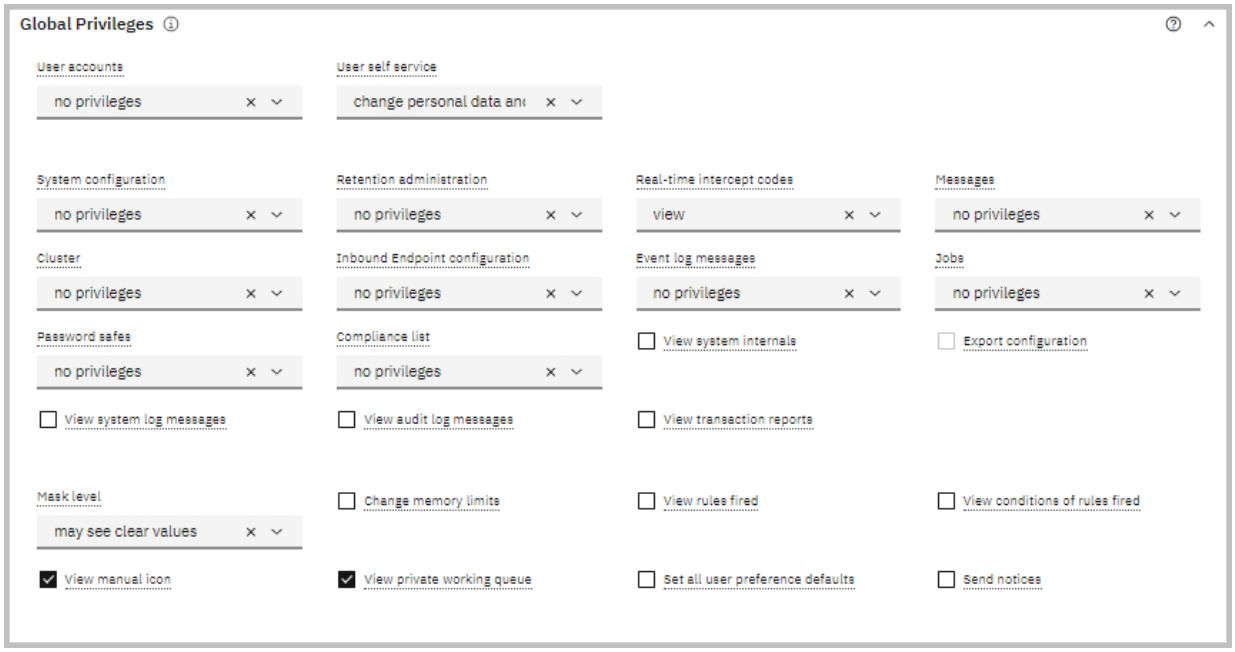


Figure 27. Global privileges section

4. Select **may see clear values** in **Mask level**.

In addition to global privileges, certain functions of Safer Payments can be accessed only by users with a legitimate business need. You can grant certain privileges to such users. More precisely, model revisions, report, and query definitions must be viewed only by privileged users. However, non-privileged users can still run reports and queries.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **User management** > **Roles** from the left navigation pane.
3. Click the user role that you want to change in the roles table.

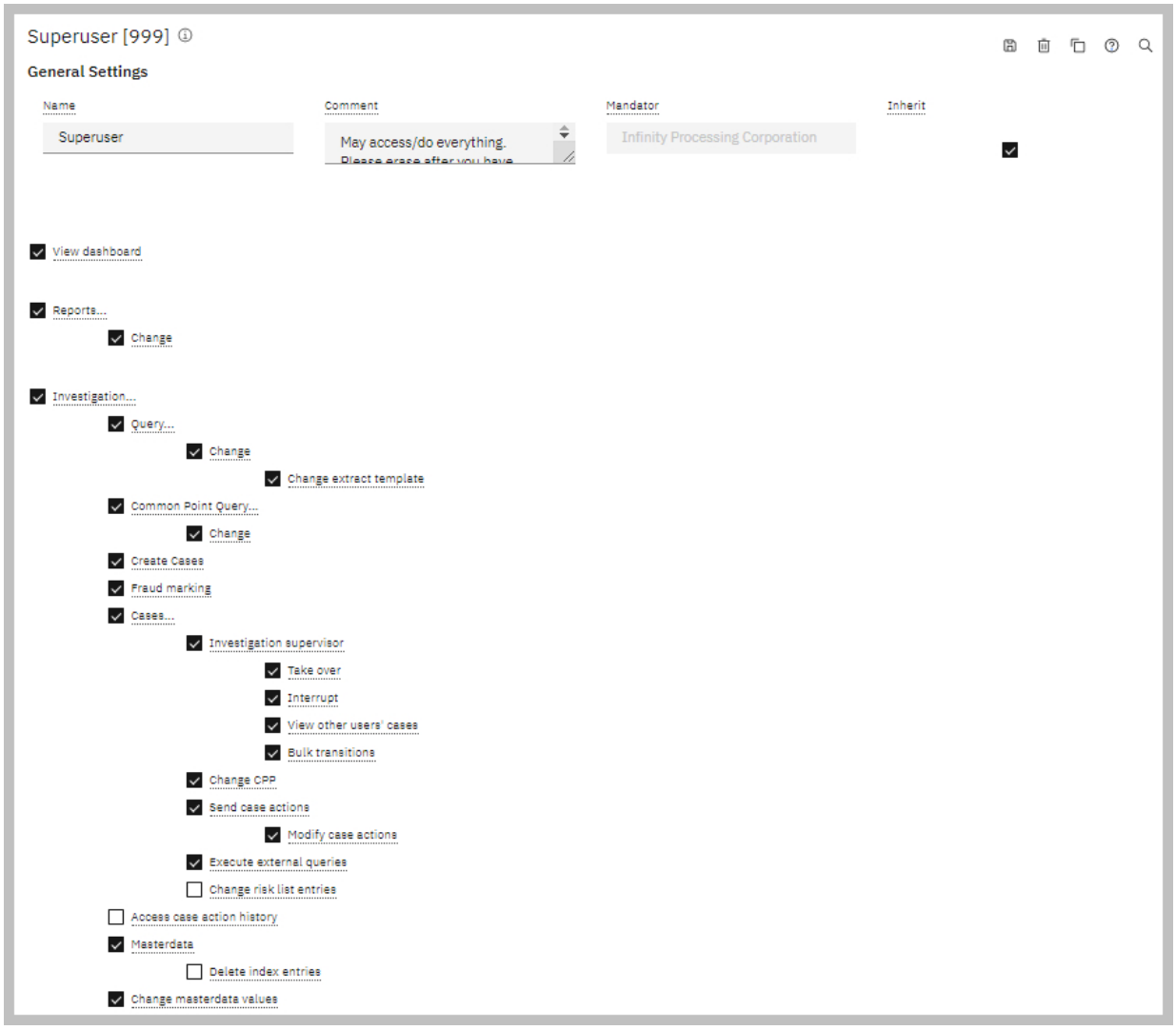


Figure 28. Superuser settings

4. Change the privileges according to your requirements by selecting the appropriate checkboxes.

Using a secure wipe tool

Various PCI DSS requirements demand the use of a secure wipe tool to securely delete sensitive authentication data and cardholder data.

According to PA-DSS requirement 1.1.4, the disk wipe tool must be in accordance with industry accepted standards for secure deletion. The National Security Agency, for example, maintains a list of approved products.

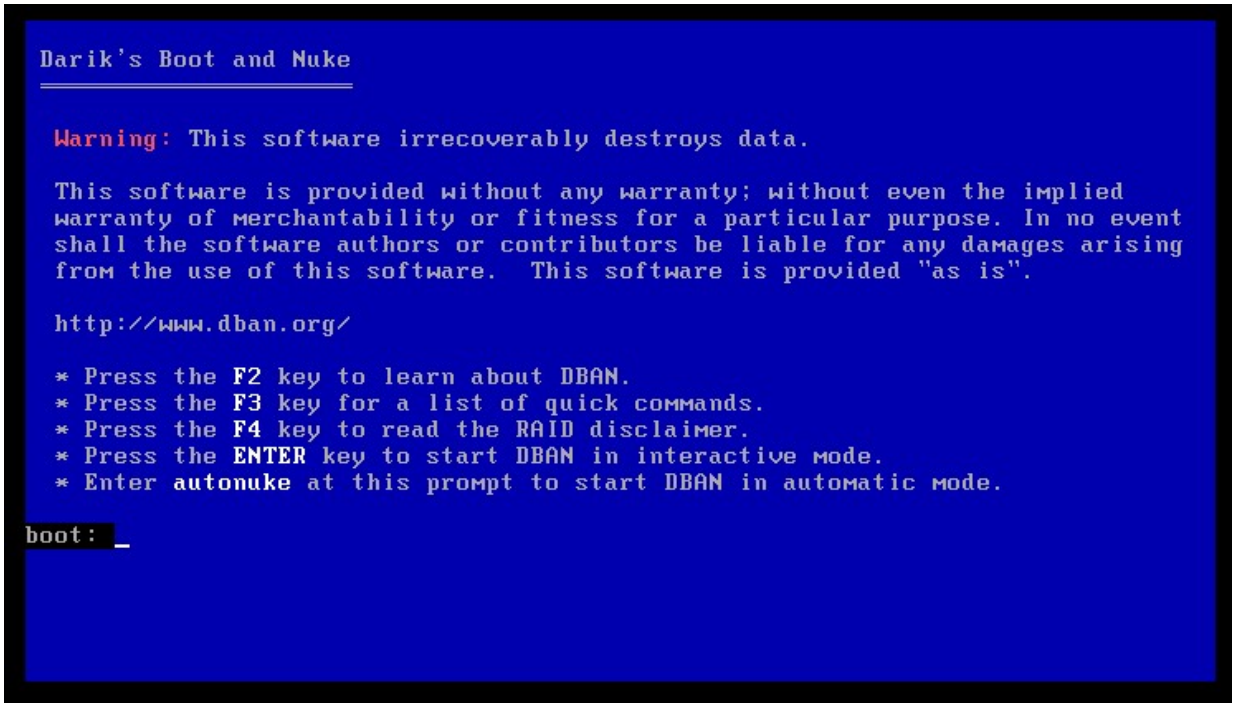
To securely wipe entire hard disks, you can use the "DBAN" tool.

To securely delete single files or directories you can use the Linux tool "Wipe".

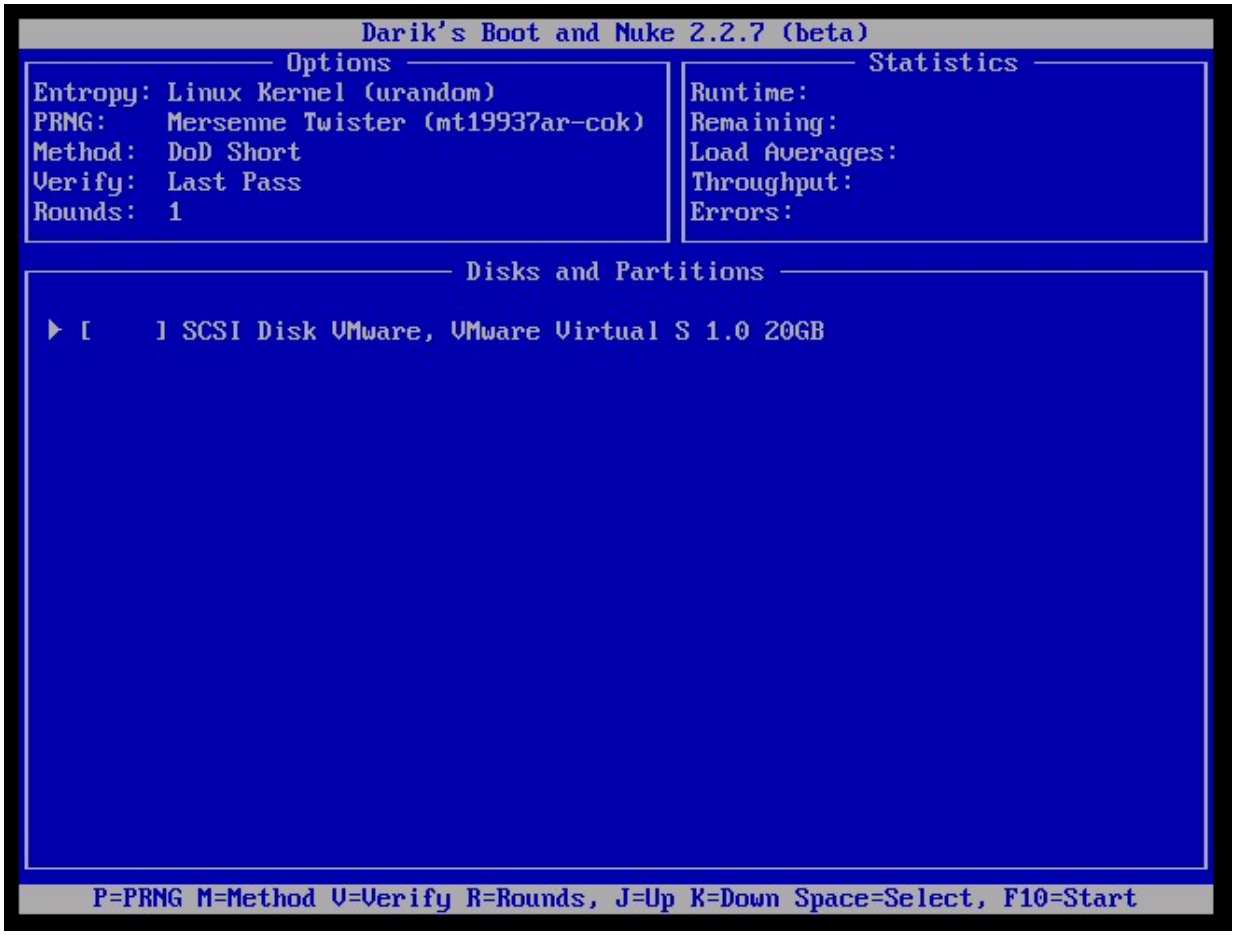
Using the DBAN tool

You can download DBAN from http://www.dban.org/.

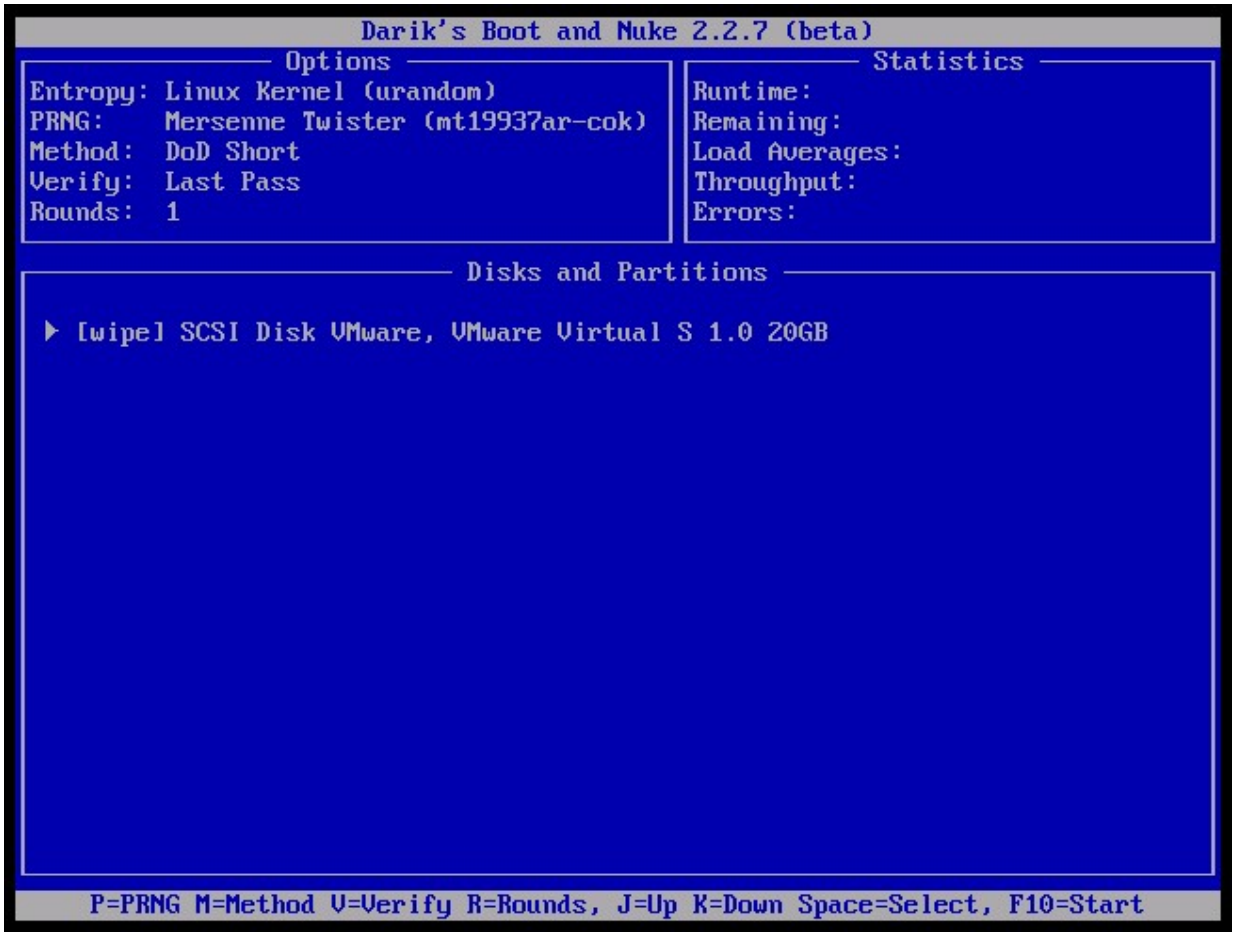
1. Create a CD with the ISO image of DBAN.
2. Boot the computer that hosts the device you want to wipe securely.
3. Press the **ENTER** key to start DBAN in interactive mode.



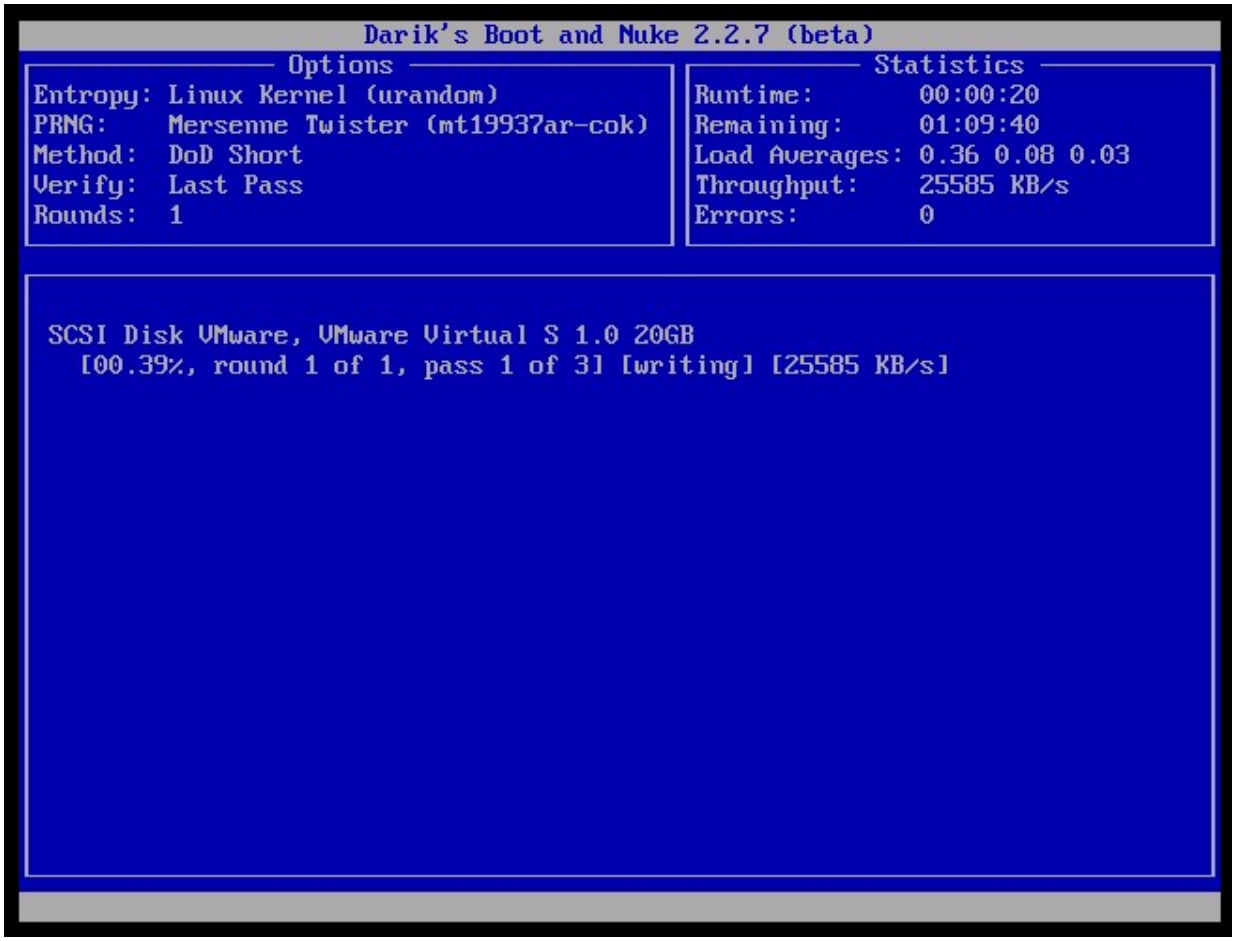
4. Type M and select the **DoD Short** method.



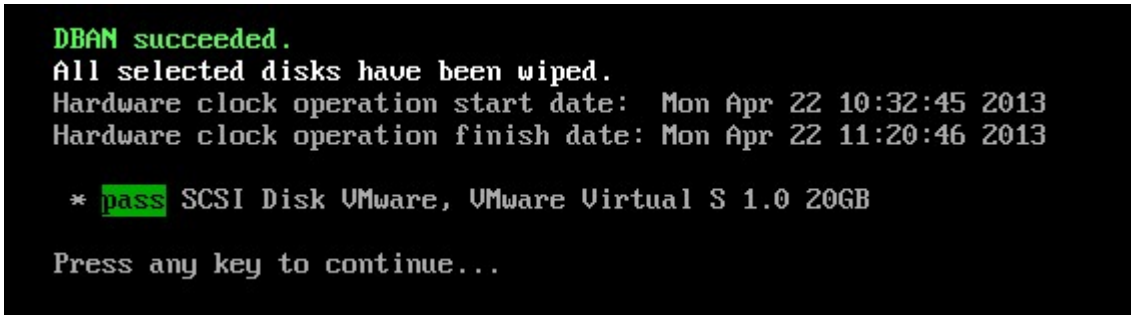
5. Select the disk or partition you want to wipe by using the up (J) and down (K) keys to move to the entry.
6. To confirm your selection, press the space bar.
7. To start wiping the disk, press F10.



8. The disk is now being wiped.



9. Make sure a dialog is displayed that confirms a successful wipe.



Using the Wipe tool

You can download Wipe from http://wipe.sourceforge.net/.

You can use the Wipe tool to securely delete single files or directories.

For example, to securely delete file *myfile.txt* run:

```
wipe -Sr -p3 myfile.txt
```

Decommission a Safer Payments instance or cluster

If a Safer Payments cluster instance or an entire Safer Payments cluster is decommissioned, you must securely delete all cardholder data by using a disk wipe tool.

Safer Payments stores cardholder data in several locations. These locations are identified and configured as described in "Configure cardholder data storage locations" on page 21.

Archived data and backups that are created by third-party applications are not in reach of the Safer Payments software itself. Therefore, they are not securely deleted automatically by Safer Payments. This aspect of this requirement must be met by organizational procedures.

PCI DSS compliance report

Safer Payments provides a built-in PCI DSS compliance report that lists all relevant configuration settings that must be changed to achieve PCI DSS compliance.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Click the  (Report PCI DSS compliance) icon to create the report.

The generated report lists potential issues with PCI DSS compliance for the current configuration of Safer Payments.

Use this report to configure Safer Payments according to the PCI DSS requirements. Refer to the online help for details on Safer Payments system configuration.

Using Safer Payments extensions

With the following extensions, Safer Payments can be used in combination with additional technologies that can be installed alongside Safer Payments on a Safer Payments server.

Security settings and logging for those software components must be set up for each of the software products in accordance to PCI DSS.

Configuration of the IBM MQ interface

Safer Payments offers the possibility to dynamically link to a local IBM MQ client to retrieve data from remote IBM MQ servers.

For more information about IBM MQ in general, see IBM Documentation. If the IBM MQ client library is installed on a local Safer Payments instance, Safer Payments loads the library at run time and uses settings in the cluster configuration to connect to remote IBM MQ servers. If no IBM MQ client is installed or loading of the IBM MQ client library fails, the function is not available.

The settings for connections to IBM MQ servers are made in **Cluster >Settings > WebSphere MQ Interface**. Each Safer Payment instance can connect to multiple queues on multiple remote IBM MQ queue managers, which are identified by a queue manager name, a target IP, and a target port. Security settings are defined in the definition of an IBM MQ channel to the queue manager. Whenever an IBM MQ connection is used to transport sensitive data over a public network, use of the "Use SSL" option is mandatory for PCI DSS compliance. Also, on the IBM MQ server side, all connection channels to a queue must be configured to use TLS 1.2 using cipher specifications listed here.

The IBM Knowledge Center article Connecting a client to a queue manager securelydescribes the process of a creating a key repository that is required for the "Use SSL" option in Safer Payments. It also describes the configurations necessary on the server side to make sure authentication using the key repository is being enforced.

IBM MQ is a product developed independently from Safer Payments, it cannot be guaranteed that the provided configuration options are always sufficient. Therefore, it is necessary to constantly monitor the IBM MQ documentation for changes to the software and the security of the used cipher suites for potential security leaks.

Configuration of a custom parser library

Safer Payments offers the possibility to dynamically link to a local library that encapsulates custom parser functionality..

As the source code within this library is user defined, it is not part of Safer Payments. To use a custom library in accordance with PCI DSS, its code needs to be developed and audited separately. If no custom parser library is installed and linked to the Safer Payments library path on the Safer Payments server, the functionality is not available.

Configuration of SSO using Kerberos

Safer Payments allows SSO login using Kerberos, which needs additional setup steps on the Safer Payments server, outside of the Safer Payments configuration.

- Your Safer Payments configuration must be connected to an existing LDAP (or Active Directory) server. After turning on LDAP in Administration, System Configuration you can select the '`Allow Single Sign On`' option.
- You must create a keytab file on your Kerberos (or Active Directory) server and deploy it to all Safer Payments servers that are used for API access.
- You must alter some system configuration files on the Safer Payments server to point to your Kerberos (or Active Directory) server.
- Finally, every user must run a setup step on the web browser to allow the browser to pass the users authentication parameters to the server.
- Detailed information on the setup process is available in the Safer Payments online help under Administration, System Configuration, LDAP.

Note: SSO is not required for operation in accordance to PA-DSS.

Single sign-on using a custom HTTP header

Safer Payments can be integrated into a single sign-on environment by using a custom HTTP header.

Safer Payments is agnostic to the authentication protocol. It does not need to directly support any of the authentication protocols, therefore, it can typically be integrated in any single sign-on environment. The authentication is handled by the authentication server, for example, ISAM, OIDC, OAuth, SAML, and so on. If the user is successfully authenticated, all requests to Safer Payments need to be enriched by a configurable custom HTTP header that has to contain the user name of the authenticated user. The HTTP communication should be secured by mutual authentication between Safer Payments and the authentication proxy server. If the user name exists within Safer Payments, the authentication is successful. If not, access with the user name is not possible.

Note: External authentication services are out of scope of Safer Payments. If you opt to implement the authentication using a custom HTTP header, you must implement proper procedures to secure this feature against unauthorized access. Enabling this authentication method will trigger a notification inside the Safer Payments PCI report. You must ensure the integrity and the safety of the networks used for this process. To initially set up Safer Payments, the integrated login mechanism must be used.

Preliminary considerations

If you choose to use a custom HTTP header for user authentication, you must ensure that the external authentication server addresses all applicable PCI DSS requirements to achieve compliance. IBM Safer Payments must be set up to perform a client certificate validation on the API to secure the communication between the application and the external authentication server. See "Configure SSL encryption" on page 14 for details.

Before you activate the custom HTTP header authentication in IBM Safer Payments, you must configure the external authentication server to properly send the required HTTP header to the application. Otherwise you won't be able to login anymore and must manually revert the configuration change on the file system.

Note: The user creation workflow in Safer Payments must be used to create additional users. PA-DSS requirement 3.2 mandates the usage of unique user IDs for each user.

Setting up custom HTTP header authentication in Safer Payments

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **System** > **Configuration** from the left navigation pane.
3. Locate the section titled **Authentication Settings**.

Figure 29. Authentication Settings

4. First select `OIDC claim` from the **Authentication method** drop-down list. Then specify the **Token name** and the **Username attribute**.

   In the **Token name** field you must specify the name of the HTTP header field that contains the JSON payload. Whereas in the **Username attribute** field you must specify the identifier inside the JSON payload that contains the user's login name.
5. Save your changes

Attention: Check the token name and the username before saving, or you will be locked out of Safer Payments.

Once this system configuration is saved, Safer Payments will accept user logins via the custom HTTP header. Other authentication mechanisms are no longer possible until you change the authentication settings again.

Figure 30 on page 48 shows an example HTTP header.

```
{
  "login_name": "jsmith@exampleMail.com",
  "last_name": "Smith",
  "first_name": "Jane",
  "id": "cl.jsmith",
  "internal_id": "25",
  "emails": {
    "personal": "jsmith@personalMail.com",
    "business": "jsmith@businessMail.com"
  },
  "phone_numbers": {
    "business": "555-555-1234"
  }
}
```

Figure 30. Example HTTP header

If you enable the custom HTTP header as an authentication method, a notification will be added to the built-in PCI report.

Python code execution

It is possible to feed data to external Python programs, which are out of scope of IBM Safer Payments. If you use external Python programs to store sensitive data outside of IBM Safer Payments, you must protect this data by fulfilling PCI DSS requirements 3.4.1, 3.5, 3.6, and all applicable subrequirements.

To help identify such cases, the Safer Payments PCI DSS report warns you about model revisions (including challenger revisions) that use external Python programs referencing encrypted attributes.

Important: Python programs are executed by the same user that runs Safer Payments. The Python program will have the same operating system privileges as that user. Therefore, the permissions of that user should be as restrictive as possible. The Safer Payments user privilege to edit mandators should also be used sparsely as those users are able to upload Python code into the application.

Kafka

Safer Payments can be connected to a Kafka cluster to retrieve messages from Kafka topics.

The settings for connections to a Kafka cluster are made in **Cluster** > **Settings** > **Kafka Instance** and **Cluster** > **Interfaces** > **Inbound**. Each Safer Payments instance can connect to multiple topics on multiple remote Kafka clusters, which are identified by a target IP and port, and a topic name. Security settings are defined in the definition of the Kafka topic on the Inbound Interface page and in the definition of the Kafka Endpoint on the Cluster Settings page. Whenever a Kafka connection is used to transport sensitive data over a public network, use of the "Use SSL" option is mandatory for PCI DSS compliance. Also, on the Kafka server side, all connection channels to a broker must be configured to use TLS 1.2 using cipher specifications listed here.

For more information about Kafka, see the Apache Kafka website.

The *Kafka documentation* describes the process of creating a certificate and key that can be used for a Kafka broker. The *librdkafka documentation* describes the process of creating a client certificate that is required for the "Use SSL" option in Safer Payments. Apache Kafka is a product that is developed independently from Safer Payments. As such, it cannot be guaranteed that the provided configuration options are always sufficient. Therefore, it is necessary to constantly monitor the Kafka documentation for changes to the software and the security of the used cipher suites for potential security leaks.

Persistent connections

Use a persistent connection channel to an external system to avoid the overhead of creating and destroying a connection for every message.

Safer Payments utilizes a persistent connection outgoing channel type inside the model in the form of external model components and the "forward to external system" option within masterdata elements. Both of these can be used to enhance transaction scoring by using information retrieved from external systems. To not impose limits on these use cases, neither the external model component nor the masterdata perform special handling on encrypted attributes. The values of these attributes are always sent in clear-text.

This means that when an external model component or masterdata is set up to send sensitive data over a public network, the corresponding persistent connections must enable the "Use SSL" option to protect the data transfer using certificates.

Chapter 3. Key management procedures for cryptographic keys

This section describes the cryptographic keys that are used by Safer Payments, how keys are generated, and how to enter and activate keys.

Cryptographic keys used by Safer Payments

Safer Payments encryption uses keys that are generated onsite by the Safer Payments user with the assistance of the **Keygen** program , which is provided as part of the Safer Payments software delivery.

Keygen uses master public keys to encrypt a random generated master key from which in subsequent steps any number of usage keys are generated. The master public key consists of two arbitrary passphrases of arbitrary length that are chosen by two master key holders. The encrypted master key is generated as a file that must be stored in a safe location.

When new usage keys are generated, **Keygen** is called with the encrypted master key. The master key is decrypted by the two master key holders who enter their passphrases. Now the entry of two usage public keys creates a usage key triplet. The usage key triplet consists of two arbitrary passphrases of arbitrary length that are chosen by two usage key holders.

Each usage key triplet consists of

- One usage private triplet subkey that is manually distributed to all instances of a Safer Payments cluster by the administrator.
- One left public subkey that is known only to one usage key holder.
- One right public subkey that is know only to another usage key holder.

To activate a usage key triplet, the Safer Payments instance must have the usage private key available locally. The two public keys must be available either locally entered by the usage key holders, or received from another Safer Payments instance of the cluster. The private triplet subkeys are never transmitted between the Safer Payments instances. Therefore, the parts of a key are never located on the same medium.

Safer Payments can keep multiple active and non-active key triplets in the key management function, and can switch between the active ones. A non-active triplet would be one where a subkey is not provided yet. While only one of the key triplets can be active at a time, it makes no difference, which of the key triplets is the active one.

Note: Generally, access to keys must be limited to the fewest number of custodians that are necessary. Also, keys must be stored securely in the fewest possible locations and forms. These are organizational duties to be met by the licensee.

Note: Key triplets are differentiated by their number.

Key generation

The following sections describe the principles of generating a master key, usage key triplets, and the actual key generation procedure in detail.

Master key generation process

Figure 31 on page 52 shows the computational actions that are involved in master key generation.

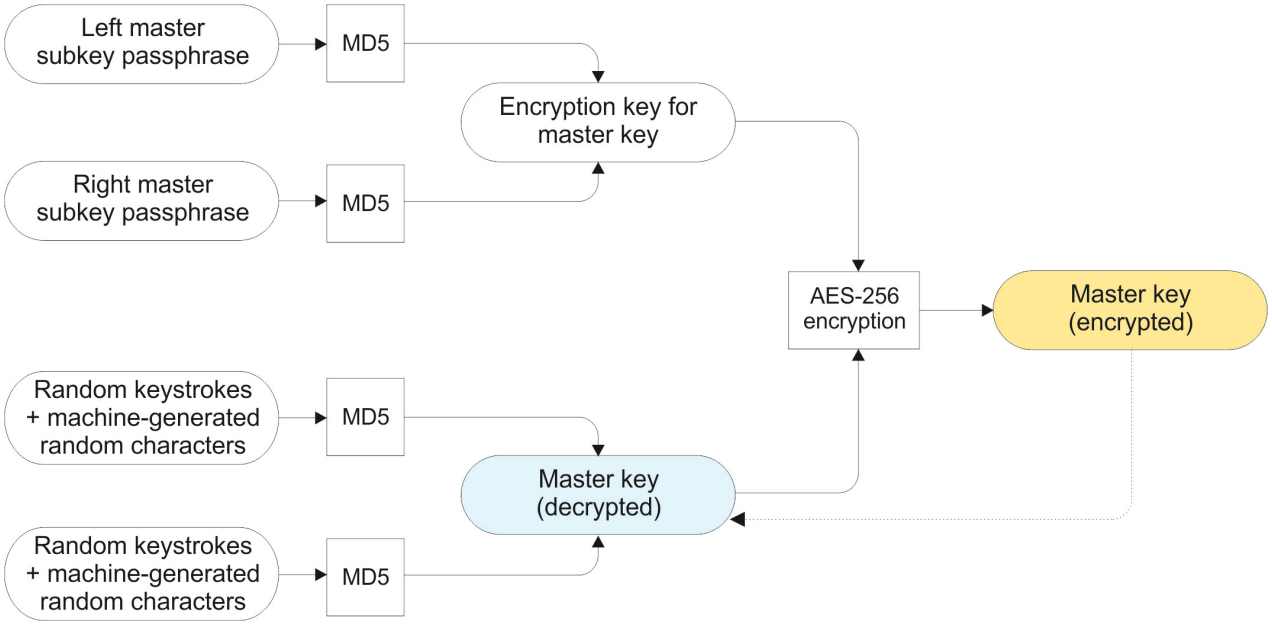


Figure 31. Master key generation process

The master key that is used by Safer Payments to encrypt and decrypt data is generated by two sets of at least 80 random characters that are hashed by MD5, creating a 256-bit length root key. The two sets of random characters are each generated by combining at least 40 random keystrokes from a user with 40 machine-generated random characters. This master key is never stored or made accessible to users. Rather, using the two passphrases of the key holders, the master key is encrypted with the AES-256 algorithm.

Important: Using the two passphrases, the encrypted master key can be decrypted. This is illustrated in Figure 31 on page 52 with the dotted line.

The encrypted master key is stored in a safe place and is used, together with the passphrases of the key holders, to create the usage key triplets. The usage key triplets are the only keys that are used during Safer Payments operations.

This is also the reason why the key generator is provided as a separate utility program rather than a part of Safer Payments. Not even the encrypted master key must ever be stored on the Safer Payments server host. Use a different computer to create the encrypted master key, store it in a safe place, and generate usage key triplets whenever needed.

Usage key triplet generation process

The usage key triplet generation requires the left and right master key passphrases, and thus the presence of the key holders. Two key holders for the two public subkeys of each usage key triplet are also required. The key holders can be the same persons.

Figure 32 on page 53 illustrates the process.

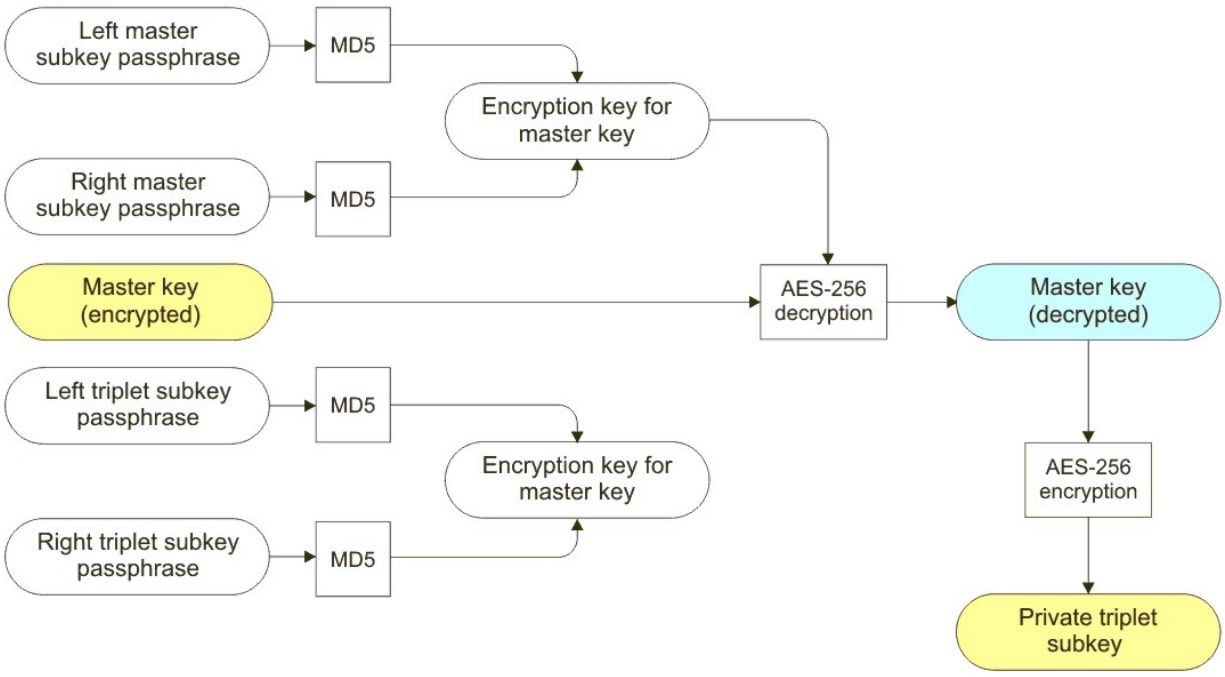


Figure 32. Private triplet subkey generation process

The encrypted master key is read from file and using the two master passphrases is decrypted in main memory only. From this decrypted version of the master key each usage key triplet is generated by encrypting the master key with a new pair of passphrases.

The result of this process is the private triplet subkey, which must be stored in the key directory of the Safer Payments installation. Because the file system of the Safer Payments server host is a protected area, this provides an added level of security.

A good key generation practice is to generate a number of usage key triplets in advance and then use them when they are needed.

Important: Safer Payments can reconstruct the master key in main memory from each private triplet subkey, using the two public subkeys for decryption.

Key generation steps

This section describes the key generation procedure step-by-step.

Key generation is conducted outside of Safer Payments with the **keygen** tool.

In summary

1. You must generate master keys.
2. The master keys are stored at a safe place and are never used by the Safer Payments software.
3. The master keys are used to generate usage keys and an empty no-fly list.
4. Only usage keys and the no-fly list are used by the Safer Payments software.
5. If you want to obtain a PA-DSS certification at a future date, keep in mind that any storage media that is used to store or distribute keys is in scope of PA-DSS requirement 2.5.2.
6. When the storage media is not required anymore, it must be securely wiped, or destroyed. See "Using a secure wipe tool" on page 42 for details.
7. You must protect and store all keys securely.

Prerequisites

Use a separate PC that is not connected to the internet to generate keys. To not block a complete PC for the occasional key generation process, you can use a PC that is started from an OS boot CD. This has the advantage that even if you disconnect the PC temporarily from the internet, no malware could have logged any of your data.

Note: You can use RHEL/CentOS 64-bit OS.

Obtain key generator

Keygen is provided as part of a Safer Payments installation and is located in `/usr/bin/keygen`. Its integrity is checked when you download and verify the installation image.

Copy the contents to a portable memory location. This can be a memory card or USB stick.

Generate master key

This topic describes how to generate the master key.

To generate the master key, run the following command from the console:

```
keygen master <masterkeypath> <tripletkeypath> <master_key_id>
```

- *masterkeypath* is the location on your portable memory device where you want to store the master key.
- *tripletkeypath* is the location on your portable memory device where you want to store the triplet keys. The triplet keys are later physically distributed to the Safer Payments instances.
- *master_key_id* is the numeric ID for the new generated master key. Every master key that is used by your Safer Payments installation must have its unique ID.

The key generator guides you through the process of generating a master key. You need two master key holders for this process and the `masterkeypath` and `tripletkeypath` subdirectories must exist.

The master key is stored as `masterkeypath/master_key_private_<master_key_id>.iris`" and is created together with `tripletkeypath/revoked_keys.iris`.

The file `revoked_keys.iris` is used during the operation of Safer Payments to store a no-fly list of keys that Safer Payments must never use. To verify authenticity of the `revoked_keys.iris` file, it must be generated together with the initial master key.

Note: The file `revoked_keys.iris` is distributed with the initial key distribution to the Safer Payments instances. The file `master_key_private_<master_key_id>.iris` must never be distributed to Safer Payments instances, or anywhere outside the portable memory device location. Never replace an existing "`revoked_keys.iris`" file in the key folder of your configuration. If you change to a usage key from another master key by the Safer Payments user interface, `revoked_keys.iris` is reencrypted as well.

If the two master key holders activate the master key that you generated, you can generate any number of usage keys.

You can now directly proceed to "Generate usage key triplets" on page 54, or shut down the PC and store the portable memory device at a safe place until you need to generate usage keys.

Generate usage key triplets

This topic describes how to generate the usage key triplets.

To generate the usage key triplets, run the following command from the console:

```
keygen triplet <masterkeyfilepath> <tripletkeypath>
```

- *masterkeyfilepath* is the file location of your master key. This location must include the file name of the master key.

- *tripletkeypath* is the location on your portable memory device where you want to store the usage key triplets. The usage key triplets are later physically distributed to the Safer Payments instances. When the first key id is specified keys are not generated in *tripletkeypath*.

The key generator guides you through the process of generating a usage key triplet. You need two master key holders and two usage key holders for this process.

Keygen generates the file `tripletkeypath/key_<usage_key_id>.iris`.

You can repeat this process at any time to generate the number of usage key triplets that you need. The master key holder passphrases do not have to be entered for each usage key triplet generation, unless you quit the key generator.

If you generated all the usage key triplets you need, shut down the PC and store the portable memory device at a safe place until you need to generate more usage keys.

Distribute keys

This topic describes how to distribute the keys.

All the `key_`*n*`.iris` files (private triplet subkeys) that you want to use with your Safer Payments installation, must be copied manually from the portable memory device to the `key` subdirectories of all Safer Payments instances.

If you copy usage key triplets to running Safer Payments instances, you must reload the keys as described in "Activate data encryption - step 2" on page 31. Safer Payments reloads keys automatically whenever it restarts. Do not overwrite or replace the `revoked_keys.iris` or the `key_`*n*`.iris` files in the `key` subdirectory.

The first time that you distribute keys to Safer Payments instances, you must include the file `revoked_keys.iris` that was generated during the initial creation of the master key. This file stores the no-fly list of revoked keys. Never overwrite this file manually once it is delivered to the Safer Payments instances. Make sure that this file is writable for Safer Payments to revoke keys or to reencrypt the file. For example, if you change to another master key.

The content of the encrypted `revoked_keys.iris` files might differ on each instance after you reencrypt or revoke a key. As the encryption of this file adds a random token, the encrypted result differs on each instance. Nevertheless, the stored no-fly list is always the same.

When you copy the files to the `key` subdirectories, make sure that you adjust the user and group access privileges so that only the Safer Payments process user can access those files.

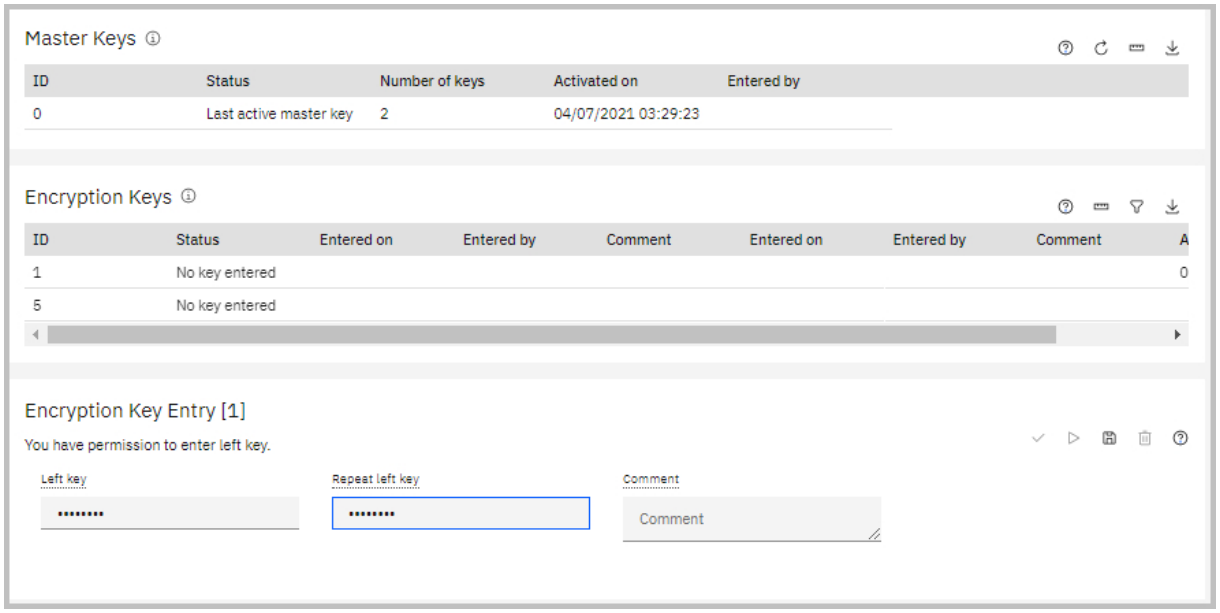
Leave a copy of the usage key triplet files on the portable memory device so that you have a reference of generated keys. You must protect and store the device securely.

Activate keys

This topic describes how to activate encryption keys.

To activate a usage key triplet, the two passphrases must be entered into the Safer Payments user interface.

1. Assuming you are the left key holder, log in.
2. The Safer Payments user interface opens and the **General Settings** for the left key user are displayed.
3. Click **Administration**. The **Key management** > **Encryption keys** form displays.
4. In the **Master Keys** section, click the row of the master key instance you want to activate.
5. In the **Encryption Keys** section, click the row of the key instance you want to activate.



6. In the **Left key** field, enter your key and repeat it for verification.
7. Click the (Save) icon.
8. Repeat the steps above for the right key holder.
9. The user who has the global privilege to activate key triplets must log in and go to **Administration > Encryption keys**.

   Note: The global privilege to activate usage key triplets can be granted to the key holders or any other user.

10. In the **Encryption Key Entry** section, click the (Activate key) icon.

You can prepare more than one key for activation, and users with respective privileges can switch between them by activating a key.

If a key is revoked, the key file is automatically securely erased on all Safer Payments instances in a cluster. The revoked key is also added to the no-fly list to ensure that this key cannot be active again in Safer Payments.

Safer Payments instances in a cluster share the passphrases over their encrypted network connection (ECI). The private triplet subkey of the usage key triplet is transferred manually by the operator. Therefore, the private key and the public keys never travel together on the same medium. Thus, spying out only one of the channels does not deliver sufficient information to decrypt Safer Payments.

Because Safer Payments instances share the public keys, the key holders do not have to enter them each time a Safer Payments instance is started. If one Safer Payments instance is still running in the cluster, passphrases do not have to be reentered. Only when you start the first Safer Payments instance, passphrases must be entered.

You can simultaneously start all Safer Payments instances because in key-entry mode the user interface is partially active to allow for key entry. When keys are entered on any Safer Payments instance of the cluster, they are shared within the cluster and the Safer Payments instances start. This might take a few minutes.

Note: A key is automatically deactivated, if you activate another key.

Precautions and possible errors

- If you use a Flash-based portable memory device, which most USB sticks or SD cards are, it is difficult to securely erase data from them. Therefore, you must store the portable memory device in a safe

location for the entire duration of the master key being valid. If you ever need to erase the master key on such a portable memory device, the safest way is physical destruction.

- If Safer Payments cannot locate the `revoked_keys.iris` file during startup, or if the file is tampered with, Safer Payments creates a log message and shuts down immediately.
- If Safer Payments finds an active key that is on the no-fly list, Safer Payments securely deletes the key from the `key` subdirectory and shuts down immediately. If the key is not active, Safer Payments creates a log message, securely deletes the key from the `key` subdirectory, and continues with startup.
- If you run a key reload from the Encryption Keys page of the Safer Payments user interface, the following problems can occur:
  - If Safer Payments cannot locate the `revoked_keys.iris` file, or the file is tampered with, an error message on the user interface and a log message are created, reloading is stopped, yet operations resume.
  - If Safer Payments finds keys that are on the no-fly list, the keys are securely deleted from the `key` subdirectory, an error message on the user interface and a log message are created.

Enforce regular key changes

This topic describes how to define regular key changes and how to set key life alerts.

Regular key changes are recommended.

The National Institute of Standards and Technology has developed recommendations for key management. Their guidelines assist you in defining the correct key retention periods for your organization.

You can download the *NIST Special Publication 800-57* here: http://csrc.nist.gov/publications/PubsSPs.html#SP%20800

Based on this, we recommend a maximum key life of 120 days, and a maximum master key life of three years.

Important: Retirement or replacement of keys is required, if the integrity of the key has been weakened, or keys are suspected of being compromised.

Define maximum key life

You can define the maximum key life and the maximum master key live as follows.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **System** > **Configuration** from the left navigation pane. Click the **System** tab.
3. Scroll down to the **Encryption** section.

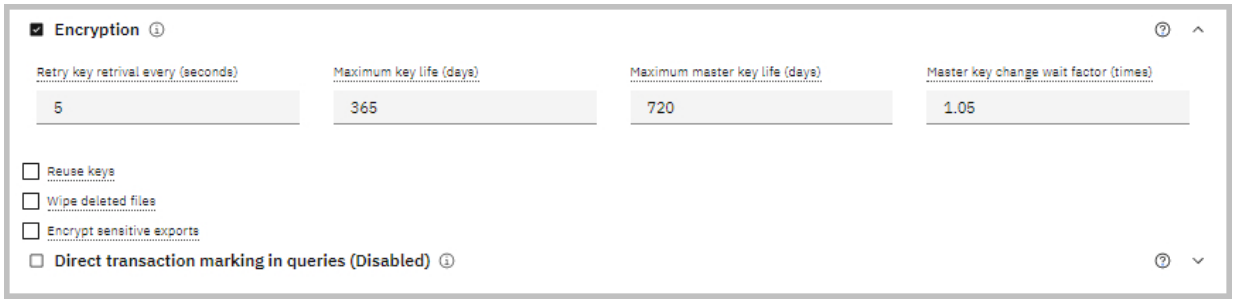


Figure 33. Encryption section

4. In the **Maximum key life (days)** field, enter the number of days you defined in your organization.

If the maximum key life is reached and no key is changed during this period, Safer Payments automatically shuts down.

Set maximum key life alerts

Safer Payments provides a **Status Alarm Indicator** (SAI) that alerts if the end of the maximum key life approaches. SAI alerts can be sent to the Safer Payments dashboard and can be distributed by email, or log messages.

You must define the following two status alarm indicators:

- One for the encryption key, it must have the alarm type **encryption key remaining lifetime**.
- One for the master key, it must have the alarm type **master key remaining lifetime**.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Select **Dashboard settings** > **Status alarm indicators** from the left navigation pane.
3. From the **Status Alarm Indicators** table, click the + (New status alarm indicator) icon to create a new status alarm indicator.

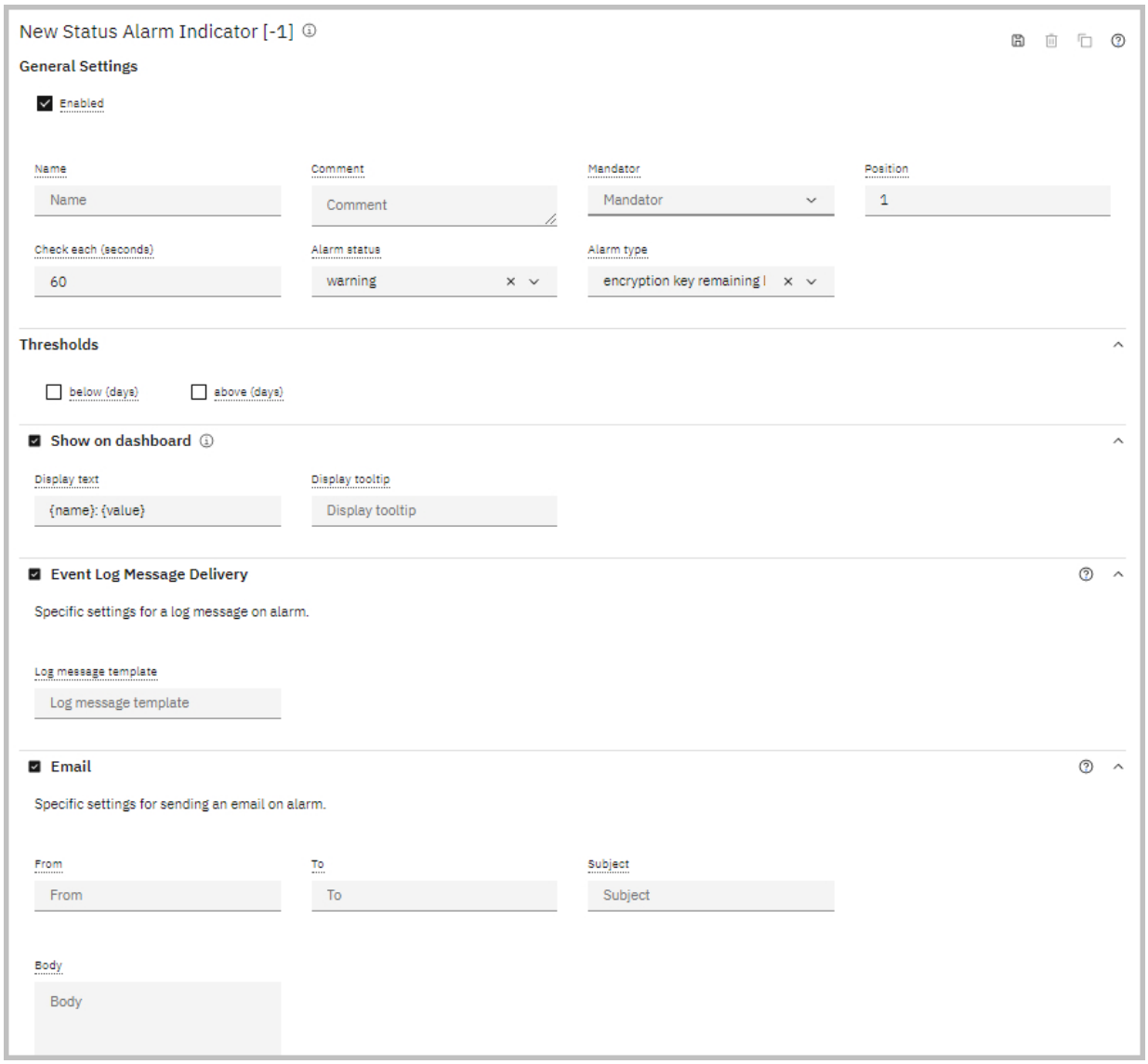


Figure 34. New Status Alarm Indicator form

Figure 34 on page 58 shows an exemplary SAI definition for monitoring the last encryption key change.

This SAI assumes a maximum key life of 120 days. If the current key is valid for only 10 more days a warning is displayed on the dashboard, a mail is sent out, and a log message is created.

Revoke Keys

Authorized users can revoke cryptographic keys in the **Encryption Key Entry** form. If a key is revoked, Safer Payments securely deletes the usage private key stored on disk, and removes the two usage public keys from main memory in all cluster instances.

Note: Only inactive keys can be revoked.

However, you must manually delete the revoked keys from all other storage locations using a secure wipe tool. For example, the media used for key distribution. See "Using a secure wipe tool" on page 42 for details.

Change the master key

Carefully consider when to change the master key. During the change of the master key, all cluster instances become inactive.

While it is still possible to score transactions, you cannot change the configuration or investigate cases during the change process. The change affects all data that is stored in Safer Payments, which means such a change process can take several hours to complete.

Changing the master key requires the global privilege to change the master key. This privilege must be granted to the user in advance.

1. On the Safer Payments user interface, click the **Administration** tab.
2. Click **User management** > **Accounts** from the left navigation pane. Select your user.
3. Scroll down to the **Global Privileges** section.
4. In the **Key Management** field, select **activate and revoke keys and view encryption management, and change master key** from the pull-down menu.
5. Save your changes.

The process to change the master key is as follows:

1. Generate a new master key with the **keygen** tool with a new master key ID. See "Generate master key" on page 54 for details.
2. Generate new private keys from the new master key. See "Generate usage key triplets" on page 54 for details.
3. Copy the new private keys (`key_<key_id_n>.iris`) into the `key` folder on all instances
4. Do not replace any file of the `key` folder while copying.
5. Click the **Administration** tab.
6. Select **Key management** > **Encryption keys** from the left navigation pane.
7. Click the  (Reload private keys from disk) icon to reload private keys from disk.
8. The new master key is displayed. Log out.
9. The left key holder must log in and insert the left key.
10. The right key holder must log in and insert the right key.
11. Log in. You must have the right to activate a key.
12. Click the  (Activate key) icon to change the master key.

Appendix A. PA-DSS requirements

This topic lists the individual PA-DSS requirements and the corresponding Safer Payments implementation.

Requirement 1: Do not retain full track data, card verification code or value (CAV2, CID, CVC2, CVV2), or PIN block data

Requirement 1.1

Do not store sensitive authentication data after authorization (even if encrypted). If sensitive authentication data is received, render all data unrecoverable upon completion of the authorization process. Sensitive authentication data includes the data as cited in the following Requirements 1.1.1 through 1.1.3.

Such data is not required by Safer Payments for full operational functionality. To meet this requirement, supplying systems must be configured in a way that they do not send such data to Safer Payments. If you send such data to Safer Payments, your installation is not PCI DSS compliant.

Requirement 1.1.4

Securely delete any track data (from the magnetic stripe or equivalent data contained on a chip), card verification values or codes, and PINs or PIN block data stored by previous versions of the payment application, in accordance with industry-accepted standards for secure deletion, as defined, for example by the list of approved products maintained by the National Security Agency, or by other State or National standards or regulations.

Note: This requirement applies only if previous versions of the payment application stored sensitive authentication data.

Such data is not required by Safer Payments for full operational functionality. To meet this requirement, supplying systems must be configured in a way that they do not send such data to Safer Payments. If you send such data to Safer Payments, your installation is not PCI DSS compliant.

If you are migrating from competing software that stores sensitive authentication data, such data must be removed. Any disk space that is previously used for storing sensitive authentication data must be deleted securely by using a disk wipe tool. See “Using a secure wipe tool” on page 42 for details.

Removal is necessary for PCI DSS compliance.

Safer Payments itself securely wipes any files that store encrypted attributes when they are deleted from the user interface.

Requirement 1.1.5

Do not store sensitive authentication data on vendor systems. If any sensitive authentication data (pre-authorization data) must be used for debugging or troubleshooting purposes, ensure the following:

- Sensitive authentication data is collected only when needed to solve a specific problem.
- Such data is stored in a specific, known location with limited access.
- The minimum amount of data is collected as needed to solve a specific problem.
- Sensitive authentication data is encrypted with strong cryptography while stored.
- Data is securely deleted immediately after use, including from:
  - Log files

- Debugging files
- Other data sources received from customers

Such data is not required by Safer Payments for full operational functionality, and would contradict with fully meeting Requirement 1.

Requirement 2: Protect stored cardholder data

Requirement 2.1

Software vendor must provide guidance to customers regarding secure deletion of cardholder data after expiration of customer-defined retention period.

Cardholder data that exceeds the customer-defined retention period must be securely deleted. According to PCI DSS Requirement 3.1, each licensee must define a retention period.

Data that is stored in the Disk Data Cache (DDC) of Safer Payments is automatically securely deleted. It is stored in a ring buffer type memory that overwrites itself when its capacity limit is reached. Therefore, in consequence, it is necessary that the DDC capacity limit is aligned with, or less than the retention period. Safer Payments provides the configuration option, and it is the responsibility of the licensee to configure Safer Payments accordingly.

However, this is not the case with indexes. If an index is created on an attribute that contains cardholder data, the "purge after" setting must be made accordingly with the definition of the index. Safer Payments deletes such index entries automatically and securely. See "Securely delete outdated index entries" on page 37 for details.

Cardholder data can be used as part of conditions in the Safer Payments configuration. Therefore, it is possible that the cfg directory also contains encrypted cardholder data.

Safer Payments fully controls and protects cardholder data within its reach. Therefore, no special configuration is required regarding underlying software or systems (such as the operating system) to prevent inadvertent capture or retention.

Archived data and backups that are created by third-party applications are not in reach of the Safer Payments software itself. Therefore, they cannot be securely deleted automatically by Safer Payments. This aspect of this requirement must be met by organizational procedures.

See "Using a secure wipe tool" on page 42 for details on how to securely delete cardholder data.

Certain operating system functions might also store encrypted cardholder data outside the reach of Safer Payments. See "Swap disk configuration" on page 25 and "Disable locate for Safer Payments folders" on page 26 on how to avoid this.

You can freely define the storage locations of cardholder data within Safer Payments. See "Configure cardholder data storage locations" on page 21 for details.

Data that is created via external Python scripts is out of scope of Safer Payments. If Python scripts are used in combination with cardholder data or encrypted attributes, organizational procedures must be implemented to cover this aspect.

Requirement 2.2

Mask PAN when displayed (the first six and last four digits are the maximum number of digits to be displayed), such that only personnel with a legitimate business need can see more than the first six/last four digits of the PAN.

Note: This requirement does not supersede stricter requirements in place for displays of cardholder data. For example, legal or payment card brand requirements for point-of-sale (POS) receipts.

If a PAN is displayed in full or only masked is determined by a user account privilege. Because the masking is run at the Safer Payments server, unmasked PAN never make it outside Safer Payments via the API. It is thus impossible for a non-authorized user to gain access to the full PAN numbers.

The licensee must implement proper operational procedures that ensure only users with "legitimate business need to see full PAN" are granted the respective privilege. Safer Payments allows for granting this privilege on a per-user basis.

Safer Payments displays PANs in the following components:

- Queries: PANs are masked depending on the user privilege.
- Defined risk lists: PANs are masked depending on the user privilege.
- Conditions: PANs are masked depending on the user privilege. Conditions can be found in many Safer Payments definitions in the sections administration, model, monitoring, investigation, and report.
- Logs: PANs are always masked independent from the user privileges.

See “Set user privileges” on page 40 for details.

Requirement 2.3

Render PAN unreadable anywhere it is stored (including data on portable digital media, backup media, and in logs) by using any of the following approaches:

- One-way hashes based on strong cryptography (hash must be of the entire PAN).
- Truncation (hashing cannot be used to replace the truncated segment of PAN).
- Index tokens and pads (pads must be securely stored).
- Strong cryptography with associated key-management processes and procedures.

Any storage of PANs uses 256-Bit AES encryption. This includes transaction attributes, indexes, and cases. Encryption is turned on by the respective configuration option after which the respective PAN attribute encryption setting must be turned on.

In Safer Payments you can export data to store outside the payment application by CSV-exports and by the RDI interface.

If you use the RDI, you thus must render all PANs unreadable. See “Miscellaneous Safer Payments configuration settings” on page 34 on how to configure masking for RDI.

If you use CSV-exports, you must enable encryption for sensitive exports. See “Enable cardholder data encryption” on page 31 on how to configure Safer Payments encryption.

Even if debugging functions are enabled, PANs are never included in debugging logs. This is ensured by the design of the software and cannot be changed by configuration options.

Requirement 2.4

Payment application must protect keys used to secure cardholder data against disclosure and misuse.

Note: This requirement applies to keys used to encrypt stored cardholder data, as well as to key-encrypting keys used to protect data-encrypting keys. Such key-encrypting keys must be at least as strong as the data-encrypting key.

Safer Payments meets this requirement. See Chapter 3, “Key management procedures for cryptographic keys,” on page 51 for details.

Generally, access to keys must be limited to the fewest number of custodians necessary. Also, keys must be stored securely in the fewest possible locations and forms. These are organizational duties to be met by the licensee.

See “Configure cardholder data storage locations” on page 21 for details on the location where Safer Payments encryption keys are stored.

Remarks:

- This location is different for each Safer Payments instance in a Safer Payments cluster. Therefore, you must address the location individually for each cluster instance.

- Archived data and backups that are created by third-party applications are not in reach of the Safer Payments software itself. This aspect of this requirement must be met by organizational procedures. If you do not intend to exclude the key path from backups, make sure that this does not contradict to PCI DSS Requirement 3.5.2.
- If the integrity of the key has been weakened, or there is a known or suspected compromise of a key you must activate a new usage key triplet and revoke the previous one. See “Revoke Keys” on page 59 for details.

Requirement 2.5

Payment application must implement key management processes and procedures for cryptographic keys used for encryption of cardholder data, including at least the following:

See Chapter 3, “Key management procedures for cryptographic keys,” on page 51 for details on how to securely generate, distribute, protect, change, store, retire, and replace cryptographic keys.

2.5.1 Generation of strong cryptographic keys

“Key generation” on page 51 discusses strong cryptographic key generation procedures that are implemented in Safer Payments.

2.5.2 Secure cryptographic key distribution

Private keys must be copied manually into the key directory of each Safer Payments instance. Therefore, it is the duty of the licensee to ensure secure distribution. Safer Payments itself does not distribute these files. See “Key generation steps” on page 53 for details.

Public keys are made available to Safer Payments by entering them in the GUI. See “Activate keys” on page 55 for details. Therefore, communication between the workstations of the users and the Safer Payments instances must be encrypted securely. For example, by using TLSv1.2. See refer to section 4.3.3.

2.5.3 Secure cryptographic key storage

Safer Payments uses the AES-256 algorithm to ensure secure cryptographic key storage.

Any cryptographic keys must be stored securely, and access must be limited to people with a legitimate business need to access them.

2.5.4 Cryptographic key changes for keys that have reached the end of their cryptoperiod (for example, after a defined period of time has passed and/or after a certain amount of cipher-text has been produced by a given key), as defined by the associated application vendor or key owner, and based on industry best practices and guidelines (for example, NIST Special Publication 800-57).

Safer Payments implements key-management procedures that allow for cryptographic key changes during operation, without service interruption. If no key change occurs during a defined period, Safer Payments automatically shuts down. See “Enforce regular key changes ” on page 57 for details about the key change procedures that are implemented in Safer Payments.

2.5.5 Retirement or replacement of keys (for example: by archiving, destruction, and/or revocation as applicable) as deemed necessary when the integrity of the key has been weakened (for example, departure of an employee with knowledge of a clear-text key, etc.) or keys are suspected of being compromised.

Safer Payments implements key-management procedures that allow for retirement or replacement of keys.

Safer Payments does not archive revoked keys, and there is no need to do so. When a cryptographic key is retired/revoked in Safer Payments, it becomes redundant right away, because the process must include activation of a replacement key. There is no need to keep a retired/revoked key for decryption/verification purpose, because the same data can be decrypted/verified with any other valid key, including this and subsequent replacement keys.

Safer Payments does not use any inactive cryptographic keys for encryption operations.

Retirement or replacement of keys is required, if the integrity of the key has been weakened, or keys are suspected of being compromised. See “Revoke Keys” on page 59 for details.

If the integrity of the master key has been weakened, a new master key should be generated and deployed to the Safer Payments cluster. See “Change the master key” on page 59 for details.

Safer Payments securely deletes all revoked keys within its reach. However, you must delete revoked keys manually from all other storage locations by using a secure wipe tool. See “Using a secure wipe tool” on page 42 for details.

2.5.6 If the payment application supports manual clear-text cryptographic key management operations, these operations must enforce split knowledge and dual control.

Note: Examples of manual key-management operations include, but are not limited to: key generation, transmission, loading, storage, and destruction.

Safer Payments enforces split knowledge and dual control, as it supports manual cryptographic key management operations. More precisely, two key holders are required for key generation and activation.

2.5.7 Prevention of unauthorized substitution of cryptographic keys

Substitution of cryptographic keys must be legitimized by two key holders. Organizational procedures must be put in place by the licensee that ensures no single user is granted two accounts. This warrants that no single user can pretend to be two distinct key holders.

In consequence, unauthorized substitution of cryptographic keys is prevented.

Requirement 2.6

Provide a mechanism to render irretrievable any cryptographic key material or cryptogram stored by the payment application, in accordance with industry-accepted standards. These are cryptographic keys used to encrypt or verify cardholder data.

Note: This requirement applies only if the payment application uses, or previous versions of the payment application used, cryptographic key materials, or cryptograms to encrypt cardholder data.

For PCI DSS compliance, cryptographic material must be rendered irretrievable. See “Using a secure wipe tool” on page 42 on how to render cryptographic material irretrievable using a secure wipe tool.

Safer Payments meets this requirement. Keys in Safer Payments are application version independent and thus remain exactly what they are in an upgrade situation. Therefore, no action is required to render previous Safer Payments version cryptographic material irretrievable. If you want to render cryptographic material irretrievable, you must revoke the respective keys. Keys that are revoked are securely deleted on disk (disk space overwritten with pattern) before the respective file contents are deleted.

Reencrypting historic data with new keys is an internal process of Safer Payments, which is automatically triggered by entering and activating a new key, and revoking the old keys. See “Enforce regular key changes ” on page 57, “Revoke Keys” on page 59, and “Change the master key” on page 59for details on these reencryption processes.

Requirement 3: Provide secure authentication features

Requirement 3.1

The payment application must support and enforce the use of unique user IDs and secure authentication for all administrative access and for all access to cardholder data. Secure authentication must be enforced to all accounts generated or managed by the application by the completion of installation and for subsequent changes after installation. The application must enforce 3.1.1 through 3.1.11.

Note: The term *subsequent changes* used throughout Requirement 3 refers to any application changes that result in user accounts reverting to default settings, changes to existing account configurations, and changes that generate new accounts or re-create existing accounts.

Note: These password controls are not intended to apply to personnel who only have access to one card number at a time to facilitate a single transaction. These controls are applicable for access by personnel with administrative capabilities, for access to systems with cardholder data, and for access controlled by the payment application. This requirement applies to the payment application and all associated tools used to view or access cardholder data.

Safer Payments meets this requirement. Safer Payments enforces secure authentication for all authentication credentials that the application generates by:

- Enforcing secure changes to authentication credentials by the completion of installation.
- Enforcing secure changes for any subsequent changes (after installation) to authentication credentials.

Remarks:

- You must not use any administrative/root user accounts for daily operational use.
- You must assign secure authentication to any default accounts (even if they are not used), and then disable them. Use the PCI DSS compliance report to verify that all default user accounts are addressed correctly.
- If you use an LDAP server to manage user passwords, you must make sure that the LDAP server is configured according to the requirements.
- If you use the SMTP-based emailing capabilities of Safer Payments, you must make sure that the SMTP server is configured according to the requirements.
- If you use OpenID connect (OIDC) token for user authentication, you must make sure that the authorization infrastructure is configured according to all applicable PA-DSS requirements.

3.1.1. The payment application does not use (or require the use of) default administrative accounts for other necessary software (for example, the payment application must not use the database default administrative account).

Safer Payments does not require any administrative privileges to run and it does not use any third-party software except the operating system.

Note: If you use OIDC tokens with an external authentication infrastructure, you must address this requirement for the communication between the user client and the authorization server.

3.1.2. The application must enforce the changing of all default application passwords for all accounts that are generated or managed by the application, by the completion of installation and for subsequent changes after installation. This applies to all accounts, including user accounts, application and service accounts, and accounts used by the vendor for support purposes.

Safer Payments meets this requirement because it is delivered with one default user account and you must change the password at your first login to access the user interface.

3.1.3. The payment application assigns unique IDs for user accounts.

Safer Payments meets this requirement. Unique IDs are assigned for user accounts.

3.1.4. The payment application employs at least one of the following methods to authenticate all users:

- Something that you know, such as a password or passphrase.
- Something that you have, such as a token device or smart card.
- Something that you are, such as a biometric.

Safer Payments meets this requirement by using password-based authentication.

3.1.5. The payment application does not require or use any group, shared, or generic accounts and passwords.

Safer Payments meets this requirement. It does not require or use any group, shared, or generic accounts and passwords.

3.1.6 The payment application requires that passwords meet the following:

- Require a minimum length of at least seven characters.
- Contain both numeric and alphabetic characters.

Safer Payments meets this requirement by providing options in the system configuration that force user passwords to have a minimum length, contain at least one upper case character, contain at least one lower case character, contain at least one digit, or contain at least one special character.

3.1.7. The payment application requires changes to user passwords at least every 90 days.
Safer Payments meets this requirement by providing an option in the system configuration where the number of days for which passwords are valid is defined.

3.1.8 The payment application keeps password history and requires that a new password is different than any of the last four passwords used.
Safer Payments meets this requirement by providing an option in the system configuration where the number of past passwords that a new password is checked against is defined.

3.1.9 The payment application limits repeated access attempts by locking out the user account after not more than six logon attempts.
Safer Payments meets this requirement by providing an option in the system configuration where the number of failed login attempts after which a user account is disabled is defined.

3.1.10 The payment application sets the lockout duration to a minimum of 30 minutes or until administrator enables the user ID.
Safer Payments meets this requirement by disabling a user account until an administrator manually reactivates it.

3.1.11 If a payment application session has been idle for more than 15 minutes, the application requires the user to reauthenticate to reactivate the session.
Safer Payments meets this requirement by providing an option in the system configuration defining the number of seconds of idle time after which a session expires causing an automatic log out.

You can use the PCI DSS compliance report to verify the compliance of your Safer Payments installation. See “PCI DSS compliance report” on page 46 for details.

Requirement 3.2

Software vendor must provide guidance to customers that all access to PCs, servers, and databases with payment applications must require a unique user ID and secure authentication.

For the organization of the licensee to be PCI DSS compliant, all access to PCs, servers, and databases with payment applications and cardholder data must require a unique user ID and PCI DSS compliant secure authentication. You must ensure that this requirement is met by organizational guidelines.

The Safer Payments application itself complies with this requirement. In particular, each user can use a unique user ID, and authentication in accordance with Requirement 3.1 is implemented. However, the licensee must ensure by using organizational guidelines that users do not share accounts, and secure authentication is ensured through PCI DSS compliant configuration of the software.

Requirement 3.3

Secure all payment application passwords (including passwords for user and application accounts) during transmission and storage.

3.3.1 Use strong cryptography to render all payment application passwords unreadable during transmission.

To render passwords unreadable during transmission, communication between the workstations of the users and the Safer Payments instances, and between Safer Payments instances, must be encrypted. For example, by using the internal SSL encryption function of Safer Payments.

If LDAP integration is used with Safer Payments, passwords are transmitted to the LDAP server. For secure transmission, turn on LDAP encryption over SSL on the **Administration** > **System** > **Configuration** > **User** page, as shown in Figure 35 on page 68 .

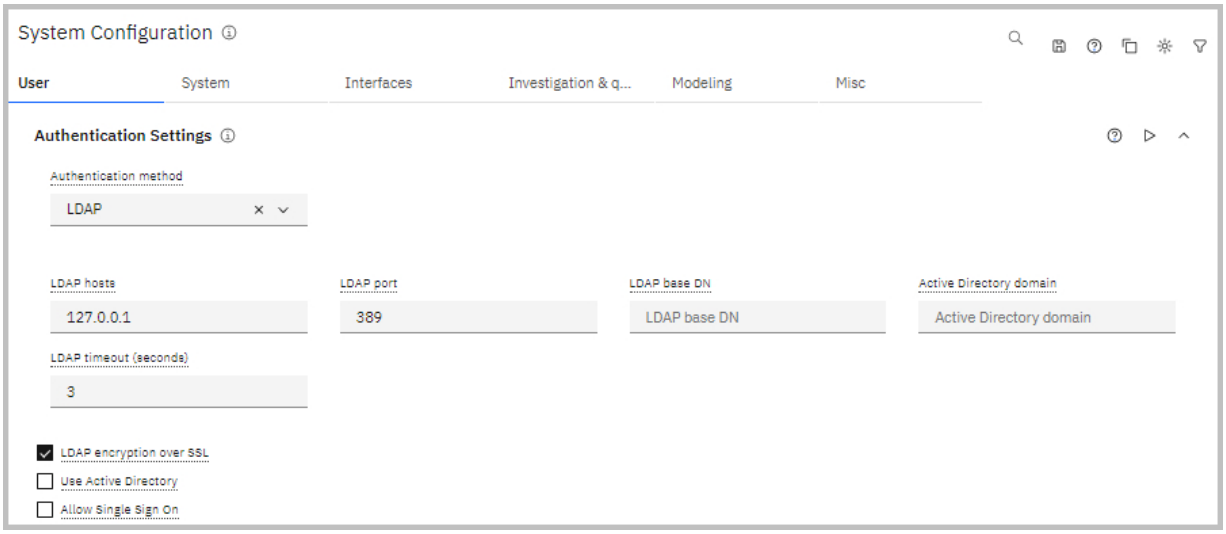


Figure 35. Authentication Settings (LDAP)

You can use the PCI DSS compliance report to verify the compliance of this option in your Safer Payments installation. See “PCI DSS compliance report” on page 46 for details.

3.3.2 Use a strong, one-way cryptographic algorithm, based on approved standards to render all payment application passwords unreadable during storage. Each password must have a unique input variable that is concatenated with the password before the cryptographic algorithm is applied.

Note: The input variable does not need to be unpredictable or secret.

Passwords are stored securely with Safer Payments as salted PBKDF2 hashes. Every user has a unique 32 characters long random salt, which is set during user creation. To generate randomness, two boost functions are used, which take entropy from "/dev/urandom". As random input characters the 62 case-sensitive alphanumeric characters are used. The strength of the salt is thus 1984 bit (32*62). The generated salt is combined with the user’s password and login and is hashed by a PBKDF2 hashing function that uses SHA512 as the message digest algorithm and performs 50027 iterations.

You can use the PCI DSS compliance report to verify the compliance of your Safer Payments installation. See “PCI DSS compliance report” on page 46 for details.

Requirement 3.4

Payment application must limit access to required functions/resources and enforce least privilege for built-in accounts.

- By default, all application/service accounts have access to only those functions/resources needed for purpose of the application/service account.
- By default, all application/service accounts have minimum level of privilege assigned for each function/resource as needed for the application/service account.

Use the Safer Payments default user only to create new personalized user accounts and disable the default user immediately afterward.

Safer Payments meets this requirement. Safer Payments has a built-in role system that allows to define permissions for different user groups.

See “Start the first Safer Payments instance” on page 12 and “Set user privileges” on page 40 for details.

Requirement 4: Log payment application activity

Requirement 4.1

At the completion of the installation process, the "out of the box" default installation of the payment application must log all user access and be able to link all activities to individual users.

The default Safer Payments configuration logs all user access, and links all activities to individual users.

You can use the PCI DSS compliance report to verify the correct configuration of your Safer Payments logging function. See "PCI DSS compliance report" on page 46 for details.

Note: All PCI DSS relevant log messages are classified accordingly in the Safer Payments software. Disabling them results in non-compliance with PCI DSS. See "Change log message settings" on page 39 for details.

You must also configure your system log according to PCI standards. You can find links to the security guides of all supported operating systems in "Miscellaneous system settings" on page 27.

Requirement 4.2

Payment application must provide an audit trail to reconstruct the following events:

- All individual user accesses to cardholder data from the application
- All actions taken by any individual with administrative privileges as assigned in the application
- Access to application audit
- Invalid logical access attempts
- Use of, and changes to the application's identification and authentication mechanisms (including but not limited to creation of new accounts, elevation of privileges, etc.), and all changes, additions, deletions to application accounts with root or administrative privileges
- Initialization, stopping, or pausing of the application audit logs
- Creation and deletion of system-level objects within or by the application

Safer Payments can be configured accordingly to meet all subrequirements of requirement 4.2.

You can use the PCI DSS compliance report to verify the correct configuration of your Safer Payments logging function. See "PCI DSS compliance report" on page 46 for details.

See "Change log message settings" on page 39 for details on adapting log message settings.

After adaptation, the report can be rerun and immediately reflects any changes made.

Note: All PCI DSS relevant log messages are classified accordingly in the Safer Payments software. Disabling them results in non-compliance with PCI DSS.

Note: Safer Payments itself cannot prevent log files to be deleted from outside the application on file level. Organizational procedures must be implemented to prevent such deletions and modifications. Therefore, centralized logging is recommended.

Requirement 4.3

Payment application must record at least the following audit trail entries for each event: [...]

Safer Payments can be configured accordingly to meet all subrequirements of requirement 4.3.

You can use the PCI DSS compliance report to verify the correct configuration of your Safer Payments logging function. See "PCI DSS compliance report" on page 46 for details.

Note: All PCI DSS relevant log messages are classified accordingly in the Safer Payments software. Disabling them results in non-compliance with PCI DSS.

Requirement 4.4

Payment application must provide centralized logging.

Safer Payments provides centralized logging and fully meets this requirement.

All Safer Payments system and audit logs can be accessed from the GUI, and third-party monitoring tools can import Safer Payments log files. Third party monitoring tools can retrieve the log files that are written by Safer Payments from the "log" directory as specified in Safer Payments' base configuration.

To facilitate centralized logging, Safer Payments supports the syslog protocol in Unix/Linux.

See "Change log message settings" on page 39 for details on how to activate centralized logging.

Note: Your central log server must collect all relevant log messages from the system log. You must implement an operational process within your organization to collect the relevant logs from the operating systems logs.

Requirement 5: Develop secure payment applications

Requirement 5.1

The software vendor has defined and implemented a formal process for secure development of payment applications, which includes:

- Payment applications are developed in accordance with PCI DSS and PA-DSS (for example, secure authentication and logging).
- Development processes are based on industry standards and/or best practices.
- Information security is incorporated throughout the software development life cycle.
- Security reviews are performed prior to release of an application or application update.

Safer Payments meets this requirement. The internal software development guidelines of Safer Payments reflect on PCI DSS and PA-DSS requirements. A regular IT security training program is established. It is mandatory for the developers of Safer Payments to follow the OWASP guidelines for secure software development and operations.

5.1.1 Live PANs are not used for testing or development.
Safer Payments meets this requirement. Safer Payments uses artificially generated data for testing, or data that is anonymized using a secure hashing method, such as SHA256 with salt.

5.1.2 Test data and accounts are removed before release to customer.
Safer Payments meets this requirement. Test data and accounts are only added to the software in the test harnesses, never to the installation media. Thus test data and accounts can never find their way to the delivered installation archives.

5.1.3 Custom payment application accounts, user IDs, and passwords are removed before payment applications are released to customers
Safer Payments meets this requirement. Test data and accounts are only added to the software in the test harnesses, never to the installation media. Thus test data and accounts can never find their way to the delivered installation archives.

5.1.4. Payment application code is reviewed prior to release to customers after any significant change, to identify any potential coding vulnerability.

- Code changes are reviewed by individuals other than the originating code author, and by individuals who are knowledgeable in code-review techniques and secure coding practices.
- Code reviews ensure code is developed according to secure coding guidelines. (See PA-DSS Requirement 5.2.)
- Appropriate corrections are implemented prior to release.
- Code-review results are reviewed and approved by management prior to release.

- Documented code-review results include management approval, code author, and code reviewer, and what corrections were implemented prior to release.

Safer Payments uses a revision policy, in which feature development occurs only in the development branch, and never in the release branches. Code changes to release branches are committed individually by issue ticket of the issue tracking system to the source code revision control system. Development managers can review them individually before they are committed to a new branch release (patch release).

5.1.5 Secure source-control practices are implemented to verify integrity of source code during the development process.

Safer Payments uses Git as a source code management system. In our intranet, we maintain a section that provides information on how to use it and how it is configured.

5.1.6 Payment applications are developed according to industry best practices for secure coding techniques, including:

- Developing with least privilege for the application environment.
- Developing with fail-safe defaults (all execution is by default denied unless specified within initial design).
- Developing for all access point considerations, including input variances such as multi-channel input to the application.

The developers of Safer Payments are provided with training materials about secure coding technologies and how to use them. Each Safer Payments developer must undergo secure coding training at least annually.

5.1.7 Provide up-to-date training in secure development practices for application developers at least annually, as applicable for the developer's job function and technology used, for example:

- Secure application design
- Secure coding techniques to avoid common coding vulnerabilities (for example, vendor guidelines, OWASP Top 10, SANS CWE Top 25, CERT Secure Coding, etc.)
- Managing sensitive data in memory
- Code reviews
- Security testing (for example, penetration-testing techniques)

The developers of Safer Payments are provided with training materials about secure coding technologies and how to use them. Each Safer Payments developer must undergo secure coding training at least annually.

Requirement 5.2

Develop all payment applications to prevent common coding vulnerabilities in software-development processes.

The Agile approach that is used by Safer Payments development meets this requirement. In addition, mandatory code reviews by trained reviewers prevent common coding vulnerabilities.

5.2.1 Injection flaws, particularly SQL injection. Also consider OS Command Injection, LDAP and XPath injection flaws as well as other injection flaws.

SQL injection in Safer Payments is impossible as there is no SQL engine in Safer Payments. In general, every entry to Safer Payments is read by an anticipatory parser that does not open any door for code injection.

5.2.2 Buffer Overflow

All buffers that are used for external input are protected against overflowing. All internal buffers do not provide external access and thus cannot be used to attack the software.

5.2.3 Insecure cryptographic storage

Safer Payments meets this requirement. Safer Payments uses industry standard AES-256 and SHA256 crypto-libraries that it keeps in source code form.

5.2.4 Insecure communications

To comply with this requirement, all sensitive and authenticated IP communication must be encrypted. This can be achieved with the internal SSL encryption functions of Safer Payments. See “Configure SSL encryption” on page 14 for details.

Note: For performance reasons, the internal SSL encryption of Safer Payments does not apply for all internal communication between Safer Payments instances. Instead, only keys and passwords are encrypted during transmission, as well as any encrypted attribute values, including the PAN.

5.2.5 Improper error handling

Error messages do not deliver passwords or transaction data values.

5.2.6 All “High” vulnerabilities as identified in the vulnerability identification process at PA-DSS Requirement 7.1

See “Requirement 7: Test payment applications to address vulnerabilities and maintain payment application updates” on page 74 for details.

5.2.7 Cross-site scripting (XSS)

For injecting JS code or any other browser executable code, Safer Payments provides save escaping routines, as described in the online help system of Safer Payments.

Requirement 5.3

Software vendor must follow change-control procedures for all application changes. Change-control procedures must follow the same software development processes as new releases (as defined in PA-DSS Requirement 5.1), and include the following:

Safer Payments uses a standard revision control system and an issue tracking system. The combination of these tools satisfies this requirement.

5.3.1 Documentation of impact

This requirement is met by Safer Payments development operational procedures in combination with the software development that are used by Safer Payments.

5.3.2 Documented approval of change by appropriate authorized parties

This requirement is met by Safer Payments development operational procedures in combination with the software development tools that are used by Safer Payments.

More precisely, each issue that potentially causes code changes in a released branch must be described (by customer or employees) in a ticket that is entered into the issue tracking system. From there, only development managers review the tickets, and if they authorize them for a code change, they forward the ticket to the respective developer or developer team. This ensures that any code change in a released branch occurs only after it has been approved by authorized parties. The entire process is fully documented by the Safer Payments issue tracking system.

5.3.3 Functionality testing to verify that the change does not adversely impact the security of the system.

This is done by using the automated factory test suite of Safer Payments. The Safer Payments software development handbook mandates that at least full factory tests are run on each branch release.

5.3.4 Back-out or product de-installation procedures

Safer Payments enables back-out after the application of a patch in full flight. Similar to the Safer Payments update process, you update one instance at a time. In a redundant (clustered) setup, the other instances assume the computation load and automatically provide the missed data to the updated instance. Back-out is the same process, only in reverse.

The key prerequisite is that you can back-out only one patch level at a time. If you want to back-out from Safer Payments 5.3.1.8 to Safer Payments 5.3.1.4, for instance, you first back-out one instance after another to Safer Payments 5.3.1.7, then to Safer Payments 5.3.1.6.

See “Installing a fix pack update” on page 10 for details.

Requirement 5.4

The payment application vendor must document and follow a software-versioning methodology as part of their system development lifecycle. The methodology must follow the procedures in the PA-DSS Program Guide for changes to payment applications.

The Safer Payments revision policy is documented in this publication under “Versioning Method” on page ix.

Requirement 5.5

Risk assessment techniques (for example, application threat-modeling) are used to identify potential application security design flaws and vulnerabilities during the software-development process.

On our intranet, a section describes the risk assessment techniques of Safer Payments.

Requirement 5.6

Software vendor must implement a process to document and authorize the final release of the application and any application updates.

This requirement is met by Safer Payments development operational procedures. A factory staging process mandates that every patch or upgrade release is authorized by the respective person.

Requirement 6: Protect wireless transmissions

This requirement does not apply to Safer Payments itself, as the software does not require wireless transmissions. If the licensee uses wireless transmission, it must be ensured that subrequirements 6.1, 6.2 and 6.3 are met.

Requirement 6.1

For payment applications using wireless technology, change wireless vendor defaults, including but not limited to default wireless encryption keys, passwords, and SNMP community strings. The wireless technology must be implemented securely.

To ensure compliance, you must verify that

- Default encryption keys are changed at installation, and are changed anytime anyone with knowledge of the keys leaves the company or changes positions.
- Default SNMP community strings on wireless devices are changed.
- Default passwords/passphrases on access points are changed.
- Firmware on wireless devices is updated to support strong encryption for authentication and transmission over wireless networks.
- Other security-related wireless vendor defaults are changed, if applicable.
- Firewalls are installed between Safer Payments (and other systems that store Cardholder Data) and wireless networks.
- Firewalls are configured to deny or control, if such traffic is necessary for business purposes, any traffic from the wireless environment into the Cardholder Data environment.

Requirement 6.2

For payment applications using wireless technology, payment application must facilitate use of industry best practices (for example, IEEE 802.11i) to implement strong encryption for authentication and transmission.

When you use wireless technology with Safer Payments, you must ensure that

- Industry best practices (for example, IEEE 802.11i) are used to include or make available strong encryption for authentication and transmission.
- PA-DSS requirement 6.1 is fully met.

Requirement 6.3

Provide instructions for customers about secure use of wireless technology.

When you use wireless technology with Safer Payments, you must ensure that

- PA-DSS requirement 6.1 is fully met.
- PA-DSS requirement 6.2 is fully met.

Requirement 7: Test payment applications to address vulnerabilities and maintain payment application updates

Requirement 7.1

Software vendors must establish a process to identify and manage vulnerabilities, as follows:

7.1.1 Identify new security vulnerabilities using reputable sources for obtaining security vulnerability information.
Organizational procedures are implemented to keep this information current.

7.1.2 Assign a risk ranking to all identified vulnerabilities, including vulnerabilities involving any underlying software or systems provided with or required by the payment application
On our intranet, we maintain a section that collects common security vulnerabilities as well a risk assessment regarding the Safer Payments software product.

7.1.3 Test payment applications and updates for the presence of vulnerabilities prior to release
A collection of automated tests verifies that there are no known vulnerabilities in the new Safer Payments release. Those tests are run before a release is delivered to customers and are documented within the factory staging.

Requirement 7.2

Software vendors must establish a process for timely development and deployment of security patches and upgrades.

7.2.1 Patches and updates are delivered to customers in a secure manner with a known chain of trust.
A process for the development and deployment of patches and upgrades is established. The same process is used whether the root cause is a security-related issue or just a technical/functional related issue.

7.2.2 Patches and updates are delivered to customers in a manner that maintains the integrity of the patch and update code.
Patches are delivered through Fixcentral. Software is delivered using a secure web server (https protocol). Installation media is protected by SHA256 hash that is provided via the Release Notes in the IBM Support Portal. Any patches and updates are integrity tested before delivery. Before the installation, the customer must manually test the integrity via the checksum as described in “Download the installation image ” on page 5.

7.2.3 Provide instructions for customers about secure installation of patches and updates.
“Where to find more information” on page ix provides a link to the IBM Support Portal and Fixcentral where Technotes, patches, and updates can be securely downloaded.

Requirement 7.3

Include Release Notes for all application updates, including details and impact of the update, and how the version number was changed to reflect the application update.

A process to include Release Notes for all patches or upgrades is established. The Release Notes and the version number are publicly available on the IBM Support Portal. If a patch for a PA-DSS certified release is delivered, a vendor change document is created. The document describes the impact of all changes according to PCI DSS compliance and why it was necessary. The vendor change analysis document can be requested from your account manager.

Requirement 8: Facilitate secure network implementation

Requirement 8.1

The payment application must be able to be implemented into a secure network environment. Application must not interfere with use of devices, applications, or configurations required for PCI DSS compliance (for example, payment application cannot interfere with anti-virus protection, firewall configurations, or any other device, application, or configuration required for PCI DSS compliance).

This requirement is met by Safer Payments.

Note: For performance reasons regular “on demand” anti-virus protection scans are preferred over “on access” scans for Safer Payments’ “ddc” directories.

Requirement 8.2

The payment application must only use or require use of necessary and secure services, protocols, daemons, components, and dependent software and hardware, including those provided by third parties, for any functionality of the payment application. For example, if NetBIOS, file-sharing, Telnet, FTP, etc., are required by the application, they are secured via SSH, S-FTP, TLSv1.2, IPSec, or other technology.

The following services, protocols, daemons, components, and dependent software and hardware are required and used by Safer Payments:

- Computer hardware that supports the operating system.
- Operating system. Refer to “System requirements ” on page 4 for the list of operating systems that are supported with this Safer Payments release in a PCI DSS compliant environment.
- IP/http networking secured by TLSv1.2
- syslog
- SMTP (optional) secured by TLSv1.2
- LDAP (optional) secured by TLSv1.2
- The following libraries are linked statically:
  - openssl-1.1.1n
  - zlib-1.2.11
    - minizip
  - boost_1_70_0
  - bzip2-1.0.8
  - snmp++ 2.6
  - minizip 1.1
  - opencv-4.1.1
  - Itx

  - rapidjson
  - librdkafka-1.3.0
- The list of dynamically linked libraries can be obtained by running the following command from a shell:

```
ldd /usr/bin/iris
```

- In case you want to use the ODBC interface in case actions or notifications Safer Payments links the following plug-in dynamically:

```
Iris_sql_util.so
```

- The plug-in itself might also link other libraries. The list of dynamically linked libraries for the plug-in can be obtained by running the following command from shell:

```
ldd iris_sql_util.so
```

- In addition, Safer Payments can link IBM MQ client libraries (libmqic.so) and a custom parser implementation (sp_custom_parser.so) at run time, if the shared libraries are deployed on the shared library search path of Safer Payments. Both are not required to run Safer Payments. They are developed and released by independent development teams. Therefore, they are not covered by the PA-DSS certification of Safer Payments.

To comply with this requirement, certain IP communication must be encrypted, and several operating system configuration settings must be made. This is addressed in detail in sections “Installation ” on page 3 and “Configure for operational use” on page 25.

To comply with this requirement, you must not use SSD type hard disks, as secure deletion cannot be assured with this technology.

Requirement 8.3

The payment application must not require use of services or protocols that preclude the use of or interfere with normal operation of multi-factor authentication technologies for securing remote access to the payment application that originates from outside the customer environment.

Safer Payments does not interfere with such technologies.

Requirement 9: Cardholder data must never be stored on a server connected to the internet

Requirement 9.1

The payment application must be developed such that any web server and any cardholder data storage component (for example, a database server) are not required to be on the same server, nor is the data storage component required to be on the same network zone (such as a DMZ) with the web server.

To meet this requirement, you must not store cardholder data on a server that is connected to the internet. Refer to “Configure cardholder data storage locations” on page 21 for details.

Systems that are used by external Python programs are out of scope of Safer Payments. External Python programs could be used to store data on external systems. You must ensure that no Cardholder Data is stored on servers that are accessible from the internet.

Requirement 10: Facilitate secure remote access to payment application

Requirement 10.1

Multi-factor authentication must be used for all remote access to the payment application that originates from outside the customer environment.

If the organization of the licensee enables remote access, this must be secured in accordance with requirement 10 by using multi-factor authentication.

Safer Payments itself implements multi-factor authentication through the distribution of personalized client certificates, which must be imported with the browser that is being used. See "Create certificates with OpenSSL" on page 18 for details.

If your installation requires a multi-factor authentication, you can either use multi-factor authentication as provided by Safer Payments or use a third party multi-factor solution. For example, a remote VPN access.

Requirement 10.2

Any remote access into the payment application must be performed securely.

10.2.1 If payment application updates are delivered via remote access into customers' systems, software vendors must tell customers to turn on remote-access technologies only when needed for downloads from vendor, and to turn off immediately after download completes. Alternatively, if delivered via virtual private network (VPN) or other high-speed connection, software vendors must advise customers to properly configure a firewall or a personal firewall product to secure "always-on" connections.

This requirement applies only if the licensee accepts updates to be delivered using remote access. To be PCI DSS compliant, such remote access must be turned on only temporarily, and when needed. It must be turned off immediately after use. Notwithstanding, PCI DSS Requirement 1 must always be met.

Use a securely configured firewall or a personal firewall product, if the computer is connected over VPN or other high-speed connection, to secure these "always-on" connections, per PCI DSS Requirement 1.

10.2.2 If vendors or integrators/resellers can access customers' payment applications remotely, a unique authentication credential (such as a password/phrase) must be used for each customer.

Currently, remote access to a customer's environment is not allowed for vendors or integrators/resellers.

10.2.3 Remote access to customers' payment applications by vendors, integrators/resellers, or customers must be implemented securely.

For any remote access, remote access security features must be used. These include but are not limited to:

- Change default settings in the remote access software. For example, change default passwords and use unique passwords for each customer.
- Allow connections only from specific (known) IP/MAC addresses.
- Use strong authentication and complex passwords for login.
- Enable encrypted data transmission according to PA-DSS Requirement 12.1.
- Enable account lockout after a certain number of failed login attempts.
- Configure the system so a remote user must establish a Virtual Private Network (VPN) connection over a firewall before access is allowed.
- Enable the logging function.
- Restrict access to customer passwords to authorized reseller/integrator personnel.

- Establish customer passwords according to PA-DSS Requirements 3.1.1 through 3.1.11.

See "Requirement 3: Provide secure authentication features" on page 65 for details.

Requirement 11: Encrypt sensitive traffic over public networks

Requirement 11.1

If the payment application sends, or facilitates sending, cardholder data over public networks, the payment application must support use of strong cryptography and security protocols (for example, TLS, IPSEC, SSH) to safeguard sensitive cardholder data during transmission over open, public networks.

To comply with this requirement, all IP communication must be protected by strong cryptography and security protocols. For example, by only using TLSv1.2 or higher, SSH-2, IPSEC, all with at least 128-bit encryption. This can be achieved by using the internal SSL encryption function of Safer Payments and using multi-factor authentication. See "Configure SSL encryption" on page 14 for details.

Requirement 11.2

If the payment application facilitates sending of PANs by end user messaging technologies (for example, email, instant messaging, chat), the payment application must provide a solution that renders the PAN unreadable or implements strong cryptography, or specify use of strong cryptography to encrypt the PANs.

Safer Payments allows to send PANs by end user messaging technologies. For such "notifications" and "case actions", a configuration option ensures that only masked PANs are sent out, according to PA-DSS Requirement 2.2. In consequence, requirement 11.2 is met.

If you use PAN in notifications or case actions, you must activate encryption and enable masked PAN. See "Data encryption" on page 29 on how to turn masking of PANs on.

Requirement 12: Secure all non-console administrative access

Requirement 12.1

If the payment application facilitates non-console administrative access, encrypt all such access with strong cryptography using technologies such as SSH, VPN, or TLS, for web-based management and other non-console administrative access.

All administrative access to Safer Payments is over the Safer Payments API and natively uses the http or https protocol. To comply with this requirement, all API communication must use the https protocol only, providing strong encryption. This can be achieved with the internal SSL encryption function of Safer Payments. See "Configure SSL encryption" on page 14 for details.

Requirement 12.1.1 Instruct customers to encrypt all non-console administrative access with strong cryptography, using technologies such as SSH, VPN, or TLSv1.2 for web-based management and other non-console administrative access.

This requirement aligns with PCI DSS requirement 2.3. To be compliant, you are required to use strong cryptography for non-console administrative access. Use technologies such as SSH2, VPN, or TLSv1.2 (use at least 128-bit encryption strength). Do not use telnet or rlogin for remote access to Safer Payments servers.

Requirement 12.2

Use multi-factor authentication for all personnel with non-console administrative access.

You must implement processes to make sure that for all non-console administrative access multi-factor authentication is used.

Multi-factor authentication consists of at least two of the following:

- Something you have, such as a token device or a smart card.
- Something you are, such as a biometric identification.
- Something you know, such as a password or a passphrase.

To achieve multi-factor authentication, you can either use a third-party solution or the built-in Safer Payments solution, which is described in "Create certificates with OpenSSL" on page 18.

Requirement 13: Maintain a PA-DSS Implementation Guide for customers, resellers, and integrators

Requirement 13.1

Develop, maintain, and disseminate a PA-DSS Implementation Guide for customers, resellers, and integrators that accomplishes the following:

13.1.1 Provides relevant information specific to the application for customers, resellers, and integrators to use.

This document is the Safer Payments PA-DSS Implementation Guide.

13.1.2 Addresses all requirements in this document wherever the PA-DSS Implementation Guide is referenced.

Appendix A, "PA-DSS requirements," on page 61 addresses all requirements.

13.1.3 Includes a review at least annually and upon changes to the application or to the PA-DSS requirements, and is updated as needed to keep the documentation current with all changes affecting the application, as well as to the requirements in this document.

The Safer Payments PA-DSS Implementation Guide is reviewed/updated at least annually, and whenever a relevant software feature is changed or introduced to the software. It is reviewed/updated at least on an annual basis with respect to PA-DSS requirements changes.

Requirement 14: Assign PA-DSS responsibilities for personnel, and maintain training programs for personnel, customers, resellers, and integrators

Requirement 14.1

Provide training in information security and PA-DSS for vendor personnel with PA-DSS responsibility at least annually.

Training is provided at least annually for personnel with PA-DSS responsibilities.

Requirement 14.2

Assign roles and responsibilities to vendor personnel including the following:

- Overall accountability for meeting all the requirements in PA-DSS.
- Keeping up-to-date within any changes in the PCI SSC PA-DSS Program Guide.
- Ensuring secure coding practices are followed.
- Ensuring integrators/resellers receive training and supporting materials.
- Ensuring all vendor personnel with PA-DSS responsibilities, including developers, receive training.

Roles for all responsibilities are assigned to Safer Payments team members. To obtain the list of all responsible persons, request it from your account manager.

Requirement 14.3

Develop and implement training and communication programs to ensure payment application resellers and integrators know how to implement the payment application and related systems and networks according to the PA-DSS Implementation Guide and in a PCI DSS compliant manner.

All staff who is involved in Safer Payments implementation and support is educated regarding PCI DSS requirements and is supplied with training materials. The same applies to staff of our resellers and integrators, if they support Safer Payments installations in scope of PCI DSS.

14.3.1 Review training materials at least annually and upon changes to the application or to PA-DSS requirements. Update the training materials as needed to keep the documentation current with new payment application versions and changes to PA-DSS requirements.

Training materials are reviewed/updated at least annually, and whenever a new software version is released.

To obtain the most current set of the training materials, request them from your account manager.

Sample key custodian form

Name of key custodian: _____

Personnel Number: _____

Position: _____

I hereby confirm and agree that

- I have read and understood the policies and procedures that are associated with key management and I will comply with them.
- I understand that non-compliance with the key management procedures can lead to disciplinary action including termination of employment and prosecution.
- I am aware that cryptographic keys and related information are sensitive information. I will treat them with due care.
- I will never divulge any key management or related security systems, passwords, processes, keys, security hardware, or secrets to any unauthorized party.
- I understand and accept my responsibilities as a key custodian.

Date: _____

Key custodian signature: _____

Supervisor signature: _____

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Red Hat, JBoss®, OpenShift®, Fedora®, Hibernate®, Ansible®, CloudForms®, RHCA®, RHCE®, RHCSA®, Ceph®, and Gluster® are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and Conditions for Product Documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein. IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed. You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Accessibility

Accessibility features help a user who has a physical disability such as restricted mobility or limited vision to use software products successfully. IBM Safer Payments supports the following accessibility features:

- Assistive technologies such as screen readers and screen magnifier software.
- Keyboard operation for specific or equivalent features.
- Customization of display attributes such as color, contrast, and font size.

Index

Special Characters

A

B

C

D

E

F

G

I

K

L

M

O

P

R

S

T

U

V

SC34-7719-00