



バージョン 1.3.2:1

序章

ActiveServerはGPayments Pty Ltdが提供する3Dセキュア2サーバーのソリューションです。

また、GPaymentsは、3Dセキュア2モバイルSDKソリューションであるActiveSDKも提供しています。SDKソリューションの詳細については、当社までお問い合わせください。

ドキュメントの使い方

バージョンと言語

デフォルトで最新のActiveServerのリリースのドキュメントが表示されます。メニューバーの**ドキュメントVer**のドロップダウンをクリックすることで以前のバージョンのドキュメントがご覧になれます。**最新**をクリックすると最新のドキュメントがご覧になれます。

言語のドロップダウンをクリックすると英語のドキュメント、日本語のドキュメントが確認できます。

移動方法

すべてのページの左側に**チャプター**の各リストがあります。メニューから任意の章を選択してそのページに移動することができます。ウィンドウが狭すぎる場合、**チャプター**の各リストが表示されない場合がありますが、左上のハンバーガーのアイコンからアクセスできます。

また、各ページには、右側に**目次**が表示されます。これにはそのページのサブセクションが表示されます。リストから任意のサブセクションを選択してスキップできます。ウィンドウ幅が

狭すぎるか、単純に小さすぎる場合、**目次**が表示されない場合があります。この場合、それらが表示されるようにウィンドウのサイズを変更する必要があります。

見出しの隣にある¶アイコンをクリックして、アドレスバーにリンクをコピーすると、セクションへのリンクを作成できます。

画面右上の**検索**ボックスに語句を入力すると、ドキュメント内を検索できます。検索結果はその語句を含むすべてのページについて表示され、ドキュメントの関連部分にリンクしています。

PDFとしてダウンロード

全てのページには**ActiveServerドキュメントをダウンロード**アイコンが右上にあり、これをクリックすることでActiveServerのドキュメントをPDFとしてダウンロードすることが可能です。このアイコンはオンラインドキュメントの全てのセクションをPDFとしてダウンロードします。

重要

オンラインのドキュメントが常に最新のドキュメントになります。PDFとしてダウンロードした場合、使用しているActiveServerのバージョンのオンラインのドキュメントに変更がないかを日頃からチェックされることを推奨します。

このドキュメントの概要

このドキュメントでは、ActiveServerの紹介、統合プロセスのガイド、トラブルシューティング手順について説明します。

このドキュメントは以下の章で構成されています：

- ・ **序章** - ActiveServerの紹介、このドキュメントの概要。
- ・ **クイックスタート** - インストール手順、ヒント、ActiveServerを稼働させるための必須情報。
- ・ **機能まとめ** - システムの主な機能の説明。
- ・ **ガイド** - 3DS2タスクの為のシステム機能の使用法とその他の機能に関する広範なガイド。

- ・ **APIリファレンス** - APIの使用方法の概要、APIリファレンス・ドキュメントおよびエラーコードリストへのリンク。
- ・ **用語集** - ドキュメント全体で使用される3DS2およびActiveServer固有の用語、略語、定義。
- ・ **ドキュメントバージョン** - ドキュメントの変更ログ。
- ・ **リリースノート** - バージョンごとのActiveServerソフトウェア・リリースノート。
- ・ **法的通知** - 機密保持、著作権、免責事項、責任に関する声明。

製品紹介

ActiveServerは、加盟店および決済代行会社向けの3Dセキュア2の **3DSサーバーソリューション** です。ActiveServerを使用すると、加盟店は、決済フローに対して3DS2を実装でき、ライアビリティ・シフトによる非対面取引（CNP）詐欺に対する保護を保証できます。主要なすべての国際ブランドがサポートされており、PCI-DSS 3.2に完全に対応しており、使いやすいAPIを用いて簡単に統合できます。

ActiveServerは**インハウスの導入に柔軟に対応**しており、GPaymentsの**ホスティング・サービス**からもご利用したりできます。

中核機能

ActiveServerには、以下の中核機能があります。

- ・ **インテリジェント・レポート** - 管理Webアプリケーションから提供されるレポート機能で主要なビジネス情報が利用できます。
- ・ **アプリケーション・サーバーとOSに依存しない** - 一般に使用されているWebコンテナを使用し、WARファイル形式でActiveServerを起動するか、またはSpringを使用してスタンドアロンアプリケーションとして展開できます。この機能は、一般的に使用されているWindowsおよびLinuxベースのシステムを含むすべてのオペレーティング・システムにまで使用できます。
- ・ **HSMに依存しない** - Thales、Gemalto、AWS KMS、またはPKCS11互換HSMを含む、主要なほとんどの暗号化のための汎用ハードウェア・セキュリティ・モジュールと互換性があります。

- ・ **容易な製品アクティベーション** - GPaymentsを使用する組織のアカウントにリンクされたトークンベースのアクティブ化手順で展開されたすべてのActiveServerインスタンスを簡単に管理できます。
- ・ **複数の3DSリクエストと加盟店** - 複数の3DSリクエストと加盟店を同じActiveServerインスタンスに追加できます。
- ・ **移行の容易さ** - 既存のお客様は、GPaymentsを使用してActiveServerへの移行に必要な計画を立て、移行ツールを特定します。

ActiveServer API

ActiveServerは、加盟店の既存のシステムとの統合のため、業界標準に基づいた使いやすいRESTful APIを提供します。リクエストはJSON形式で送受信できます。APIには詳細なドキュメントとサンプルコードが付属しており、シームレスな体験を提供します。

認証API

ActiveServerは認証コンポーネントを公開しており、加盟店はブラウザとモバイルの既存のチェックアウト・プロセスにAPIコードを埋め込むことができます。このコードはActiveServerを呼び出して、認証を実行し、認証レスポンスを返します。これは柔軟なモデルであり、加盟店自身のネットワークからインターネット経由でActiveServerを遠隔操作する機能を加盟店に提供します。

管理API

ActiveServerは、システム管理者や開発者が加盟店およびアクワイアラーを既存のインフラストラクチャに統合できるようにする管理サービスへのAPIを公開しています。管理APIは、すでに加盟店情報を維持管理している加盟店アグリゲーターおよび決済代行会社に特に役に立ちます。これにより、加盟店自身のシステム内でActiveServerの加盟店管理タスクが統合され、管理上のオーバーヘッドが大幅に削減されます。

GPaymentsについて

GPayments

GPaymentsはオーストラリアに本社を置く企業で東京にも支社があります。世界中のお客様に向けて、オンライン取引に関する決済認証製品の提供に特化しています。弊社は国際ブランド、金融機関（イシューアとアクワイアラーの両方）、オンラインサービスプロバイダー、加

盟店、カード会員向けにさまざまなソリューションを提供しています。弊社の3DS2アプリケーション・スイートには、ActiveAccess（ACS）、ActiveServer（3DSサーバー）、ActiveSDK（3DSモバイルSDK）が含まれています。また、GPaymentsはお客様に対して、エンドツーエンドのシステム統合テストのために、3DS2 TestLabsへのアクセスも提供しています。TestLabsには、新しい完全に開発済みのディレクトリ・サーバー、モバイルSDK、EMVCo準拠ACSがあります。

GPaymentsの詳細については、弊社のWebサイト<https://www.gpayments.com/jp>にアクセスするか、sales@gpayments.co.jpまでお問い合わせください。

このドキュメントで誤りを見つけた場合や、追加のサポートについてお問い合わせいただく場合は、support@gpayments.co.jpのGPaymentsテクニカルサポートまで電子メールでお問い合わせください。

クイックスタート

クイックスタートは、ActiveServerのダウンロード、セットアップ、および実行を円滑に行うことを目的としたガイドです。ActiveServerの設定や管理の方法に関する特定のユーザーガイドについては、[ガイドセクション](#)を参照してください。(クイックスタートの2つ下のメニュー項目にあります。)

前提条件

仕様：

仕様	詳細
オペレーティング・システム(OS)	Linux、Windows Server
メモリ	2 GB RAM (推奨)
ディスク容量	最低要件はありませんが、すべてのデータがデータベースに格納されるため、データベース用に十分なディスク容量を用意してください。
Java Development Kit	Java SE Development Kit 8 (Open JDK v1.8)
Javaコンテナ	<code>.jar</code> ファイルは、Servlet 2.4/JSP 2.0をサポートする任意のコンテナで実行できます。デフォルト・コンテナは <code>UnderTow</code> です。
Webブラウザ	Web管理インターフェイス は、Chrome、Firefox、またはEdgeブラウザを使用してアクセスできます。

データベース：

データベース	互換バージョン
MySQL / Amazon Aurora MySQL	5.7
Oracle	11g、12c

データベース	互換バージョン
Microsoft SQL Server	2008 R2、2012、2014、2016、2017
PostgreSQL	8.4 and later
IBM Db2	11.1 and later

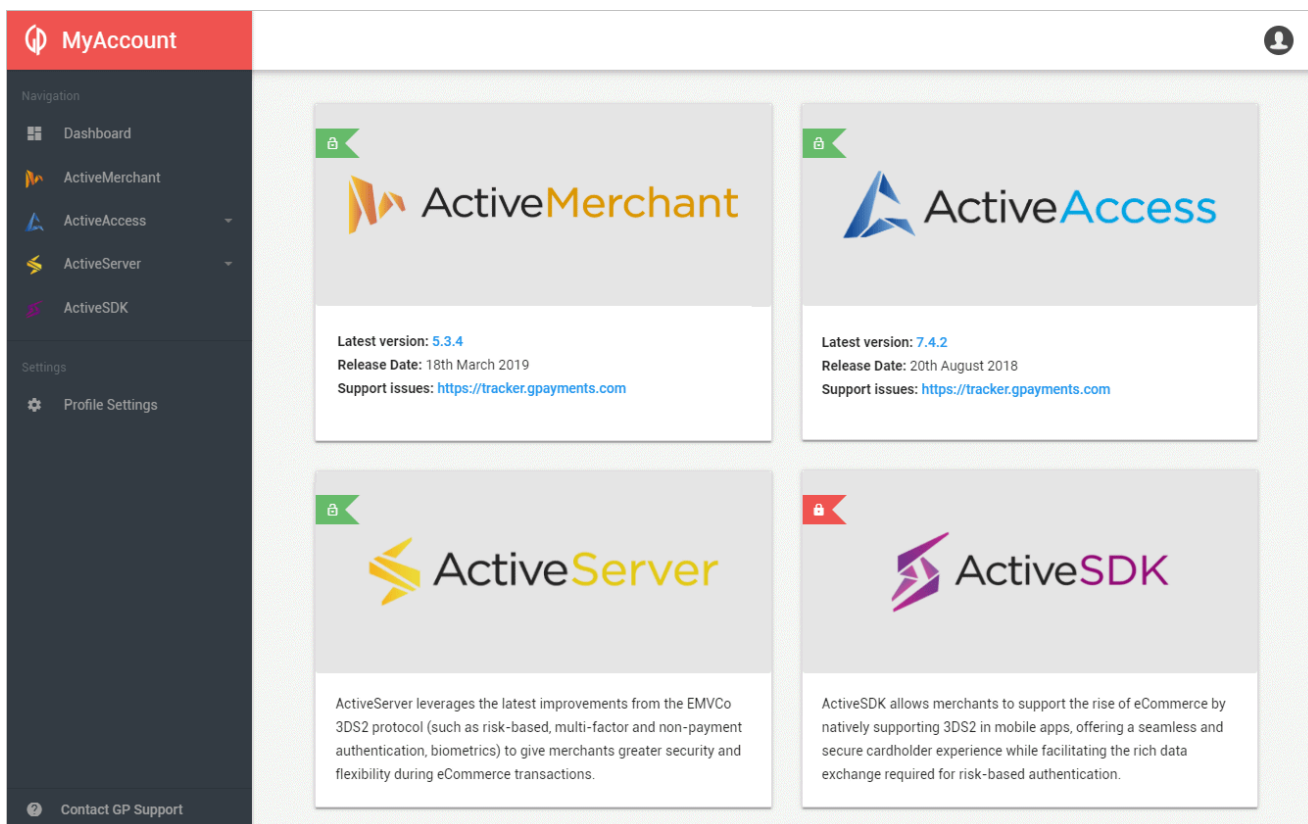
ActiveServerのダウンロード

1. <https://login.gpayments.com/login>からGPaymentsのMyAccountにログインします。

The screenshot shows the GPayments My Account login page. The header includes the GPayments logo and navigation links: HOME, ABOUT, SOLUTIONS, CLIENTS, RESOURCES, CONTACT, and LOGIN. The main heading is 'MY ACCOUNT'. Below this, there is a section titled 'Login to your Account' with a lock icon. It contains input fields for 'EMAIL ADDRESS' and 'PASSWORD', a 'REMEMBER ME' checkbox, a 'LOGIN' button, and a 'Reset my password' link. At the bottom, there is a 'Don't have an account? Register' link. The footer contains the GPayments logo, social media icons (Facebook, Twitter, LinkedIn), and a menu with categories: COMPANY, PRODUCT, RESOURCES, and LANGUAGE. The footer also includes the text 'Copyright © GPayments Pty Ltd. All rights reserved.' and a 'Privacy Policy' link.

!!! "アカウントをお持ちでない場合" アカウントをお持ちでない場合は、次のURLから登録してください。 <https://login.gpayments.com/register>

2. ログインすると、MyAccountダッシュボードが表示されます。



3. 左のメニューから **ActiveServer > Download** を選択して下さい。
4. **ActiveServer > Download** を選択して、リリースパッケージをダウンロードします。

MyAccount 会社設定

MyAccountではユーザーが所属している会社(Organisation)によってソフトウェアへのアクセス権が得られるようになっています。ソフトウェアにアクセス権がある会社に所属しているユーザーはソフトウェアのダウンロードやインスタンスのアクティブ化、ライセンスの確認等の権限が得られます。

ソフトウェアをダウンロードする際に、会社に所属していない場合は新規の会社を**登録**するか、GPaymentsのMyAccountにすでに登録済みで会社を作成したユーザーに招待(invite)してもらう必要があります。**MyAccount > Profile Settings > My Organisation**からユーザーを招待できます。

⚠ 重要

ソフトウェアを既に購入しインスタンスの管理をする場合は既にMyAccountに登録された会社があるはずですので、新しい会社を登録する必要はありません。

会社に既に所属しており、貴社が**ActiveServer**を購入している場合、パッケージのダウンロードができます。既に会社に所属しており、貴社が**ActiveServer**を購入しているのにも関わらずパッケージがダウンロードできない場合はtechsupport@gpayments.co.jpにお問い合わせください。新規のお客様は弊社の営業部までsales@gpayments.co.jpお気軽にお問い合わせください。

インストール

ダウンロードした **.zip** ファイルを展開すると、以下のファイルが表示されます。

```
ActiveServer_vX.XX/  
├── application-prod.properties  
├── as.jar  
├── README.txt  
├── release.txt  
├── startup.bat  
└── startup.sh
```

ファイル：

- `application-prod.properties` - ActiveServerを初期化するための設定ファイル。
- `as.jar` - メインのActiveServer Javaパッケージ。
- `README.txt` - ActiveServer、ドキュメント、ライセンス、サポートに関する一般的な情報。
- `release.txt` - ActiveServerのすべてのバージョンのリリースノート。
- `startup.bat` - Windows用のスタートアップ・スクリプト。
- `startup.sh` - Linux用のスタートアップ・スクリプト。

設定

`application-prod.properties` ファイルを編集することで、ActiveServerのシステムプロパティを設定します。

⚠ アプリケーションプロパティを設定せずにActiveServerは実行する際の注意点

ダウンロードしたパッケージの `application-prod.properties` ファイルには、デフォルトのアプリケーションプロパティが含まれています。これらのデフォルト・アプリケーションプロパティを使用してActiveServerのインスタンスを起動すると、ActiveServerは：

- ・ 一時的にのみRAMに保存され、ActiveServerがシャットダウンするとクリアされるデフォルト・データベースを使用します。
- ・ `${AS_HOME}/conf/security/` にローカルに保存されているSunJCEを使用してキーストア・ファイルを作成します。
- ・ 電子メールサーバー設定をスキップするため、システムがユーザーに電子メール通知を送信しなくなります。

デフォルト・プロパティを使用すると、設定を変更する必要なくActiveServerのインスタンスを迅速に起動できるため、ソフトウェアやインターフェイスを試す際に便利ですが、本番環境のインスタンスをセットアップする前にアプリケーションプロパティを設定する必要があります。

デフォルト・アプリケーション・プロパティを変更するには：

ファイル `application-prod.properties` を開き、各関連パラメータに関連付けられている対応する値を変更します。

システムプロパティには4つのカテゴリーがあります。

- ・ [データベース設定](#)
- ・ [Webサーバー設定](#)
- ・ [キーストア設定](#)
- ・ [電子メールサーバー設定](#)

データベース設定

ActiveServerでは、以下のデータベースがサポートされています。

- ・ [MySQL / Amazon Aurora MySQL](#)
- ・ [Oracle](#)
- ・ [Microsoft SQL Server](#)
- ・ [PostgreSQL](#)
- ・ [IBM Db2](#)

各データベースには、設定可能な以下の一連のプロパティがあります。

- `as.db.vendor=`
データベース・ベンダー/タイプ。デフォルト値は空であり、メモリ内のテスト・データベースが使用されます。メモリ内のテスト・データベースを使用して本番環境に移行することはできません。可能な値は、`mysql`、`oracle`、または `sqlserver` です。
- `as.db.url=`
データベースへの接続に使用されるデータベース接続URL。URLはJDBC形式である必要があります。
- `as.db.username=`
データベースユーザー名。データベース管理者によって設定されたユーザー名を入力します。
- `as.db.password=`
パスワード。データベース管理者によって設定されたパスワードを入力します。
- `as.db.password-plain=`
上記のパスワードを暗号化するか否か。`application-prod.properties` ファイルに格納されている場合、ActiveServerはパスワードを暗号化できます。パスワード暗号化を有効化するには、`false` と入力します。パスワードをプレーン・テキストのままにするには、`true` と入力します。

MySQL / Amazon Aurora MySQL

MySQLデータベースを使用するには、以下のプロパティが必要です。

MySQL データベース・プロパティ(例)

```
as.db.vendor=mysql
as.db.url=jdbc:mysql://<SQL DB ホスト名を入力>:3306/<DB名を入力>?
useUnicode=true&characterEncoding=utf8&useSSL=false
as.db.username=<ユーザー名>
as.db.password=<パスワード>
as.db.password-plain=true
```

Oracle

Oracleデータベースを使用するには、以下のプロパティが必要です。

Oracle データベース・プロパティ(例)

```
as.db.vendor=oracle
as.db.url=jdbc:oracle:thin:@//<Oracle DBホスト名を入力>:1521/<Oracle DB名を入力>
as.db.username=<ユーザー名>
as.db.password=<パスワード>
as.db.password-plain=true
```

Microsoft SQL Server

MS SQLデータベースを使用するには、以下のプロパティが必要です。

MS SQL データベース・プロパティ(例)

```
as.db.vendor=sqlserver
as.db.url=jdbc:sqlserver://<MS SQL DBホスト名を入力>:<ポート番号>;databaseName=<DB名>
または jdbc:sqlserver://<MS SQL DBホスト名を入力>\<インスタンス名>;databaseName=<DB名>
as.db.username=<ユーザー名>
as.db.password=<パスワード>
as.db.password-plain=true
```

PostgreSQL

PostgreSQLデータベースを使用するには、以下のプロパティが必要です。

PostgreSQL データベース・プロパティ(例)

```
as.db.vendor=postgresql
as.db.url=jdbc:postgresql://<PostgreSQLホスト名を入力>:5432/<DB名を入力>
as.db.username=<ユーザー名>
as.db.password=<パスワード>
as.db.password-plain=true
```

IBM Db2

DB2データベースを使用するには、以下のプロパティが必要です。

DB2 データベース・プロパティ(例)

```
as.db.vendor=db2
as.db.url=jdbc:db2://<DB2ホスト名を入力>:50000/<DB名>
as.db.username=<ユーザー名>
as.db.password=<パスワード>
as.db.password-plain=true
```

Webサーバー設定

Webサーバー設定では、デフォルトのサーバー・ポートやその他のネットワーク関連の値を設定できます。ネットワーク・セットアップによっては、HTTPを使用するかHTTPSを使用するかを選択できます。HTTPを使用する場合、エントリー・ポイントがインターネットからアクセス可能である必要があるため、HTTPSトラフィックやSSLターミネーションを処理するためにロードバランサーまたはリバースプロキシが必要です。

デフォルトでは、**サーバーポート** は、**認証コールバックページ**および管理UIインターフェースのリクエストを含むすべてのウェブページリクエストに対応します。

サーバーポート、プロトコル、SSL設定

```
## サーバーポート、プロトコル、SSL設定
## protocol http|https|both
as.server.protocol=http
as.server.http.port=8080
as.server.https.port=8443
as.server.https.key-store=<キーストアファイルパスを入力>
## キーストア・タイプ、 pkcs12またはjks
as.server.https.key-store-type=pkcs12
as.server.https.key-store-password=<キーストアファイルのパスワードを入力>
##trueにセットすることでコネクターを作成します
# as.server.enabled=false
```

- `as.server.https.key-store=`
キーストア・ファイルパス。HTTPSの場合、キーストアが必須です。キーストアには、指定されたHTTPSコネクターのサーバー証明書が含まれている必要があります。本番環境のインスタンスの場合は、サーバー証明書がCAによって商業的に署名されている必要があることに注意してください。

- `as.server.https.key-store-type=`
キーストア・タイプ。ActiveServerでは、2種類のキーストアがサポートされています。可能な値は、`pkcs12`、または `jks` です。CAが発行した商業的に署名された証明書は、通常「pkcs12」形式であり、ファイル拡張子は `.p12` または `.pfx` です。
- `as.server.enabled=`
サーバー・コネクタを有効化または無効化します。サーバー・コネクタを有効化するには、`true` と入力します。サーバー・コネクタを無効化するには、`false` と入力します。

ネットワーク設定によっては、個別のポートで管理アクセスをセットアップすることをお勧めします。そのためには、以下の設定を適用する必要があります。デフォルトでは、管理ポート番号が無効化されています。有効化する場合、以下で設定したポート番号が他のポート番号と競合しないようにしてください。

この設定を有効にすると、すべての管理UIインターフェイストラフィックが指定されたポートに制限されます。 **サーバーポート** は **認証コールバックページ** にのみ対応します。

管理ポート (例)

```
# 管理ポート、プロトコル、SSL設定
# 管理ポートの設定のデフォルトはサーバーポートの設定になります
# trueにセットすることでコネクタを作成します
as.admin.enabled=false
as.admin.http.port=9090
as.admin.https.port=9443
#プロトコル http|https|both
as.admin.protocol=http
as.admin.https.key-store=<キーストアファイルパスを入力>
#キーストア・タイプ、 pkcs12またはjks
as.admin.https.key-store-type=pkcs12
as.admin.https.key-store-password=<キーストアファイルのパスワードを入力>
```

認証および管理APIのポート。相互認証のHTTPSでのみ利用できます。このポートは、ActiveServerインスタンスがアクティブ化されると有効になります。このポートを有効化するには、サーバーの再起動が必要です。

認証APIポート (例)

```
#認証APIポート, HTTPSのみ設定可能  
as.api.port=7443
```

以下のディレクトリ・サーバー・コネクタ設定は、ActiveServerが相互認証でディレクトリ・サーバーとリクエストを送受信するためのものです。これらのコネクタでは常にHTTPSが有効です。ディレクトリ・サーバーのサーバーおよびクライアント証明書は、[DS証明書の管理](#)で説明されているように、後で設定できます。

備考

管理者UIでの証明書の設定が完了したら、サーバーの再起動が必要です。

各ディレクトリ・サーバーには、設定可能な以下の一連のプロパティがあります。

- `as.<Card Scheme>.port=`

各国際ブランドのディレクトリ・サーバーに対してリスンするポート番号です。デフォルト値は以下にあります。

備考

ポート番号が他のポート番号と競合しないようにしてください。

- `# as.<Card Scheme>.enabled=false`

このパラメータはデフォルトでコメント化されています。ディレクトリ・サーバー・コネクタのステータス（無効または有効）を決定します。

ディレクトリ・サーバー・コネクタを無効化するには、`false`と入力します。そうでない場合は、コメント化したままにします。

American Express

American Expressディレクトリ・サーバープロパティ(例)

```
as.amex.port=9600
## falseにセットすることでDSのリスニングポートはオフになります
# as.amex.enabled=false
```

China UnionPay

China UnionPayディレクトリ・サーバープロパティ(例)

```
as.chinaunionpay.port=9601
## falseにセットすることでDSのリスニングポートはオフになります
# as.chinaunionpay.enabled=false
```

China UnionPayのサポートは今後のリリースで追加されます。

Discover / Diners Club International

Discover / Diners Club Internationalディレクトリ・サーバープロパティ(例)

```
as.discover.port=9602
## falseにセットすることでDSのリスニングポートはオフになります
# as.discover.enabled=false
```

JCB

JCB ディレクトリ・サーバープロパティ(例)

```
as.jcb.port=9603
## falseにセットすることでDSのリスニングポートはオフになります
# as.jcb.enabled=false
```

Mastercard

Mastercard ディレクトリ・サーバープロパティ(例)

```
as.mastercard.port=9604
## falseにセットすることでDSのリスニングポートはオフになります
# as.mastercard.enabled=false
```

Visa

Visa ディレクトリ・サーバープロパティ(例)

```
as.visa.port=9605
## falseにセットすることでDSのリスニングポートはオフになります
# as.visa.enabled=false
```

キーストア設定

ActiveServerでは暗号化キーの格納のオプションを3つ提供しています。

- ローカル・キーストア (SunJCE)
- Amazon S3キーストア
- PKCS11 HSM

以下のプロパティを使用してキーストア・タイプを設定します。

```
as.keystore.type=<キーストア・タイプを入力>
```

- `as.keystore.type=`
キーストア・タイプ - 可能な値は、`local`、`s3`、`pkcs11` です。

ローカル・キーストア (SunJCE)

ローカル・キーストア・ファイルを使用するには、以下のプロパティを使用します。

ローカル・キーストア(SunJCE) (例)

```
as.keystore.local.path=${AS_HOME}/security/kestores/
```

- `as.keystore.local.path=`
キーストア・ファイルパス。キーストア・ファイルのファイルパスを入力します。これは、キーストア・ファイルを含むフォルダを参照している必要があります。

警告

Windowsベースのマシンを使用している場合、キーストアフォルダへのフルパスを設定するときにエスケープ文字 (\) がおそらく必要になることに注意してください。

例：Windows共有フォルダがパス `\\ActiveServer\keystores` で使用されている場合、キーストアパスは次のように設定されます。 `as.keystore.local.path=\\\\ActiveServer\\keystores`。

Amazon S3キーストア

ActiveServerでは、キーストアとしてのAmazon S3の使用がサポートされています。Amazon S3キーストアを使用するには、*AWS Bucket*、*AWS Region*、*AWS Credentials*設定を設定する必要があります。

AWSバケット

以下のプロパティでAWSバケットパスを設定します。

Amazon S3キーストア(例)

```
as.keystore.s3.bucket-name=<Amazon S3バケツ名を入力>  
...
```

AWSのリージョン

AWS Regionはいくつかの方法で設定できます。リージョン・コードのリストは以下の表の *Region*列にあります：[Amazon AWS - Regions and Availability Zones](#)。

1. 以下のプロパティでAWSのリージョンコードを設定します。

Amazon S3キーストア(例)

```
...  
as.keystore.s3.region=<Amazon S3のリージョンコードを入力>  
...
```

2. あるいは、ローカル・システムのAWS設定ファイルでAWSのリージョンを設定します。設定ファイルは以下の場所にあります：Linux、macOS、Unixの場合は `~/.aws/config`、Windowsの場合は `C:\Users\USERNAME\.aws\config`。このフィールドには、以下の形式で行を含める必要があります。

```
[default]  
region = <S3のリージョンコード>
```

アクセス権限

指定した鍵へのリードおよびライト権限があるIAMユーザーであることをご確認ください。認証情報の設定に関する詳しい情報は[AWS認証情報](#)をご参照ください。

AWS Key Management Service (KMS)

設定

1. AWSの[ドキュメント](#)に従って、KMSに対称カスタマーマスターキー(CMK)を作成します。
2. キーARNをAWSダッシュボードから `properties` ファイルへコピーします。

AWS KMS (例)

```
as.keystore.kms.key-arn=<Your AWS Key ARN> e.g. arn:aws:kms:us-east-1:  
123456789012:key/19c7b3dc-c49d-401f-bb97-f10bf3e116c9
```

アクセス権限

指定した鍵へのリードおよびライト権限があるIAMユーザーであることをご確認ください。認証情報の設定に関する詳しい情報は[AWS認証情報](#)をご参照ください。

警告

AWS KMSはAPI v2のみに対応することにご留意ください。API v1で取引を実行するとエラーコード1027になります。

PKCS11 HSM

ActiveServerでは、キーストアとしてのHSMの使用がサポートされています。HSMはPKCS11 APIをサポートする必要があります。PKCS11 HSMでハードウェア暗号化を使用するには、以下のプロパティが必要です：

PKCS11 HSM (例)

```
as.keystore.pkcs11.library=<pkcs11ドライバーのライブラリーを入力>  
as.keystore.pkcs11.slot=<スロット番号を入力>
```

- `as.keystore.pkcs11.library=`
HSMドライバ・ライブラリ。Linuxの場合、これは通常 `.so` ファイルです。Windowsの場合、これは通常 `.dll` ファイルです。使用する必要があるライブラリの詳細については、HSMのドキュメントを参照してください。
- `as.keystore.pkcs11.slot=`
HSMのスロット番号。使用する必要があるスロット番号の詳細については、HSMのドキュメントを参照してください。

Info

HSMのセットアップと設定はこのドキュメントの対象外であることに注意してください。ActiveServerで設定する前に、HSMが完全に機能していることを確認してください。

HSMトークンログインは必須です

キー管理にHSMを使用する場合、ASはHSMが「常にトークンログインが必要 (Always require Token Login)」の設定を有効にしている必要があります。通常、この設定はほとんどのHSMで自動的にオンに設定されますが、SafeNetなどのHSMでは、この設定はデフォルトでオンにならない場合があります。手順については、HSMのドキュメントを参照してください。

SafeNet HSMの場合、これは **パブリック暗号なし (No Public Crypto)** セキュリティフラグを設定することで実行できます。管理者は、提供されたコマンドラインユーティリティ `ctconf` を使用してフラグを設定できます。

AWS 認証情報

`as.keystore.type=s3` もしくは `as.keystore.type=kms` を設定した場合、ActiveServerがAWSサービスへアクセスできるようにAWS認証情報を設定する必要があります。AWS認証情報には `access_key_id` と `secret_access_key` があります。AWS認証情報は以下の通り、いくつかの設定方法があります。

1. S3バケットと違う認証情報を使用する場合は、下記の通りAWS認証情報を設定してください。

Amazon S3 keystore (例)

```
as.keystore.s3.credentials.access-key-id=<Your Amazon S3 access key ID>
as.keystore.s3.credentials.secret-access-key=<Your Amazon S3 secret access key>
```

警告

この設定方法はActiveServer v1.3.0より古いバージョンを使用するお客様のために引き続き対応しますが、推奨できません。下記の他の認証情報設定方法を使用することを強く推奨します。

2. 下記と通りAWS認証情報を設定してください。

AWS認証情報 (例)

```
as.aws.credentials.access-key-id=<Your AWS access key ID>  
as.aws.credentials.secret-access-key=<Your AWS secret access key>
```

- ローカル・システムのAWS認証情報プロファイル・ファイルでAWS認証情報を設定してください。認証情報プロファイル・ファイルは以下の場所にあります：Linux、macOS、Unixの場合は `~/.aws/credentials`、Windowsの場合は `C:\Users\USERNAME\.aws\credentials`。AWS認証情報プロファイル・ファイルには、以下の形式で行を含める必要があります。

```
[default]  
aws_access_key_id = <Your Amazon S3 access key ID>  
aws_secret_access_key = <Your Amazon S3 secret access key>
```

この方法を使用する場合、ActiveServer `properties` ファイルの

`as.aws.credentials.access-key-id` と `as.aws.credentials.secret-access-key` パラメーターを使わずに空白のままにすることができます。

- AWS EC2インスタンス上に**ActiveServer**を展開する場合、IAMロールを指定してからそのロールにEC2インスタンス・アクセス権を付与できます。この場合、[Amazon AWS - Using IAM Roles to Grant Access to AWS Resources on Amazon EC2](#)ガイドに従う必要があります。

この方法を使用する場合、ActiveServer `properties` ファイルの

`as.aws.credentials.access-key-id` と `as.aws.credentials.secret-access-key` パラメーターを使わずに空白のままにすることができます。

電子メールサーバー設定

ActiveSeverでは、ユーザーに電子メール通知を送信できます。電子メール通知は、アクティブ化URLをユーザーに通知したり、ライセンスの期限が近づいたらユーザーにリマインドしたりするのに使用できます。

電子メール通知をセットアップするには、認証情報と関連付けられている電子メールアカウントとサーバー詳細が必要です。

Email Server Properties (例)

```
as.mail.host=<SMTPサーバーホストを入力>
as.mail.port=<SMTPサーバーポートを入力>
as.mail.user-name=<メールアドレス>
as.mail.password=<メールアドレス>
as.mail.auth=true
as.mail.start-tls=true
```

- `as.mail.host=`
電子メールサーバーのSMTPドメイン。
- `as.mail.port=`
電子メールサーバーのポート番号。
- `as.mail.user-name=`
電子メールの送信元のアカウントの電子メールアドレス。
- `as.mail.password=`
電子メールアカウントのパスワード。
- `as.mail.auth=`
電子メールアカウントにSMTP認証が必要かどうか。電子メールアカウントに認証が必要な場合、`true` と入力します。そうでない場合は、`false` と入力します。よくわからない場合は、詳細について電子メールサーバーの管理者と相談してください。
- `as.mail.start-tls=`
SMTPサーバーにTLSが必要かどうか。SMTPサーバーにTLSが必要な場合、`true` と入力します。そうでない場合は、`false` と入力します。よくわからない場合は、詳細について電子メールサーバーの管理者と相談してください。

Info

電子メールサーバーのセットアップと設定はこのドキュメントの対象外であることに注意してください。ActiveServerで設定する前に、電子メールサーバーが完全に機能していることを確認してください。

ログ構成

デフォルトでは、**ActiveServer**はすべてのログファイルを `{AS_HOME}/logs/` フォルダに出力します。もし、フォルダが存在しない場合は自動的に作成されます。別のログフォルダの場所を指定する場合は、コメントを外して、次の設定をログを主力したいパスに編集します。

```
# as.logging.path=<Your log file path>
```

マルチノードセットアップの場合、ファイルの命名規則の制約により、ノードごとに個別のフォルダーが必要です。

警告

この値を変更した場合、セットアップでエラーが発生すると、ログファイルが正しく記録されない場合があることに注意してください。パスが正しいこと、および**ActiveServer**インスタンスからアクセスでき、指定されたパスの読み取り/書き込み権限があることを確認してください。

ローカルファイルへのログ出力を無効にする

ローカルファイルへのログ出力が不要な場合、**ActiveServer**にはこの機能を無効にするオプションがあります。

ActiveServerのアプリケーションログをローカルファイルに保存するのを無効にするには、アプリケーションを起動する前に、`startup.sh` (Linux) または `startup.bat` (Windows) ファイルの `AS_PROFILES` の最後にコマンド `disableFileLogs` を追加します、例えば

```
startup.sh  startup.bat
```

```
export AS_PROFILES=prod,disableFileLogs
```

セキュリティーおよびアプリケーションのテクニカルサポートの要件から、この設定に関係なく、ログは引き続き標準のコンソール出力に送信されます。

TLSバージョンの設定

警告

`java.security` ファイルを編集すると、サーバー上のすべてのJavaアプリケーションに影響することに注意してください。GPaymentsは、この変更から生じる可能性のある問題について責任を負いません。

特定のプロトコルを無効にするには、`<jdk directory>/jre/lib/security`にある `java.security` ファイルを編集し、`jdk.tls.disabledAlgorithms` エントリを更新します。元のエントリは次のようになります。

```
jdk.tls.disabledAlgorithms=SSLv3, RC4, DES, MD5withRSA, DH keySize < 1024, \
    EC keySize < 224, 3DES_EDE_CBC, anon, NULL
```

`SSLv2Hello`、`TLSv1`、`TLSv1.1` のように、デフォルトでは含まれていない脆弱と見なされるプロトコルを無効にするには、これらの値を以下のようにエントリに追加します。

```
jdk.tls.disabledAlgorithms=SSLv3, RC4, DES, MD5withRSA, DH keySize < 1024, \
    EC keySize < 224, 3DES_EDE_CBC, anon, NULL, SSLv2Hello, TLSv1, TLSv1.1
```

重要

無効にする推奨プロトコルは単なる提案です。**ActiveServer**のセキュリティ構成を決定する際には、セキュリティチームに相談してください。

`java.security` ファイルを編集したら、実行中の**ActiveServer**インスタンスを再起動して、変更を有効にします。Linux端末で次のコマンドを実行することにより、有効になっているプロトコルを確認できます。

```
nmap --script +ssl-enum-ciphers -p 7443 127.0.0.1
```

ActiveServerの起動

すべてのプロパティが正しく設定され、ActiveServerのインスタンスを起動できるようになりました。

Linuxを使用している場合は、**ターミナル**を開きます。Windowsを使用している場合は、**コマンドプロンプト**を開きます。

スタートアップ・スクリプト（`.sh` または `.bat` ファイル）が含まれているフォルダに作業ディレクトリを変更します。

以下のコマンドを使用してActiveServerを起動できるようになりました。

```
Linux  Windows  
./startup.sh
```

✘ `as.jar` ファイルがスタートアップ・スクリプトと同じディレクトリ内にあることを確認してください

`as.jar` ファイルがスタートアップ・スクリプトと同じディレクトリにないと、スタートアップ・コマンドは動作しません。

ターミナルまたはコマンドプロンプトに以下の出力が表示されるはずです。次のステップで必要になるため、**AdministrationURL**をメモします。

ActiveServerインスタンスの情報

```
-----
ActiveServer by GPayments
```

```
-----
--- |----- /----(-)--- ----- -- /----- -----
-----
-- /| | _ ___/_ ___/_ / _ _ | / / _ \____ \ _ \__ ___/_ _ | / / _
\_ _ ___/
- ___ || /_ / /_ - / _ _ || / / ___/_/_ / / / ___/_ / _ _ || / / ___/
- /
/_/ |_\___/ \_/ / / _ ___/_ \___/ /___/ \___/ / / _ ___/_ \___/ / /
```

```
-----
.
.
.
```

```
-----
ActiveServer by GPayments is up and running.
```

```
Version:                1.0.0
Git Commit Id:          da369ec

Activation:              NOT ACTIVATED, please contact GPayments
Authentication API Port: 7443

Server:                  http://10.0.75.1:8081
Administration:         http://10.0.75.1:8081

Key Store Type:         SUNJCE

Profile(s):              [prod]
-----
```

スタートアップ・スクリプト

スタートアップ・スクリプトでは、環境変数 `AS_HOME` が、`application-prod.properties` が存在するディレクトリに設定されています。ActiveServerは、`AS_HOME` を使用して、設定ファイルを探したり、キーストアを管理したり、ログを出力したりします。別の場所を参照するように `AS_HOME` を設定すると、同じサーバーで複数のActiveServerインスタンスを実行できます。これは、別のディレクトリにパッケージをコピーするか、別のスタートアップ・スクリプトを作成

して、それらのスクリプトで別の場所を参照するように `AS_HOME` を設定することで実行できます。

備考

同じサーバーに複数のインスタンスがある場合、`application-prod.properties` ファイルのいずれかでポート番号が競合しないようにしてください。

ActiveServerプロファイル

`AS_HOME` の設定に加えて、スタートアップ・スクリプトは、環境変数 `AS_PROFILES` も設定します。これは、プロファイルベースの設定を指定するための便利な仕組みです。

デフォルトでは、プロファイルは `prod` に設定されています。

ActiveServerはパターン `application-<profile>.properties` を使用して、プロファイルの設定ファイルを読み込んでいます。そのため、デフォルトの `prod` プロファイルでは、`application-prod.properties` が読み込まれます。ただし、新しいプロファイル（`test` など）を作成して、ActiveServerの異なる設定をセットアップできます。

新しいプロファイルを作成するには：

- `application-test.properties` という名前の新しい設定ファイルを作成し、`prod` 設定ファイルと同じディレクトリに貼り付けます。
- スタートアップ・スクリプトを開き、`AS_PROFILES` の値を `test` に設定します。

ActiveServerは、古い `prod` プロパティの代わりに新しいプロファイルを読み込みます。

ActiveServerは同時に複数のプロファイルを読み込むこともできます。このためには、`AS_PROFILES` の値を `prod, test` に設定することで、ActiveServerが `prod` と `test` の両方のプロファイルからプロパティ・ファイルを読み込みます。

これらのオプションが利用可能な場合、個別の `.properties` ファイルで別々に設定を管理できます。

Tip

すべてのデータベース設定を `application-db.properties` で、Webサーバー設定を `application-web.properties` で管理する場合、`AS_PROFILES` の値を `db, web` に設定すると、さまざまな管理者が管理するためのActiveServer設定を提案できます。

セットアップ・ウィザード

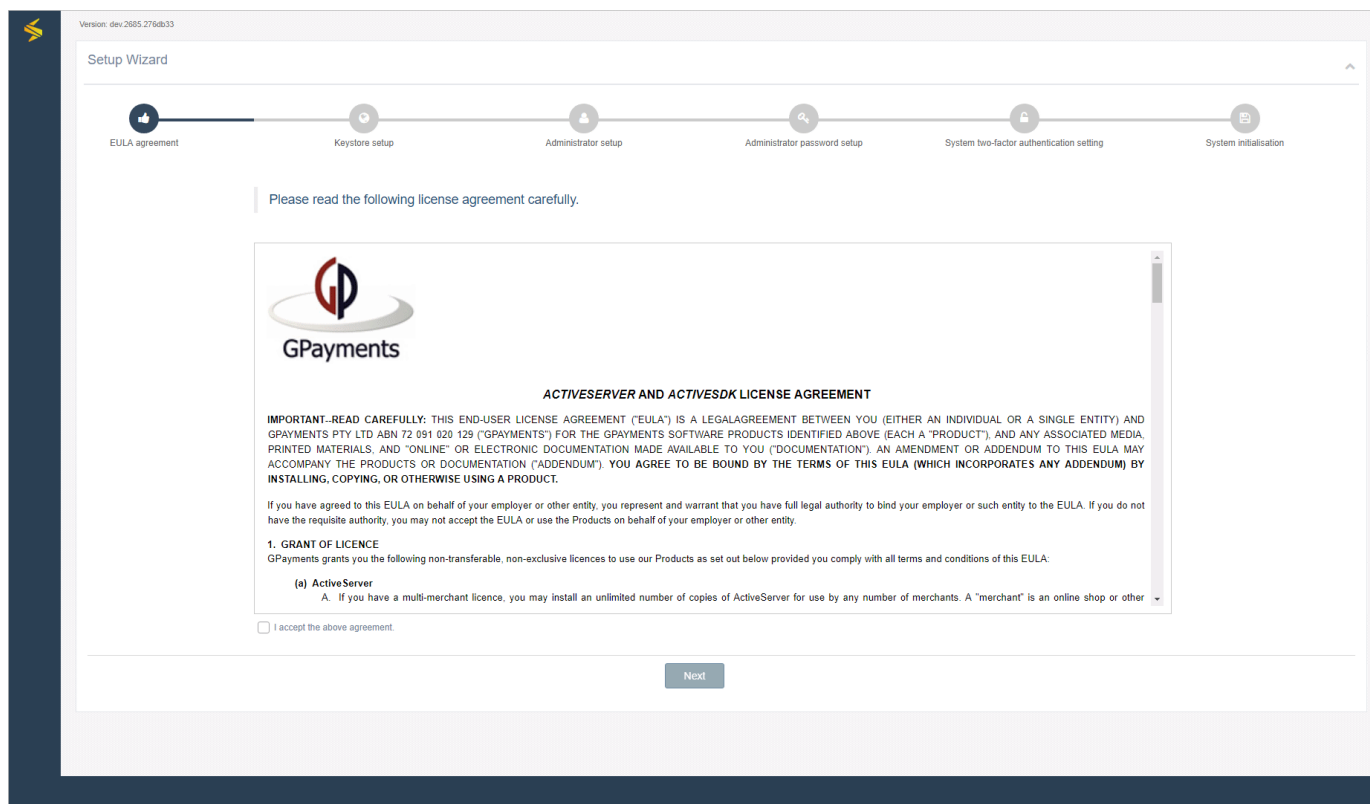
ActiveServerが稼働すると、**Administration URL**から管理者UIにアクセスできます。

初めてActiveServerを実行する場合、セットアップ・ウィザードが表示され、セットアップ・プロセスがガイドされます。

セットアップ・ウィザードには、以下の手順が含まれます。

- EULA契約
- キーストア・セットアップ
- 管理者セットアップ
- 管理者パスワード・セットアップ
- システム2要素認証設定
- システム初期化

EULA契約



EULA契約を確認します。利用契約に同意する場合は、**I accept the above agreement.** チェックボックスを選択して進んでください。

キーストア・セットアップ

The screenshot shows a progress bar at the top with six steps: EULA agreement, Keystore setup (current), Administrator setup, Administrator password setup, System two-factor authentication setting, and System initialisation. Below the progress bar, a message states: "You have chosen the following keystore type for data encryption keys." Underneath, there is a label "Keystore type" followed by a radio button selected for "Software (JCE)". At the bottom, there are "Previous" and "Next" buttons.

- **Keystore type**を選択します。

セットアップ中に**Software**が選択された場合、SUNJCEが使用されます。

application-prod.properties ファイルに適切な詳細を入力することで、PKCS#11 HSMを使用するオプションも利用できます。PKCS#11 HSMのセットアップ方法については、[暗号化モジュール](#)を参照してください。

- 続行するには、**次**ボタンを選択します。

管理者セットアップ

The screenshot shows a progress bar at the top with six steps: EULA agreement, Keystore setup, Administrator setup (current), Administrator password setup, System two-factor authentication setting, and System initialisation. Below the progress bar, a message states: "Please enter user details." There are five input fields: "Username" (text), "First name" (text), "Last name" (text), "Email" (text), and "Time zone" (dropdown menu with "Select from the list" selected). At the bottom, there is a "Create" button and "Previous" and "Next" buttons.

- 管理者アカウントのユーザー詳細を入力します。
- **Create**ボタンを選択すると、アカウントが作成されます。
- 続行するには、**次**ボタンを選択します。

管理者パスワード・セットアップ

Please choose a password for user.

Password *

Re-enter password *

Enter password

Re-enter password

Save

Previous Next

- ・ 管理者アカウントのパスワードを入力します。
- ・ パスワードを再入力して確認します。
- ・ **Save**ボタンを選択すると、パスワードが作成されます。
- ・ 続行するには、**次**ボタンを選択します。

システム2要素認証設定

ActiveServer supports the use of Two Factor Authentication for user login utilising Google Authenticator. Google Authenticator can be installed and setup for mobile [here](#).

You can set whether 2FA should be force enabled for all users below. This setting can be changed later from the System Settings in the administration page.

If 2FA is force enabled now, you will be required to set it up for this account before proceeding.

Force 2FA for all users

Enabled

Previous Next

ActiveServerでは、管理者UIへのサインイン用に2要素認証がサポートされています。

備考

この機能を使用するには、モバイル・デバイスにGoogle認証システムがインストールされている必要があります。

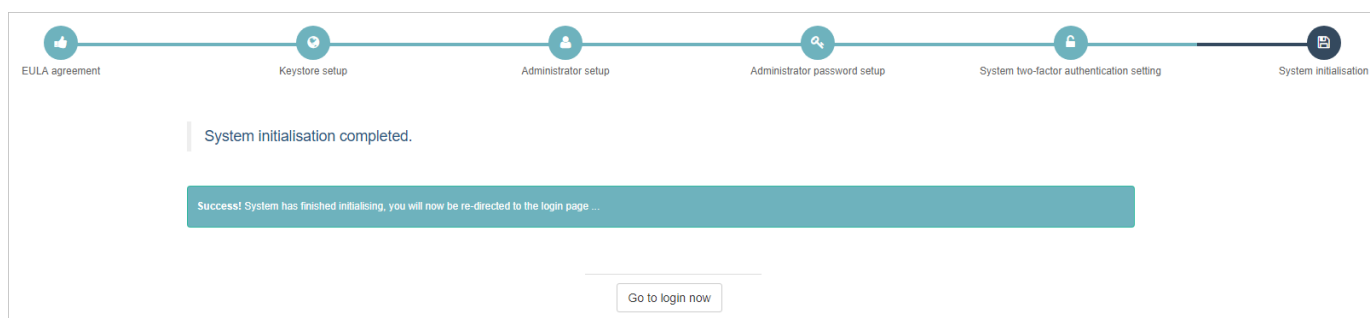
Google Authenticatorのセットアップ手順については、[Google Authenticatorをインストール](#)を参照してください。

デフォルトでは、ActiveServerはユーザーに2要素認証の使用を強制しません。

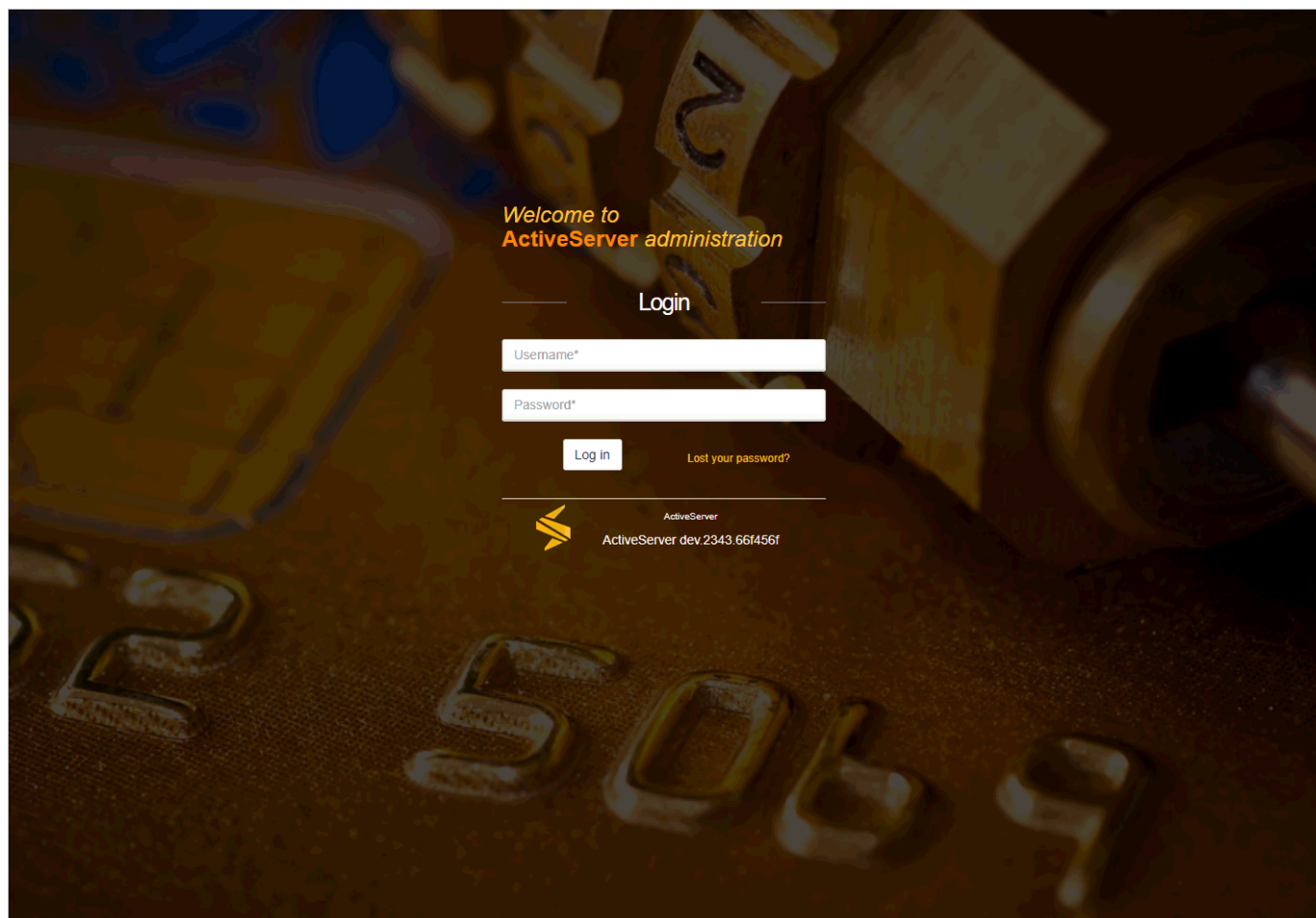
すべてのユーザーに2要素認証を強制するには：

- **Force 2FA for all users**に隣接するトグルを **Enable**にします。
- 続行するには、**次**ボタンを選択します。

システム初期化



セットアップ・ウィザードは、システムの初期化が完了したことを通知し、ActiveServerのログインページにリダイレクトします。



次の手順

この後は、以下を実行できます。

- [ActiveServerのアクティブ化](#)
- [システム設定の管理](#)
- [ActiveServerのAPIを統合する](#)

TestLabs

GPayments TestLabsは、GPaymentsのDirectory ServerおよびAccess Control Serverで構成されています。クライアントがActiveServerインスタンスで機能テストを実行するためのさまざまなカード会員のシナリオがセットアップされています。ActiveServerでサポートされているすべての国際ブランドは、TestLabsでサポートされています。

ActiveServerとの統合

ActiveServerと統合してテスト取引を作成する方法については、[統合ガイド](#)を参照してください。

TestLabsセットアップ

すべてのTestLabs取引では、デフォルトのマーチャント（**Test Merchant**、マーチャントID **123456789012345**）を使用する必要があります。これには、その加盟店プロフィールで利用可能な[クライアント証明書](#)の使用が含まれます。

取引を実行するときは、次のフィールドも使用する必要があります。

- ・ **カード会員名** - 値は `Test Card` または空の値である必要があります
- ・ **有効期限 (YYMM)** - 値は `2508` または空の値である必要があります

警告

上記の値を正しく設定しないと、取引が失敗します。

TestLabsシナリオ

GPayments TestLabsでは以下のテストが実施可能です。取引を実行する際には、指定されたカード番号を使用してください。

認証成功 - フリクションレス

- ・ **説明** - 取引はACSからのチャレンジなしで完了します。

- **ARes結果:**
 - Transaction Status = Y
 - ECI = 05 or 02 (Mastercard)
 - Authentication Valueあり

- **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000000100	5100000000000107	3528000000000106	3400000000000108	6440000000

認証成功 - チャレンジ

- **説明** - 取引は静的パスワードを利用したチャレンジに移行します。パスワード"123456"と入力して取引を完了してください。

- **ARes結果:**
 - Transaction Status = C

- **RReq結果:**
 - Transaction Status = Y
 - ECI = 05 or 02 (Mastercard)
 - Authentication Valueあり

- **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
41000000000005000	51000000000005007	35280000000005006	34000000000005008	6440000000

認証が実施された

- **説明** - 取引は認証を実施しようとし、Attempts応答を返します。

- **ARes結果:**
 - Transaction Status = A
 - ECI = 06 or 01 (Mastercard)
 - Authentication Valueあり

・ **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000100009	5100000000100006	3528000000100005	340000000100007	6440000000

カード番号が登録されていない

- ・ **説明** - 3DS2に登録していないカード番号で取引を実施しようとしています。認証失敗を応答します。

・ **ARes結果:**

- Transaction Status = N
- Transaction Status Reason = 08 (カードレコードが見つからなかった)
- ECIなし
- Authentication Valueなし

・ **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000200007	5100000000200004	3528000000200003	340000000200005	6440000000

認証失敗

- ・ **説明** - 取引は静的パスワードを利用したチャレンジに移行します。パスワード"111111"を入力し、カード会員の認証は失敗します。

・ **ARes結果:**

- Transaction Status = C

・ **RReq結果:**

- Transaction Status = N
- ECI = 00
- Authentication Valueなし

・ **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000300005	5100000000300002	3528000000300001	340000000300003	6440000000

認証を実行できなかった

- **説明** - シミュレーションされたACSとのテクニカルエラーによって取引において認証を実行できなかった。
- **ARes結果:**
 - Transaction Status = U
 - ECIなし
 - Authentication Valueなし

- **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000400003	5100000000400000	3528000000400009	340000000400001	6440000000

認証拒否

- **説明** - 取引はACSによって認証が拒否されます。
- **ARes結果:**
 - Transaction Status = R
 - ECIなし
 - Authentication Valueなし

- **カード番号:**

Visa	Mastercard	JCB	American Express	Discover
4100000000500000	5100000000500007	3528000000500006	340000000500008	6440000000

機能まとめ

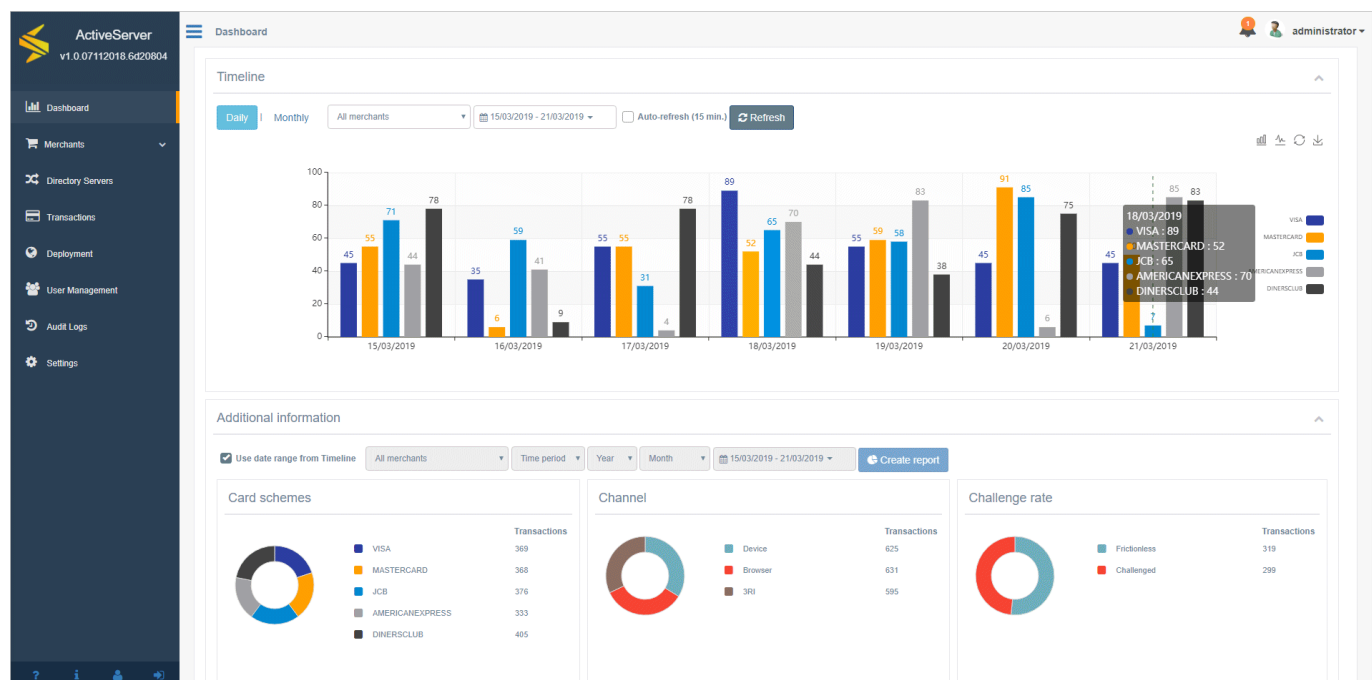
以下はユーザー・インターフェースの概要を含む **ActiveServer**のすべての機能の概要です。各セクションの下には、タスクの実行に役立つ関連ガイドへのリンクがあります。

ダッシュボード

Dashboardには、認証の統計グラフが表示されます。これらの統計情報は、期間をカスタマイズして使用することができ、システム全体の統計を表示したり、加盟店ごとに分割したりすることができます。

ダッシュボードは加盟店の管理用に設計されたロール(**Business Admin**、**Merchant Admin**、**Merchant** など)のユーザーにのみ表示されます。

詳細については、[ガイド > 管理UI > ダッシュボードの使い方](#)を参照してください。



加盟店

Merchantsには2つのセクションがあります：**Merchants**および**Acquirers**。

Merchantsページは、**ActiveServer**の加盟店エンティティを管理するのに使用されます。加盟店を作成、検索、表示、編集、削除できます。これは、3DSリクエスター **クライアント証明書**をダウンロードする場所でもあり、加盟店の暗号化キーをローテーションできる場所でもあります。

Acquirersページは、**ActiveServer**のアクワイアラー・エンティティを管理するのに使用されます。3DS2認証リクエスト用に加盟店プロフィールを割り当てる前に、アクワイアラーを作成、検索、表示、編集、削除できます。

加盟店ページは加盟店の管理用に設計されたロール(**Business Admin**、**Merchant Admin**、**Merchant**ロールなど)のユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#)以下を参照してください：

- [加盟店を検索](#)
- [加盟店を管理](#)
- [アクワイアラを管理](#)

ディレクトリ・サーバー

Directory Serversページは、**ActiveServer**のさまざまな国際ブランドディレクトリ・サーバーを管理するのに使用されます。

それぞれの国際ブランド用のタブがあります。証明書セクションから、国際ブランド固有の接続詳細を入力し、タイムアウト設定を調整し、SSL接続を管理できます。

詳細については、[ガイド](#) > [管理UI](#)以下を参照してください：

- [DS設定の管理](#)
- [DS証明書の管理](#)

取引

Transactionsページは、**ActiveServer**によって処理されるすべての取引のレコードへのアクセスに使用されます。取引はさまざまなフィールドでフィルタリングでき、取引詳細と3DSメッセージの両方を参照するためにアクセスできます。

このメニュー項目とページは加盟店の管理用に設計されたロール(**Business Admin**、**Merchant Admin**、**Merchant**ロールなど)を持つユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#) > [取引を見る](#)を参照してください。

デプロイ

Deploymentページは、現在オンラインの**Node**（ノード）を管理できる場所です。また、インスタンスの**アクティベーションのステータス**を確認する場所でもあり、インスタンスが最初にアクティブ化される場所でもあります。

このメニュー項目とページはシステムアーキテクチャの管理用に設計されたロール(**System Admin**ロールなど)を持つユーザーにのみ表示されます。

詳細については、以下を参照してください：

- ・ [ガイド](#) > [管理UI](#) > [ノードの管理](#)
- ・ [Guides](#) > [インスタンスのアクティブ化](#)

ユーザー管理

User Managementページは、管理者インターフェイスへのユーザーアクセスを付与および管理できる場所です。ユーザーを **作成**、**検索**、**表示**、**編集**、**削除** できます。特に、ここはユーザーがさまざまなシステム機能にアクセスするための **ロール**が付与できる場所です。

このメニュー項目とページはシステム全体のユーザーの管理用に設計されたロール(**User Admin**ロールなど)を持つユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#) 以下を参照してください：

- ・ [ユーザーの管理](#)
- ・ [ロールと権限](#)

監査ログ

Audit Logsページは、システムイベントおよび変更が参照用に記録される場所です。

このメニュー項目とページはシステムアーキテクチャの管理用に設計されたロール(**System Admin**ロールなど)を持つユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#) > [ログ](#)を参照してください。

設定

Settingsページは、ユーザーがインスタンスの**システム**、**セキュリティ**、**3Dセキュア2**関連の設定を管理できる場所です。

このメニュー項目とページはシステムアーキテクチャの管理用に設計されたロール(**System Admin**ロールなど)を持つユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#) > [システム設定の管理](#)を参照してください。


システム情報

Aboutページは、インスタンスの技術仕様が表示される場所です。ここは、GPaymentsサポートチームに技術サポートを問い合わせる際にユーザーに役立つ情報です。

このメニュー項目とページはシステムアーキテクチャの管理用に設計されたロール(**System Admin**ロールなど)を持つユーザーにのみ表示されます。

詳細については、[ガイド](#) > [管理UI](#) > [ActiveServerの情報を見る](#)を参照してください。

通知

Notificationsセクションは、重要なシステム通知をユーザーに連絡する場所です。通知は管理インターフェイスの右上の  アイコンの下に表示されます。

詳細については、[ガイド](#) > [管理UI](#) > [通知](#)を参照してください。

ユーザープロフィール

User profileページは、現在のユーザーがアカウントに関する詳細を編集したり、パスワードを変更したりできる場所です。管理インターフェイスの左下の **Profile** アイコンを選択することでアクセスできます。

詳細については、[ガイド](#) > [管理UI](#) > [ユーザープロフィール](#)を参照してください。

ログファイル

ActiveServerは、毎日ログファイルを作成し、`as_home/logs` ディレクトリに保存します。ログファイルの名前は "**as.yyyy-mm-dd.log**"形式です(例：2019年11月23日に作成されたログファイルは **as.2019-11-23.log**と命名されます)。

ログファイルには、**ActiveServer**コンソールウィンドウに表示されたものと同じメッセージ、警告、エラーが含まれます。

ActiveServerをデバッグ・モードで実行中の場合、ログファイルには取引に関する詳細な情報が含まれるため、サイズが非常に大きくなる可能性があります。必ずログ記録用に十分なディスク容量があることを確認してください。3ヶ月ごとに古いログファイルを削除（またはアーカイブ）することをお勧めします。

ログファイルの詳細度は、[システム](#)で設定できます。詳細については、[ガイド](#) > [管理UI](#) > [システム設定の管理](#)を参照してください。

インスタンスのアクティブ化

⚠ アクティブ化が必須です

新規のActiveServerインスタンスから認証を行う場合はアクティブ化が必須です。

ActiveServerインスタンスをアクティブ化するには：

1.GPaymentsからライセンスを購入する

インスタンスをアクティブ化するためのMyAccount機能にアクセスするには、GPaymentsからライセンスを購入する必要があります。詳細については、sales@gpayments.co.jpにお問い合わせください。

2.インスタンスのセットアップ

クイックスタートに従って、ActiveServerインスタンスがセットアップされていることと、管理インターフェイスにアクセスできることを確認してください。

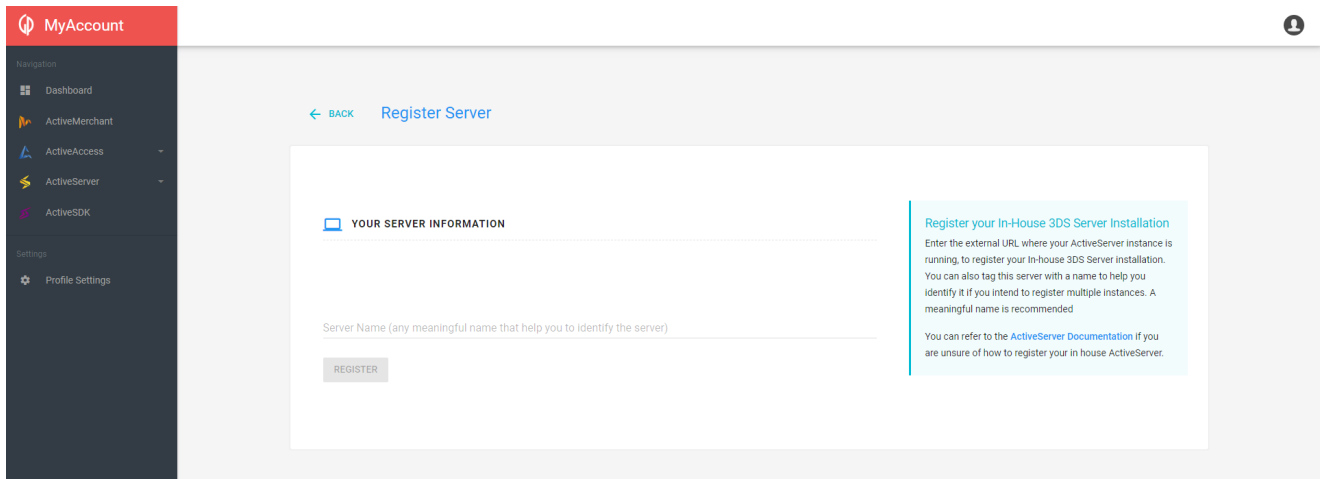
3.外部URLおよび認証API URLの設定

管理インターフェイスで、**Settings > System**に移動し、**External URL**および**API URL**の値を入力します。**Save**ボタンを選択します。

- **External URL** - ActiveServerインスタンスが実行中で、`as.server.https.port` でリスンするように設定した、パブリックにアクセスできるURL。注意点: **外部URL**はロードバランサーのセットアップによってはポート番号を必要としません。例：`https://admin.myserverinstance.com`。
- **API URL** - 認証および管理API呼び出しの受信に使用されるURL。このURLのドメイン名は、API (x.509) の認証用のクライアント証明書の生成にも使用されます。提供されなかった場合、**ActiveServer**はデフォルトではクライアント証明書の生成に外部URLのドメイン名を使用します。このURLは外部からアクセス可能である必要はありません。URLの形式は**外部URL**と同じで、**ポート番号**を指定できます。

4.サーバーを登録し、アクティブ化の方法を選択

1. **MyAccount**にログインします。GPaymentsからライセンスを購入している場合は、すでにActiveServerセクションにアクセスできるはずです。
2. サイドメニューの**ActiveServer > My Instances**を選択します。
3. **ADD NEW SERVER**を選択します。以下のような画面が表示されます。この画面には、**Server Name**の入力フィールドが表示されます。



4. **REGISTER**を選択します。入力したサーバー情報と **Activation State**が表示されます。間違えてしまい、このインスタンスを削除する場合は、**REMOVE**を選択します。
5. **ACTIVATE 3DS SERVER**を選択します。以下のアクティブ化方法からいずれかを選択するように求められます。

オプション1：セッションを使用したアクティブ化

この方法を選択する場合、**前のステップ**で指定した **External URL** がパブリックにアクセスできることを確認してください。

ライセンス・サーバーは、この**External URL**にリクエストを送信し、指定した**External URL**でインスタンスが実行中であることを確認し、インスタンスをアクティブ化します。

オプション2：DNSを使用したアクティブ化

このアクティブ化プロセスでは、GPaymentsのライセンス・サーバーによって生成された **CNAME** レコードを検証することで、ActiveServerインスタンスをアクティブ化します。

DNSレコードは以下のように表示されます：

DNS Record ^

Add the following CNAME record to the DNS configuration for your domain to verify the domain ownership. The procedure for adding CNAME records depends on your DNS service Provider.

Name	<code>_n4xi8anlpzdopxhps0yhutxov3av75xv.[EXTERNAL_URL]</code>
Type	CNAME
Value	<code>_c03ocacrxizwqd2hk1vvczk3anppiwb.41bhl6zhct.gp-validations.myaccount.</code>

DNSレコードを作成するには :

- (1). ドメインのDNSレコードに移動します。
- (2). DNS設定にレコードを追加し、レコードのタイプとしてCNAMEを選択します。
- (3). **Name**の値（上記のスクリーンショットの `_n4xi8anlpzdopxhps0yhutxov3av75xv`）をコピーして、DNSレコードの**Label/Host/Name**に貼り付けます。これはドメイン・ホストによって異なります。
- (4). **Value**の値（上記のスクリーンショットの `_c03ocacrxizwqd2hk1vvczk3anppiwb.41bhl6zhct.gp-validations.myaccount.`）をコピーして、**Destination/Target/Value**に貼り付けます。これはドメイン・ホストによって異なります。
- (5). レコードを保存します。**CNAME**レコードの変更が有効になるには最大72時間かかる場合がありますが、通常はより短い時間で反映されます。

備考

ドメイン・ホストは通常、ドメイン名の購入元です（AWS Route 53, GoDaddy®, Enom®, Name.comなど）。

6. すべてのデータ要素を送信するか、送信したデータ要素をカスタマイズするかを選択することで、ライセンス・サーバーに送信されるデータ要素を選択します。

Transaction data (core) : 請求のために必要な情報であり、送信するために必須（または条件付き必須）です。

ID	名前	必須	グループ	コメント
ADE001	ディレクトリ・サーバー・タイプ	Y	コア	認証リクエストがProductionまたはGPayments TestLabsのディレクトリ・サーバーに送信されたかどうかを追跡するのに使用されます。
ADE002	3DSサーバー取引ID	Y	コア	3DSサーバーが取引に割り当てたID。請求の紛争が発生した場合に取引を相互参照するのに使用されます。
ADE003	SDK取引ID	C	コア	条件付き必須：SDK取引に対してのみ割り当てられ、値が存在する場合に指定する必要があります。請求の紛争が発生した場合に取引を相互参照するのに使用されます。
ADE004	ACS取引ID	Y	コア	ACSが取引に割り当てたID。請求の紛争が発生した場合に相互参照するのに使用されます。
ADE005	取引ステータス	Y	コア	取引ステータス（“Y”、“A”、“N”など）。これは、請求のための最終取引ステータスを決定するのに使用されます（すなわち、取引中にエラーが発生）。
ADE006	取引ステータスの理由	C	コア	条件付き必須：取引が失敗した理由であり、請求のために失敗した正確な理由を特定するのに役立ちます。値が存在（すなわち取引が失敗）する場合に指定する必要があります。
ADE007	取引開始時間	Y	コア	取引開始時間。請求サイクルを決定するときに必要です。
ADE008	取引終了時間	C	コア	条件付き必須：取引終了時間。取引が失敗したり早期に終了したりした場合はnullとなり、利用可能な場合は必須です。

Transaction data (extended) : 請求目的で条件付き必須の場合を除き、任意の情報です。この情報をオプトインすると、GPaymentsが匿名の業界の見識を、参加するクライアントと共有することを許可したことになります。

ID	名前	必須	グループ	コメント
ADE009	ペイメント・ネットワーク	N	拡張	取引に使用されるペイメント・ネットワーク（American Express、China UnionPay、Discover、JCB、Mastercard、Visaなど）。請求の仕組み上この情報が必要な場合を除き、クライアントによる提供は任意です。
ADE010	デバイス・チャネル	N	拡張	取引に使用されるデバイス（BRW、APP、3RIなど）。請求の仕組み上この情報が必要な場合を除き、クライアントによる提供は任意です。
ADE011	認証タイプ	N	拡張	取引に使用される認証タイプ（NPA（非決済）、PA（決済）など。請求の仕組み上この情報が必要な場合を除き、クライアントによる提供は任意です。
ADE012	加盟店ID	C	拡張	内部加盟店ID（アクワイアラーが割り当てたIDではありません）。請求の仕組み上この情報が必要な場合は、クライアントによる提供は条件付き必須です。これは、ライセンス・サーバーが（個別の加盟店IDの計算によって）決済代行会社の規模を判断するのに使用されます。
ADE013	加盟店アクワイアラーIDインデックス	C	拡張	加盟店のアクワイアラー加盟店IDのインデックス番号。請求の仕組み上この情報が必要な場合は、クライアントによる提供は条件付き必須です。これは、ライセンス・サーバーが（個別の加盟店IDの計算によって）決済代行会社の規模を判断するのに使用されます。

Tech support data (core)： GPaymentsがトラブルシューティングおよびプランニング目的で使用される情報です。インスタンス・サーバで条件付きで利用できない場合を除き、送信が必要です。

ID	名前	必須	グループ	コメント
AD001	ActiveServerバージョン	Y	コア	ActiveServerのバージョン（v1.0など）
AD002	OS名	C	コア	OSの名前（Ubuntuなど）
AD003	OSバージョン	C	コア	OSのバージョン（16.04.5 LTSなど）

ID	名前	必須	グループ	コメント
AD004	データベース名	C	コア	データベースの名前 (MySQLなど)
AD005	データベース・バージョン	C	コア	データベースのバージョン (5.7など)
AD006	Javaエディションおよびバージョン	C	コア	使用されているJavaのバージョンのエディション (OpenJDK 1.8.120など)
AD007	ノード数	C	コア	インスタンスのノードの数 (2など)

7. 指定した情報を確認し、インスタンスをアクティブ化します。変更が必要な場合は **BACK** を、そうでない場合は **FINISH**を選択します。

5. アクティブ化

以下のような製品アクティブ化キー (PAK) が表示されるはずです。

- この後使用するため、この値をクリップボードにコピーします。

Product Activation Key (PAK) ^

Copy and paste the following product activation key in the 3DS server administration page.

```
eyJraWQiOiIyMDE3MTEyNSIsImFsZyI6IiVMTmJlU2InOyJ2ZXJzaW9uIjoiaMS4wIiwicGFrfjoiVGwyeWpNUjA0OURNRjQ2SjU2MVpBdktKakFoSII3Rjh0OUtXaURDWU5LSTc2QmxubHBjNWxBWmJyTGZvVkr0aUF4NFI4MnRvS2RsODJaUUs2VlIPMTIIRUx4cjVqMTRvemtNWVZCYkRLeIRVVHF0Tk9sWjZwVmhwNnFYZDVZTjEiLCJhY3RvcmwioiJodHRwOi8vbG9jYXRob3N0OjcwNzAvYXBPbL3YxL2FjdGI2YXRpb24vZGZjb2duaTc3MWpmZ2tiMXdpcTNoc2xpMjduY2Zqancvc2VydmVyliwicHVivXJsljoiaHR0cDovL2xvY2FsaG9zdDo3MDcwL2FwaS92MS9hY3RpdmF0aW9uL3B1YmtleT92PTlwMTcxMTI1Iiwic3RhZ2UiOm51bGwslmV4cGlyZVRpbWUiOiIyMDE5LTA1LTAzVDEwOjUyOjlyInOjI8tRQywB_LsiuXlhw9gpcmphKkyU5Rn_qeM_stH2no-38VmEvCnBRYGRD1bly3l-OzB8PaGcXb2wD05Bq_xA
```

- ActiveServerダッシュボードに戻り、**Deployment > Activation Status**に移動して、MyAccountから詳細情報を入力します。
 - MyAccount Login Name** : インスタンスをアクティブ化したアカウントに登録されている電子メールアドレス。
 - PAK** : クリップボードにコピーした製品アクティブ化キー。

3. **ACTIVATE** ボタンを選択します。成功すると、**Activation Status**が *Waiting to restart* に変更されます。
4. 変更を有効化し、アクティブ化プロセスを完了するには、インスタンスを**再起動**します。以下のスクリーンショットは、**アクティベーションステータス**の例を示しています。再起動後に管理UIのセクション `Deployment` -> `Activation status` をご確認ください。

The screenshot shows a web interface for 'Deployment'. Under the 'Activation status' tab, there is a 'Product details' section. The details are as follows:

Activation Status	Activated
MyAccount Login Name	admin@gpayments.com
Main contact	Admin GPayments

✓ 成功

おめでとうございます！ **ActiveServer**のアクティブ化は以上で終了です。

インスタンスをアップグレード

アップグレード

ActiveServerのアップグレードは、`as.jar` を新しいバージョンに置き換えるだけのシンプルなプロセスです。

既存の**ActiveServer**インスタンスを最新バージョンにアップグレードするには：

1. **ActiveServer**インスタンスノードを停止します。
2. **ActiveServer**ディレクトリを開き、ロールバックが必要な場合は、古い `as.jar` ファイルをバックアップ（単純にコピーするか、アーカイブで保存）します。
3. **ActiveServer**データベースをバックアップします（バックアップ・プロセスとデータベース固有の要件については、データベースのドキュメントを参照してください）。
4. 新しい**ActiveServer**パッケージをダウンロードし、一時ディレクトリに展開し、`as.jar` ファイルを**ActiveServer**ディレクトリにコピーします。
5. **ActiveServer**インスタンスノードを起動します。**ActiveServer**は、起動中、必要に応じてデータベースを自動的にアップグレードします。アップグレードプロセスが完了します。

クラスタリング環境でのアップグレード

複数の**ActiveServer**ノードが同じデータベースで展開されているクラスタリング環境では、データベースのアップグレードプロセスは次のようになります。一度に1つのノードのみがデータベースを移行できるように自動的に処理されます。クラスタ内の残りのノードの起動プロセスは、最初のノードによってデータベースの移行が完了するまでブロックされます。

ロールバック

前のバージョンの**ActiveServer**にロールバックする必要がある場合：

1. **ActiveServer**インスタンスノードを停止します。
2. **ActiveServer**ディレクトリを開き、必要に応じて、古い `as.jar` ファイルをバックアップ（単純にコピーするか、アーカイブで保存）します。
3. **ActiveServer**ディレクトリに以前の `as.jar` をリストアします。

4. 以前バックアップした**ActiveServer**データベースをリストアします。
5. **ActiveServer**を起動します。

ロールと権限

ロールと権限は、業務ロールに関連するさまざまなシステム機能への適切なアクセス権をユーザーに付与するために使用されます。ユーザーは複数のロールを持つことができます。

ActiveServerには、以下の事前定義済みのユーザーロールがあります。

- **System admin** - 展開とライセンス認証、ディレクトリ・サーバー接続管理、システム設定管理、システム通知の監視を含む、インスタンスの技術的な維持管理を管理するためのロールです。
 - ページ・アクセス： *Directory servers*、*Deployment*、*Audit logs*、*Settings*、*About*、*Profile*、*System notifications*
- **User admin** - ロールの割り当てを含む、インスタンスのユーザーを管理するためのロールです。このロールは、システム内のすべての加盟店を表示でき、単一スコープ・ユーザーへの加盟店の割り当てを可能にします。常にこのロールを持ったユーザーが1人は存在する必要があります。
 - ページ・アクセス： *Merchants*、*User Management*
- **Business admin** - ダッシュボード統計の表示、加盟店機能の管理、取引履歴の表示を含む、インスタンス上のすべての加盟店の事業プロセスを管理するためのロールです。
 - ページ・アクセス： *Dashboard*、*Merchants*、*Transactions*、*Profile*
- **Merchant admin** - ダッシュボード統計の表示、加盟店詳細の管理、取引履歴の表示を含む、インスタンス上の単一の加盟店の事業プロセスを管理するためのロールです。
 - ページ・アクセス： *Dashboard*、*Merchants*、*Transactions*、*Profile*
- **Merchant** - ダッシュボード統計の表示、加盟店詳細の表示、取引履歴の表示を含む、インスタンス上の単一の加盟店への読み取り専用アクセスが必要なユーザー用のロールです。
 - ページ・アクセス： *Dashboard*、*Merchants*、*Transactions*、*Profile*

権限スコープ

各ユーザーロールには、**User admin**ユーザーがシステム内のエンティティに対する適切なアクセスレベルを定義できるようにするために添付されたスコープのレベルがあります。

加盟店スコープ

Merchantsについては、スコープはユーザーが**すべての加盟店**とその情報（例：統計情報、詳細、取引）にアクセスできるかどうか、または**単一の加盟店**の情報にアクセスできるかどうかを示します。

- **Allスコープ - Business admin**ロールには**すべての加盟店**での権限があります。これによって、ダッシュボード統計の表示時に**すべての加盟店**を選択したり、すべての加盟店を検索/編集/作成/削除したり、システム内のすべての加盟店の取引を表示したりできます。**User admin**には、単一スコープ・ユーザーに加盟店を割り当てるため、加盟店の詳細を表示するアクセス権があります。
- **Single スコープ - Merchant admin**および**Merchant**ロールには、**単一の加盟店**のみでの権限があります。加盟店がプロフィールに割り当てられると、その加盟店のダッシュボード統計、加盟店詳細、および取引に対してのみ、アクセスできます。
- **No スコープ - System admin**ロールには加盟店の管理に関する権限がないため、加盟店機能のページにアクセスできません。

この役割の分割によって、決済代行会社などのクライアントは単一システムで複数の加盟店を管理し、必要に応じて個別の加盟店を詳細に制御できるようになります。

重要

ユーザーに**All**と**Single**の両方のスコープを持つロールが割り当てられている場合、**All**スコープが優先されます。

加盟店の割り当て

ユーザーの加盟店に関するスコープ・レベル・アクセスが**Single**の場合、**User admin**はそれらをすでに作成済みの加盟店に**割り当て**て管理できます。

ユーザーがすでに加盟店を割り当てている場合、プロフィールでこれを上書きできますが、一度に複数の加盟店を持つことはできません。

権限リストの表

以下の表は、ユーザーロールに付与されている特定の権限の詳細を示しています。スコープ列は、必要に応じてスコープが付随している権限を示します。

User Note

このドキュメントを通じて、特定のユーザーロールで利用可能な機能を示す **User Note** ボックスが表示されます。

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
Dashboard		すべての加盟店統計の表示	すべての加盟店			✓		
		加盟店統計の表示	単一の加盟店				✓	✓
Merchants	Search	すべての加盟店詳細の表示	すべての加盟店		✓	✓		
		加盟店詳細の表示	単一の加盟店				✓	✓

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
		複数の加盟店の作成	すべての加盟店			✓		
		複数の加盟店の削除	すべての加盟店			✓		
	Merchant Settings	すべての加盟店詳細の表示	すべての加盟店		✓	✓		
		加盟店詳細の表示	単一の加盟店				✓	✓
		すべての加盟店詳細の編集	すべての加盟店			✓		
		加盟店詳細の編集	単一の加盟店				✓	
		すべての加盟店メモの表示	すべての加盟店			✓		
		すべての加盟店メモの編集	すべての加盟店			✓		

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
		すべての加盟店の有効ステータスの編集	すべての加盟店			✓		
		すべての加盟店証明書のダウンロード	すべての加盟店			✓		
		加盟店証明書のダウンロード	単一の加盟店				✓	✓
		すべての加盟店証明書の失効	すべての加盟店			✓		
		加盟店証明書の失効	単一の加盟店				✓	
		すべての加盟店暗号化キーのローテーション	すべての加盟店			✓		
		加盟店暗号化キーのローテーション	単一の加盟店				✓	
	Acquirer	アクワイアラーの表示				✓		
		アクワイアラーの作成				✓		

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
		アクワイアラーの編集				✓		
		アクワイアラーの削除				✓		
Directory Servers		ディレクトリ・サーバー設定の表示		✓				
		ディレクトリ・サーバー設定の編集		✓				
		ディレクトリ・サーバー証明書を表示		✓				
		ディレクトリ・サーバー証明書の編集		✓				
Transactions		すべての加盟店取引の表示	すべての加盟店			✓		
		加盟店取引の表示	単一の加盟店				✓	✓
Deployment	Nodes	展開情報の表示		✓				

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
		展開情報の編集		✓				
	Activation Status	アクティブ化詳細の表示		✓				
		製品アクティブ化情報の編集		✓				
User Management	Search	すべてのユーザー詳細の表示	すべてのユーザー		✓			
		ユーザーの追加			✓			
		ユーザーの削除			✓			
	Details	すべてのユーザー詳細の編集	すべてのユーザー		✓			
		すべてのユーザーロールの編集	すべてのユーザー		✓			
		すべてのユーザー・ステータスの編集	すべてのユーザー		✓			

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
Audit Logs		すべての監査ログの表示		✓				
Settings	System	システム設定の表示		✓				
		システム設定の編集		✓				
	Security	セキュリティ設定の表示		✓				
		セキュリティ設定の編集		✓				
	3D Secure 2	3Dセキュア2設定の表示		✓				
		3Dセキュア2設定の編集		✓				
About		詳細の表示		✓	✓	✓		
User profile	プロフィールの編集	ユーザー詳細の表示	単一のユーザー	✓	✓	✓	✓	✓
		ユーザー詳細の編集	単一のユーザー	✓	✓	✓	✓	✓
Notifications		システム通知の表示		✓				

ページ	サブページ	権限	スコープ	System Admin	User Admin	Business Admin	Merchant Admin	Merchant
		ユーザー通知の表示		✓	✓	✓	✓	✓
Reset Password		パスワードのリセット		✓	✓	✓	✓	✓

ActiveMerchantから移行

ActiveMerchantマイグレーション機能により、**Business管理者**ユーザーはGPayments ActiveMerchant(3DS1 MPI)から加盟店とアクワイアラーのインポートが可能となり、3DS1から3DS2へ容易に移行(マイグレーション)できます。この機能は**Administration interface > Settings > ActiveMerchant Migration**タブからアクセスすることができます。

The screenshot displays the 'ActiveMerchant Migration' settings page. It includes a search form with fields for 'Merchant name' and 'Merchant ID', and a 'Search' button. Below the search form is a table with the following data:

Merchant name	Merchant ID	Status	Import status
	C1gE5LNbP	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.
IEUK3X5KpL	C1G6pPjlv	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: 3DS Requestor URL. Select merchant to manually import.
	jbUZ1ub3sQ	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.
	nAjyngqYIT	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name, 3DS Requestor URL. Select merchant to manually import.
tsZY0aW9EG	RhJ1ijZRO	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: 3DS Requestor URL. Select merchant to manually import.

サポートされているActiveMerchantのバージョン

ActiveMerchant v5.1.12以降がサポートされています。

加盟店のマイグレーション

すべての加盟店には、加盟店をインポートできるかどうかを示す**インポートステータス**が割り当てられます。また、**ActiveMerchant**データベースで**Enabled**または**Disabled**であったかどうかを示す**ステータス**も表示されます。これらのオプションの両方を使用して、検索結果をフィルタリングできます。

インポートステータス

インポートステータスは、以下の値のいずれかになります：

- **Already imported** - 加盟店はすでにインポートされています。*加盟店名*と*加盟店*IDのペアはすでにシステムに存在します。このステータスの加盟店のチェックボックスは無効化されています。
- **Unavailable** - 加盟店を自動的にインポートできないため、手動でインポートする必要があります。[手動で加盟店をインポート](#)をご参照ください。このステータスの加盟店のチェックボックスは無効化されています。
- **Warning** - 加盟店は*国名*、もしくは*デフォルト通貨*の値がありませんので、デフォルトの値でインポートされます。デフォルトの値はのインポート開始時のポップアップ画面にて指定できます。インポートにより、この加盟店の*国名*もしくは*デフォルト通貨*の値は上書きされます。
- **Available** - 加盟店は通常の値で自動的にインポートできます。

手動で加盟店をインポート

任意の加盟店の行をクリックし、未入力または不正確な欄に値を割り当て、あるいはその他あらゆる欄を編集することで、手作業で加盟店のインポートを行うことができます。ステータスが**Unavailable**の加盟店については、この方法がインポートの唯一のオプションです。

アクワイアラーのマイグレーション

すべてのアクワイアラーには**Import status**が割り当てられ、アクワイアラーのインポート可否が表示されます。

インポートステータス

インポートステータスは、以下の値のいずれかになります：

- **Already imported** - このアクワイアラーはすでにインポート済みです。アクワイアラーの名前はすでにシステム内に存在します。このステータスのアクワイアラーのチェックボックスは無効化されています。
- **Available** - このアクワイアラーは、通常の値で自動的にインポートすることが可能です。

マイグレーションの手順

加盟店とアクワイアラーのマイグレーションプロセスは酷似しています。以下に、加盟店のマイグレーションプロセスの手順を概説します。

1. ActiveMerchantのデータベースの詳細を `application-prod.properties` ファイルに構成してください。

```
as.migration.db.vendor=<ActiveMerchantデータベースのベンダー> e.g. mysql,  
oracle, mssql, db2 or postgres  
as.migration.db.url=<ActiveMerchantデータベースのJDBCURL> e.g. jdbc:mysql://  
<Your My SQL DB Host>:3306/<Your DB Name>  
as.migration.db.username=<ActiveMerchantデータベースのユーザーネーム>  
as.migration.db.password=<ActiveMerchantデータベースのパスワード>
```

2. `application-prod.properties` ファイルに変更が加えられた場合には、ActiveServerインスタンスを再起動する必要があります。
3. **Administration interface > Settings > ActiveMerchant Migration** タブに進み、**Connect** を選択し、**ActiveServer**と**ActiveMerchant**データベースとの接続を確立します。接続できなかった場合には、エラーが表示されます。
4. インポートしたい加盟店 / アクワイアラーのチェックボックスにチェックを入れます。すべての加盟店をインポートしたい場合には、テーブルヘッダーのチェックボックスにチェックを入れます。これにより、テーブル内のすべてエントリーが選択されます。

ステータスがUnavailableの加盟店

インポートステータスが**Unavailable**の加盟店は、必須欄のいずれかが未入力または無効となっているため、自動的にインポートを行うことはできません。これら加盟店のインポートについては、[マニュアルインポート](#)セクションをご覧ください。

5. インポートボタンを選択すると、以下のようなダイアログがポップアップします。

Import merchants

Summary

- 23 merchants will be imported.
- 11 merchants have complete data fields and can be imported without changes.
- 4 merchants are missing **Country** and will be overridden with the value below.
- 8 merchants are missing **Default currency** and will be overridden with the value below.

Country *

Default currency *

Import Back

ID	Merchant name	Status	Error Message
C1igE5LNbP		ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.
UK3X5KpL	C1G6pfYjv	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: 3DS Requestor URL. Select merchant to manually import.
	jbUZ1ub3sQ	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.
	nAJyngqYIT	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: 3DS Requestor URL, Merchant name. Select merchant to manually import.
ZYOaW9EG	RhU1hjtZRO	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: 3DS Requestor URL. Select merchant to manually import.
	IRaK3gzE30	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.
	3ovbaFXsCb	ENABLED	Unavailable - cannot import automatically due to field condition mismatch: Merchant name. Select merchant to manually import.

インポートステータスが**Warning**の加盟店を選択している場合には、このダイアログで、デフォルト値を選択してインポートに使用するよう求められます。未入力欄がある加盟店が複数存在し、同じデフォルト通貨または国名の入力が必要である場合に、この機能は便利です。

6. **Import**ボタンを選択して、インポートプロセスを開始します。プロセスが終了して、確認ダイアログが表示されるまでお待ちください。確認ダイアログが表示される前にページから離れるなどの理由で、プロセスが中断してしまった場合、すでにインポート済みの加盟店はすべて保存済みとなり、次回改めてインポートを実行する際には、**Import Status**が**Already imported**となっています。

よくある質問

ActiveMerchantは実行中にしておく必要がありますか？

マイグレーションの際、ActiveMerchantを実行中にしておく必要はありません。**ActiveServer**が、**ActiveMerchant**データベースにアクセスできればマイグレーションを実行できます。

マイグレーションによって、ActiveMerchantデータベースは影響を受けますか？

いいえ。マイグレーションプロセスでは、構成済みのActiveMerchantデータベースから読み取りを行うだけで、現在の**ActiveMerchant**のデータベースのデータは影響されません。

加盟店またはアクワイアラーは、すべて1回のセッションでインポートする必要がありますか？

いいえ。マイグレーションプロセスは複数回実施することができます。**ActiveServer**が**ActiveMerchant**データベースに接続するたびに、加盟店またはアクワイアラーの情報の存在が検出された場合には、自動的に**Import Status**が**Already imported**に設定されます。

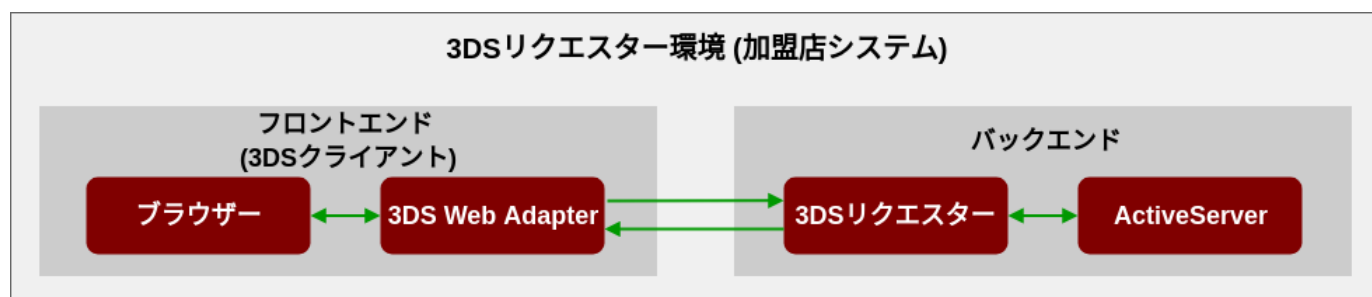
統合まとめ

加盟店または決済代行会社に3DS2認証を組み込むためには、eコマースサイト、eコマースサイトのバックエンドシステムに**ActiveServer's**認証APIを実装する必要があります。

API呼び出しは、アプリが動作中に特定のタスクを実行するために起動可能な処理です。すべてのAPIリクエストは、動作の軽いデータ転送形式であるJSON形式で作成されます。API文書の詳細は、[APIドキュメントまとめ](#)を参照してください。

この章では、**ActiveServer**に接続できるように加盟店の**ウェブサーバー**を組み込んでテスト用の取引を実行する方法について、概要を説明します。加盟店の**App**の組み込みについては、[ActiveSDKドキュメント](#)を参照してください。

3DS2を利用するためには、加盟店サイトに2つの機能、すなわちフロントエンドの**3DS web adapter**とバックエンドの**3DSリクエスター**を実装する必要があります。次の図は、ブラウザー、3DS webアダプター、3DSリクエスター、3DSサーバーの間の関係を示します。



- ・ **3DS web adapter** - 3DS webアダプターは加盟店サイトの3DS2コンポーネントであり、利用者のデバイスから3DSリクエスターに3Dセキュアデータを渡すために使用されます。3DS webアダプターとなり得るものの例としては、ブラウザーでの操作に対する応答を実行し3DS認証リクエストを3DSリクエスターに送信する `javascript (.js)` があります。
- ・ **3DSリクエスター** - 3DSリクエスターはコントローラーであり、3DS webアダプターと3DSサーバーの間のブリッジとして使用されます。3DSリクエスターは3DS webアダプターからの3DS認証リクエストを受信し、それらのリクエストの形式を整えて3DSサーバーに送信します。また、3DSリクエスターは3DSサーバーからの認証結果を受信して3DS webアダプターに転送します。

取引の実行

3DS2を使用した取引をシミュレートするため、この[デモ用加盟店サイト](#)を使用して、認証APIがどのように動作するかを確認できます。

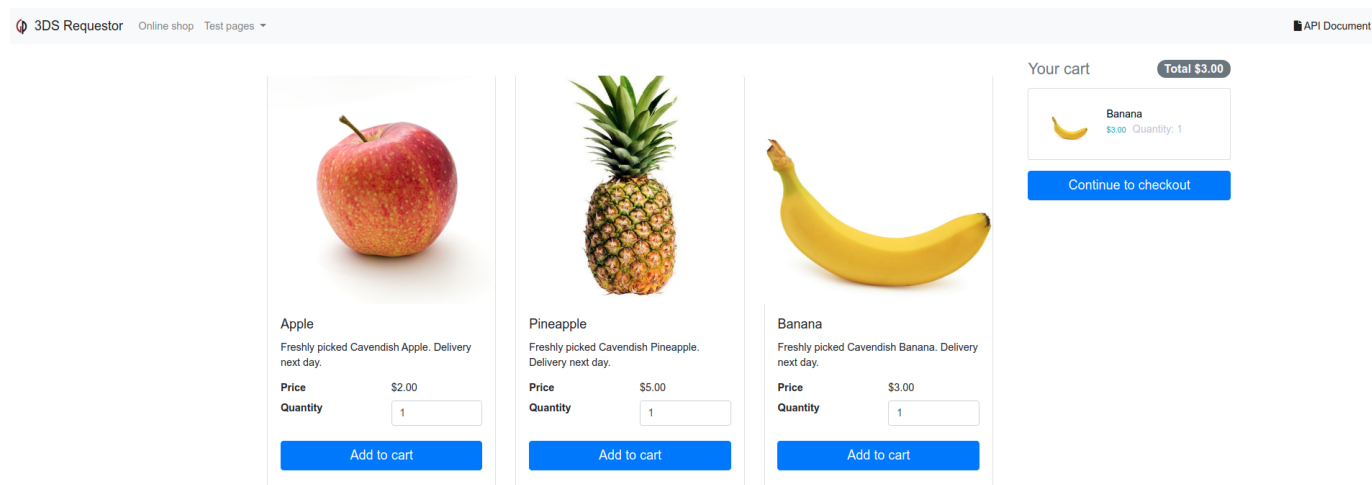
Tip

この「組み込みガイド」のサンプルでは、この[デモ用加盟店サイト](#)を使用しますので、先に進む前に使用してみてください。

フリクションレス・フロー

フリクションレス取引を開始するには、[デモ用加盟店サイト](#)を開き、果物をカートに追加します。

画面右上の**Cart**アイコンを選択すると、カートの内容が表示されます。



The screenshot shows a web interface for a demo online shop. At the top, there is a navigation bar with '3DS Requestor', 'Online shop', and 'Test pages'. On the right, there is a link for 'API Document'. The main content area displays three product cards: 'Apple' (Freshly picked Cavendish Apple, Delivery next day, Price \$2.00, Quantity 1), 'Pineapple' (Freshly picked Cavendish Pineapple, Delivery next day, Price \$5.00, Quantity 1), and 'Banana' (Freshly picked Cavendish Banana, Delivery next day, Price \$3.00, Quantity 1). Each card has an 'Add to cart' button. On the right side, there is a 'Your cart' section with a 'Total \$3.00' badge, a small cart icon, and a 'Continue to checkout' button.

Checkoutボタンを選択して決済ページに移動します。

カード番号など、決済と請求書送付先に関するデフォルトの情報があらかじめセットされており、取引の実行にこれらの情報を使用できます。**Continue to checkout**ボタンを選択すると3DS2認証処理が開始されます。

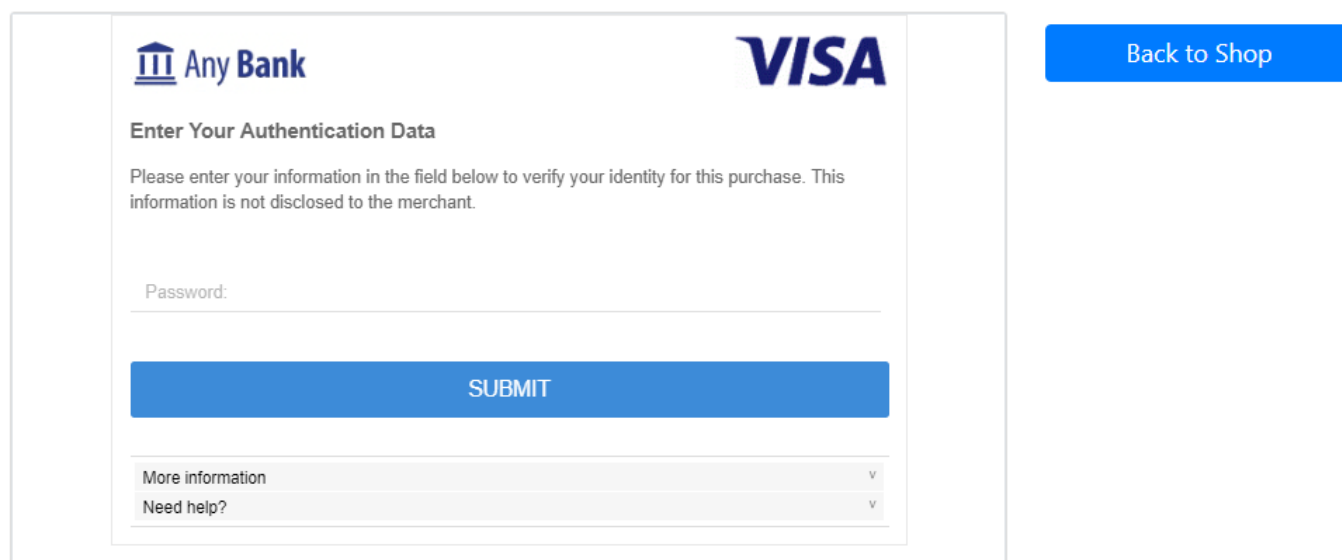
3DS web adapterはカード会員情報を収集して**3DSリクエスター**に送信します。**3DSリクエスター**はこの情報を使用してAPIリクエストを作成して**3DSサーバー**に送信し、**3DSサーバー**は3DS2メッセージングを開始します。**3DSリクエスター**は認証結果を待つ**3DS web adapter**に結果を返し、結果が次のようなwebページに表示されます。

これで、**フリクションレス・フロー**を使用した取引は完了です。シミュレートした取引は低リスクと見なされたためチャレンジは要求されませんでした。

チャレンジ・フロー

チャレンジ・フローをテストするには、**Back to Store**ボタンを選択してチャレンジシナリオをテストしたい場合は、チェックアウトページでカード番号を**4100000000005000**に設定しましょう。このシミュレーションではACSは、取引がハイリスクであると判断しカード会員の追加の

アクションが必要と判断しますのでチャレンジ・フローに移行します。次のようなチャレンジ画面が表示されます。このデモで使用するパスワードは**123456**です。



パスワードを入力すると正しく取引が処理されるはずですが。実際の状況における、このチャレンジの方法としては、イシューアのACSやカード会員に対応する登録済み認証方法に応じて**OTP**や**生体認証**などさまざまな方法が考えられます。

次のチャプター

下のフッターの**次**を選択し、**認証処理**に関する詳細を学習してください。

認証処理

3DSリクエスターは、認証中に次の3つの処理を実行します。

1. **認証の初期化**—3DSリクエスターは**ActiveServer**にリクエストを送信して認証を初期化し、認証を行えるよう**ActiveServer**を準備します。
2. **認証の実行**—**ActiveServer**は認証を実行します。3DS2には2つの主要な認証フロー**リクシオンレス・フロー**と**チャレンジ・フロー**があります。これらについては**処理フロー**の項で説明します。
3. **認証結果の取得**—3DSリクエスターに認証結果が返されます。

処理 1: 認証の初期化

このステップで、フロントエンドの**3DS web adapter**はカード会員が入力した情報を取り込み、バックエンドの**3DSリクエスター**に渡します。次に**3DSリクエスター**は、3DS2から要求されたすべての情報を**ActiveServer**に渡し、認証処理が開始されます。

実装方法まとめのサンプルで、利用者が**Checkout**を選択すると、**3DS web adapter**が**3DSリクエスター**に **認証の初期化** メッセージを送信します。

3DSリクエスターは **認証の初期化** メッセージを受信し、ActiveServer認証APIに適合する形式にし、一意の**3DS Requestor transaction ID**(`threeDSRequestorTransID`)を生成して、メッセージへ追加します。メッセージのデータ項目が揃いましたら、メッセージは**ActiveServer**へ送信されます(`/api/v2/auth/brw/init`)。

ActiveServerが **認証の初期化** メッセージを受信するとチェックアウトページで**3DSリクエスター**がページフォワーディングをセットアップするためのコールバック

URL(`threeDSServerCallbackUrl`)を返却します(**3DS Web Adapter** のコールバックは隠されたiframeを使用します) 。このiframeが設置されることで、**ActiveServer**は、ブラウザー情報を収集し、認証処理を行える状態になります。

備考

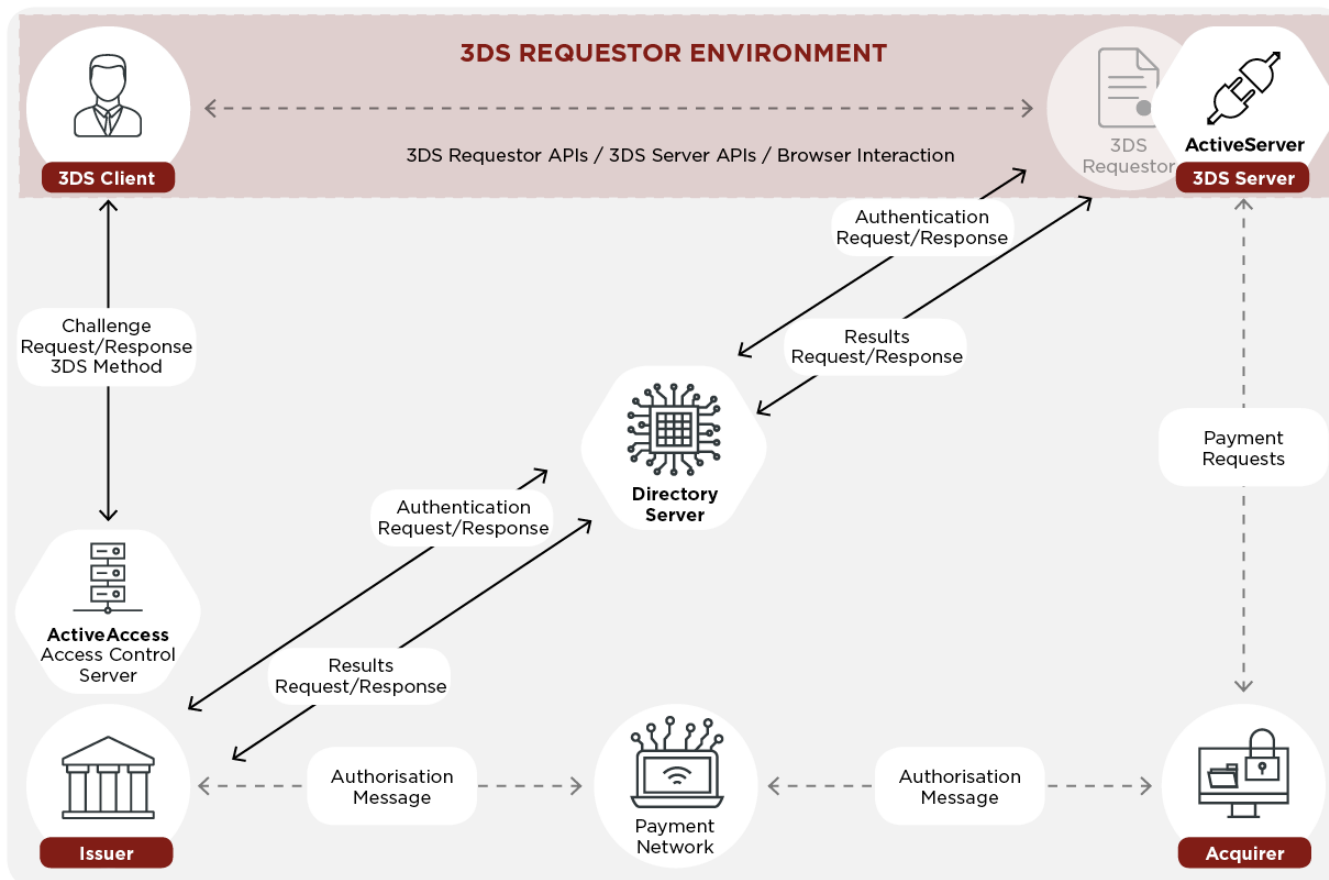
3DSサーバーとACSは自動的にブラウザー情報を収集します。この処理の概要は**3DSリクエスター**には含まれません。

処理 2: 認証の実行

ブラウザ情報の収集が完了すると、加盟店は `/api/v2/auth/brw` を呼び出して認証を実行できます。この処理が実行されると、**ActiveServer**が3DS2メッセージング処理を開始します。3DS2には2つの主要な認証フローフリクションレス・フローとチャレンジ・フローがあります。

- ・ **フリクションレス・フロー**—**AReq/ARes**認証メッセージからなる3Dセキュア認証フローを開始します。与えられた情報から取引が低リスクであるとACSが判断した場合は、直ちに認証が承認されます。
- ・ **チャレンジ・フロー**—取引が特定の許容限界値より高リスクであるとACSが判断した場合、または法律によってチャレンジが必須である場合は、カード会員がさらに操作を行うことが必要な、**フリクションレス・フロー**が**チャレンジ・フロー**に切り替わります。**チャレンジ・フロー**はフリクションレス・フローでもあった**AReq/ARes**メッセージ、**CReq/CRes**チャレンジメッセージと**RReq/RRes**結果メッセージから構成されます。

チャレンジ・フローを次の図に示します。



点線は、クライアント/3DSリクエスターと信用承認機能の間の通信など、3DS2プロトコルの範囲外のメッセージングを示します。

処理 3: 認証結果の取得

3DS2処理が完了すると、加盟店は認証結果を取得します。認証結果（チャレンジのステータスによりAResまたはRRes）には、ECI、認証値（CAVVなど）、および3DSリクエスターへの最終取引ステータスなどの情報が含まれています。

次のチャプター

次を選択し、**認証シーケンス**の詳細をご覧ください。

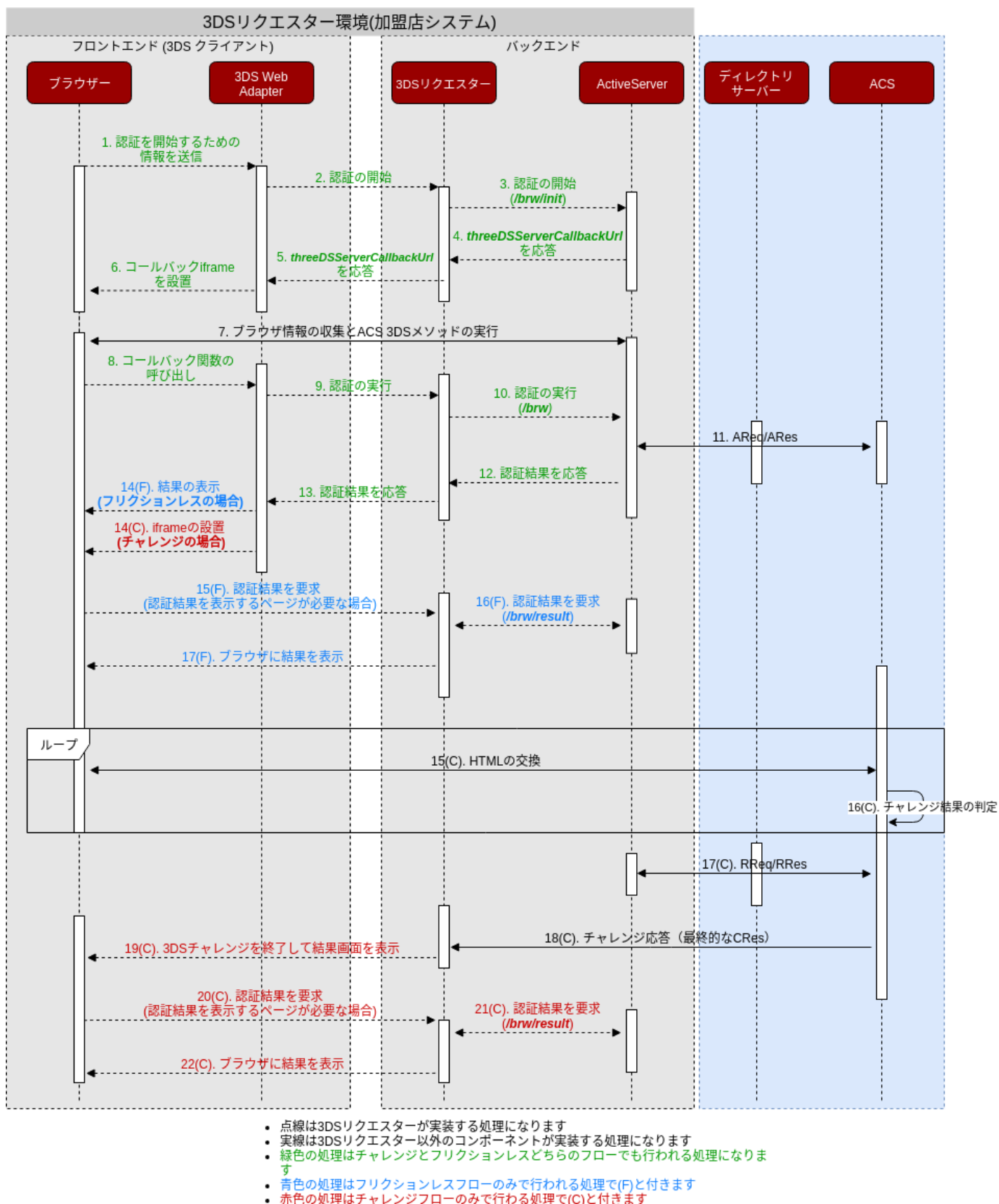
認証シーケンス

次のシーケンス図は、3DS2認証処理について、特に3DS2フローにおけるGPaymentsの APIを使用した3DSリクエスターの役割に焦点を当てて各ステップを段階を追って示したものです。

備考

ActiveServerを統合するには、フロントエンドに3DS web adapterを、バックエンドに3DSリクエスターを実装する必要があります。

- **処理 1: 認証の初期化**
 - ステップ1～ステップ7
- **処理 2: 認証の実行**
 - フリクションレス・フロー: ステップ8～ステップ13、およびステップ14(F)
 - チャレンジ・フロー: ステップ8～ステップ13、およびステップ14(C)～ステップ19(C)
- **処理 3: 認証結果の取得**
 - フリクションレス・フロー: ステップ15(F)～ステップ17(F)
 - チャレンジ・フロー: ステップ20(C)～ステップ22(C)



1. 認証の初期化用の情報を送信 ←3DSリクエスターの処理

- カード番号やカード会員の氏名など、決済ページで得られたカード会員情報が、3DS web adapterに送信されます。これは加盟店のフロントエンドシステムをシミュレートした簡潔なJavaScriptのコードです。

2. 認証の初期化 ←3DSリクエストの処理

- **3DS web adapter**は決済ページから収集した情報を使用して**3DSリクエスト**へのPOSTリクエストを行い、**3DSリクエスト**に認証の初期化を要求します。

3. 認証の初期化 (`initAuth`) ←3DSリクエストの処理

- **3DSリクエスト**はフロントエンドから情報を取得し、`initAuth` へのPOST API呼び出しを行って認証を初期化します。
- ここで送信される重要なフィールドは `eventCallbackUrl` であり、**ActiveServer**がこのURLへのコールバックを行ってブラウザの情報収集完了を通知できるようステップ8を開始するために必要です。

4. `threeDSserverCallbackUrl` を応答

- `initAuth` からの正常な応答には `threeDSserverCallbackUrl` と `threeDSserverTransID` が含まれています。

5. `threeDSserverCallbackUrl` を応答 ←3DSリクエストの処理

- **3DSリクエスト**は**3DS web adapter**に `threeDSserverCallbackUrl` を返します。

6. コールバック `iframe` を設置 ←3DSリクエストの処理

- `src` 属性を `threeDSserverCallbackUrl` に設定した非表示の `iframe` を挿入します。これにより、**ActiveServer**は**3DSリクエスト**に接続できる状態になります。**ActiveServer**は、この `iframe` へのコールバックを行います。

7. ブラウザ情報の収集

- **ActiveServer**は `iframe` を使用してブラウザ情報を収集し、**ACS**が**3DSメソッド**データを収集できるようにします。次に、**ACS**は、用意された `iframe` を使用して**3DSメソッド**データを収集します。

8. コールバック関数の呼び出し ←3DSリクエストの処理

- 認証の初期化中、**3DSメソッド**が終了またはスキップされたときに**ACS**が**3DSリクエスト**に通知できるよう、**3DSリクエスト**は `eventCallbackUrl` を送信します。ステップ7で設置した `iframe` からPOST要求がこの `eventCallbackUrl` に行われます。
- **3DSリクエスト**は、この要求を受け `iframe` 内に必要な `callbackFn` 含めたパラメータと一緒に `notify-3ds-events.html` を表示します。 `notify_3ds_events.html` は表示後 `3ds-web-adater` に定義された `callbackFn` を呼び出します。

9. 認証の実行 ←3DSリクエスターの処理

- `callbackFn` は `_on3DSMethodSkipped()` または `_on3DSMethodFinished()` のいずれかであり、どちらも `doAuth()` を呼び出します。**3DS web adapter**は `doAuth()` を呼び出し、認証を実行するよう3DSリクエスターに要求します。
- `_on3DSMethodSkipped` はブラウザの情報がなんらかの理由によってACSが取得できなかったことを意味します。なので、もしこのコールバック関数が呼ばれた場合加盟店は認証を続行しない選択をすることもできます。

10. 認証の実行 (auth)

- **3DSリクエスター**は `auth` を呼び出して認証処理を開始します。

11. AReq/ARes

- 認証リクエスト (AReq) は、ディレクトリ・サーバーを介して**ActiveServer**からACSに送信されます。ACSからは、認証結果を含む認証応答 (ARes) が**ActiveServer**に送信されます。

12. 認証結果を応答

- `auth` は、**3DSリクエスター**に `tranStatus` を返します。

13. 認証結果を応答 ←3DSリクエスターの処理

- 認証結果を3DS webアダプターに返します。

Info

返された `transStatus` が"Y"の場合は **ステップ 14(F)** に、"C"の場合は **ステップ 14(C)** に進んでください。

フリクシオンレス・フローの場合

14(F). 結果の表示 (フリクシオンレスの場合) ←3DSリクエスターの処理

- 認証結果の `transStatus` が"Y"の場合は `authSuccess()` が呼び出され、ページを `/auth/result?transId` にリダイレクトします。

15(F). 認証結果を要求 (認証結果を表示するページが必要な場合) ←3DSリクエスターの処理

- ブラウザーが `transId` で**3DSリクエスター**に通知し、取引結果がリクエストに使用できる状態になります。

16(F). 認証結果を要求(`result`)

- ・ 3DSリクエスターは `result` を呼び出し、ActiveServerから結果を受信するように要求します。

17(F). ブラウザに結果を表示

- ・ 結果画面が開き、認証結果が表示されます。

チャレンジフローの場合

14(C). `iframe` の設置 (チャレンジの場合)。

- ・ 認証結果の `transStatus` が"C"である場合は `startChallenge()` が呼び出され、チャレンジ用の `iframe` を挿入します。

15(C). HTMLの交換

- ・ ACSは `iframe` 内にチャレンジ画面を埋め込み、カード会員は認証チャレンジを実行します。

16(C). チャレンジ結果の判定

- ・ ACSは、実行されたチャレンジが成功したか否かを判定します。

17(C). RReq/RRes

- ・ ACSは、ディレクトリ・サーバーを介してActiveServerに、認証結果を含む結果リクエスト (RReq) を送信します。ActiveServerは、結果応答 (RRes) を使用して受信確認通知を送信します。

18(C). チャレンジ応答 (最終的なCRes)

- ・ ACSは、最終的なチャレンジ応答 (CRes) を3DSリクエスターに送信します。

19(C). 3DSチャレンジを終了して結果画面を表示 ←3DSリクエスターの処理

- ・ チャレンジが終了したため、3DSリクエスターはページを `/auth/result?transId` にリダイレクトします。

20(C). 認証結果を要求(認証結果を表示するページが必要な場合) ←3DSリクエストの処理


- ・ ブラウザーが transId で3DSリクエストに通知し、取引結果がリクエストに使用できる状態になります。

21(C). 認証結果を要求(result) ←3DSリクエストの処理

- ・ ステップ16(F)と同様に、3DSリクエストは result からの結果受信を要求します。

22(C). ブラウザに結果を表示 ←3DSリクエストの処理

- ・ ブラウザーで結果画面が開き、認証結果が表示されます。

 **次のチャプター**

下のフッターの**次**を選択して**実装ガイド**にアクセスし、**ActiveServer**を使用した加盟店の決済処理機能を組み込んでください。

認証APIv1からv2へのAPI移行

このページはActiveServer **API v1**から**API v2**へ移行するためのガイドです。ActiveServer API v1とはGPayments社が以前公開しましたActiveServerのAPI仕様です。

ActiveServerインハウス（オンプレミス）をご導入いただいているお客様に対して、**API v1**のサポート期限は2020年Q4（欧米カレンダー）までと予定されております。詳細については改めてお知らせいたします。なお、**ActiveServerホスティング・サービス（SaaS）**は**API v2**のみ対応しますので、ホスティング・サービスをご利用いただくお客様はAPI v2でご導入いただきます。

主な変更

- `/api/v2/auth/` 認証APIのエントリーポイントを追加しました。
- ActiveServerはカード番号（PAN）を保持しないようになりました。ActiveServerはPANの上6桁と下4桁のみ、プレーンテキストでデータベースに保存します。よって、万が一、サイバー攻撃に遭う際の情報漏洩を最小限に抑えることができます。また、PCI審査の要件項目を減らすことも期待されております。なお、管理UI上、取引PANの検索機能は上6桁と下4桁のみマッチングするようになります。
- 暗号化PANは保存されないため、**API v2**を使用する場合は加盟店ごとの暗号化キーは不要になります。**API v1**においては引き続き必須です。
- 弊社の3DSリクエスター・サンプル・コードは、**API v1**ならびに**API v2**へ対応できるように更新されました。

ActiveServer変更

ActiveServerバージョンv1.3.0以降、API v2を通じて認証を実行しますと、ActiveServerは暗号化されたカード番号（PAN）を保存しないようになります。代わりに、トランケート形式（PANを部分的に切り捨てる形式）で、暗号化せずにプレーンテキストでデータベースに保存します。PANの上6桁と下4桁のみが保存され、全桁のPANは復元できなくなります。例えば、**4123456789876543** のPANは **412345XXXXXX6543** のようにデータベースに保存されます。また、管理UI上も同様、**412345XXXXXX6543** と表示されます。

その結果、加盟店ごとの暗号化キーは不要となり、将来、この機能は削除される**予定**です。

なお、全桁のPANを保持しなくなるため、PAN全桁での取引検索はできなくなります。ですが、PANの残り桁での検索することは可能です。その場合、検索したいPANの全桁を検索欄に入力しますと、検索結果は上6桁と下4桁のみ一致する取引が返されます。仮にそれ以外の桁が違ってても、その取引は検索結果に返されます。その場合、他の絞り込み条件（Transaction ID、Date/Time、Purchase amount、Currency、など）を使用し、お探しの取引を特定することを推奨します。API v2で実行された取引は引き続きPAN全桁での検索が可能です。

上記の変更により、API v2をご導入いただきますと以下のメリットが期待されております：

- ・ パフォーマンスの向上：全取引にPANを暗号化する処理が省かれる
- ・ セキュリティーの向上：万が一、サイバー攻撃に遭う際の情報漏洩を最小限にできる
- ・ PCI受審期間の短縮化、または受審コストの削減：PCI受審時のカード会員データ環境(英：Cardholder Data Environment)のスコープを縮小できる

Important

上記の変更はAPI v2を使用する場合のみ適応されます。API v1に関する変更は下記をお読みください。

API v1への影響、およびAPI v2へのマイグレーション計画

API v1によって実行された取引は上記の変更に影響されません。現在統合済みのAPI v1システムは引き続き稼働可能です。その間、API v2へのマイグレーションの計画をご検討いただくようお願い申し上げます。API v1のサポート期限は2020年Q4（欧米カレンダー）までと予定されております。

以前API v1で実行された取引認証は引き続き、全桁のPANを暗号化し、加盟店ごとの暗号化キーと共にデータベースに保存されます。また、API v2へのマイグレーションをスムーズに行えるよう、弊社はAPI v2マイグレーション・ツールをリリースする予定です。このツールは以下の機能が予定されております：

- ・ API v1によって実行された取引をAPI v2形式へ変換します。
- ・ 暗号化キーストアをクリアします。
- ・ API v1を無効化します。

弊社はお客様になるべく早くAPI v2へ移行することを推奨します。

APIの変更 - BRW認証API v2へのマイグレーション

API v2のBRW APIに関する主な変更は、カード会員情報は `/api/v2/auth/brw` (`Execute Authentication`)のAPIによって送信されるようになります。API v1においては、カード会員情報は `/api/v2/auth/brw/init` (`Initialise Authentication`)のAPIによって送信されていました。

API v2において、`/api/v2/auth/brw/init` (`Initialise Authentication`)は単に認証前のブラウザ情報収集、および3DS Method (ACSから追加のブラウザ情報収集処理)の実行 (適用される場合のみ)を行います。

下記はリクエストのJSONメッセージ変更：

- ・ **赤ハイライト** - このフィールドは削除されました。
- ・ **緑ハイライト** - このフィールドは追加されました。
- ・ **ハイライトなし** - このフィールドはAPI v1と同様です。

フィールド変更に関する詳細は[認証APIドキュメント](#)をご参照ください。

Initialise Authentication の変更

- ・ `{messageCategory}` パラメーターはリクエストメッセージURLから `Execute Authentication` (`/api/v2/auth/brw`) リクエストの本文へ移行されました。
- ・ 下記、削除された取引情報フィールドは `Execute authentication` のAPIへ移行されました。
- ・ `authUrl` フィールドはレスポンス・メッセージに追加され、`Execute authentication` リクエストの送信先となります。
- ・ エラーコードのフィールドは成功した取引認証のレスポンス・メッセージから削除されました。

リクエスト本文

以下は `/api/v1/auth/brw/init/{messageCategory}` と `/api/v2/auth/brw/init` のリクエスト本文の相違点を記します。

```
{
- "acctID": "ActiveServer 3DS Test Account 000000001",
- "acctInfo": {
-   "chAccAgeInd": "03",
-   "chAccChange": "20160712",
-   "chAccChangeInd": "04",
-   "chAccDate": "20140328",
-   "chAccPwChange": "20170328",
-   "chAccPwChangeInd": "02",
-   "nbPurchaseAccount": "11",
-   "paymentAccAge": "20160917",
-   "paymentAccInd": "04",
-   "provisionAttemptsDay": "3",
-   "shipAddressUsage": "20160714",
-   "shipAddressUsageInd": "04",
-   "shipNameIndicator": "02",
-   "suspiciousAccActivity": "01",
-   "txnActivityDay": "1",
-   "txnActivityYear": "21"
- },
  "acctNumber": "7654310438720050",
- "acctType": "03",
- "authenticationInd": "01",
- "authenticationInfo": {
-   "threeDSReqAuthData": "validlogin at UL TS BV",
-   "threeDSReqAuthMethod": "02",
-   "threeDSReqAuthTimestamp": "201711071307"
- },
- "cardExpiryDate": 1910,
- "cardHolderInfo": {
-   "billAddrCity": "Bill City Name",
-   "billAddrCountry": 840,
-   "billAddrLine1": "Bill Address Line 1",
-   "billAddrLine2": "Bill Address Line 2",
-   "billAddrLine3": "Bill Address Line 3",
-   "billAddrPostCode": "Bill Post Code",
-   "billAddrState": "C0",
-   "cardholderName": "Cardholder Name",
-   "email": "example@example.com",
-   "homePhone": {
-     "cc": "123",
-     "subscriber": "123456789"
-   },
-   "mobilePhone": {
-     "cc": "123",
-     "subscriber": "123456789"
-   },
-   "shipAddrCity": "Ship City Name",
-   "shipAddrCountry": "840",
-   "shipAddrLine1": "Ship Address Line 1",
```

```
- "shipAddrLine2": "Ship Address Line 2",
- "shipAddrLine3": "Ship Address Line 3",
- "shipAddrPostCode": "Ship Post Code",
- "shipAddrState": "C0",
- "workPhone": {
-   "cc": "123",
-   "subscriber": "123456789"
- },
- },
- "challengeInd": "02",
  "eventCallbackUrl": "https://example.requestor.com/3ds-notify",
  "merchantId": "1234567890123456789012345678901234",
- "merchantName": "string",
- "merchantRiskIndicator": {
-   "deliveryEmailAddress": "deliver@email.com",
-   "deliveryTimeframe": "01",
-   "giftCardAmount": "337",
-   "giftCardCount": "02",
-   "giftCardCurr": "840",
-   "preOrderDate": "20170519",
-   "preOrderPurchaseInd": "02",
-   "reorderItemsInd": "01",
-   "shipIndicator": "02"
- },
- "payTokenInd": true,
- "priorTransID": "59ae264e-b0f4-43c7-870e-4d14bd52806e",
- "purchaseAmount": "12345",
- "purchaseCurrency": "978",
- "purchaseDate": "20180122153045",
- "purchaseInstalData": "024",
- "recurringExpiry": "20180131",
- "recurringFrequency": "2",
  "threeDSRequestorTransID": "2409a5df-b777-4ebc-ad59-2a61091187f1",
- "transType": "03"
}
```

レスポンス

以下は `/api/v1/auth/brw/init/{messageCategory}` と `/api/v2/auth/brw/init` のレスポンス本文の相違点を記します。

```
{
- "errorCode": "1005",
- "errorComponent": "A",
- "errorDescription": "Data element not in the required format. Not numeric or
wrong length.",
- "errorDetail": "billAddrCountry,billAddrPostCode,dsURL",
- "errorMessageType": "AReq",
+ "authUrl": "https://demo.3dsserver.com/api/v2/auth/brw?
t=dbb270aa890028817afe58fda659526e",
  "monUrl": "https://demo.3dsserver.com/brw/init/mon?
t=6afa6072-9412-446b-9673-2f98b3ee98a2",
  "threeDSSTServerCallbackUrl": "https://demo.3dsserver.com/brw/callback?
transId=6afa6072-9412-446b-9673-2f98b3ee98a2",
  "threeDSSTServerTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2"
}
```

Important

API v2において、Execute Authentication 処理はActiveServerが作成した動的（ダイナミック）URL（authUrl）を使用することが必須となります。3DSリクエスターはこのURLを使用します。以前API v1のみ対応したデモ用3DSリクエスター・サンプル・コードはハードコードされたURLがありますが、そのURLは使用しません。

Execute Authentication の変更

- Execute Authentication (`/api/v2/auth/brw`) のエントリーポイントは、Initialise authentication レスポンス内の authUrl へ変更されました。
- Initialise Authentication から削除されたカード会員情報と取引情報のパラメーターはリクエスト本文へ移行されました。
- threeDSRequestorTransID と threeDSSTServerTransID のフィールドは不要になったため、削除されました。
- acctNumber フィールドが追加されました。

- `browserInfo` フィールドが追加されました。このフィールドはブラウザ情報収集処理によって収集された情報です。詳細は[3DS Requestor changes](#)をご参照ください。
- **API v1**にありました `Initialise Authentication {messageCategory}` パラメーターは**API v1** URLから削除され、レスポンス本文へ移行されました。
- `addrMatch` オプション・フィールドは `cardHolderInfo` JSONオブジェクトに追加されました。請求先住所はお届け先住所と同じの場合は `Y` を設定、同じでない場合は `N` を設定します。請求先住所とお届け先住所は必須情報です。
- `threeDSRequestorTransID` フィールドは不要になったため、削除されました。
- エラーコードのフィールドは成功した取引認証のレスポンス・メッセージから削除されました。

リクエスト本文

以下は `/api/v1/auth/brw` と `/api/v2/auth/brw` のリクエスト本文の相違点を記します。

```
{
+ "acctID": "ActiveServer 3DS Test Account 000000001",
+ "acctInfo": {
+   "chAccAgeInd": "03",
+   "chAccChange": "20160712",
+   "chAccChangeInd": "04",
+   "chAccDate": "20140328",
+   "chAccPwChange": "20170328",
+   "chAccPwChangeInd": "02",
+   "nbPurchaseAccount": "11",
+   "paymentAccAge": "20160917",
+   "paymentAccInd": "04",
+   "provisionAttemptsDay": "3",
+   "shipAddressUsage": "20160714",
+   "shipAddressUsageInd": "04",
+   "shipNameIndicator": "02",
+   "suspiciousAccActivity": "01",
+   "txnActivityDay": "1",
+   "txnActivityYear": "21"
+ },
+ "acctNumber": "7654310438720050",
+ "acctType": "03",
+ "authenticationInd": "01",
+ "authenticationInfo": {
+   "threeDSReqAuthData": "validlogin at UL TS BV",
+   "threeDSReqAuthMethod": "02",
+   "threeDSReqAuthTimestamp": "201711071307"
+ },
+ "browserInfo": "Base64 encoded browser information", --New field added from
V2
+ "cardExpiryDate": "1910",
+ "cardHolderInfo": {
+   "addrMatch": "N",
+   "billAddrCity": "Bill City Name",
+   "billAddrCountry": 840,
+   "billAddrLine1": "Bill Address Line 1",
+   "billAddrLine2": "Bill Address Line 2",
+   "billAddrLine3": "Bill Address Line 3",
+   "billAddrPostCode": "Bill Post Code",
+   "billAddrState": "C0",
+   "cardholderName": "Cardholder Name",
+   "email": "example@example.com",
+   "homePhone": {
+     "cc": "123",
+     "subscriber": "123456789"
+   },
+   "mobilePhone": {
+     "cc": "123",
+     "subscriber": "123456789"
+   },
+ }
```

```
+ "shipAddrCity": "Sh"ip" City Name",
+ "shipAddrCountry": 840,
+ "shipAddrLine1": "Ship Address Line 1",
+ "shipAddrLine2": "Ship Address Line 2",
+ "shipAddrLine3": "Ship Address Line 3",
+ "shipAddrPostCode": "Ship Post Code",
+ "shipAddrState": "C0",
+ "workPhone": {
+   "cc": "123",
+   "subscriber": "123456789"
+ }
+ },
+ "challengeInd": "02",
+ "merchantName": "Test Merchant",
+ "merchantRiskIndicator": {
+   "deliveryEmailAddress": "deliver@email.com",
+   "deliveryTimeframe": "01",
+   "giftCardAmount": "337",
+   "giftCardCount": "02",
+   "giftCardCurr": "840",
+   "preOrderDate": "20170519",
+   "preOrderPurchaseInd": "02",
+   "reorderItemsInd": "01",
+   "shipIndicator": "02"
+ },
+ "messageCategory": "pa",
+ "payTokenInd": true,
+ "priorTransID": "59ae264e-b0f4-43c7-870e-4d14bd52806e",
+ "purchaseAmount": "12345",
+ "purchaseCurrency": "978",
+ "purchaseDate": "20180122153045",
+ "purchaseInstalData": "024",
+ "recurringExpiry": "20180131",
+ "recurringFrequency": "2",
- "threeDSRequestorTransID": "2409a5df-b777-4ebc-ad59-2a61091187f1",
  "threeDSServerTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2",
+ "transType": "03"
}
```

レスポンス

以下は `/api/v1/auth/brw` と `/api/v2/auth/brw` のレスポンス本文の相違点を記します。

```
{
  "acsChallengeMandated": "Y",
  "acsReferenceNumber": "3DS_GP_ACS_201_13579",
  "acsTransID": "375d90ad-3873-498b-9133-380cbbc8d99d",
  "authenticationType": "02",
  "authenticationValue": "MTIzNDU2Nzg5MDA5ODc2NTQzMjE=",
  "cardholderInfo": "For example, Additional authentication is needed for this
transaction, please contact (Issuer Name) at xxx-xxx-xxxx. \n Length: Maximum
128 characters\n Optional",
  "challengeUrl": "https://demo.acs.com/challenge",
  "dsReferenceNumber": "string",
  "dsTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2",
  "eci": "02",
  - "errorCode": "1005",
  - "errorComponent": "A",
  - "errorDescription": "Data element not in the required format. Not numeric or
wrong length.",
  - "errorDetail": "billAddrCountry,billAddrPostCode,dsURL ",
  - "errorMessageType": "AReq",
  "messageVersion": "string",
  "threeDSServerTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2",
  "transStatus": "Y",
  "transStatusReason": 11
}
```

Result の変更

- `authenticationType` と `interactionCounter` フィールドを追加しました。これはカード会員へのチャレンジに関して情報をもっと提供するためです。
- `acsReferenceNumber` と `dsReferenceNumber` フィールドは削除されました。これは `Execute Authentication` 処理にすでに提供されているからです。
- エラーコードのフィールドは成功した取引認証のレスポンス・メッセージから削除されました。

リクエスト

リクエスト本文への変更はありません。

レスポンス

以下は `/api/v1/auth/brw/result` と `/api/v2/auth/brw/result` のレスポンス本文の相違点を記します。

```
{
- "acsReferenceNumber": "3DS_GP_ACS_201_13579",
  "acsTransID": "375d90ad-3873-498b-9133-380cbbc8d99d",
+ "authenticationType": "02",
  "authenticationValue": "MTIzNDU2Nzg5MDA5ODc2NTQzMjE=",
- "dsReferenceNumber": "string",
  "dsTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2",
  "eci": "02",
+ "interactionCounter": "02",
- "errorCode": "1005",
- "errorComponent": "A",
- "errorDescription": "Data element not in the required format. Not numeric or
wrong length.",
- "errorDetail": "billAddrCountry,billAddrPostCode,dsURL",
- "errorMessageType": "AReq",
  "messageVersion": "string",
  "threeDSServerTransID": "6afa6072-9412-446b-9673-2f98b3ee98a2",
  "transStatus": "Y",
  "transStatusReason": 11
}
```

APIの変更 - APP認証API v2へのマイグレーション

- `/api/v1/auth/app/{messageCategory}` のパスは `/api/v2/auth/app` へ変更されました。
`{messageCategory}` フィールドはURLからリクエスト本文へ移行されました。
- `addrMatch` オプション・フィールドは `cardHolderInfo` JSONオブジェクトに追加されました。請求先住所はお届け先住所と同じの場合は `Y` を設定、同じでない場合は `N` を設定します。請求先住所とお届け先住所は必須情報です。
- `interactionCounter` フィールドは `/api/v2/auth/app/result` のレスポンスへ追加されました。この値はカード会員が試した認証回数を示すため、ACSから返される場合があります。

APIの変更 - 3RI認証API v2へのマイグレーション

- `/api/v1/auth/3ri/{messageCategory}` のパスは `/api/v2/auth/3ri` へ変更されました。
`{messageCategory}` フィールドはURLからリクエスト本文へ移行されました。

3DSリクエストの変更

- 3DSリクエストのバックエンド・コードは、返された `authURL` をWebサーバー・セッションの `initAuth` レスポンスに保存します。後に、`authURL` はBRW認証チャンネルの `Execute Authentication` 処理に使われます。
- 3DSリクエストは `eventCallbackUrl` にて `3DSMethodFinished` もしくは `3DSMethodSkipped` のeventを受信する際、`param` パラメーターの中にはActiveServerが収集したブラウザー情報がBase64エンコード方式で含まれています。そのため、弊社の3DSリクエスト・サンプル・コードには新しいAPI v2対応の `3ds-web-adapter` が追加されました。3DSリクエストは、`Execute Authentication` リクエストにある `browserInfo` フィールド内に、上記のBase64エンコードされたブラウザー情報を送信します。
- 3DSリクエストはPANを二回ActiveServerへ送信します。`/api/v2/auth/brw/init` と `/api/v2/auth/brw` のAPIコールを通じてPANを送信します。これは、**ActiveServer**は暗号化されたPANを保存しなくなったためです。

序章

統合ガイドのドキュメントは、サンプルの加盟店サイトを用いて、3DS2認証フローを統合するプロセスを一から説明します。これは**3DSリクエスター**と**3DSクライアント**を、**3DSリクエスター環境**と呼ばれる加盟店チェックアウトプロセスに、EMV 3Dセキュア2.0の仕様を実装するためのものです。

この統合ガイドは、提供された3DSリクエスターデモのプロジェクトに基づいており、二つのパートで構成されています:

- **HTML/JavaScript** のフロントエンド - デモのオンラインショップと、BRW、3RI、EnrolのAPI呼び出しのためのテストページ
- バックエンド - フロントエンドページとリソースのホスト。また、ページ転送の実行とx509 認証による**ActiveServer**認証APIの呼び出しを行います。

フロントエンドWebページとJavaScriptの実装は以下のフレームワーク、コンポーネントを使用します。

- [Bootstrap 4.1.3](#)
- [Font Awesome 5.2.0](#)
- [Moment.js](#)
- [jQuery 3.3.1](#)

前提条件

このガイドを使用するための前提条件は以下の通りです:

- Webフロントエンドの開発知識 (HTML、CSS、JavaScript)
- Gitクライアント
- アクティブ化され、実行中の**ActiveServer**インスタンス
- Java、PHP もしくは C#のプログラミングの重要知識

サンプルコードのチェックアウト

3DSリクエスターデモのコードはGitHubにホストされており、以下のレポジトリから複製する事が出来ます:

<https://github.com/gpayments/gp-3ds-requestor-demo.git>

サンプルコードをチェックアウトするには、以下のコマンドをローカル環境で実行してください:

```
git clone https://github.com/gpayments/gp-3ds-requestor-demo.git
```

レポジトリが複製されると、このチュートリアル全ての必要なデモコードが以下のディレクトリ内に見つかります:

```
$ cd gp-3ds-requestor-demo
gp-3ds-requestor-demo $ ls
dotnet go java php README.md
```

GPaymentsは現在**Java**、**PHP**、**C#**、**Go**向けのバックエンドのサンプルコードを提供しています。統合ガイドを通して、バックエンドの言語間で説明が異なる場所では、その言語のタブをクリックすることで言語別の説明が表示されます。

すべてのバックエンドのサンプルは、**JavaScript**を使用して書かれテンプレート化に**Mustache**を利用した実装方法になっています。

注釈

簡潔化とデモの目的の為、3DSリクエスターデモのコードは、Webフロントエンドのコード内に大半のプロセスとページのコールバックのロジックが実装されています。バックエンドのコードは単純なサーバー側のページ転送と**ActiveServer**認証APIに接続する為のx509認証クライアントになっています。実際の実装では、異なるフロントエンドとバックエンドのコード構造をもった、あなたの既存のチェックアウトプロセスとワークフローに合わせる為に、自分自身の3DSリクエスターコードをデザインし、実装することが可能です。

サーバー側の依存性

サーバー側の3DSリクエスターデモのコードは、言語により異なりますが、以下の依存性とライブラリをデモの3DSリクエスターを実行する前にインストールする必要があります:

言語	依存性とツール	注釈
Java	JDK 1.8 Apache Maven - https://maven.apache.org/install.html	
C#	ASP.NETとWeb開発ワークロードを含めたVisual Studio 2013またはそれ以降 Nugetをインストール。参照： https://docs.microsoft.com/en-us/nuget/install-nuget-client-tools#visual-studio	Windows system required
PHP	PHP 7.2とcURL (Client URL Library) Composer - https://getcomposer.org Guzzle - http://docs.guzzlephp.org OpenSSL	
Go	gin-gonicとGo 1.12	Go 1.12をインストール https://golang.org/

クライアント証明書の取得

バックエンドが**ActiveServer**認証APIを呼び出し自身を認証する前に、クライアント証明書をセットアップする必要があります。このクライアント証明書により、バックエンドは**ActiveServer**で相互に認証されたTLS接続をセットアップする事が出来ます。

クライアント証明書を取得するには、[こちら](#)を参照してください。ActiveServerインスタンスへのアクセスを持っておらず、**GPayments TestLabs**を使用する場合は、techsupport@gpayments.com宛てにメールを送り、証明書を取得してください。

クライアント証明書キーストアファイルの用意が出来たら、好みのディレクトリにそれをコピーし、そのファイルパスをバックエンドにて指定してください。

証明書は3DSリクエスターデモプロジェクトディレクトリ内にコピーする必要はありませんが、クライアント証明書をプロジェクト内に保存する事で、管理しやすくなります。

プロジェクトのディレクトリ構造を確認するには、[ディレクトリ構造](#)で詳細を参照してください。

3DSリクエスターデモの設定

3DSリクエスターデモを実行するために、システムを起動する前に以下の設定を行う必要があります:

1. 3DSリクエスターは、**ActiveServer**に自身のエントリーポイントをコールバックURLとして送信する必要があります。さらに、認証APIを実行するために**ActiveServer**の認証API URLも設定する必要があります。

AsAuthUrlと**BaseUrl**を[バックエンドリソースファイル](#)で設定してください。これらの値はアプリケーションが起動され再起動しないと変更出来ない時に時に使用されます:

- **AsAuthUrl** - **ActiveServer**インスタンスのAPI URLを設定します。GPayments TestLabsを利用する場合はデフォルトの値をお使いください。
- **BaseUrl** - アプリケーションが起動されアクセスされるURLになります。

Java C# PHP Go

```
//application.yml
server:
  port: 8082
  ...
gpayments:
  asAuthUrl: https://api.as.testlab.3dsecure.cloud:7443
  baseUrl: http://localhost:8082
  certFileName: # Client Certificate file (.p12 or .pfx) path
  groupAuth: false
  merchantToken: # groupAuth = trueの場合のmerchantToken
```

注釈

もしサンプル内で使用される標準のポートが使用されている場合はエラーが発生する事があります。その場合は `port` と `baseUrl` を適切な値に設定してください。

2. バックエンドリソースファイルの `certFileName` をダウンロードした証明書に設定します。

- **加盟店クライアント証明書**を利用する場合:
 - `certFileName` をクライアント証明書の完全なファイル名とパスに設定。
 - `groupAuth` を `false` に設定。
 - `merchantToken` を空白に設定。
- **認証APIマスタークライアント証明書**:
 - `certFileName` をクライアント証明書の完全なファイル名とパスに設定。
 - `groupAuth` を `true` に設定。
 - `merchantToken` を設定する。マーチャントトークンについては[こちら](#)を参照してください。

⚠ 警告

`groupAuth = true` の場合、バックエンドはHTTPリクエストのHTTPヘッダーに `AS-Merchant-Token` フィールドを追加し`merchantToken`の値を設定する必要があります。バックエンド実装の詳細を確認するには、[こちら](#)を参照してください。

Java C# PHP Go

```
//application.yml
server:
  port: 8082
  ...
gpayments:
  asAuthUrl: https://api.as.testlab.3dsecure.cloud:7443
  baseUrl: http://localhost:8082
  certFileName: # Client Certificate file (.p12 or .pfx) path
  groupAuth: false
  merchantToken: # groupAuth = trueの場合のmerchantToken
```

✎ 備考

クライアント証明書は任意の場所に保存することが出来ます。例えば、クライアント証明書を `C:/Downloads` ディレクトリ(Windowsの場合)に `client_certificate.p12` のファイル名で保存した場合は、`certFileName` を `C:/Downloads/client_certificate.p12` に変更してください。

3. クライアント証明書のパスワードを設定して下さい。[バックエンドリソースファイル](#)を参照してください:

Java C# PHP Go

```
//RestClientConfig.java
private static final String KEYSTORE_PASSWORD = "123456";
private static final String KEY_ENTRY_PASSWORD = "123456";
private static final String CA_CERTS_FILE_NAME = "certs/cacerts.pem";
```

4. **Terminal**(Linux)または**コマンドプロンプト**(Windows)を開き以下のコマンドラインをプロジェクトのルートディレクトリで実行して下さい。これは初回実行時にはすべての必要な依存性をダウンロードする為に数分かかる事があります。

Java C# PHP Go

```
$ cd java
$ mvn spring-boot:run
```

Javaネットワークアクセスによるファイアウォール警告

WindowsではJavaのネットワークアクセスによりファイアウォールセキュリティアラートが発生する可能性があります。その場合は、アクセスを許可して続けてください。

3DSリクエストWebサイトを閲覧する

クライアント証明書とURLが適切に設定され、スタートアップコマンドが実行されると、3DSリクエストデモが正常に起動されるはずです。3DSリクエストページは<http://localhost:8082>(もしくは上記で指定したBaseURL)にアクセスする事で閲覧する事が出来ます。

The screenshot shows the homepage of the 3DS Requestor website. At the top, there is a navigation bar with '3DS Requestor', 'Online shop', and 'Test pages' on the left, and 'API Document' on the right. The main heading is '3DS Requestor' in a large blue font. Below the heading, a welcome message reads: 'Welcome to GPayments sample 3DS Requestor website. Check the online shop to see a sample merchant integration, or use the test pages to test ActiveServer's authentication APIs.' There are two main sections: 'Online shop' with a 'Launch' button and 'Test pages' with buttons for 'BRW test', '3RI test', and 'Enrol test'. A copyright notice '© GPayments Ltd. 2019' is at the bottom.

ショップページを開くには**Online Shop**内の**Launch**ボタンを選択してください。いくつか商品をカートに追加し、デフォルトのカード会員の情報を使用してチェックアウトしてみてください。

The screenshot shows the online shop interface. It features three product cards: 'Apple' (Freshly picked Cavendish Apple, Delivery next day, Price \$2.00, Quantity 1), 'Pineapple' (Freshly picked Cavendish Pineapple, Delivery next day, Price \$5.00, Quantity 1), and 'Banana' (Freshly picked Cavendish Banana, Delivery next day, Price \$3.00, Quantity 1). Each card has an 'Add to cart' button. On the right, a 'Your cart' section shows 'Total \$3.00' and a single item 'Banana \$3.00 Quantity: 1' with a 'Continue to checkout' button.

すると3DSリクエスターデモはチェックアウトページを表示します:

3DS Requestor Online shop Test pages ▾ API Document

Cardholder Information

Payment

Name on card	Test Card
Card Number	4100000000000100
Expiry Date (YYMM)	2508
Currency	036

Billing details


Address Line 1	Unit 1
Address Line 2	123 Street
Address Line 3	123 Suburb
City	Sydney
State	NSW
ZIP	2000
Country Code	036

Is this address also your shipping address?

Yes
 No

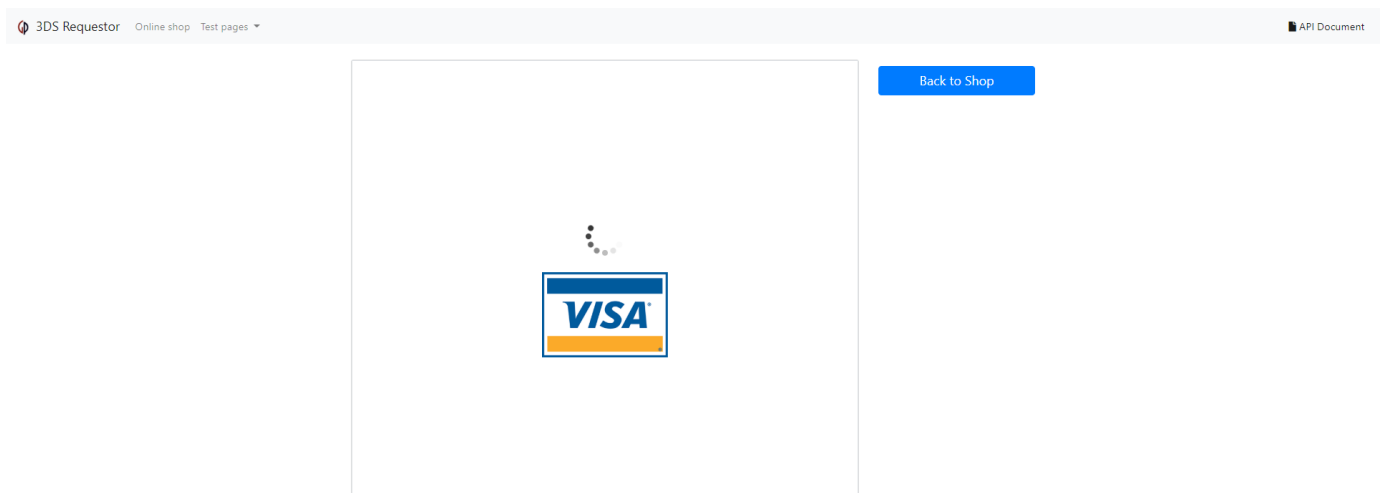
Your cart

Total \$3.00

 Banana
\$3.00 Quantity: 1

[Checkout \(v2\)](#)

Continue to Checkoutボタンで続行すると、3DSリクエスターデモは進捗画面を表示します:



そして最後に結果ページが表示されます(使用したカード番号により、セキュリティ確認画面が表示される事があります。詳細は[サンプルコード機能](#)を確認してください):

Test Results

[Back to Shop](#)

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAYVnCUQgAAAAAWcJAAAAAAAA=
transStatus	Y
threeDSServerTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

[Show results in separate page »](#)

おめでとうございます！初めての3DSリクエストのデモをローカル環境で正常に実行してテストしました。

警告

3DSリクエストがHTTPSではなくHTTP上で実行されています。これは単なるデモの目的の為に、本番環境ではHTTPSプロトコルを使用されるのが望ましいでしょう。

サポート

この文書についてご質問のある場合は、お気軽にお問合わせ下さい。
techsupport@gpayments.co.jpにEメールをお待ちしております。

次のチャプター

次へボタンを選択して、3DSリクエストの**フロントエンド実装**についてご覧ください。

フロントエンド実装(v1)

このセクションでは、[サンプルコード](#)を使用して加盟店アプリケーションのためのフロントエンドの実装方法を示します。

🔥 重要

GPayments社は今から実装を始める場合は認証APIV2を実装されることを強く推奨いたします。将来的に認証APIV1はサポートされなくなります。どのようにしてAPI V1からV2に移行するのかは[API V1からV2に移行ガイド](#)をご覧ください。認証APIV2のフロントエンドの実装をご覧になりたい場合は[こちら](#)をご覧ください。

サンプルコードパッケージ内の以下のファイルはフロントエンド内の3DS2認証に必須です。必要な場合は[ディレクトリツリー](#)で詳細を確認してください。

- `v1/process.html` - すべての[認証シーケンス](#)を実装しています。
- `v1/3ds-web-adapter.js` - 3DS2データをフロントエンドからバックエンドへ渡し、コールバックURLの為に必要な `iframe` を生成する**3DSクライアント**の重要なコンポーネントです。
- `notify_3ds_events.html` - **ActiveServer**の為に使用されるコールバックページで、認証において([ステップ 7](#)と[ステップ 18\(C\)](#))を開始する際に必要になります。このページはチェックアウト処理に認証イベントを渡します。

以下は[認証処理](#)と[認証シーケンス](#)に基づいたフロントエンド実装の詳細です。

処理1: 認証の初期化

認証を初期化する為に、フロントエンドが必要な事:

- **認証の初期化** メッセージを**3DSリクエスター**へ送信する ([ステップ 1](#)と[ステップ 2](#))。
- 応答メッセージを受信しコールバック `iframe` をセットアップする ([ステップ 5](#)と[ステップ 6](#))。

ユーザーがチェックアウトページ内の**チェックアウトを続ける**ボタンをクリックすると、ブラウザは必要なデータをセッションストレージに保存し `process.html` ページに移動します。

```
//checkout.html
function checkout() {
  var apiVersion = getApiVersion();
  switch (apiVersion) {
    case "v1":
      goApiV1();
      break;
    ...
  }
}

function goApiV1() {
  var sessionData = getSessionData();
  sessionData.messageCategory = "pa";
  sessionStorage.setItem("sessionData", JSON.stringify(sessionData));
  window.location.href = "/v1/process";
}

function getSessionData() {
  var sessionData = {};
  sessionData.channel = "brw";
  sessionData.authData = genAuthData();
  sessionData.backButtonType = "toShop";
  return sessionData;
}
```

注目

sessionData が含むもの:

- **channel** : brw または 3ri。上記の例では、ブラウザベースの認証を実行するために brw チャンネルを使用しました。
- **messageCategory** : pa -決済認証または npa -非決済認証のいずれか。上記の例では、pa を使用しました。
- **authData** : JSON形式のすべての必要なデータ。データ構造については、[APIドキュメント](#)を参照してください。
- **backButtonType** : プロセスページの back ボタンタイプを示します。上記の例では **戻る** ボタンを **ショップに戻る** として作成します。

注意点: ページ間通信に sessionStorage を使用するの、デモ用です。3DSリクエストの実装では、既存のチェックアウトプロセスと統合するために、ページ間でパラメーターを転送するための最適なアプローチを選択しましょう。

process.html ページ内で、sessionData 受信し3DS2処理を開始します。まず、認証を初期化する為の情報を3ds-web-adapterに送信します (ステップ 1)。

```
//process.html
var sessionData = JSON.parse(sessionStorage.getItem("sessionData"));
...
switch (sessionData.channel) {
  case "brw":
    var container = $('#iframeDiv');
    brw(sessionData.authData, container, _callbackFn,
    sessionData.messageCategory,
    sessionData.options);
    break;
  ...
}
```

情報

3ds-web-adapter内の brw() 関数は以下のパラメータを含みます:

- authData : JSON形式のすべての必要なデータ。データ構造については、[APIドキュメント](#)を参照してください。
- container : 複数の iframe を生成する3ds-web-adapterの事前定義されたコンテナ。
- callbackFn : 認証結果を処理するコールバック関数。
- messageCategory : pa -決済認証または npa -非決済認証。
- options : optional parameters for 3DS Requestor. For example, the 3DS Requestor can choose to cancel the challenge by setting `options.cancelChallenge=true`.

次に3ds-web-adapter内の brw() メソッドから3DSリクエスターバックエンドに認証を初期化する為の情報を送信します(ステップ 2)。

```
//3ds-web-adapter.js
function brw(authData, container, callbackFn, messageCategory, options) {

  _callbackFn = callbackFn;
  iframeContainer = container;
  if (options) {
    _options = options;
  }
  //iframeのためにランダムな番号を作成する
  iframeId = String(Math.floor(100000 + Math.random() * 900000));

  //認証を初期化するための3DSリクエストのURL
  var initAuthUrl;
  if (messageCategory) {
    if (messageCategory === "pa" || messageCategory === "npa") {
      initAuthUrl = "/v1/auth/init/" + messageCategory;
    } else {
      _onError({"Error": "Invalid messageCategory"});
    }
  } else {
    initAuthUrl = "/v1/auth/init/" + "pa";
  }

  console.log('init authentication', authData);

  // /auth/init/{messageCategory}に認証を初期化するためにデータを送信する(ステップ 2)
  doPost(initAuthUrl, authData, _onInitAuthSuccess, _onError);
}
```

`brw()` 関数はバックエンドへ `/api/v1/auth/init/{messageCategory}` POSTリクエストを送信します。APIメッセージオブジェクトはJSON形式で送信されます。バックエンドがどのようにこのリクエストを処理するかを確認するには [こちら](#) を参照してください。

`3ds-web-adapter` は成功した応答を処理する為に `_onInitAuthSuccess()` 関数を使用します(ステップ 5)。

```

//3ds-web-adapter.js
function _onInitAuthSuccess(data) {
  console.log('init auth returns:', data);
  if (data.threeDSServerCallbackUrl) {

    serverTransId = data.threeDSServerTransID;
    $('<iframe id="' + "3ds_" + iframeId
      + '" width="0" height="0" style="visibility: hidden;" src="'
      + data.threeDSServerCallbackUrl + '"></iframe>')
      .appendTo(iframeContainer);

    if (data.monUrl) {
      // 任意でモニタリング用iframeを追加する
      $('<iframe id="' + "mon_" + iframeId
        + '" width="0" height="0" style="visibility: hidden;" src="'
        + data.monUrl + '"></iframe>')
        .appendTo(iframeContainer);
    }
  } else {
    _onError(data);
  }
}

```

3ds-web-adapterは2つの隠し `iframe` をチェックアウトページに挿入します ([ステップ 6](#))。1つ目の `iframe` は([ステップ 7](#)) 用で、`threeDSServerCallbackUrl` を使用してブラウザの情報がACSとActiveServerにより収集出来るようにします。

2つ目はオプションのモニタリング `iframe` で、ブラウザ情報を収集する間もしくは3DSメソッドの処理中に何かしらのエラーが発生した時に、`InitAuthTimedOut` イベントを**ActiveServer**受信できるようになります。このイベントのタイムアウトは**15秒**です。

注目

シーケンス図の[ステップ 1](#)から[ステップ 7](#)はこの処理で実装されています。

処理 2: 認証の実行

認証を実行するためにはフロントエンドはブラウザ情報の収集、3DSメソッドデータの収集（収集が可能な場合のみ）を終える必要があります。これらの2つの処理が完了した後、

`notify_3ds_events.html` は `3ds-web-adapter` に認証を続けるよう通知します。この処理内で、何らかの理由によりデータ収集が失敗したか、もしくは完了出来なかった場合、`InitAuth`処理内でセットアップした別のモニタリング `iframe` によって `InitAuthTimedOut` のイベントが `3ds-web-adapter` に通知され、認証処理は終了されます。

以下は認証を実行するためのステップです。

- `notify_3ds_events.html` を実装もしくは提供されたものを再利用し、データ収集についてのイベントを受信します。
- `_on3DSMethodSkipped` もしくは `_on3DSMethodFinished` イベントが通知された後、**認証の実行** メッセージを **3DSリクエスター** に送信します (**ステップ 8**と**ステップ 9**)。
- 認証結果をフリクシオンレスフローもしくはチャレンジフローで処理します。 (**ステップ 13** に続き、**ステップ 14(F)**または**ステップ 14(C)**)。

`notify_3ds_events.html` は認証処理を開始するのに使用されます (**ステップ 8**)。 **3DSリクエスター**は `notify_3ds_events.html` に `transId`、`callbackName` とオプションの `param` 変数を提供します。バックエンド実装を確認するには[こちら](#)を参照してください。

```

<!--notify_3ds_events.html-->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>3DSecure 2.0 Authentication</title>
</head>
<body>

<form>
  <input type="hidden" id="notifyCallback" name="notifyCallback"
value={{callbackName}}>
  <input type="hidden" id="transId" name="transId" value={{transId}}>
  <input type="hidden" id="param" name="param" value={{callbackParam}}>
</form>

<script src="https://code.jquery.com/jquery-3.3.1.min.js"
  integrity="sha256-FgpCb/KJQLNfO91ta32o/NMZxltwRo8QtmkMRdAu8="
  crossorigin="anonymous"></script>
<script>
  //チェックアウトページに通知を送り、以降の処理を実行する。

  var callbackFn = parent[$('#notifyCallback').val()];
  //callbackFnは3ds-notifyハンドラーメソッドにより定義されています。
  if (typeof callbackFn === 'function') {
    callbackFn($('#transId').val(), $('#param').val());
  }
</script>

</body>
</html>

```

callbackName に応じて **3ds-web-adapter** 内の異なるメソッドを呼んでいる事が分かります(ステップ 8)。callbackName の値は `_onThreeDSMethodFinished`、`_onThreeDSMethodSkipped`、`_onAuthResult`、もしくは `_onInitAuthTimedOut` です。それぞれのメソッドの説明は以下になります:

イベント	説明
<code>_onThreeDSMethodFinished</code>	3DSメソッドがACSにより終了し、 <code>_doAuth()</code> を呼び出す時である事を通知します

イベント	説明
<code>_onThreeDSMethodSkipped</code>	3DSメソッドが(利用出来ない、もしくは別の理由により) スキップされ、 <code>_doAuth()</code> を呼び出す時である事を通知します3DSメソッドがスキップされたか否かに関わらず、3DSサーバーのブラウザ情報収集は3DSメソッドよりも前に実行される事に注意してください。
<code>_onAuthResult</code>	このイベントは認証結果が ActiveServer から取得可能になったことを通知します。フリクションレスフローとチャレンジフローで使用されます (ステップ 17(F) と 19(C)) 。
<code>_onInitAuthTimedOut</code>	3DSメソッド中もしくはブラウザ情報収集中にエラーが発生した事を通知します。デモ内ではこのイベントが発生した場合、認証処理は終了されません。

Test Results

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

Error	InitAuth timeout
--------------	------------------

Back to BRW

ここ[ステップ 8](#)で**ActiveServer**により返される `callbackName` は `_onThreeDSMethodFinished` もしくは `_onThreeDSMethodSkipped` のどちらかです。**3ds-web-adapter**はこれらのイベントを受信すると、POSTリクエストを送信し認証を実行する為に `_doAuth()` を呼び出します ([ステップ 9](#)) 。

そしてバックエンドはAPI `/api/v1/auth/brw`の呼び出しを作成します。バックエンドがどのようにそのリクエストを処理するか確認するには[こちら](#)を参照してください。

```
//3ds-web-adapter.js
function _doAuth(transId) {

  console.log('Do Authentication for transId', transId);

  //最初に3DSメソッド用のiframeを取り除く
  $("#3ds_" + iframeId).remove();
  var authData = {};
  authData.threeDSRequestorTransID = transId;
  authData.threeDSSTransID = serverTransId;

  doPost("/v1/auth", authData, _onDoAuthSuccess, _onError);
}
```

3ds-web-adapterは成功した応答を処理する為に `_onDoAuthSuccess()` 関数を使用します(ステップ 13)。

```
//3ds-web-adapter.js
function _onDoAuthSuccess(data) {
  console.log('auth returns:', data);

  if (data.transStatus) {
    if (data.transStatus === "C") {
      // 3Dリクエストはチャレンジフローに移行するか否かを選択できます。
      if (_options.cancelChallenge) {
        if (_options.cancelReason) {
          var sendData = {};
          sendData.threeDSSTransID = serverTransId;
          sendData.status = _options.cancelReason;
          doPost("/v1/auth/challenge/status", sendData, _onCancelSuccess,
            _onCancelError)
        } else {
          var returnData = _cancelMessage();
          _callbackFn("onAuthResult", returnData);
        }
      } else {
        data.challengeUrl ? startChallenge(data.challengeUrl) : _onError(
          {"Error": "Invalid Challenge Callback Url"});
      }

    } else {
      _callbackFn("onAuthResult", data);
    }
  } else {
    _onError(data);
  }
}
```

返された `transStatus` に基づいて異なるフローを実行します。

注目

transStatusは Y、C、N、U、A とです。

- Y: 認証 / 口座確認に成功
- C: チャレンジが必要
- N: 未認証 / 口座未確認
- U: 認証 / 口座確認を実行できなかった
- A: 処理の試行が実施された
- R: 未認証 / 口座未確認または取引拒否

詳細は[APIドキュメント](#)を参照してください。

フリクシヨレス認証結果

もし transStatus が C ではない場合（つまりフリクシヨレスの場合）、**3ds-web-adapter**はフリクシヨレスフローへ移行し([ステップ 14\(F\)](#))、`_callbackFn("onAuthResult", data)` を呼び結果を表示します (11行目)。

```
//process.html
function _callbackFn(type, data) {

  switch (type) {
    case "onAuthResult":
      // "Show results in separate page"を表示する。
      $("#sepButton").removeClass("d-none");
      showResult(data);
      break;
  }
}
```

`showResult(data)` 関数は `process.html` ページ内に結果を表示します:

Test Results

[Back to Shop](#)

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAYVnCUQgAAAAAWcJAAAAAAAA=
transStatus	Y
threeDSServerTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

[Show results in separate page »](#)

フリクションレス認証処理はこの時点で終了になります。そして加盟店チェックアウト処理は、認証結果情報を使用して通常のオーソリ処理に移動できます。

情報

認証結果は別のページにも表示されます。これは[処理 3](#)で説明されます。

チャレンジ処理の続行

`transStatus` が `C` の場合、ACSがチャレンジを要求したことを示します。3DS2認証を継続してチャレンジプロセスに進むか、チャレンジが不要な場合は認証プロセスを終了するかを決定するのは、3DSリクエスター次第です。このデモでは、`options.cancelChallenge` パラメーターを使用して、チャレンジフローに関する3DSリクエスターの意思決定を示します。この機能は、[BRWテストページ](#)で概説されています。

備考

デモの目的で、キャンセルチャレンジプロセスはフロントエンドの `javascript (3ds-web-adapter)` で実装されています。セキュリティ上の理由から、このプロセスは本番環境ではバックエンドで実装する必要がある場合があります。

`3ds-web-adapter`は、`options.cancelChallenge` パラメーターをチェックします。`options.cancelChallenge = true` の場合、`3ds-web-adapter`はチャレンジをキャンセルします。オプションで、指定されたキャンセルの理由に応じて、`options.cancelReason` を設定して `ActiveServer` に送信することもできます。指定された理由は、`3ds-web-adapter`によってパッ

クエンドから `/auth/challenge/status` に送信されます。バックエンドがこのリクエストを処理する方法を確認するには、[こちら](#)を参照してください。

チャレンジ結果のキャンセル画面は、次のスクリーンショットのようになります。

Test Results

Back to BRW

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

Challenge cancelled	You can get further challenge results by select the "Show results in separate page" button after at least 30 seconds
Cancel reason	CReqNotSent
threeDSserverTransID	932ecfb3-e768-4dd7-8103-6a00b6192b9f

Show results in separate page »

`options.cancelChallenge = true` が存在しない場合（または値が `false` に設定されている場合）、**3ds-web-adapter**は `startChallenge()` を呼び出し、チャレンジウィンドウに `src` を `challengeUrl` に設定した `iframe` を挿入します(ステップ. 14 (C))

```
//3ds-web-adapter.js
function startChallenge(url) {

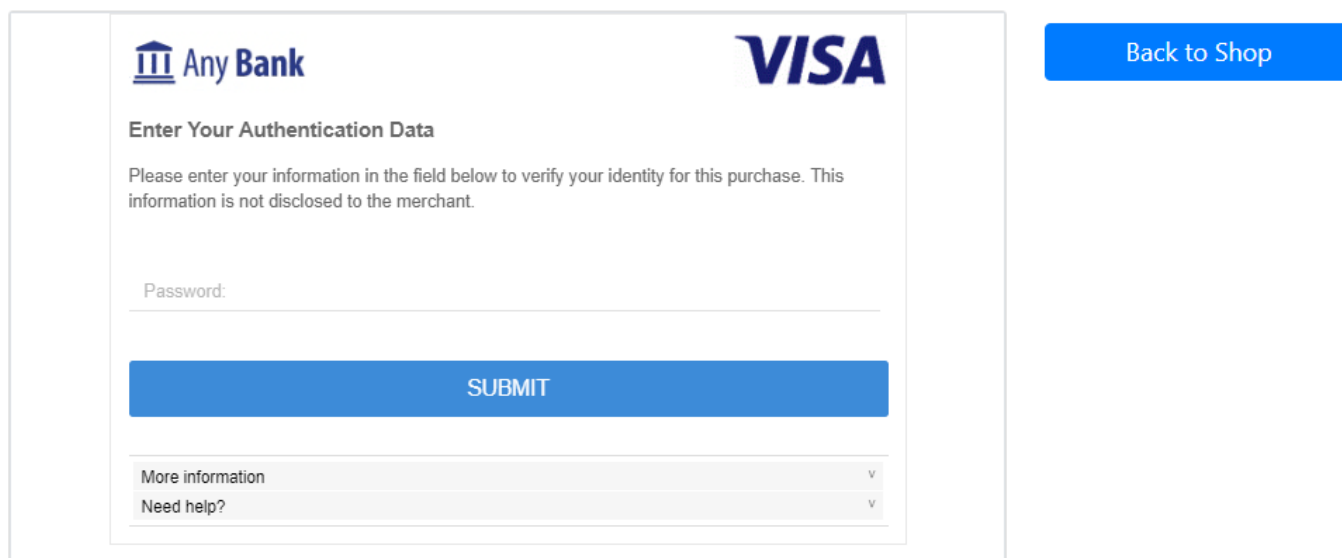
  _callbackFn("onChallengeStart");
  // チャレンジ用iframeを生成する
  $('<iframe src="' + url
    + '" width="100%" height="100%" style="border:0" id="' + "cha_" + iframeId
    + '"></iframe>')
  .appendTo(iframeContainer);
}

```

注目

チャレンジシナリオを試したい場合は、[こちら](#)のガイドに従ってください。

`iframe` の `class` 属性が `width="100%"` と `height="100%"` に設定される事が分かります。これは `iframe` がACSに提供される内容に従ってサイズ変更をする為に必要です。チャレンジスクリーンは以下のスクリーンショットに似た表示になるはずですが。



Any Bank VISA

Enter Your Authentication Data

Please enter your information in the field below to verify your identity for this purchase. This information is not disclosed to the merchant.

Password:

SUBMIT

More information v
Need help? v

Back to Shop

そしてカード会員はワンタイムパスワードのような認証データを入力してチャレンジフォームを送信する事が出来ます。ACSは取引とカード会員を認証してチャレンジ結果を返します。

チャレンジフローが終了した後、**ActiveServer**は**3DSリクエスター**バックエンド内の `/3ds-notify` エントリーポイントを `iframe` を通して呼び出し、`_onAuthResult` が呼ばれます。**3DSリクエスター**は以前 ([ステップ 19\(C\)](#))と同様に `notify_3ds_events.html` ページをレンダリングします。**3ds-web-adapter**は認証結果を取得しそれを `process.html` ページに表示する為に `_onAuthResult()` 関数を呼び出します。

i 情報

本番環境ではACSはカード会員から得られた情報から複雑なリスクベースの認証を実行します。同様に、認証メソッド（例えばワンタイムパスワード、生体認証など）がカード会員のイシューアにより決定され、実行されます。

処理3: 認証結果の取得

チャレンジ処理が終了した後（もしくはフリクションレス処理結果を別ページに表示する必要がある場合）、オーソリ処理に必要な最終認証結果を取得するために、認証結果を要求する必要があります。

⚠ なぜ別の認証結果要求が必要なのか？

処理 2の後で利用可能な認証の結果を既に持っているのに、なぜ結果をもう一度要求する必要があるのか不思議に思うかもしれません。

これは、[ステップ 13](#)で認証結果がページに転送されているためです。認証結果ページは、チェックアウトページとは別のページとして表示されるのが一般的です。この場合、**3ds-web-adapter**は結果を3DSリクエスターまたは結果ページに転送できますが、認証結果はクライアント側コードによって再転送されるため、推奨されるデータフローではありません。一般的に安全ではないと見なされます。

3DSリクエスターのサーバー側には、常に元のソース(**ActiveServer**)から結果を取得する独自のメカニズムが必要です。3DSリクエスターのクライアント側が認証結果を提供するのではなく、3DSリクエスターのサーバー側が認証結果を取得し結果ページを提供することでデータがより安全であることが保証できます。したがって、このステップで認証結果を再要求する必要があります。

このデモでは、`process.html` 内の処理 2の最後の `Show results in separate page` を選択して別のページに結果を表示する事も出来ます。

認証結果を取得し別ページに表示する為に、フロントエンドが必要な事は:

- **認証結果の取得** メッセージを3DSリクエスターに送信する([ステップ 15\(F\)](#)もしくは[ステップ 20\(C\)](#))。
- 画面に結果を表示する ([ステップ 17\(F\)](#) もしくは [ステップ 22\(C\)](#))。

まず、`process.html` ページ内で `serverTransId` を `sessionStorage` 内に保存し、`result.html` ページに移動します。

```
//process.html
function openResultInNewWindow() {
if (serverTransId) {
var url = '/v1/result';

sessionStorage.setItem("serverTransId", serverTransId);
window.open(url, 'newwindow', 'height=800,width=1000');
}
}
```

`v1/result.html` ページ内で、認証結果を取得する為に **3ds-web-adapter**内の `result()` メソッドを呼び出します。`result()` メソッドは **認証結果の取得** メッセージをバックエンドへ送信します。バックエンドはこのリクエストを受信し、`/api/v1/auth/brw/result`メッセージを **ActiveServer**に送信します。バックエンドがどのようにこのリクエストを処理するか確認するに

は、[こちら](#)を参照してください。コールバック関数 `showData()` はページに結果を表示します。

```
//result.html
var serverTransId = sessionStorage.getItem("serverTransId");

// 認証結果を要求する。(Step 15(F)またはStep 20(C))
result(serverTransId, showData);

// 認証結果を別ページに表示する (Step 17(F)またはStep 22(C))
function showData(type, data) {
  var toShow = "<dl class='row'>";
  Object.keys(data).forEach(function (k) {
    toShow = toShow + "<dt class='col-sm-4'>" + k + "</dt>" + "<dd class='col-sm-8'>" + data[k]
      + "</dd>";
  });
  toShow += "</dl>";
  $('#showResult').empty().append(toShow);
  $('#resultCard').removeClass("d-none");
}
```

新しい結果画面は以下のスクリーンショットのようになります:

Test Results	
Test result values are displayed below	
These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.	
dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAyVnCUQgAAAAAWcJAAAAAA=
transStatus	Y
threeDSServerTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

✓ 成功

この文書ではフロントエンド統合について説明しました。認証が完了した後、チェックアウト処理は取引ステータス、ECIとCAVVを使用してオーソリ処理を続け、取引を完了出来ます。

次のチャプター

次へボタンを選択し3DSリクエスターの **バックエンド実装**について確認できます。

バックエンド実装(v1)

この章では、[サンプルコード](#)を使用して、加盟店サイトのバックエンドへの実装方法について説明します。

重要

GPayments社は今から実装を始める場合は認証APIV2を実装されることを強く推奨いたします。将来的に認証APIV1はサポートされなくなります。どのようにしてAPI V1からV2に移行するのかは[API V1からV2に移行ガイド](#)をご覧ください。認証APIV2のバックエンドの実装をご覧になりたい場合は[こちら](#)をご覧ください。

バックエンドには**3DSリクエスター**を実装する必要があります。**3DSリクエスター**は**3ds-web-adapter**から情報を受信し**ActiveServer**へリクエストを送信します。また**ActiveServer**からの認証結果も受信し、結果を**3ds-web-adapter**へ転送します。

3DSリクエスターのデモのコードは、バックエンドの実装を以下のサーバーサイドの言語付きで提供します:

- Java - JavaのバージョンはSpringbootフレームワークを使用して実装されています。Springbootの詳細は<https://spring.io/projects/spring-boot>を参照してください。
- C# - C#のバージョンはASP.netを使用して実装されています。
- PHP - PHPのバージョンはcURL(Client URL Library)とPHP7.2を使用して実装されています。
- Go - GoのバージョンはGo 1.12とGoモジュールサポートを使用して実装されます。すべての依存性は [go.mod](#) ファイルに記載されています。

なぜバックエンドの実装が必要なのか？

EMVCo's 3Dセキュア2.0の仕様で定義されているように、3DSサーバーと3DSリクエスターの環境が分かれている時、これらの2つのコンポーネント間の通信は相互認証されていなければなりません。

[Req 300] もし3DSリクエスターと3DSサーバーが別のコンポーネントである場合は、コンポーネント間で転送されるデータが、支払いシステムのセキュリティ要件を満たすレベルで保護され両方のサーバーで相互認証されている事を確保してください。

ActiveServerと認証プロセスを行う前に、3DSリクエスターはActiveServer と相互TLS接続を確立する必要があります。[クライアント証明書](#)を所持し3DSリクエスターでそれが設定されている事を確認してください。そうでない場合は、導入ページにある [クライアント証明書の取得と3DSリクエスター詳細の設定](#) の説明に従ってください。

HTTPクライアントの為のTLS設定の導入は下記にあります:

次に、認証プロセスと認証シーケンスに基づいたバックエンド実装の詳細について記述します。

- Java - TLS構成とクライアント証明書のロード方法は、クラス `RestClientConfig` にあります。
- C# - TLS構成とクライアント証明書のロード方法は、クラス `RestClientHelper.cs` にあります。
- PHP - TLS構成とクライアント証明書のロード方法は、ファイル `RestClientConfig.php` にあります。
- Go - TLS構成とクライアント証明書のロード方法は、ファイル `https.go` にあります。

次に、[認証プロセス](#)および[認証シーケンス](#)に基づいたバックエンド実装の詳細を説明します。

処理 1: 認証の初期化

認証を初期化するためには、3DSリクエスターが以下のように動作する必要があります。

- **認証の初期化** リクエストを3DS-web-adapter([Step. 2](#))から受信する。
- **ActiveServer**にリクエストを送信する ([Step. 3](#))。
- **ActiveServer**([Step. 4](#))から応答を受信する。
- 受信したデータを3DS-web-adapter ([Step. 5](#))に返答する。

Java C# PHP Go

```
//AuthControllerV1.java
/**
 * 認証の初期化要求を3DS-web-adapterから受信する。(ステップ 2)
 * ActiveServerに認証を初期化するためのデータを送信する。
 */
@PostMapping("/v1/auth/init/{messageCategory}")
public Message initAuth(@RequestBody Message request,
    @PathVariable(value = "messageCategory") String messageCategory) {

    // リクエスターのトランザクションIDを生成する。
    String transId = UUID.randomUUID().toString();
    request.put("threeDSRequestorTransID", transId);
    // イベントコールバックURLを「リクエスターURL + /3ds-notify」に設定する。
    String callBackUrl = config.getBaseUrl() + "/3ds-notify";
    request.put("eventCallbackUrl", callBackUrl);

    // 認証を初期化するためのActiveServerのURL
    String initAuthUrl = config.getAsAuthUrl() + "/api/v1/auth/brw/init/" +
messageCategory;
    logger.info("initAuthRequest on url: {}, body: \n{}", initAuthUrl, request);

    // ActiveServerに認証を初期化するためのデータを送信する。(ステップ 3)
    // ActiveServerから応答を受け取る (ステップ 4)
    Message response =
        sendRequest(initAuthUrl, request, HttpMethod.POST);
    logger.info("initAuthResponseBRW: \n{}", response);

    // 3ds-web-adapterデータを返答する。(ステップ 5)
    return response;
}
```

 注目

eventCallbackUrl を `{baseUrl}/3ds-notify` に設定しています。このURLを設定することで **ActiveServer** からブラウザ情報の収集が完了した時通知を受ける事ができます。(ステップ. 7)。

baseUrl is set [here](#).

initAuth リクエストが `{ActiveServer auth url}/api/v1/auth/brw/init/{messageCategory}` のURLに送信されます。initAuth リクエストのデータ構造をご覧になりたい場合は [認証APIドキュメント](#) を参照してください。

認証APIマスタークライアント証明書用のHTTPヘッダー

認証APIマスタークライアント証明書を使用してビジネス管理者ユーザーとして加盟店に代わって認証を行う場合、バックエンドは、HTTPヘッダーに `AS-Merchant-Token` フィールドを追加する必要があります。このフィールドは加盟店プロファイルから取得できる `マーチャントトークン` を設定します。

Java C# PHP Go

```
//注目: sendRequest()がActiveServerにリクエストを送信します。
//これがgroupAuthの場合、リクエストにはAS-Merchant-Tokenのフィールドを持つHTTPヘッダーが含まれている必要があります。
private Message sendRequest(String url, Message request, HttpMethod method) {

    HttpEntity<Message> req;
    HttpHeaders headers = null;

    if (config.isGroupAuth()) {
        //the certificate is for groupAuth, work out the header.
        headers = new HttpHeaders();
        headers.add("AS-Merchant-Token", config.getMerchantToken());
    }

    switch (method) {
        case POST:
            req = new HttpEntity<>(request, headers);
            return restTemplate.postForObject(url, req, Message.class);
        case GET:
            if (headers == null) {
                return restTemplate.getForObject(url, Message.class);
            } else {
                req = new HttpEntity<>(headers);
                return restTemplate.exchange(url, HttpMethod.GET, req,
Message.class).getBody();
            }
        default:
            return null;
    }
}
```

処理2: 認証の実行

認証を実行する為に3DSリクエストが必要な事は:

- ステップ 7の後、ActiveServerからの /3ds-notify メッセージを処理する。
- 3DS-web-adapter からの 認証の実行 リクエストを処理する (ステップ 9とステップ 10)。
- 認証結果を受信し、それを3DS-web-adapterへ返答する (ステップ 12とステップ 13)。

ブラウザ情報の収集 (ステップ7)が完了した時、ActiveServer はあなたが eventCallbackUrl に設定した `http://localhost:8082/3ds-notify` に通知を行います。3DSリクエストはこの通知を処理し必要なパラメータを `notify-3ds-events.html` ページへ渡します。

Java C# PHP Go

```
//MainController.java
@PostMapping("/3ds-notify")
public String notifyResult(
    @RequestParam("requestorTransId") String transId,
    @RequestParam("event") String callbackType,
    @RequestParam(name = "param", required = false) String param,
    Model model) {

    String callbackName;
    if ("3DSMethodFinished".equals(callbackType)) {

        callbackName = "_on3DSMethodFinished";

    } else if ("3DSMethodSkipped".equals(callbackType)) {

        callbackName = "_on3DSMethodSkipped";

    } else if ("AuthResultReady".equals(callbackType)) {
        callbackName = "_onAuthResult";
    } else {
        throw new IllegalArgumentException("invalid callback type");
    }

    model.addAttribute("transId", transId);
    model.addAttribute("callbackName", callbackName);
    model.addAttribute("callbackParam", param);

    return "notify_3ds_events";
}
```

 注目

このハンドラーメソッドは**ActiveServer**により呼び出され、**ActiveServer**により `requestorTransId`、`event`、オプションの `param` が提供されます。 `event` は `3DSMethodFinished`、`3DSMethodSkipped`、もしくは `AuthResultReady` になります。そしてハンドラーメソッドはページコンテキスト内の適切な属性値を設定し `notify_3ds_events.html` ページへ返します。そして **Mustache** テンプレートエンジンを使用してページにレンダリングします。

フロントエンドの `notify_3ds_events.html` の実装を確認したい場合は [こちら](#) から確認できます。

そして3DSクライアントがブラウザ情報の収集を終えた時、`auth` エンドポイントを呼び出し認証を開始します。**3DSリクエスター**は **認証の実行** リクエストを処理します。(ステップ 9、ステップ 10、ステップ 12とステップ 13)。

Java C# PHP Go

```
//AuthControllerV1.java
/**
 * 認証の実行要求を3DS-web-adapterから受信する。(ステップ 9)
 * 受信したデータをActiveServerに送信し認証を実行する。
 */
@PostMapping("/v1/auth")
public Message auth(@RequestBody Message request) {

    // 認証を実行するためのActiveServerのURL
    String authUrl = config.getAsAuthUrl() + "/api/v1/auth/brw";
    logger.info("requesting BRW Auth API {}, body: \n{}", authUrl, request);

    // データをActiveServerに送信し認証を実行する。(ステップ 10)
    // 応答データをActiveServerから受信する。(ステップ 12)
    Message response =
        sendRequest(authUrl, request, HttpMethod.POST);
    logger.info("authResponseBRW: \n{}", response);

    //3ds-web-adapterデータを返答する。(ステップ 13)
    return response;
}
```

注目

3DSリクエスターは、フロントエンドへ返答します。返されるメッセージには、フリクションレスフローまたはチャレンジフローのいずれかによって **Y** または **C** の `transStatus` 値を含みます。フロントエンドがどのように `transStatus` を処理するかを確認するには、[こちら](#)を参照してください。応答データの構造を確認するには、[APIドキュメント](#)を参照してください。

チャレンジフローをキャンセル

`transStatus = C` の場合、3DSクライアントはチャレンジを開始するか否かを選択できます。3DSクライアントがチャレンジをキャンセルすることを選択した場合、`/auth/challenge/status` エンドポイントを呼び出して **キャンセルした理由** を指定できます。**3DSリクエスター**が **ActiveServer** に送信します。フロントエンドがこのリクエストを処理する方法を確認するには、[こちら](#)を参照してください。

Java C# PHP Go

```
//AuthControllerV1.java
@PostMapping("/auth/challenge/status")
public Message challengeStatus(@RequestBody Message request) {

    String challengeStatusUrl = config.getAsAuthUrl() + "/api/v1/auth/challenge/status";
    logger.info("request challenge status API {}", body: "\n{}", challengeStatusUrl, request);

    Message response =
        sendRequest(challengeStatusUrl, request, HttpMethod.POST);
    logger.info("challengeStatus response: \n{}", response);

    return response;
}
```

処理 3: 認証結果の取得

認証結果を取得する為に**3DSリクエスター**が必要な事は:

- **3DS-web-adapter**からの **認証結果の取得** リクエストを処理する (**ステップ 15(F)**もしくは**ステップ 20(C)**) 。

- 結果を取得しフロントエンドへ返す為に**ActiveServer**へリクエストを送信する (ステップ 16(F)、ステップ 17(F) もしくは ステップ 21(C)、22(C))。

Java C# PHP Go

```
//AuthControllerV1.java
/**
 * 認証結果の取得要求を受信する。(ステップ15(F)とステップ20(C))
 * ActiveServerから認証結果を取得する。
 */
@GetMapping("/v1/auth/result")
public Message result(@RequestParam("txid") String serverTransId) {

    // ActiveServerからURLを取得する
    String resultUrl = config.getAsAuthUrl() +
        "/api/v1/auth/brw/result?threeDSServerTransID=" +
        serverTransId;

    // ActiveServerから認証結果を取得する。(ステップ16(F)とステップ21(C))
    Message response = sendRequest(resultUrl, null, HttpMethod.GET);
    logger.info("authResponse: \n{}", response);

    //認証結果をresult.htmlに表示する。(ステップ17(F)とステップ22(C))
    return response;
}
```

✓ 成功

この文書ではバックエンド実装についての説明は以上で終了です。認証が完了した後、チェックアウト処理は取引ステータス、ECIとCAVVを使用して承認の実行を続け、取引を完了出来ます。

🔗 次のチャプター

次へボタンを選択しGPayments 3DSリクエスターの為のサンプルコードのすべての機能の説明をご覧ください。

フロントエンド実装(v2)

このセクションでは、[サンプルコード](#)を使用して加盟店アプリケーションのためのフロントエンドの実装方法を示します。

サンプルコードパッケージ内の以下のファイルはフロントエンド内の3DS2認証に必須です。必要な場合は[ディレクトリツリー](#)で詳細を確認してください。

- `v2/process.html` - すべての[認証シーケンス](#)を実装しています。
- `v2/3ds-web-adapter.js` - 3DS2データをフロントエンドからバックエンドへ渡し、コールバックURLの為に必要な `iframe` を生成する**3DSクライアント**の重要なコンポーネントです。
- `notify_3ds_events.html` - **ActiveServer**の為に使用されるコールバックページで、認証において([ステップ 7](#)と[ステップ 18\(C\)](#))を開始する際に必要になります。このページはチェックアウト処理に認証イベントを渡します。

以下は[認証処理](#)と[認証シーケンス](#)に基づいたフロントエンド実装の詳細です。

処理1: 認証の初期化

認証を初期化する為に、フロントエンドが必要な事:

- **認証の初期化** メッセージを**3DSリクエスター**へ送信する ([ステップ 1](#)と[ステップ 2](#))。
- 応答メッセージを受信しコールバック `iframe` をセットアップする ([ステップ 5](#)と[ステップ 6](#))。

ユーザーがチェックアウトページ内の**チェックアウトを続ける**ボタンをクリックすると、ブラウザは必要なデータをセッションストレージに保存し `process.html` ページに移動します。

```
//checkout.html
function checkout() {
  var apiVersion = getApiVersion();
  switch (apiVersion) {
    case "v2":
      goApiV1();
      break;
    ...
  }
}

function goApiV1() {
  var sessionData = getSessionData();
  sessionData.messageCategory = "pa";
  sessionStorage.setItem("sessionData", JSON.stringify(sessionData));
  window.location.href = "/v2/process";
}

function getSessionData() {
  var sessionData = {};
  sessionData.channel = "brw";
  sessionData.authData = genAuthData();
  sessionData.backButtonType = "toShop";
  return sessionData;
}
```

注目

sessionData が含むもの:

- **channel** : brw または 3ri。上記の例では、ブラウザベースの認証を実行するために brw チャンネルを使用しました。
- **messageCategory** : pa -決済認証または npa -非決済認証のいずれか。上記の例では、pa を使用しました。
- **authData** : JSON形式のすべての必要なデータ。データ構造については、[APIドキュメント](#)を参照してください。
- **backButtonType** : プロセスページの back ボタンタイプを示します。上記の例では **戻る** ボタンを **ショップに戻る** として作成します。

注意点 : ページ間通信に sessionStorage を使用するのには、デモ用です。3DSリクエストの実際の実装では、既存のチェックアウトプロセスと統合するために、ページ間でパラメーターを転送するための最適なアプローチを選択しましょう。

process.html ページ内で、sessionData 受信し3DS2処理を開始します。まず、認証を初期化する為の情報を3ds-web-adapterに送信します (ステップ 1)。

```
//process.html
var sessionData = JSON.parse(sessionStorage.getItem("sessionData"));
...
switch (sessionData.channel) {
  case "brw":
    var container = $('#iframeDiv');
    brw(sessionData.authData, container, _callbackFn,
    sessionData.messageCategory,
    sessionData.options);
    break;
  ...
}
```

情報

3ds-web-adapter内の brw() 関数は以下のパラメータを含みます:

- authData : JSON形式のすべての必要なデータ。データ構造については、[APIドキュメント](#)を参照してください。
- container : 複数の iframe を生成する3ds-web-adapterの事前定義されたコンテナ。
- callbackFn : 認証結果を処理するコールバック関数。
- messageCategory : pa -決済認証または npa -非決済認証。
- options : optional parameters for 3DS Requestor. For example, the 3DS Requestor can choose to cancel the challenge by setting `options.cancelChallenge=true`.

次に3ds-web-adapter内の brw() メソッドから3DSリクエスターバックエンドに認証を初期化する為の情報を送信します(ステップ 2)。

```
//3ds-web-adapter.js
function brw(authData, container, callbackFn, messageCategory, options) {

  _callbackFn = callbackFn;
  iframeContainer = container;
  if (options) {
    _options = options;
  }
  //iframeのためにランダムな番号を作成する
  iframeId = String(Math.floor(100000 + Math.random() * 900000));

  //認証を初期化するための3DSリクエストのURL
  var initAuthUrl;
  if (messageCategory) {
    if (messageCategory === "pa" || messageCategory === "npa") {
      initAuthUrl = "/v2/auth/init/" + messageCategory;
    } else {
      _onError({"Error": "Invalid messageCategory"});
    }
  } else {
    initAuthUrl = "/v2/auth/init/" + "pa";
  }

  console.log('init authentication', authData);

  // /auth/init/{messageCategory}に認証を初期化するためにデータを送信する(ステップ 2)
  doPost(initAuthUrl, authData, _onInitAuthSuccess, _onError);
}
```

`brw()` 関数はバックエンドへ `/api/v2/auth/init/{messageCategory}` POSTリクエストを送信します。APIメッセージオブジェクトはJSON形式で送信されます。バックエンドがどのようにこのリクエストを処理するかを確認するには [こちら](#) を参照してください。

3ds-web-adapterは成功した応答を処理する為に `_onInitAuthSuccess()` 関数を使用します(ステップ 5)。

```
//3ds-web-adapter.js
function _onInitAuthSuccess(data) {
  console.log('init auth returns:', data);
  if (data.threeDSServerCallbackUrl) {

    serverTransId = data.threeDSServerTransID;
    $('<iframe id="" + "3ds_" + iframeId
      + '" width="0" height="0" style="visibility: hidden;" src=""
      + data.threeDSServerCallbackUrl + '"></iframe>')
      .appendTo(iframeContainer);

    if (data.monUrl) {
      // 任意でモニタリング用iframeを追加する
      $('<iframe id="" + "mon_" + iframeId
        + '" width="0" height="0" style="visibility: hidden;" src=""
        + data.monUrl + '"></iframe>')
        .appendTo(iframeContainer);
    }
  } else {
    _onError(data);
  }
}
```

3ds-web-adapterは2つの隠し `iframe` をチェックアウトページに挿入します ([ステップ 6](#))。1つ目の `iframe` は([ステップ 7](#)) 用で、`threeDSServerCallbackUrl` を使用してブラウザの情報がACSとActiveServerにより収集出来るようにします。

2つ目はオプションのモニタリング `iframe` で、ブラウザ情報を収集する間もしくは3DSメソッドの処理中に何かしらのエラーが発生した時に、`InitAuthTimedOut` イベントを**ActiveServer**受信できるようになります。このイベントのタイムアウトは**15秒**です。

注目

シーケンス図の[ステップ 1](#)から[ステップ 7](#)はこの処理で実装されています。

処理 2: 認証の実行

認証を実行するためにはフロントエンドはブラウザ情報の収集、3DSメソッドデータの収集（収集が可能な場合のみ）を終える必要があります。これらの2つの処理が完了した後、

`notify_3ds_events.html` は `3ds-web-adapter` に認証を続けるよう通知します。この処理内で、何らかの理由によりデータ収集が失敗したか、もしくは完了出来なかった場合、`InitAuth`処理内でセットアップした別のモニタリング `iframe` によって `InitAuthTimedOut` のイベントが `3ds-web-adapter` に通知され、認証処理は終了されます。

以下は認証を実行するためのステップです。

- `notify_3ds_events.html` を実装もしくは提供されたものを再利用し、データ収集についてのイベントを受信します。
- `_on3DSMethodSkipped` もしくは `_on3DSMethodFinished` イベントが通知された後、**認証の実行** メッセージを **3DSリクエスター** に送信します (**ステップ 8**と**ステップ 9**)。
- 認証結果をフリクシオンレスフローもしくはチャレンジフローで処理します。 (**ステップ 13**に続き、**ステップ 14(F)**または**ステップ 14(C)**)。

`notify_3ds_events.html` は認証処理を開始するのに使用されます (**ステップ 8**)。 **3DSリクエスター**は `notify_3ds_events.html` に `transId`、`callbackName` とオプションの `param` 変数を提供します。バックエンド実装を確認するには[こちら](#)を参照してください。

```

<!--notify_3ds_events.html-->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>3DSecure 2.0 Authentication</title>
</head>
<body>

<form>
  <input type="hidden" id="notifyCallback" name="notifyCallback"
value={{callbackName}}>
  <input type="hidden" id="transId" name="transId" value={{transId}}>
  <input type="hidden" id="param" name="param" value={{callbackParam}}>
</form>

<script src="https://code.jquery.com/jquery-3.3.1.min.js"
  integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
  crossorigin="anonymous"></script>
<script>
  //チェックアウトページに通知を送り、以降の処理を実行する。

  var callbackFn = parent[$('#notifyCallback').val()];
  //callbackFnは3ds-notifyハンドラーメソッドにより定義されています。
  if (typeof callbackFn === 'function') {
    callbackFn($('#transId').val(), $('#param').val());
  }
</script>

</body>
</html>

```

callbackName に応じて **3ds-web-adapter** 内の異なるメソッドを呼んでいる事が分かります(ステップ 8)。callbackName の値は `_onThreeDSMethodFinished`、`_onThreeDSMethodSkipped`、`_onAuthResult`、もしくは `_onInitAuthTimedOut` です。それぞれのメソッドの説明は以下になります:

イベント	説明
<code>_onThreeDSMethodFinished</code>	3DSメソッドがACSにより終了し、 <code>_doAuth()</code> を呼び出す時である事を通知します

イベント	説明
<code>_onThreeDSMethodSkipped</code>	3DSメソッドが(利用出来ない、もしくは別の理由により) スキップされ、 <code>_doAuth()</code> を呼び出す時である事を通知します3DSメソッドがスキップされたか否かに関わらず、3DSサーバーのブラウザ情報収集は3DSメソッドよりも前に実行される事に注意してください。
<code>_onAuthResult</code>	このイベントは認証結果が ActiveServer から取得可能になったことを通知します。フリクションレスフローとチャレンジフローで使用されます (ステップ 17(F) と 19(C)) 。
<code>_onInitAuthTimedOut</code>	3DSメソッド中もしくはブラウザ情報収集中にエラーが発生した事を通知します。デモ内ではこのイベントが発生した場合、認証処理は終了されません。

Test Results

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

Error	InitAuth timeout
--------------	------------------

Back to BRW

ここステップ 8で**ActiveServer**により返される `callbackName` は `_onThreeDSMethodFinished` もしくは `_onThreeDSMethodSkipped` のどちらかです。**3ds-web-adapter**はこれらのイベントを受信すると、POSTリクエストを送信し認証を実行する為に `_doAuth()` を呼び出します ([ステップ 9](#)) 。

そしてバックエンドはAPI [/api/v2/auth/brw](#) の呼び出しを作成します。バックエンドがどのようにそのリクエストを処理するか確認するには[こちら](#)を参照してください。

```
//3ds-web-adapter.js
function _doAuth(transId) {

  console.log('Do Authentication for transId', transId);

  //最初に3DSメソッド用のiframeを取り除く
  $("#3ds_" + iframeId).remove();
  var authData = {};
  authData.threeDSRequestorTransID = transId;
  authData.threeDSSTransID = serverTransId;

  doPost("/v2/auth", authData, _onDoAuthSuccess, _onError);
}
```

3ds-web-adapterは成功した応答を処理する為に `_onDoAuthSuccess()` 関数を使用します(ステップ 13)。

```
//3ds-web-adapter.js
function _onDoAuthSuccess(data) {
  console.log('auth returns:', data);

  if (data.transStatus) {
    if (data.transStatus === "C") {
      // 3Dリクエストはチャレンジフローに移行するか否かを選択できます。
      if (_options.cancelChallenge) {
        if (_options.cancelReason) {
          var sendData = {};
          sendData.threeDSSTransID = serverTransId;
          sendData.status = _options.cancelReason;
          doPost("/v2/auth/challenge/status", sendData, _onCancelSuccess,
            _onCancelError)
        } else {
          var returnData = _cancelMessage();
          _callbackFn("onAuthResult", returnData);
        }
      } else {
        data.challengeUrl ? startChallenge(data.challengeUrl) : _onError(
          {"Error": "Invalid Challenge Callback Url"});
      }

    } else {
      _callbackFn("onAuthResult", data);
    }
  } else {
    _onError(data);
  }
}
```

返された `transStatus` に基づいて異なるフローを実行します。

注目

transStatusは Y、C、N、U、A とです。

- Y: 認証 / 口座確認に成功
- C: チャレンジが必要
- N: 未認証 / 口座未確認
- U: 認証 / 口座確認を実行できなかった
- A: 処理の試行が実施された
- R: 未認証 / 口座未確認または取引拒否

詳細は[APIドキュメント](#)を参照してください。

フリクシヨレス認証結果

もし transStatus が C ではない場合（つまりフリクシヨレスの場合）、**3ds-web-adapter**はフリクシヨレスフローへ移行し([ステップ 14\(F\)](#))、`_callbackFn("onAuthResult", data)` を呼び結果を表示します (11行目)。

```
//process.html
function _callbackFn(type, data) {

  switch (type) {
    case "onAuthResult":
      // "Show results in separate page"を表示する。
      $("#sepButton").removeClass("d-none");
      showResult(data);
      break;
  }
}
```

`showResult(data)` 関数は `process.html` ページ内に結果を表示します:

Test Results

[Back to Shop](#)

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAYVnCUQgAAAAAWcJAAAAAAAA=
transStatus	Y
threeDSServerTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

[Show results in separate page »](#)

フリクションレス認証処理はこの時点で終了になります。そして加盟店チェックアウト処理は、認証結果情報を使用して通常のオーソリ処理に移動できます。

情報

認証結果は別のページにも表示されます。これは[処理 3](#)で説明されます。

チャレンジ処理の続行

`transStatus` が `C` の場合、ACSがチャレンジを要求したことを示します。3DS2認証を継続してチャレンジプロセスに進むか、チャレンジが不要な場合は認証プロセスを終了するかを決定するのは、3DSリクエスター次第です。このデモでは、`options.cancelChallenge` パラメーターを使用して、チャレンジフローに関する3DSリクエスターの意思決定を示します。この機能は、[BRWテストページ](#)で概説されています。

備考

デモの目的で、キャンセルチャレンジプロセスはフロントエンドの `javascript (3ds-web-adapter)` で実装されています。セキュリティ上の理由から、このプロセスは本番環境ではバックエンドで実装する必要がある場合があります。

`3ds-web-adapter`は、`options.cancelChallenge` パラメーターをチェックします。`options.cancelChallenge = true` の場合、`3ds-web-adapter`はチャレンジをキャンセルします。オプションで、指定されたキャンセルの理由に応じて、`options.cancelReason` を設定して `ActiveServer` に送信することもできます。指定された理由は、`3ds-web-adapter`によってパッ

クエンドから `/auth/challenge/status` に送信されます。バックエンドがこのリクエストを処理する方法を確認するには、[こちら](#)を参照してください。

チャレンジ結果のキャンセル画面は、次のスクリーンショットのようになります。

Test Results

Back to BRW

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

Challenge cancelled	You can get further challenge results by select the "Show results in separate page" button after at least 30 seconds
Cancel reason	CReqNotSent
threeDSServerTransID	932ecfb3-e768-4dd7-8103-6a00b6192b9f

Show results in separate page »

`options.cancelChallenge = true` が存在しない場合（または値が `false` に設定されている場合）、**3ds-web-adapter**は `startChallenge()` を呼び出し、チャレンジウィンドウに `src` を `challengeUrl` に設定した `iframe` を挿入します(ステップ. 14 (C))

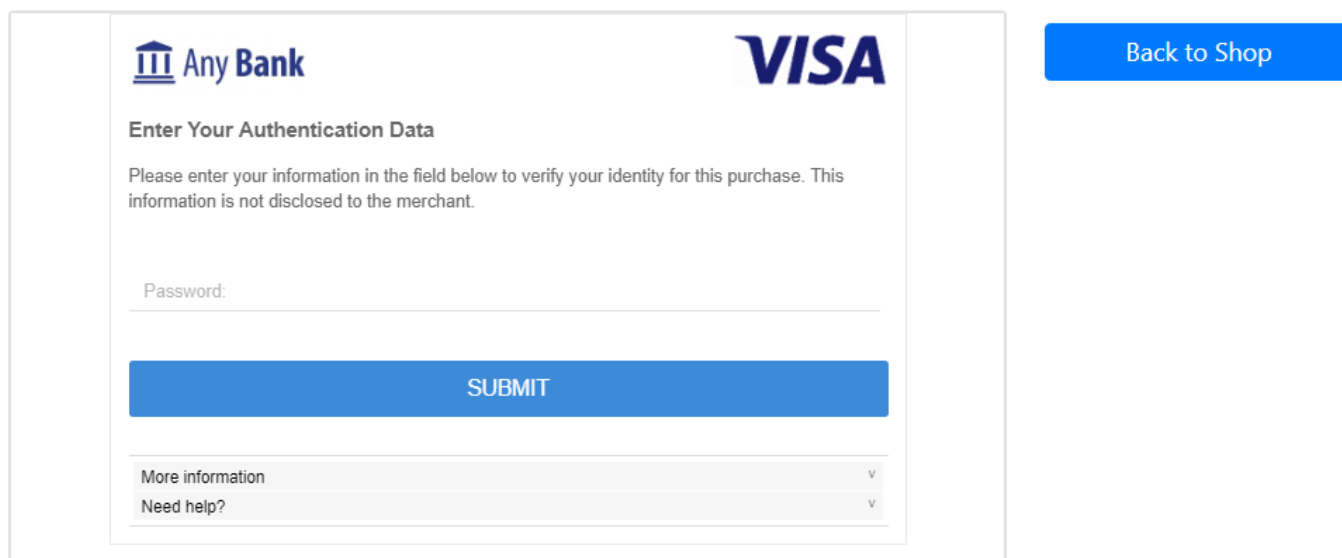
```
//3ds-web-adapter.js
function startChallenge(url) {

  _callbackFn("onChallengeStart");
  // チャレンジ用iframeを生成する
  $('<iframe src="' + url
    + '" width="100%" height="100%" style="border:0" id="' + "cha_" + iframeId
    + '"></iframe>')
  .appendTo(iframeContainer);
}
```

注目

チャレンジシナリオを試したい場合は、[こちら](#)のガイドに従ってください。

`iframe` の `class` 属性が `width="100%"` と `height="100%"` に設定される事が分かります。これは `iframe` がACSに提供される内容に従ってサイズ変更をする為に必要です。チャレンジスクリーンは以下のスクリーンショットに似た表示になるはずです。



Any Bank **VISA**

Enter Your Authentication Data

Please enter your information in the field below to verify your identity for this purchase. This information is not disclosed to the merchant.

Password: _____

SUBMIT

[More information](#) v

[Need help?](#) v

[Back to Shop](#)

そしてカード会員はワンタイムパスワードのような認証データを入力してチャレンジフォームを送信する事が出来ます。ACSは取引とカード会員を認証してチャレンジ結果を返します。

チャレンジフローが終了した後、**ActiveServer**は**3DSリクエスター**バックエンド内の `/3ds-notify` エントリーポイントを `iframe` を通して呼び出し、`_onAuthResult` が呼ばれます。**3DSリクエスター**は以前 ([ステップ 19\(C\)](#))と同様に `notify_3ds_events.html` ページをレンダリングします。**3ds-web-adapter**は認証結果を取得しそれを `process.html` ページに表示する為に `_onAuthResult()` 関数を呼び出します。

i 情報

本番環境ではACSはカード会員から得られた情報から複雑なリスクベースの認証を実行します。同様に、認証メソッド（例えばワンタイムパスワード、生体認証など）がカード会員のイシューアにより決定され、実行されます。

処理3: 認証結果の取得

チャレンジ処理が終了した後 (もしくはフリクションレス処理結果を別ページに表示する必要がある場合)、オーソリ処理に必要な最終認証結果を取得するために、認証結果を要求する必要があります。

⚠ なぜ別の認証結果要求が必要なのか？

処理 2の後で利用可能な認証の結果を既に持っているのに、なぜ結果をもう一度要求する必要があるのか不思議に思うかもしれません。

これは、[ステップ 13](#)で認証結果がページに転送されているためです。認証結果ページは、チェックアウトページとは別のページとして表示されるのが一般的です。この場合、**3ds-web-adapter**は結果を3DSリクエスターまたは結果ページに転送できますが、認証結果はクライアント側コードによって再転送されるため、推奨されるデータフローではありません。一般的に安全ではないと見なされます。

3DSリクエスターのサーバー側には、常に元のソース(**ActiveServer**)から結果を取得する独自のメカニズムが必要です。3DSリクエスターのクライアント側が認証結果を提供するのではなく、3DSリクエスターのサーバー側が認証結果を取得し結果ページを提供することでデータがより安全であることが保証できます。したがって、このステップで認証結果を再要求する必要があります。

このデモでは、`process.html` 内の処理 2の最後の [Show results in separate page](#) を選択して別のページに結果を表示する事も出来ます。

認証結果を取得し別ページに表示する為に、フロントエンドが必要な事は:

- **認証結果の取得** メッセージを3DSリクエスターに送信する([ステップ 15\(F\)](#)もしくは[ステップ 20\(C\)](#))。
- 画面に結果を表示する ([ステップ 17\(F\)](#) もしくは [ステップ 22\(C\)](#))。

まず、`process.html` ページ内で `serverTransId` を `sessionStorage` 内に保存し、`result.html` ページに移動します。

```
//process.html
function openResultInNewWindow() {
  if (serverTransId) {
    var url = '/v2/result';

    sessionStorage.setItem("serverTransId", serverTransId);
    window.open(url, 'newwindow', 'height=800,width=1000');
  }
}
```

`v2/result.html` ページ内で、認証結果を取得する為に **3ds-web-adapter**内の `result()` メソッドを呼び出します。`result()` メソッドは **認証結果の取得** メッセージをバックエンドへ送信します。バックエンドはこのリクエストを受信し、`/api/v2/auth/brw/result`メッセージを **ActiveServer**に送信します。バックエンドがどのようにこのリクエストを処理するか確認するに

は、[こちら](#)を参照してください。コールバック関数 `showData()` はページに結果を表示します。

```
//result.html
var serverTransId = sessionStorage.getItem("serverTransId");

// 認証結果を要求する。(Step 15(F)またはStep 20(C))
result(serverTransId, showData);

// 認証結果を別ページに表示する (Step 17(F)またはStep 22(C))
function showData(type, data) {
  var toShow = "<dl class='row'>";
  Object.keys(data).forEach(function (k) {
    toShow = toShow + "<dt class='col-sm-4'>" + k + "</dt>" + "<dd class='col-sm-8'>" + data[k]
      + "</dd>";
  });
  toShow += "</dl>";
  $('#showResult').empty().append(toShow);
  $('#resultCard').removeClass("d-none");
}
```

新しい結果画面は以下のスクリーンショットのようになります:

Test Results	
Test result values are displayed below	
These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.	
dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAyVnCUQgAAAAAWcJAAAAAAAA=
transStatus	Y
threeDSServerTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

✓ 成功

この文書ではフロントエンド統合について説明しました。認証が完了した後、チェックアウト処理は取引ステータス、ECIとCAVVを使用してオーソリ処理を続け、取引を完了出来ます。

次のチャプター

次へボタンを選択し3DSリクエスターの **バックエンド実装**について確認できます。

バックエンド実装(v2)

この章では、**サンプルコード**を使用して、加盟店サイトのバックエンドへの実装方法について説明します。

バックエンドには**3DSリクエスター**を実装する必要があります。**3DSリクエスター**は**3ds-web-adapter**から情報を受信し**ActiveServer**へリクエストを送信します。また**ActiveServer**からの認証結果も受信し、結果を**3ds-web-adapter**へ転送します。

3DSリクエスターのデモのコードは、バックエンドの実装を以下のサーバーサイドの言語付きで提供します:

- Java - JavaのバージョンはSpringbootフレームワークを使用して実装されています。Springbootの詳細は<https://spring.io/projects/spring-boot>を参照してください。
- C# - C#のバージョンはASP.netを使用して実装されています。
- PHP - PHPのバージョンはcURL(Client URL Library)とPHP7.2を使用して実装されています。
- Go - GoのバージョンはGo 1.12とGoモジュールサポートを使用して実装されます。すべての依存性は `go.mod` ファイルに記載されています。

なぜバックエンドの実装が必要なのか？

EMVCo's 3Dセキュア2.0の仕様で定義されているように、3DSサーバーと3DSリクエスターの環境が分かれている時、これらの2つのコンポーネント間の通信は相互認証されていなければなりません。

[Req 300] もし3DSリクエスターと3DSサーバーが別のコンポーネントである場合は、コンポーネント間で転送されるデータが、支払いシステムのセキュリティ要件を満たすレベルで保護され両方のサーバーで相互認証されている事を確保してください。

ActiveServerと認証プロセスを行う前に、3DSリクエスターは**ActiveServer**と相互TLS接続を確立する必要があります。**クライアント証明書**を所持し3DSリクエスターでそれが設定されている事を確認してください。そうでない場合は、導入ページにある **クライアント証明書の取得と3DSリクエスター詳細の設定** の説明に従ってください。

HTTPクライアントの為のTLS設定の導入は下記にあります:

次に、認証プロセスと認証シーケンスに基づいたバックエンド実装の詳細について記述します。

- Java - TLS構成とクライアント証明書のロード方法は、クラス `RestClientConfig` にあります。
- C# - TLS構成とクライアント証明書のロード方法は、クラス `RestClientHelper.cs` にあります。
- PHP - TLS構成とクライアント証明書のロード方法は、ファイル `RestClientConfig.php` にあります。
- Go - TLS構成とクライアント証明書のロード方法は、ファイル `https.go` にあります。

次に、[認証プロセス](#)および[認証シーケンス](#)に基づいたバックエンド実装の詳細を説明します。

処理 1: 認証の初期化

認証を初期化するためには、**3DSリクエスター**が以下のように動作する必要があります。

- **認証の初期化** リクエストを**3DS-web-adapter**([Step. 2](#))から受信する。
- **ActiveServer**にリクエストを送信する ([Step. 3](#))。
- **ActiveServer**([Step. 4](#))から応答を受信する。
- 受信したデータを**3DS-web-adapter** ([Step. 5](#))に返答する。

Java C# PHP Go

```
//AuthControllerV2.java
/**
 * 認証の初期化要求を3DS-web-adapterから受信する。(ステップ 2)
 * ActiveServerに認証を初期化するためのデータを送信する。
 */
@PostMapping("/v2/auth/init")
public Message initAuth(@RequestBody Message request,
    @PathVariable(value = "messageCategory") String messageCategory) {


    // リクエスターのトランザクションIDを生成する。
    String transId = UUID.randomUUID().toString();
    request.put("threeDSRequestorTransID", transId);
    // イベントコールバックURLを「リクエスターURL + /3ds-notify」に設定する。
    String callBackUrl = config.getBaseUrl() + "/3ds-notify";
    request.put("eventCallBackUrl", callBackUrl);

    // 認証を初期化するためのActiveServerのURL
    String initAuthUrl = config.getAsAuthUrl() + "/api/v2/auth/brw/init/" +
messageCategory;
    logger.info("initAuthRequest on url: {}, body: \n{}", initAuthUrl, request);

    // ActiveServerに認証を初期化するためのデータを送信する。(ステップ 3)
    // ActiveServerから応答を受け取る (ステップ 4)
    Message response =
        sendRequest(initAuthUrl, request, HttpMethod.POST);
    logger.info("initAuthResponseBRW: \n{}", response);

    if (response != null) {
        //save initAuth response into session storage
        session.setAttribute((String) response.get("threeDSSEServerTransID"),
            response);
    } else {
        logger.error("Error in initAuth response");
    }

    // 3ds-web-adapterデータを返答する。(ステップ 5)
    return response;
}
```

 注目

eventCallbackUrl を `{baseUrl}/3ds-notify` に設定しています。このURLを設定することで **ActiveServer** からブラウザ情報の収集が完了した時通知を受ける事ができます。(ステップ. 7)。
baseUrl is set [here](#).

initAuth リクエストが `{ActiveServer auth url}/api/v2/auth/brw/init/{messageCategory}` のURLに送信されます。initAuth リクエストのデータ構造をご覧になりたい場合は [認証APIドキュメント](#) を参照してください。

認証APIマスタークライアント証明書用のHTTPヘッダー

[認証APIマスタークライアント証明書](#) を使用して **ビジネス管理者** ユーザーとして加盟店に代わって認証を行う場合、バックエンドは、HTTPヘッダーに `AS-Merchant-Token` フィールドを追加する必要があります。このフィールドは加盟店プロフィールから取得できる [マーチャントトークン](#) を設定します。

Java C# PHP Go

```
//注目: sendRequest()がActiveServerにリクエストを送信します。
//これがgroupAuthの場合、リクエストにはAS-Merchant-Tokenのフィールドを持つHTTPヘッダーが含まれている必要があります。
private Message sendRequest(String url, Message request, HttpMethod method) {

    HttpEntity<Message> req;
    HttpHeaders headers = null;

    if (config.isGroupAuth()) {
        //the certificate is for groupAuth, work out the header.
        headers = new HttpHeaders();
        headers.add("AS-Merchant-Token", config.getMerchantToken());
    }

    switch (method) {
        case POST:
            req = new HttpEntity<>(request, headers);
            return restTemplate.postForObject(url, req, Message.class);
        case GET:
            if (headers == null) {
                return restTemplate.getForObject(url, Message.class);
            } else {
                req = new HttpEntity<>(headers);
                return restTemplate.exchange(url, HttpMethod.GET, req,
Message.class).getBody();
            }
        default:
            return null;
    }
}
}
```

処理2: 認証の実行

認証を実行する為に3DSリクエスターが必要な事は:

- ステップ 7の後、ActiveServerからの /3ds-notify メッセージを処理する。
- 3DS-web-adapter からの 認証の実行 リクエストを処理する (ステップ 9とステップ 10)。
- 認証結果を受信し、それを3DS-web-adapterへ返答する (ステップ 12とステップ 13)。

ブラウザ情報の収集 (ステップ7)が完了した時、**ActiveServer** はあなたが `eventCallbackUrl` に設定した `http://localhost:8082/3ds-notify` に通知を行います。**3DSリクエスター**はこの通知を処理し必要なパラメータを `notify-3ds-events.html` ページへ渡します。

Java C# PHP Go

```
//MainController.java
@PostMapping("/3ds-notify")
public String notifyResult(
    @RequestParam("requestorTransId") String transId,
    @RequestParam("event") String callbackType,
    @RequestParam(name = "param", required = false) String param,
    Model model) {

    String callbackName;
    if ("3DSMethodFinished".equals(callbackType)) {

        callbackName = "_on3DSMethodFinished";


    } else if ("3DSMethodSkipped".equals(callbackType)) {

        callbackName = "_on3DSMethodSkipped";

    } else if ("AuthResultReady".equals(callbackType)) {
        callbackName = "_onAuthResult";
    } else {
        throw new IllegalArgumentException("invalid callback type");
    }

    model.addAttribute("transId", transId);
    model.addAttribute("callbackName", callbackName);
    model.addAttribute("callbackParam", param);

    return "notify_3ds_events";
}
```

 注目

このハンドラーメソッドは**ActiveServer**により呼び出され、**ActiveServer**により `requestorTransId`、`event`、オプションの `param` が提供されます。 `event` は `3DSMethodFinished`、`3DSMethodSkipped`、もしくは `AuthResultReady` になります。そしてハンドラーメソッドはページコンテキスト内の適切な属性値を設定し `notify_3ds_events.html` ページへ返します。そして **Mustache** テンプレートエンジンを使用してページにレンダリングします。

フロントエンドの `notify_3ds_events.html` の実装を確認したい場合は [こちら](#) から確認できます。

そして3DSクライアントがブラウザ情報の収集を終えた時、`auth` エンドポイントを呼び出し認証を開始します。**3DSリクエスター**は **認証の実行** リクエストを処理します。(ステップ 9、ステップ 10、ステップ 12とステップ 13)。

Java C# PHP Go

```
//AuthControllerV2.java
/**
 * 認証の実行要求を3DS-web-adapterから受信する。(ステップ 9)
 * 受信したデータをActiveServerに送信し認証を実行する。
 */
@PostMapping("/v2/auth")
public Message auth(@RequestBody Message request) {

    // 認証を実行するためのActiveServerのURL
    String authUrl = config.getAsAuthUrl() + "/api/v2/auth/brw";
    logger.info("requesting BRW Auth API {}, body: \n{}", authUrl, request);

    // データをActiveServerに送信し認証を実行する。(ステップ 10)
    // 応答データをActiveServerから受信する。(ステップ 12)
    Message response =
        sendRequest(authUrl, request, HttpMethod.POST);
    logger.info("authResponseBRW: \n{}", response);

    //3ds-web-adapterデータを返答する。(ステップ 13)
    return response;
}
```

注目

3DSリクエスターは、フロントエンドへ返答します。返されるメッセージには、フリクションレスフローまたはチャレンジフローのいずれかによって **Y** または **C** の `transStatus` 値を含みます。フロントエンドがどのように `transStatus` を処理するかを確認するには、[こちら](#)を参照してください。応答データの構造を確認するには、[APIドキュメント](#)を参照してください。

チャレンジフローをキャンセル

`transStatus = C` の場合、3DSクライアントはチャレンジを開始するか否かを選択できます。3DSクライアントがチャレンジをキャンセルすることを選択した場合、`/auth/challenge/status` エンドポイントを呼び出して **キャンセルした理由** を指定できます。**3DSリクエスター**が **ActiveServer** に送信します。フロントエンドがこのリクエストを処理する方法を確認するには、[こちら](#)を参照してください。

Java C# PHP Go

```
//AuthControllerV2.java
@PostMapping("/auth/challenge/status")
public Message challengeStatus(@RequestBody Message request) {

    String challengeStatusUrl = config.getAsAuthUrl() + "/api/v2/auth/challenge/status";
    logger.info("request challenge status API {}", body: "\n{}", challengeStatusUrl, request);

    Message response =
        sendRequest(challengeStatusUrl, request, HttpMethod.POST);
    logger.info("challengeStatus response: \n{}", response);

    return response;
}
```

処理 3: 認証結果の取得

認証結果を取得する為に**3DSリクエスター**が必要な事は:

- **3DS-web-adapter**からの **認証結果の取得** リクエストを処理する (**ステップ 15(F)**もしくは**ステップ 20(C)**) 。

- 結果を取得しフロントエンドへ返す為に**ActiveServer**へリクエストを送信する (ステップ 16(F)、ステップ 17(F) もしくは ステップ 21(C)、22(C))。

Java C# PHP Go

```
//AuthControllerV2.java
/**
 * 認証結果の取得要求を受信する。(ステップ15(F)とステップ20(C))
 * ActiveServerから認証結果を取得する。
 */
@GetMapping("/v2/auth/result")
public Message result(@RequestParam("txid") String serverTransId) {

    // ActiveServerからURLを取得する
    String resultUrl = config.getAsAuthUrl() +
        "/api/v2/auth/brw/result?threeDSServerTransID=" +
        serverTransId;

    // ActiveServerから認証結果を取得する。(ステップ16(F)とステップ21(C))
    Message response = sendRequest(resultUrl, null, HttpMethod.GET);
    logger.info("authResponse: \n{}", response);

    //認証結果をresult.htmlに表示する。(ステップ17(F)とステップ22(C))
    return response;
}
```

✓ 成功

この文書ではバックエンド実装についての説明は以上で終了です。認証が完了した後、チェックアウト処理は取引ステータス、ECIとCAVVを使用して承認の実行を続け、取引を完了出来ます。

🔗 次のチャプター

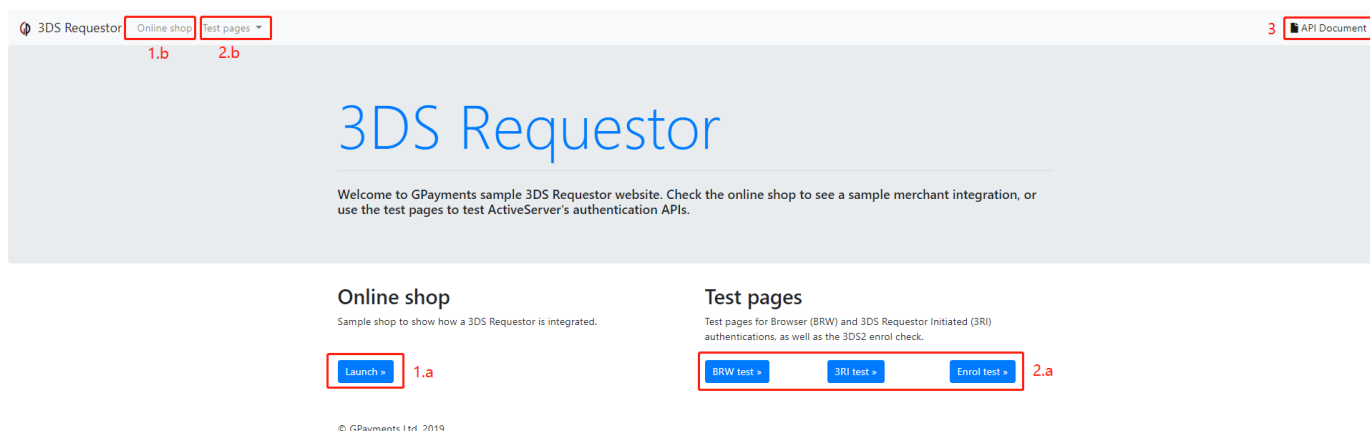
次へボタンを選択しGPayments 3DSリクエスターの為のサンプルコードのすべての機能の説明をご覧ください。

サンプルコード機能

このセクションでは、**3DSリクエスターデモプロジェクト**のすべての機能を紹介します。弊社の3DSリクエスターにアクセスするには:

- ・ [こちら](#)から3DSリクエスターデモのコードをダウンロードして実行するか、
- ・ 弊社のオンラインの[TestLabsリクエスター](#)を使用してください。

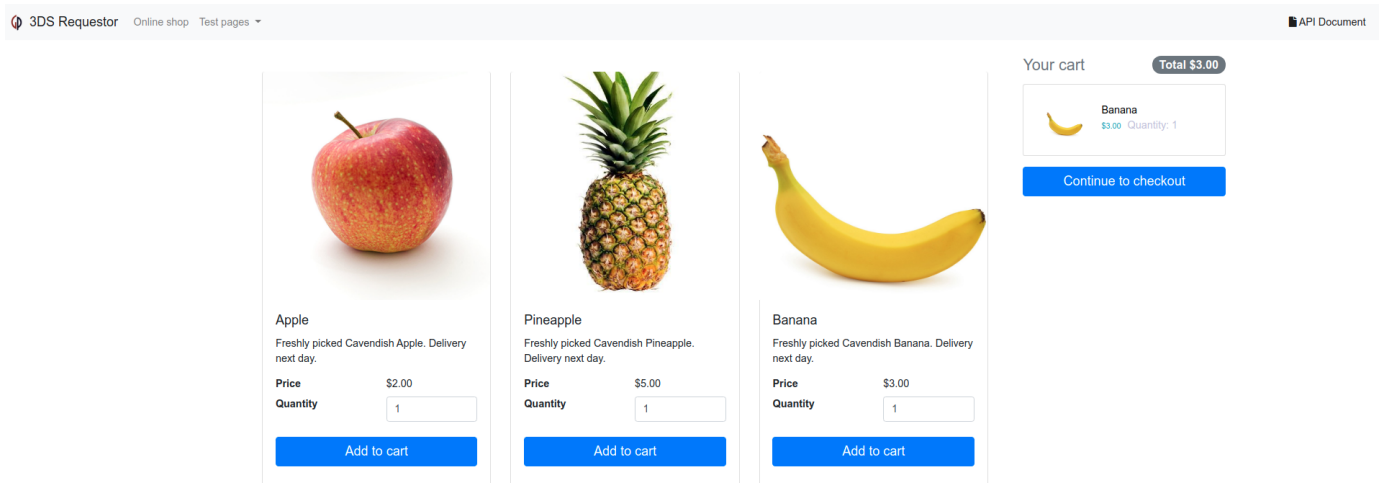
以下に表示されているのは弊社の3DSリクエスターサイトのホームページです。 **オンラインショップ**と**テストページ**の2つのセクションがあります。オンラインショップは **Launch** ボタン **(1.a)**を選択する、もしくは左上の **Online shop** ボタン **(1.b)**を選択する事でアクセス出来ます。それぞれのテストページは関連ボタン **(2.a)**を選択する、もしくは左上の **Test pages** ドロップダウン **(2.b)**を選択する事でアクセス出来ます。加えて、右上に**ActiveServer認証APIドキュメント**への **API Document** のリンク **(3)**があります。



オンラインショップ

オンラインショップは、どのように3DS2を電子取引サイトに統合するかを加盟店視点で提供する、デモの加盟店のチェックアウトページです。統合の実装については、[統合ガイド](#)に従ってください。

商品をショップページ内のカートに追加してください:

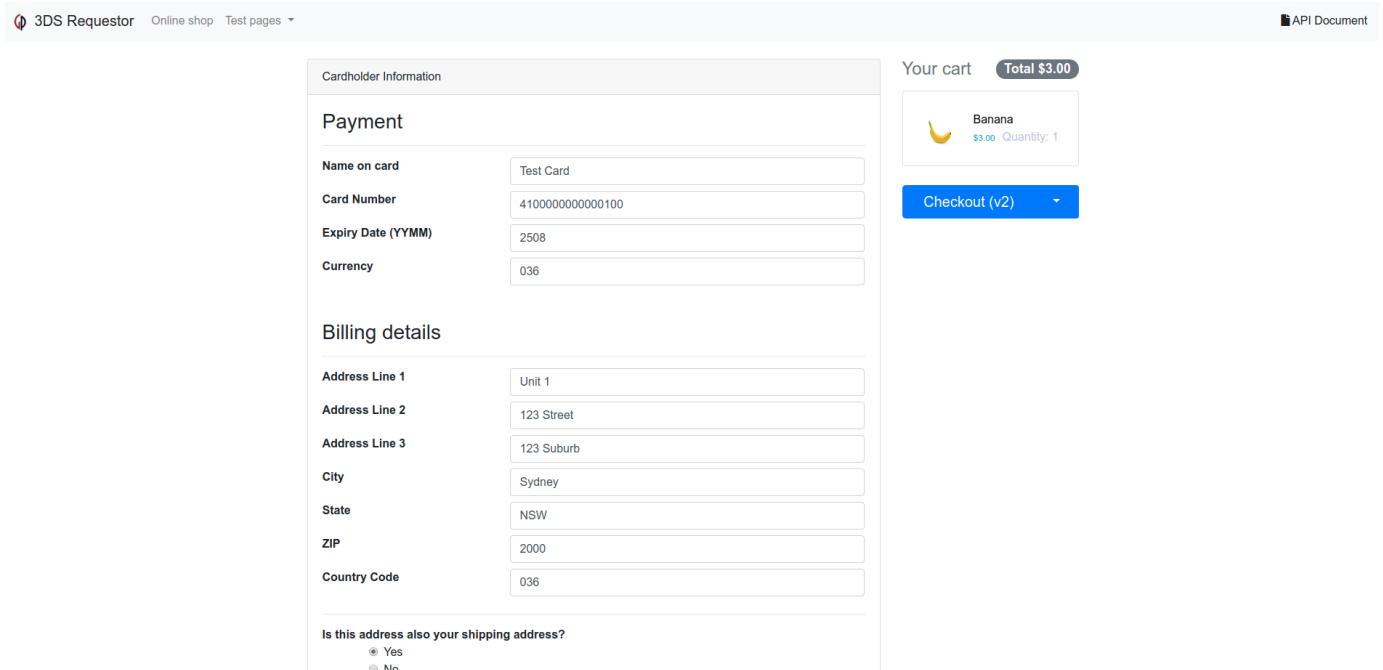


Continue to Checkoutボタンを選択しチェックアウトページへ移動してください:

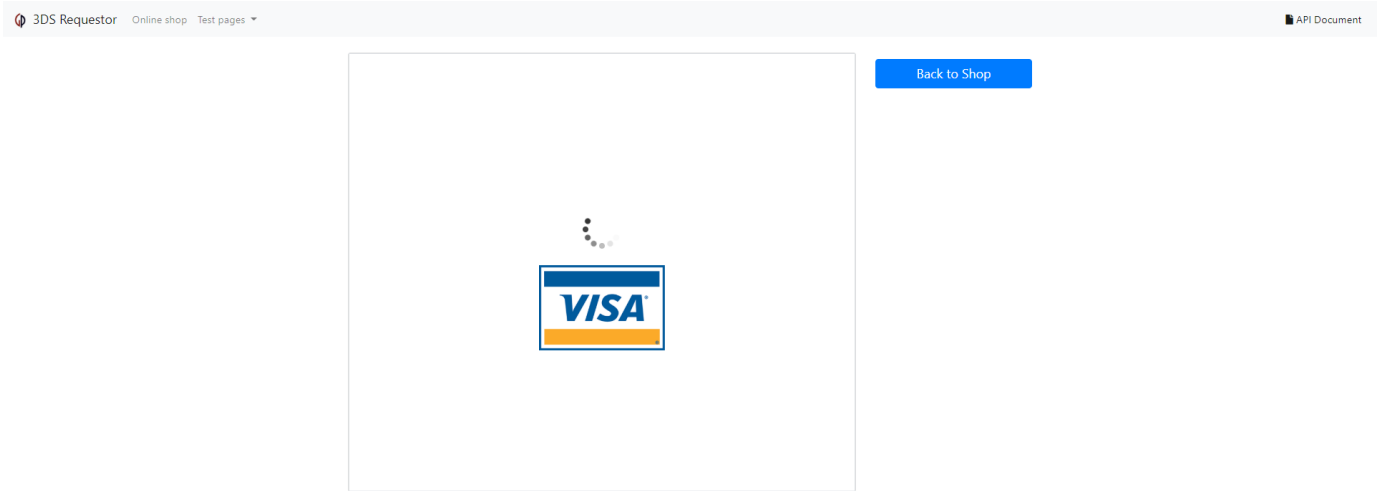
取引を完了するのに使用されるカード番号を含む、標準の支払いと請求の情報が予め入力されています。**Continue to Checkout**ボタンを選択して3DS2認証プロセスを開始してください:

注目

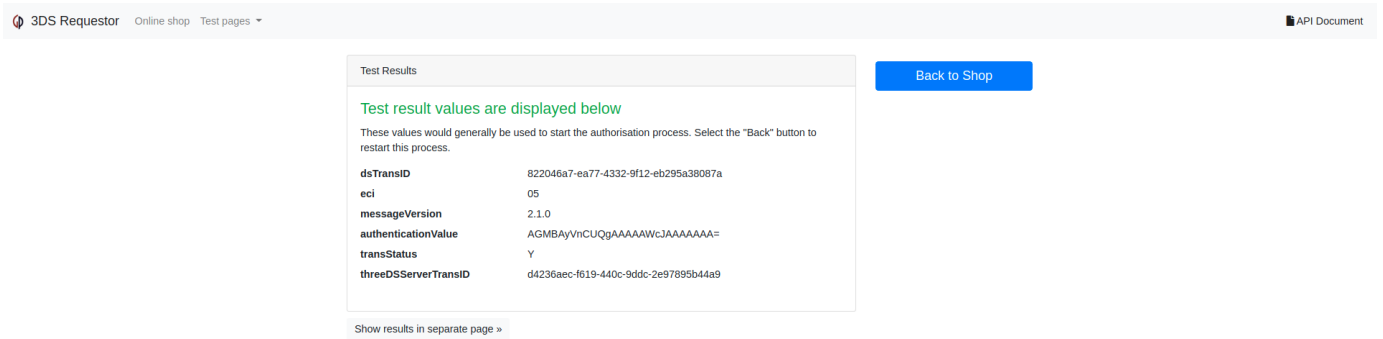
Checkoutボタンの右側にある矢印をクリックすると、3DS2認証プロセスを実行するAPIバージョンを選択できます。



3DSリクエスターが3DS2認証を実行している間、処理ページにスピナーが表示されます。3DS2認証が終了すると、結果が表示されます。



Show results in separate pagesを選択すると、新しいページが開き結果が表示されます。



テストページ

テストページでブラウザベース認証 (BRW)、3DS Requestor Initiated (3RI)、EnrolAPIドキュメントに定義されたすべてのパラメータで試験を実施できます。

BRWテストページ

基礎情報

BRWテストページには Basic Info(基礎情報)、Cardholder(カード会員)、Additional Risk (追加リスク) の3つのタブがあります。

Basic Info Cardholder Additional Risk Test Options

Channel

Auth API URL *

Requestor URL *

Message Category * Payment Auth Non-payment Auth

Required Field

Provider: **Scenario:**

Account Number *

Merchant ID *

Authentication Indicator *

Additional Field

Purchase Amount

Purchase Currency

Account ID

Account Type

Basic Info(基礎情報) タブには Channel(チャンネル)、Required Field(必須項目)、Additional Field(追加項目) の3つのセクションがあります。

Basic Info	Cardholder	Additional Risk
------------	------------	-----------------

Channel

Auth API URL *

Requestor URL *

Message Category * Payment Auth Non-payment Auth

Required Field

Provider: **Scenario:**

Account Number *

Merchant ID *

Authentication Indicator *

Additional Field

Purchase Amount

Purchase Currency

Account ID

Account Type

Channel (チャンネル) では、アプリケーションに読み込まれている3DS Server URLとRequestor URLを確認する事が出来ます。さらに、pa もしくは npa のどちらのMessage Categoryがテストに使用されるかを選択出来ます。

Required Field (必須項目) では、カード番号、マーチャントID、認証インジケータを入力する必要があります。これらは3DS2ブラウザベース認証に必要なパラメータです。

カード会員

Additional Field (追加項目) では、**購入数量**、**金額**、**通貨**、**有効期限**といった追加の情報を入力する事が出来ます。

GPayments TestLabsシナリオの使い方

カード番号 は、自分のカード番号を入力するか、もしくは予め設定された弊社のシナリオから選ぶ事が出来ます。例えば、VisaのFrictionless authentication、そしてカード番号4100000000000001がカード番号フィールドに自動的に入力されます。

Cardholder(**カード会員**) タブでは、**名前**、**住所**、**Eメール**、**電話番号**を含むカード会員の情報を入力する事が出来ます。

Basic Info	Cardholder	Additional Risk	Test Options
------------	------------	-----------------	--------------

Card details

Card Holder Name	Test Card	
Email	abc@123.com	
Mobile Number	61	0421522329
Home Phone		
Work Phone		

Billing details

Address Line 1	Unit 1	
Address Line 2	123 Street	
Address Line 3		
City	Sydney	
State	NSW	
ZIP	2000	
Country Code	036	

Is this address also your shipping address?

Yes

No

追加リスク

Additional Risk (追加リスク) タブでは、アカウント情報、リクエスターの認証情報、加盟店リスク情報のような追加のリスク情報を入力する事が出来ます。

加盟店リスク情報

この情報の提供は必須ではありません。しかし、イシューアのACSは提供された情報が多いほどリスクベース認証の精度は上がり、フリクションレスのレートが高くなる可能性があります。なので、加盟店がこの情報を提供することは非常に重要です。

Basic Info

Cardholder

Additional Risk

Test Options

Account Information

Cardholder Account Age Indicator	<input type="text"/>
Cardholder Account Change	<input type="text"/>
Cardholder Account Change Indicator	<input type="text"/>
Cardholder Account Date	<input type="text"/>
Cardholder Account Password Change	<input type="text"/>
Cardholder Account Password Change Indicator	<input type="text"/>
Number of Purchase Account	<input type="text"/>
Payment Account Age	<input type="text"/>
Payment Account Indicator	<input type="text"/>
Provision Attempts Day	<input type="text"/>
Ship Address Usage	<input type="text"/>
Ship Address Usage Indicator	<input type="text"/>
Ship Name Indicator	<input type="text"/>
Suspicious Account Activity	<input type="text"/>
Transaction Activity Day	<input type="text"/>
Transaction Activity Year	<input type="text"/>

テストオプション

Test [Options](#) タブでは、Cancel Challenge チェックボックスをオンにすることで、ACSが取引に対してチャレンジを要求した場合にチャレンジをキャンセルすることを選択できます。チャレンジをキャンセルすると、`3ds-web-adapter` は `iframe` でCReqコールバックページを実

行しません。必要に応じて、challengeStatus のエンドポイントを使用して、Cancel Reason (キャンセルの理由) を指定し、ActiveServer にチャレンジをキャンセルした理由を通知します。

- **CReq Not Sent (CReqが送信されなかった)** - 3DSリクエスターがチャレンジをオプトアウトすることを選択したため、チャレンジリクエストが開始されなかったことを示します。CReqNotSent のステータスを challengeStatus に送信します。
- **Auth Result Not Delivered (認証結果が届かなかった)** - チャレンジリクエストを3DSリクエスターに配信できなかったことを示します。例：3DSリクエスターが認証リクエストを実行するとき (/api/v1/auth/brw/ 等) にActiveServer が応答しないなどの技術的なエラー等の理由。AuthResultNotDelivered のステータスを challengeStatus に送信します。
- **No Reason Sent (理由が送信されなかった)** - 3DSリクエスターがチャレンジを開始せず、challengeStatus を呼び出さないシナリオをシミュレートします。キャンセル理由はActiveServer に送信されません。

Basic Info Cardholder Additional Risk **Test Options**

API Options

Api Version *

Challenge Options

3DS requestor can decide whether to proceed the challenge or not. By select the following checkbox, the requestor will cancel the challenge.

Cancel Challenge

Cancel Reason

3RIテストページ

BRWテストページと同様に、3RIテストページにも Basic Info(基礎情報)、Cardholder(カード会員)、Additional Risk (追加リスク) の3つのカードがあります。1つの違いは 3RI Indicator パラメータが Required Field (必須項目) 内にあり、

Additional Field (追加項目) 内に無い事です。その他の違いとして、3DS Requestor Initiated は非決済認証のみに使用される為、Message Category を選択する事が出来ません。

Basic Info Cardholder Additional Risk

Channel

Auth API URL *

Requestor URL *

Message Category * Payment Auth Non-payment Auth

Required Field

Account Number *

Merchant ID *

3RI Indicator *

Additional Field

Account ID

Account Type

Expiry Date (YYMM)

Overridden Merchant Name

Enrolテストページ

カード番号とマーチャントIDのみがEnrolテストでは必要です。

Enrol Information

Channel

Auth API URL *

Requestor URL *

Required Field

Account Number *

Merchant ID *

テスト結果

テストページのどれか1つで **TestXXX** ボタンを選択すると、`process.html` 移動します。すべての3DS2プロセスは `process.html` ページで処理されます。3DS2プロセスが実行中の間はスピナーが表示されます。3DS2プロセスが完了すると、同じページ内に結果が表示されます。

Test Results

[Back to Shop](#)

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

dsTransID	822046a7-ea77-4332-9f12-eb295a38087a
eci	05
messageVersion	2.1.0
authenticationValue	AGMBAyVnCUQgAAAAAwcJAAAAAA=
transStatus	Y
threeDSserverTransID	d4236aec-f619-440c-9ddc-2e97895b44a9

[Show results in separate page »](#)

例えば、`BRW test` ページを選択し、そして `Test BRW` ボタンを予め入力された情報で選択すると、成功した応答を取得します。

Test Results

Test result values are displayed below

These values would generally be used to start the authorisation process. Select the "Back" button to restart this process.

errorCode	1026
errorDetail	MerchantId does not match.
errorDescription	MerchantId does not match.
errorComponent	S
errorMessageType	AReq
messageType	Erro

もしくは、不正なパラメータを BRW test ページ内で入力すると（例えば Merchant ID に 000）、エラーを返します。

ディレクトリ構造

以下はJava、C#、PHPとGoのディレクトリツリーです。

Java

21 ディレクトリ, 52 ファイル

```
.
├── mvnw
├── mvnw.cmd
├── pom.xml
├── src
│   ├── main
│   │   ├── java
│   │   │   ├── com
│   │   │   │   ├── gpayments
│   │   │   │   │   ├── requestor
│   │   │   │   │   │   ├── testlab
│   │   │   │   │   │   │   ├── AuthControllerV1.java
│   │   │   │   │   │   │   ├── AuthControllerV2.java
│   │   │   │   │   │   │   ├── config
│   │   │   │   │   │   │   │   ├── Config.java
│   │   │   │   │   │   │   │   └── RestClientConfig.java
│   │   │   │   │   │   │   ├── dto
│   │   │   │   │   │   │   │   └── Message.java
│   │   │   │   │   │   │   ├── exception
│   │   │   │   │   │   │   │   └── GlobalExceptionHandler.java
│   │   │   │   │   │   │   ├── MainController.java
│   │   │   │   │   │   └── RequestorApplication.java
│   │   └── resources
│   │       ├── application.yml
│   │       ├── certs
│   │       │   └── cacerts.pem
│   │       ├── static
│   │       │   ├── css
│   │       │   │   ├── cart.css
│   │       │   │   ├── spinner.css
│   │       │   │   └── style.css
│   │       │   ├── favicon.ico
│   │       │   ├── images
│   │       │   │   ├── amex-logo.png
│   │       │   │   ├── apple.jpeg
│   │       │   │   ├── banana.jpg
│   │       │   │   ├── discover-logo.png
│   │       │   │   ├── jcb-logo.png
│   │       │   │   ├── left-icon.ico
│   │       │   │   ├── mastercard-logo.png
│   │       │   │   ├── pineapple.jpeg
│   │       │   │   └── visa-logo.png
│   │       │   └── js
│   │       │       ├── cart.js
│   │       │       ├── check-credit-card-type.js
│   │       │       ├── common.js
│   │       │       ├── test-lab-scenarios.js
│   │       │       ├── v1
│   │       │       │   └── 3ds-web-adapter.js
│   │       │       └── v2
```

```
├── 3ds-web-adapter.js
├── templates
│   ├── 3ri.html
│   ├── brw.html
│   ├── checkout.html
│   ├── contents
│   │   ├── acct_info.html
│   │   ├── authentication_info.html
│   │   ├── cardholder_info.html
│   │   ├── deps.html
│   │   ├── merchant_risk_indicator.html
│   │   ├── nav_bar.html
│   │   ├── process_head.html
│   │   └── process_main_body.html
│   ├── enrol.html
│   ├── error.html
│   ├── index.html
│   ├── notify_3ds_events.html
│   ├── shop.html
│   ├── v1
│   │   ├── process.html
│   │   └── result.html
│   └── v2
│       ├── process.html
│       └── result.html
```

C#

17 ディレクトリ, 61 ファイル

```
.
├── 3ds2RequestorDemo.csproj
├── 3ds2RequestorDemo.sln
├── App_Start
│   ├── FilterConfig.cs
│   ├── RouteConfig.cs
│   └── WebApiConfig.cs
├── Certs
│   └── cacerts.pem
├── Controllers
│   ├── AuthV1Controller.cs
│   ├── AuthV2Controller.cs
│   ├── EnrollController.cs
│   └── MainController.cs
├── css
│   ├── cart.css
│   ├── spinner.css
│   └── style.css
├── favicon.ico
├── Global.asax
├── Global.asax.cs
├── Helpers
│   ├── Config.cs
│   └── RestClientHelper.cs
├── images
│   ├── amex-logo.png
│   ├── apple.jpeg
│   ├── banana.jpg
│   ├── cart.png
│   ├── discover-logo.png
│   ├── jcb-logo.png
│   ├── left-icon.ico
│   ├── mastercard-logo.png
│   ├── pineapple.jpeg
│   └── visa-logo.png
├── js
│   ├── cart.js
│   ├── check-credit-card-type.js
│   ├── common.js
│   ├── test-lab-scenarios.js
│   ├── v1
│   │   └── 3ds-web-adapter.js
│   ├── v2
│   │   └── 3ds-web-adapter.js
├── Models
│   └── dto
│       └── Message.cs
├── packages.config
├── Properties
│   └── AssemblyInfo.cs
```

```
|— startup.bat
|— Views
|   |— Main
|   |   |— 3ri.html
|   |   |— brw.html
|   |   |— checkout.html
|   |   |— contents
|   |   |   |— acct_info.html
|   |   |   |— authentication_info.html
|   |   |   |— cardholder_info.html
|   |   |   |— deps.html
|   |   |   |— merchant_risk_indicator.html
|   |   |   |— nav_bar.html
|   |   |   |— process_head.html
|   |   |   |— process_main_body.html
|   |   |— enrol.html
|   |   |— error.html
|   |   |— index.html
|   |   |— notify_3ds_events.html
|   |   |— v1
|   |   |   |— process.html
|   |   |   |— result.html
|   |   |— v2
|   |   |   |— process.html
|   |   |   |— result.html
|   |— Shared
|   |   |— Error.cshtml
|   |— _ViewStart.cshtml
|   |— Web.config
|— Web.config
```

PHP

14 ディレクトリ, 53 ファイル

```
.
├── composer.json
├── composer.lock
├── config
│   ├── Config.php
│   ├── RestClientConfig.php
│   ├── Router.php
│   └── Utils.php
├── controllers
│   ├── AuthControllerV1.php
│   ├── AuthControllerV2.php
│   └── MainController.php
├── css
│   ├── cart.css
│   ├── spinner.css
│   └── style.css
├── favicon.ico
├── images
│   ├── amex-logo.png
│   ├── apple.jpeg
│   ├── banana.jpg
│   ├── cart.png
│   ├── discover-logo.png
│   ├── jcb-logo.png
│   ├── left-icon.ico
│   ├── mastercard-logo.png
│   ├── pineapple.jpeg
│   └── visa-logo.png
├── index.php
├── js
│   ├── cart.js
│   ├── check-credit-card-type.js
│   ├── common.js
│   ├── test-lab-scenarios.js
│   ├── v1
│   │   └── 3ds-web-adapter.js
│   ├── v2
│   │   └── 3ds-web-adapter.js
├── README.md
├── resources
│   ├── application.ini
│   ├── certs
│   │   └── cacerts.pem
│   └── templates
│       ├── 3ri.html
│       ├── brw.html
│       ├── checkout.html
│       ├── contents
│       │   ├── acct_info.html
│       │   └── authentication_info.html
```


Go

11 ディレクトリ, 47 ファイル

```
.
├── conf
│   ├── application.yaml
│   └── cacerts.pem
├── conf.go
├── go.mod
├── go.sum
├── https.go
├── main.go
└── web
    ├── 3ri.html
    ├── brw.html
    ├── checkout.html
    ├── contents
    │   ├── acct_info.html
    │   ├── authentication_info.html
    │   ├── cardholder_info.html
    │   ├── deps.html
    │   ├── merchant_risk_indicator.html
    │   ├── nav_bar.html
    │   ├── process_head.html
    │   └── process_main_body.html
    ├── css
    │   ├── cart.css
    │   ├── spinner.css
    │   └── style.css
    ├── enrol.html
    ├── error.html
    ├── favicon.ico
    ├── images
    │   ├── amex-logo.png
    │   ├── apple.jpeg
    │   ├── banana.jpg
    │   ├── cart.png
    │   ├── discover-logo.png
    │   ├── jcb-logo.png
    │   ├── left-icon.ico
    │   ├── mastercard-logo.png
    │   ├── pineapple.jpeg
    │   └── visa-logo.png
    ├── index.html
    ├── js
    │   ├── cart.js
    │   ├── check-credit-card-type.js
    │   ├── common.js
    │   ├── test-lab-scenarios.js
    │   ├── v1
    │   │   ├── 3ds-web-adapter.js
    │   │   └── v2
    │   │       └── 3ds-web-adapter.js
```

```
|— notify_3ds_events.html
|— shop.html
|— v1
|   |— process.html
|   |— result.html
|— v2
|   |— process.html
|   |— result.html
```

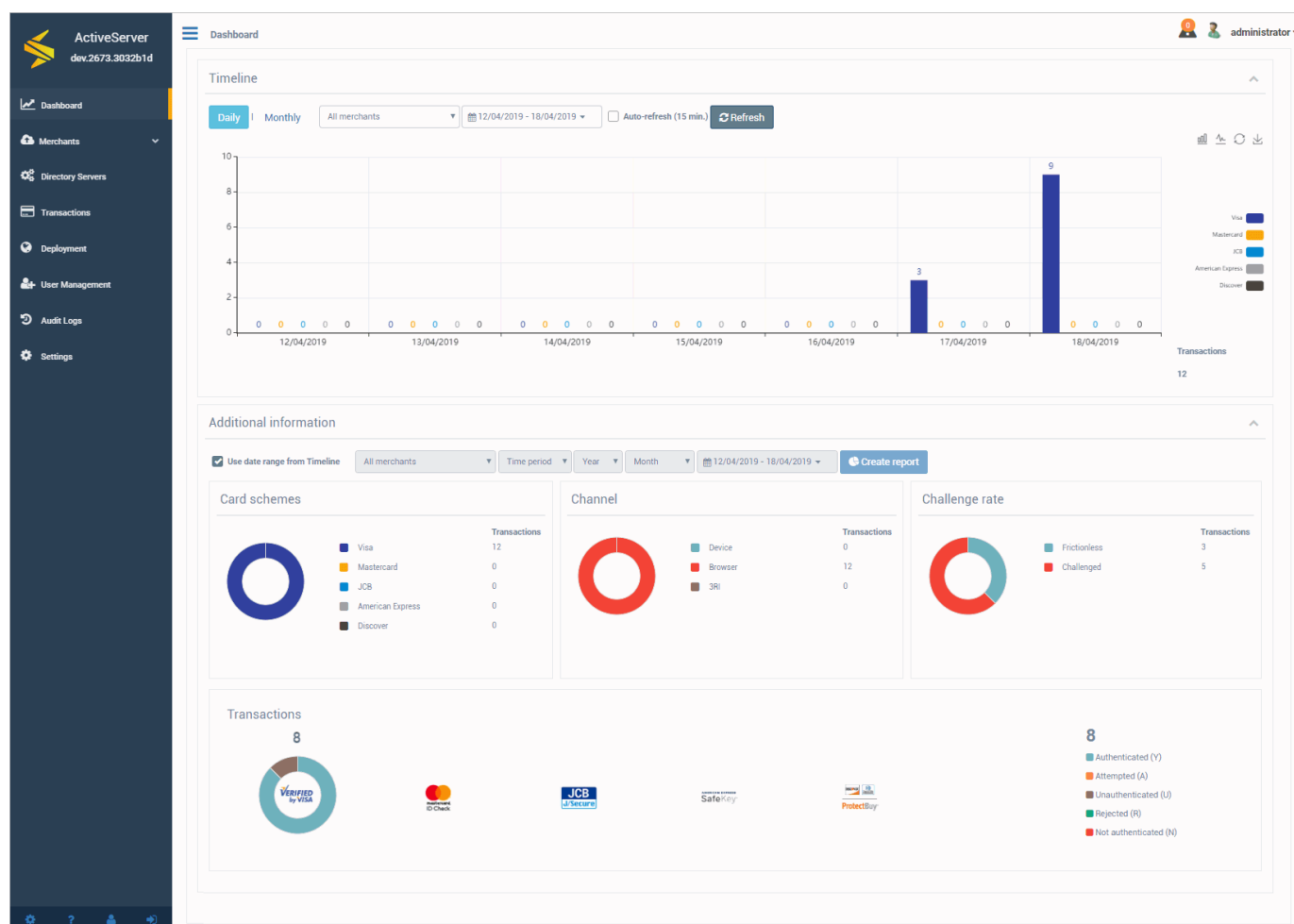
ダッシュボード使い方

Dashboardには2つのセクションがあります：

- ・ タイムライン
- ・ 追加の情報

Tip

各ダッシュボード・セクションは、セクションの見出しをクリックすることで折りたたんだり展開したりできます。

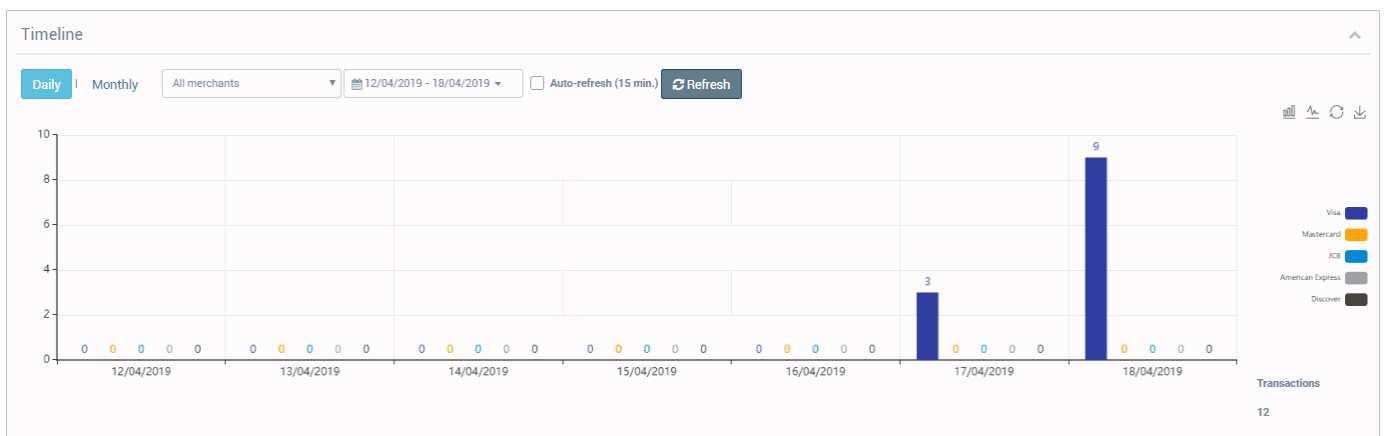


タイムライン

Timelineには、指定された期間中に実行された取引の数の履歴的でグラフィカルな内訳が表示されており、これは国際ブランド別に分割されています。各国際ブランドビューは、グラフの右側のアイコンをクリックすることでオフにできます。グラフは、折れ線グラフと棒グラフの両方の形式で利用できます。これは、グラフの右上の関連するボタンを使用して切り替えることができます。

インターフェイスは、**Auto-refresh**オプションを選択することで、15分ごとに更新するように設定できます。**Refresh**ボタンは、画面上のデータを最新の収集データに更新するのに使用できます。

データは、**Daily**または**Monthly**ビューで表示できます。これは列の値が何を表しているかを示します。



日毎

Dailyビューを選択すると、グラフの各列はDD/MM/YYYY形式で1カレンダー日を表します。**Daily**ビューには、すぐに確認できるように、事前に選択可能な日付範囲が含まれています。

- **Last 7 days** - 当日を含む過去7カレンダー日。
- **Last 30 days** - 当日を含む過去30カレンダー日。
- **Current month** - 当日を含む、現在のカレンダー月のすべての日。
- **Last month** - 前のカレンダー月のすべての日。

日付範囲ピッカーを使用して、**カスタム日付範囲**を選択することもできます。このカスタム日付範囲は、最低1日、最大31日とすることができます。

月毎

Monthlyビューを選択すると、グラフの各列は**MM/YYYY**形式で1カレンダー月を表します。**Monthly**ビューには、すぐに確認できるように、事前に選択可能な日付範囲が含まれています。

- **Current month** - 当日を含む現在のカレンダー月。
- **Current year** - 当日を含む現在のカレンダー一年。
- **Last month** - 前のカレンダー月。
- **Last year** - 前のカレンダー一年。

日付範囲ピッカーを使用して、**カスタム日付範囲**を選択することもできます。このカスタム日付範囲は、最低1か月、最大12か月とすることができます。

追加の情報

ダッシュボードの**Additional information**セクションでは、**ActiveServer**で実行される各取引の最終ステータスのより詳細な情報が表示されます。この情報は、**Timeline**セクションに表示される情報の補足としたり、システムの過去の日付範囲の取引統計を表示したりするのに使用できます。

期間

Use date range from Timelineを選択すると、日付範囲および加盟店選択オプションがグレーアウトし、**Timeline**セクションで選択されたオプションが使用され、両方のセクションで同期されたビューが表示されます。

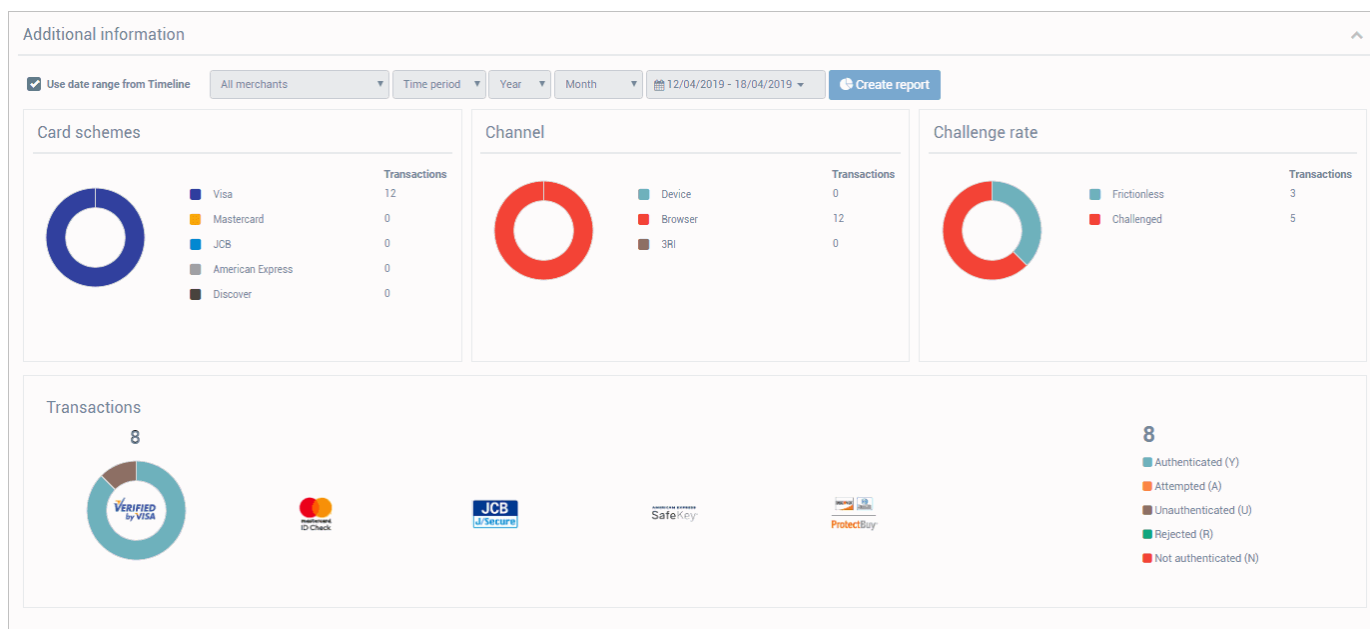
Use date range from Timelineが選択されていない場合、ユーザーは表示する必要がある加盟店統計を指定できます。これは以下の期間で表示されます。

- **Year** - カレンダー一年の期間。
- **Month** - 特定の年のカレンダー一月の期間。
- **Custom** - カスタム期間の特定の日付範囲。プリセットオプションから選択されます：
 - **Current month** - 当日を含む現在のカレンダー月。
 - **Current year** - 当日を含む現在のカレンダー一年。
 - **Last month** - 前のカレンダー月。

- Last year - 前のカレンダー一年。
- Custom - カスタム期間

グラフ

各グラフは、取引のデータの異なるサブセクションを表示します。



国際ブランド

Card schemesセクションでは、Transaction 列の国際ブランドあたりの合計取引数の数値的な内訳が表示されます。

グラフにポインタを合わせると、国際ブランドあたりの合計取引数の割合的な内訳が表示されます。

チャネル

Channelセクションでは、使用されるチャネル値の合計取引数の数値的な内訳が表示されます。加盟店が取引のほとんどを実行するプラットフォームに関する情報が示されます。Browser エントリは、Webベースのチェックアウトプロセスを使用してユーザーが認証されたことを示します。Device エントリは、チェックアウトプロセス中にネイティブ・モバイルアプリを使用してユーザーが認証されたことを示します。3RI エントリは、加盟店が3DS Requestor Initiated 認証を実行したことを示します（例：アカウントの有効性の確認や取引の分離など）。

取引の合計数は、**Transactions**列に表示されます。グラフにポインタを合わせると、使用されるチャンネルの割合的な内訳が表示されます。

チャレンジの割合

Challenge rateセクションでは、取引のチャレンジステータスに基づいて、合計取引数の数値的な内訳が表示されます。 **Frictionless** エントリは、リスクベース認証 (RBA) を使用してカード会員がイシューアのACSによって認証でき、ステップアップ・チャレンジが不要だったことを示します。 **Challenged** エントリは、ACSがカード会員に自分自身を認証するようリクエストしたことを示します。これは、時間経過に対してフリクションレス・フロー機能の進捗状況を監視するのに役立ちます。

取引

Transactionsセクションでは、取引と認証ステータスの数値的な内訳が表示されます。右側の列には、最終ステータスレスポンスに分割された合計取引数が表示されます。個々のグラフには、国際ブランドあたりの取引ステータスの割合的な内訳が表示されます。認証ステータスは以下のように説明されます。 **Transactions**セクションでは、取引と認証ステータスの数値的な内訳が表示されます。右側の列には、最終ステータスレスポンスに分割された合計認証数が表示されます。個々のグラフには、国際ブランドあたりの取引ステータスの割合的な内訳が表示されます。認証ステータスは以下のように説明されます。

- **Authenticated (Y)** - 認証確認が成功しました。
- **Attempted (A)** - 試行の処理が実行されました。認証/検証されませんでしたでしたが、認証/検証が試行されたことの証明が提供されます。
- **Unauthenticated (U)** - 認証/アカウント確認を実行できませんでした。技術的またはその他の問題。
- **Rejected (R)** - 認証/アカウント確認が拒否されました。イシューアは取引/検証を拒否し、オーソリゼーションを試行しないように要求しています。
- **Not Authenticated (N)** - 認証/アカウントが確認されませんでした。取引が拒否されました。

加盟店を検索

加盟店のリストは、管理インターフェイスの**Merchants**メニューからアクセスできます。

The screenshot shows the 'Merchants > Merchants' page. At the top right, the user is identified as 'administrator'. Below the breadcrumb, there is a search bar with three input fields: 'Merchant name', 'Merchant ID', and 'Acquirer BIN'. A 'Click to Search' label points to the search button. Below the search bar, there is a 'Display 10 Records' dropdown and a 'Showing 1 to 3 of 3 Merchants.' indicator. The main content is a table with columns: 'Merchant name', 'Merchant ID', 'Acquirer', and 'Status'. The table contains three rows of test data. A 'Click to view Merchant details' label points to the first row. At the bottom of the table, there are 'Delete' and 'New' buttons.

Merchant name	Merchant ID	Acquirer	Status
Test Merchant	123456789012345		ENABLED
Test Merchant Three	123456789036789		ENABLED
Test Merchant Two	123456789054321		ENABLED

デフォルトでは、ユーザーが割り当てられているすべての加盟店が加盟店リストに表示されます。以下のパラメータを使用して、リストをフィルタリングし、特定の加盟店を検索できます。

- **Merchant name** - 加盟店プロフィールで提供される**加盟店の名前**で完全または部分一致検索します。
- **Merchant ID** - 加盟店プロフィールで提供される、アクワイアラー割り当ての**加盟店のID**で完全または部分一致検索します。
- **Acquirer BIN** - 加盟店プロフィールで提供されるいずれかの**アクワイアラーのBIN**で完全または部分一致検索します。

検索結果は、**Merchant name**、**Merchant ID**、**Acquirer BIN**、および**Enabled status**の見出しと共に表に表示されます。詳細を**表示**または**編集**する加盟店を選択します。

ユーザーアクセス

Merchantsページは、**Business Admin**、**Merchant Admin**、または**Merchant**ユーザーなど、加盟店エンティティを管理するユーザーのみがアクセスできます。

 Hint

Merchant Adminまたは**Merchant**ロールを持つユーザーに加盟店が表示されない場合、**User Management > User details**ページで加盟店に割り当てられていることをダブルチェックしてください。

加盟店の管理

加盟店が認証APIを介して認証リクエストを行えるようにするには、加盟店エンティティを作成し、3DSリクエスターのクライアント証明書をダウンロードする必要があります。認証リクエストに含まれていて、頻繁に変更されない詳細は、API機能の単純化のためにデータベースに格納されています。加盟店エンティティの**作成**、**表示**、**編集**、**削除**プロセスは以下のとおりです。

加盟店の作成

加盟店を作成するには、まず管理インターフェイスの**Merchants**ページに移動し、**New**ボタンを選択します。

New merchant画面で、以下のフィールドを使用し、新しい加盟店を作成します。

ユーザーアクセス

ユーザーが加盟店を作成するには、**Business admin**ロールが必要です。

重要

加盟店を作成するとき、または加盟店の詳細を編集するときは、**加盟店名**と**加盟店ID**の組み合わせは一意でなければならないことに注意してください。

詳細

Detailsは認証リクエストに使用される一般的な加盟店の詳細です。項目の説明は以下です。

- **Merchant name** - アクワイアラーによって割り当てられた加盟店の名前。これはオーソリゼーションメッセージ・リクエストで使用されるものと同じである必要があります。最大40文字
- **Merchant ID** - アクワイアラーによって割り当てられた加盟店の識別子。これはオーソリゼーションメッセージ・リクエストで使用されるものと同じである必要があります。最大35文字

- **Country** - 加盟店の運営元の国。認証リクエストの一環として、ActiveServerはこのエンターリーを使用しこれを**Merchant Country Code**に変換します。これはオーソリゼーションメッセージ・リクエストで使用される値と一致している必要があります。
- **Default currency** - 認証リクエストで使用されるデフォルト通貨。この値は、`purchaseCurrency` を指定することで、[ブラウザベースの初期APIコール](#)で上書きできます。
- **3DS Requestor URL** - 3DSリクエスターWebサイトまたは顧客ケアサイトの完全修飾URL。このデータ要素は、問題が発生した場合に受信した3-Dセキュア・システムに追加情報を提供します。これには、連絡先情報を含める必要があります。
- **Status** - 加盟店が**enabled**（有効）か**disabled**（無効）かを示すステータス。加盟店を無効にすると、その特定の加盟店に対して認証APIリクエストが許可されなくなります。
- **Notes** - 管理者ユーザーが加盟店のメモにアクセスして編集できるようにするためのセクション。

ユーザーアクセス

ユーザーが**Status**および**Notes**フィールドを表示および編集するには、**Business admin**ロールが必要です。

国際ブランド

以下は認証リクエストに使用される国際ブランド固有の詳細です。

- **Acquirer BIN** - AReqメッセージを受信するDSによって割り当てられたアクワイアリング機関の識別コード。 [ドロップダウンリスト](#)から既存の**アクワイアラー**を選択して入力するか、手動で入力する事ができます。 **最大11文字**
- **Requestor ID** - DSが割り当てた3DSリクエスター識別子。各DSが、3DS2加盟店の登録が完了した後に各3DSリクエスターに個別に一意的識別子を提供します。 **最大35文字**
- **Requestor name** - DSが割り当てた3DSリクエスター名。各DSが、3DS2加盟店の登録が完了した後に各3DSリクエスターに個別に一意的名前を提供します。 **最大40文字**
- **Category code** - 加盟店の事業、製品、またはサービスの種類を説明するDS固有のコード。 **最大4文字**

警告

上記のすべての国際ブランド固有の詳細は、認証リクエストの提供に必須です。これらのいずれかが存在しない場合、認証リクエストが失敗します。

加盟店詳細の表示

加盟店詳細を表示するには、管理インターフェイスの**Merchants**ページで加盟店を**検索**し、加盟店リストから加盟店を選択します。このページでは、加盟店のセキュリティも管理されます。

Merchants > Merchants administrator

Test Merchant [← Back to Merchant list](#)

Details

Merchant name *	<input type="text" value="Test Merchant"/>	Merchant ID *	<input type="text" value="123456789012345"/>
Country *	<input type="text" value="Australia"/>	Default currency *	<input type="text" value="Australian Dollar (AUD)"/>
3DS Requestor URL *	<input type="text" value="https://gpayments.com"/>	Status *	<input checked="" type="checkbox" value="Enabled"/>

Notes
Contact number : +61-2-9999999, Contact person: Merchant Contact Name, City: Sydney

Card schemes



	Acquirer BIN		Requestor ID	Requestor name	Category code
American Express	<input type="text" value="40001"/> or choose from list		<input type="text" value="123456789.amex"/>	<input type="text" value="3dsclient.local.amex"/>	<input type="text" value="3200"/>
Discover	<input type="text" value="40001"/> or choose from list		<input type="text" value="123456789.discover"/>	<input type="text" value="3dsclient.local.discover"/>	<input type="text" value="3200"/>
JCB	<input type="text" value="Enter BIN"/> or choose from list		<input type="text" value="Requestor ID"/>	<input type="text" value="Requestor name"/>	<input type="text"/>
Mastercard	<input type="text" value="Enter BIN"/> or choose from list		<input type="text" value="Requestor ID"/>	<input type="text" value="Requestor name"/>	<input type="text"/>
Visa	<input type="text" value="Enter BIN"/> or choose from list		<input type="text" value="Requestor ID"/>	<input type="text" value="Requestor name"/>	<input type="text"/>

加盟店のセキュリティ

加盟店の**クライアント証明書**および**マーチャントトークン**、および**サーバーCA証明書**は、このページからアクセスできます。さらに、ユーザーはセキュリティのために**データ暗号化キー**を管理できます。

注釈

証明書管理は、インスタンスが**アクティブ化**された後にのみ使用可能です。

Client certificate	
Merchant token 	acdd4ac4-b30b-4aee-869f-e2480e11243a
Download	Download the client certificate to be used for authenticating the merchant during a transaction API call. Download
Revoke	Revoke client certificate if the security has been compromised and re-issue a certificate with new ID. Revoke
CA certificates	
Download	Download the CA certificate bundle used for Admin and Authentication API authorisation. Download
Data encryption key 	
Generation date	13/09/2019 13:59:12 (3 hours ago) Rotate key

クライアント証明書

3DSリクエストのクライアント証明書は、加盟店がSSL認証の認証APIリクエストに含めるために必要です。

- **Merchant token** - 認証APIリクエストのHTTPヘッダーに追加されるトークン。 [認証APIマスタークライアント証明書](#)を使用して加盟店に代わって認証する場合にのみ必要です。 [認証APIマスタークライアント証明書](#)は **ビジネス管理者** ロールを持つユーザーにのみアクセスできるので、このフィールドは **ビジネス管理者** ロールを持つユーザーにのみ表示されます。) マーチャントトークンでマスター証明書を使用する方法については、 [こちら](#) を参照してください。
- **Download** - ユーザーがパスワードの指定後に、**.p12**形式で3DSリクエストのクライアント証明書をダウンロードすることを許可します。
- **Revoke** - セキュリティ違反または証明書の喪失が発生した場合に現在の3DSリクエスト・クライアント証明書を無効化し、加盟店にダウンロードおよび提供可能な新しい証明書を再発行します。

警告

クライアント証明書を **失効** させると、過去に発行された証明書は無効化され、代替証明書がインストールされるまで、加盟店はAPIリクエストを送信できなくなります。

CA証明書

- **Download** - 認証APIにリクエストを送信する際に必要なサーバーCA証明書をダウンロードします。この機能の詳細については、 [APIドキュメント概要](#) を参照してください。

バージョン1.0.5

CA証明書のダウンロードはバージョン1.0.5でリリースされました。

ユーザーアクセス

ユーザーが証明書をダウンロードするには、**Business admin**、

Merchant adminまたはMerchantロールが必要です。
ユーザーが証明書を失効させるには、**Business admin**または**Merchant admin**ロールが必要です。

データ暗号化キー¶

データベースに保存する前にすべての認証に対して**ActiveServer**がリクエストとレスポンスの暗号化に使用するキーがすべての加盟店に割り当てられています。このキーは、取引の検索時に取引に使用されるアカウント番号を復号化するのにも使用されます。

- **Rotate key** - 内部または外部ポリシーで暗号化キーのローテーションが要求される場合など、必要に応じて、使用中の現在のデータ暗号化キーを変更するのに使用されます。以前の鍵は破壊されるわけではなく以前の取引の暗号化/非暗号化のために使用されます。新しい鍵はローテーション後の取引において使用されます。

ユーザーアクセス

ユーザーがキーをローテーションするには、**Business admin**または**Merchant admin**ロールが必要です。

加盟店詳細の編集¶

加盟店を編集するには、プロフィールを表示したり、利用可能な項目を編集したりします。

利用可能な加盟店プロフィール詳細は、ユーザーロールに固有です。

- **Status** - **Business admin**ロールのユーザーのみが有効化されたステータスを利用できます。
- **Notes** - **Business admin**ロールのユーザーのみがメモセクションを利用できます。

ユーザーアクセス

加盟店詳細を表示するには、**Business admin**、**Merchant admin**または**Merchant**ロールが必要です。
ユーザーが加盟店詳細を編集するには、**Business admin**または**Merchant admin**ロールが必要です。

加盟店の削除¶

加盟店を削除するには、まず管理インターフェイスの**Merchants**ページに移動し、加盟店を検索して、検索結果の表の加盟店名に隣接する**delete**チェックボックスを選択します。**Delete**ボタンを選択し、ダイアログボックスで確認します。

重要

デフォルトの**Test Merchant**テストに使用されるのでは削除できません。

ユーザーアクセス

ユーザーが加盟店を削除するには、**Business admin**ロールが必要です。

アクワイアラの管理

すべてのアクワイアラのリストは、管理インターフェイスの**Merchants > Acquirers**メニューからアクセスできます。リストには**アクワイアラの名前**と関連するすべての**BIN番号**が表示されます。

ユーザーアクセス

ユーザーがアクワイアラを作成、表示、編集、削除するには、**Business admin**ロールが必要です。

The screenshot displays the 'Merchants > Acquirers' management page. At the top, there is a breadcrumb 'Merchants > Acquirers' and a user profile 'administrator'. Below this is a table titled 'Acquirer list' with a 'Select a row to view details.' instruction. The table has columns for 'Acquirer name' and 'Acquirer BINs'. One row is visible: 'Test Acquirer' with BIN '40001'. Below the table are 'Delete' and 'New' buttons. Below the table is a detail view for 'Test Acquirer'. It contains a form with 'Acquirer name' (Test Acquirer) and 'Acquirer BINs' (40001 x Add a BIN). A 'Save' button is at the bottom.

アクワイアラの作成

アクワイアラを**作成**するには、**New**ボタンを選択し、フィールドに入力します。

- **Acquirer name** - ActiveServerでアクワイアラを識別するのに使用される名前。認証メッセージには使用されません。
- **Acquirer BINs** - アクワイアラに割り当てることができるBIN。1つ以上指定可能。このフィールドは認証メッセージで送信され、決済システム、DSへ加盟店が登録時に使用されたものと同じである必要があります。

Createボタンを選択すると、アクワイアラーが作成されます。

アクワイアラー詳細の表示と編集

アクワイアラー詳細を表示および編集するには、リストからアクワイアラーを選択します。必要に応じてアクワイアラー詳細を変更し、**Save**ボタンを選択します。

アクワイアラーの削除

アクワイアラーを削除するには、アクワイアラーのリストの隣にある**delete**チェックボックスを選択します。**Delete**ボタンを選択し、ダイアログボックスで確認します。

DS設定の管理

ActiveServerでサポートされるすべての国際ブランドは**Settings**タブの**Directory Servers**ページで管理できます。

ユーザーアクセス

ユーザーがDS設定を管理するには、**System admin**ロールが必要です。

The screenshot shows the 'Directory Servers' management interface. At the top, there are tabs for different international brands: American Express, Discover, JCB, Mastercard, and Visa. The 'American Express' tab is selected. Below the tabs, there are two sub-tabs: 'Settings' and 'Certificates'. The 'Settings' tab is active, displaying various configuration fields:

- Cache refresh interval: 1 (hr)
- 3DS Server Operator ID: AS_XXXX_XXXX_XXXX_0001
- HTTPS listening port: 8451
- 3DS Server URL: https://api.xxxx.xxxx.3dsecure.cloud:10443
- Timeouts section:
 - Preparation Response (PRes): 300 (sec)
 - Authentication Response (ARes): 20 (sec)
- Server URL's section:
 - Type: URL
 - Default: https://ds.xxxx.xxxx.3dsecure.cloud:4444/ds/xxxx
 - PReq: URL

A 'Save' button is located at the bottom center of the settings area.

国際ブランド設定を編集するには：

詳細を表示するには、ページ上部の適切な**国際ブランド**を選択します。

設定

Settingsセクションでは以下の設定を編集できます：

- **Cache refresh interval** - 使用可能なすべての国際ブランドのPResキャッシュが更新される間隔。PReq/PResメッセージは、**ActiveServer**によって利用され、利用可能なACS、DS、および3DSメソッド呼び出しに使用されるURLでサポートされるプロトコルバージョン番号

に関する情報をキャッシュします。データは、DSによって設定されたカード範囲ごとに整理されます。ACSとDSがサポートするプロトコルバージョン番号で提供される情報は、アプリベース、ブラウザーベース、および3RIのフローで利用できます。この交換は少なくとも24時間に1回、最大でも1時間に1回行われることが3DS2仕様の要件です。(単位：時間)

- **3DS Server Operator ID** - DSが割り当てた3DSサーバー識別子。各DSは各3DSサーバーに個別に一意のIDを提供できます。これは通常、国際ブランド・コンプライアンス・プロセス中、あるいは終了時に提供されます。このフィールドがAReqおよびPReqメッセージに存在する要件はDS固有です。
- **HTTPS callback port** - ASが認証中にHTTPS通信をリスンするポート。
- **External URL** - DSが認証中にコールバックするURL。

注釈

外部URLの更新時に3DSサーバーURLが既に設定されている場合があります。

重要

上記のURLの場合、システム機能が正しく実行されることを確認するためにURL妥当性確認が実行されます。妥当性確認では、URLにパスまたはクエリ文字列が含まれていないことを確認します。

例： `https://domainname<:port>` は妥当性確認に成功しますが、 `https://domainname<:port>/path?queryString` は失敗します。

タイムアウト

Timeoutsセクションでは以下の設定を編集できます：

- **Preparation Response (PRes)** - PResメッセージのタイムアウト時間
- **Authentication Response (ARes)** - AResメッセージのタイムアウト時間

サーバーURLリスト

Server URLsセクションでは、国際ブランドDSのアドレスのDS URLを入力できます。

Default URLはAReqとPReqのメッセージをDSのプロバイザーに送信する際に使用されます。

DSのプロバイダーによってはAReqとPReqのエンドポイントが違う場合があるので、**PReq URL**はオプションで入力できます。この項目に値を設定した場合は**Default URL**の値を上書きし、設定していない場合は**Default URL**がPReqにしようされます。

DS証明書の管理

ActiveServerでサポートされるすべての国際ブランドは、**Certificates**タブの**Directory Servers**ページで管理できます。

ユーザーアクセス

ユーザーがDS証明書を管理するには、**System admin**ロールが必要です。

The screenshot displays the 'JCB' directory server configuration page. It features a navigation bar with tabs for 'American Express', 'Discover', 'JCB', 'Mastercard', and 'Visa'. The 'JCB' tab is selected, and the 'Certificates' sub-tab is active. The page is divided into three main sections: 'Client certificate', 'Server certificate', and 'CA certificate'. Each section contains a table with columns for 'Certificate Information', 'Status', 'Validity', 'Issuer', and 'Hash algorithm'. Below the 'Client' and 'Server' certificate tables, there are buttons for 'Create CSR', 'Download CSR', 'Delete CSR', and 'Install Certificate'. The 'CA certificate' section includes a 'Display' dropdown set to '10' and a 'Records' label, followed by a table listing two CA certificates and an 'Install Certificate' button.

国際ブランド証明書を管理するには：

詳細を表示するには、ページ上部の適切な**国際ブランド**タブを選択します。

クライアントとサーバー証明書

3DS2.0の認証ではDSへのインバウンドとアウトバンドの接続は相互SSL認証される必要があり、**ActiveServer**はそれぞれHTTPSサーバーとクライアントとして役割を担います。

ActiveServerから国際ブランドのDSに接続する際にはActiveServerはクライアントとしての役割を担います。AReqを送信する際に**クライアント証明書**を使用しAResを受信します。

もし認証においてチャレンジが必要な場合は**ActiveServer**はサーバーとしての役割を担います。チャレンジ終了後にDSからRReqを受信し最終的な認証結果をActiveServerに通知されます。この接続を相互認証するためにサーバー証明書が使用されます。

証明書は、通常、証明書署名リクエスト（CSR）を提供した後に国際ブランドからダウンロードできます。

証明書の表には以下の情報が記載されています：

- **Certificate Information** - インストールされている証明書の情報
- **Status** - CAによってサインされたか否か。インストールされている**CA証明書**によってサインされている場合は**Valid**、されていない場合は**Invalid**を表示します。
- **Validity** - 証明書の有効期限
- **Issuer** - 証明書をサインしたCAのイシューアー名
- **Hash algorithm** - 証明書をサインする際に使用されたハッシングアルゴリズム
- **Export | Delete** - 証明書の**ダウンロードと削除**

以下のクライアント証明書の管理ができます：

CSRの作成

CSRの生成を支援するため、**ActiveServer**は**Create CSR**ボタンからこの機能を提供しています。ただし、ご希望であれば、**Java keytool**のような別の方法を使用して、手動でこのプロセスを実行することもできます。

証明書の内容は、国際ブランドの要件に応じて入力する必要があります。以下のオプションが利用可能です。

- **Key size** - リクエストのキーのサイズ（ビット単位）
- **Common Name** - 証明書に使用されるホスト名。通常、完全修飾ドメイン名が使用されます。サーバー証明書の場合はこれは**ActiveServer**のホスト名になります。クライアント証明書の場合は通常サーバー証明書と同じになりますが、国際ブランドによっては違う場合があります。**Common Name**の値はデフォルトでDSに設定されている**3DS Server URL**のドメイン名になります。
- **Organization** - 企業または組織の法的な名前
- **Organization Unit** - グループの部署または部門の名前
- **City** - 企業がある市区町村

- **Province** – 企業がある都道府県
- **Two letter country code** – 国の2文字の略称
- **Hash algorithm** - CSRの署名に使用されるハッシュアルゴリズム

CSRの作成では、生の証明書コンテンツが作成され、**.p10**形式で**Download certificate**のボタンが提供されます。

重要

各DSにはクライアントCSRとサーバーCSRの2つのみ保存できます。

CSRをエクスポート

CSRをエクスポートはCSRはCSRのコンテンツを **.csr** としてダウンロードします。ファイル名は **"Common Name"_"国際ブランド名".csr**のフォーマットになります。例： `api.testlab.3dsecure.cloud_JCB.csr` .

CSRを作成した後のみ**CSRをエクスポート**できます。

CSRを削除

CSRを削除した場合CSRのコンテンツとCSRを作成するのに使用された秘密鍵の両方を削除します。

CSRを作成した後のみ**CSRを削除**できます。

警告

CSRを削除した場合削除した後に署名された証明書はインストールできなくなります。

証明書をインストール

証明書をインストールは署名された**certificate content**または**certificate file**をインストールできます。

サポートされている証明書のフォーマット：**.pfx** , **.p7b** , **.p12** , **.jks** , **.pem** 。**ActiveServer** は各ファイルタイプを読み込みます。もし、ファイルにパスワードが必要な場合はCertificateの

ページでパスワードを入力して下さい。例えば：**.p12** はパスワードが必要なファイル形式ですので、インストールする際にパスワードを入力する必要があります。

ActiveServerは **.pfx**、**.p12** または **.jks** のファイルがインストールされた場合はファイルに含まれている秘密鍵を使用して証明書をインストールしようとします。もし、ファイルに秘密鍵が含まれていない場合は現在インストールされている秘密鍵を使用します。秘密鍵の作成の仕方については[こちらを参照](#)下さい。

もし、国際ブランドがクライアントとサーバーの接続に必要な証明書が1つだけの場合、**Server certificate is the same as the client certificate** オプションを選択できます。これにより、クライアントセクションとサーバーセクションの両方に証明書がインストールされます。

重要

正しい証明書タイプが正しいセクションにインストールされていることを確認するために、**ActiveServer** は証明書の拡張子 **X509v3 Extended Key Usage** をチェックし、次のチェックを実行します。* クライアント証明書- **TLS Web Client Authentication** の値が必要です。**TLS Web Server Authentication** の値がない場合、*Server certificate is the same as the client certificate* オプションが無効になります。* サーバー証明書- **TLS Web Server Authentication** の値が必要です

インストール (Install)

署名済みの証明書は、**Install** ボタンを使用することでインストールできます。

Warning

一度に**1つ**のクライアント証明書のみを持つことができ、別の証明書をインストールまたはインポートすると、現在の証明書が上書きされます。

エクスポートと削除 (Export and Delete)

クライアントとサーバーの証明書は、バックアップのためにエクスポートしたり、必要に応じて証明書テーブルから削除アイコンを選択して削除したりできます。

証明書は、次の2つの形式で**エクスポート**できます。

- **PKCS12キーストア (秘密鍵を含む.p12)** - 証明書と関連する秘密鍵を含む **.p12** ファイルを作成します。オプションで、ファイルのパスワードを含めることができます。

- ・ **証明書のみ** (.pem - 証明書のみを含む **.pem** ファイルを作成します。

Deleteは、システムから証明書を削除する前に、証明書の削除を確認するようユーザーに求めるプロンプトを表示します。

警告

証明書の削除は永続的であり、最初にバックアップとして証明書をエクスポートすることをお勧めします。

CA証明書

CA証明書は、サーバー/クライアント証明書のCA署名者を検証し、それらが有効なCAからのものであることを確認するために使用されます。CA証明書はサーバー/クライアント証明書をインストールした際にはCAチェーンが見つかった際に自動的にインストールされます。または、手動でインストールする事も可能です。

関連するCAがインストールされていない場合、クライアントまたはサーバー証明書の**Status**は**Not Valid**です。CAを削除すると、以前のインストールも無効になります。

[証明書のインストール

]ボタンには、ローカル証明書ファイルを検索するプロンプトが表示されます。表示される証明書情報と機能は、クライアントおよびサーバー証明書にCA証明書のエイリアス値を追加したものになります。

証明書管理に外部ツールを使用する (Using External Tool)¶

選択したツールを使用して、CSRと秘密キーを生成できます。OpenSSLを使用した例を以下に示します。

OpenSSLがインストールされていることを確認し、ターミナルを開いて以下を実行します。

1. RSA秘密鍵を作成します

```
1 openssl genrsa -out privateKey.key 2048
```

2. CSRを生成し、プロンプトに従ってCSRの詳細を入力します

```
1 openssl req -new -key privateKey.key -out yourCSR.csr
```

3. CSRが国際ブランドによって署名されたら、提供された署名済み証明書と国際ブランドCA証明書チェーンを生成された秘密キーと結合します

```
1 openssl pkcs12 -export -out certificate.p12 -inkey privateKey.key -in
```

4. 証明書をインストールする。

取引を表示

Transactionsは、左側のメニューからアクセスでき、3つのセクションがあります。

- 取引を検索
- 取引のリスト
- 取引の詳細

Transactions
administrator

Search transactions

From Transaction date from	To Transaction date to	3DS Server Transaction ID 3DS Server Transaction ID
Account number Account number	Min purchase amount Min purchase amount	Max purchase amount Max purchase amount
Merchant name Merchant name	Provider(s) Provider(s)	Status ▼
Purchase currency Purchase currency	Message category ▼	

Clear
Search

Transaction list Select a row to view details.

Display 10 Records Showing 1 to 10 of 47,787 Transactions. Previous 1 2 3 4 5 ... 4779 Next

Provider	Date (dd/mm/yyyy)	3DS Server Transaction ID	Account number	Amount	Currency	Merchant name	Status
Visa	06/02/2020 15:58:09	3492d142-5838-4618-a1d5-733f63bc82f9	410000XXXXXX0009			Test Merchant	A (ARes: A)
Visa	06/02/2020 15:58:03	b99a0667-1453-4039-b905-838cef9819f3	410000XXXXXX5000		036	Test Merchant	Y (ARes: C, RReq: Y)
Visa	06/02/2020 15:58:01	b8a31d62-787a-4864-ac7a-9f142803b4d8	410000XXXXXX0100	1.00	036	Test Merchant	Y (ARes: Y)
Visa	06/02/2020 15:57:52	755c5589-628b-4e1c-ba22-2fab3c1e2f1f	410000XXXXXX0100			Test Merchant	Y (ARes: C, RReq: Y)
Visa	06/02/2020 15:57:49	fe2d9f58-7564-4fd8-be6b-05a8c95fc12a	410000XXXXXX0100			Test Merchant	Y (ARes: Y)
Visa	06/02/2020 15:57:47	01ad0f90-c47d-4e46-af91-844f4af9ba09	410000XXXXXX0100			Test Merchant	Y (ARes: Y)
Visa	06/02/2020 15:57:42	bb1414a4-da5e-44c8-b82d-828e00a1e564	410000XXXXXX5000	200.00	036	Test Merchant	Y (ARes: C, RReq: Y)
Visa	06/02/2020 15:57:39	3eccb7b3-06be-4cb2-84be-e8087fb714d1	410000XXXXXX0100	1.00	036	Test Merchant	Y (ARes: Y)
Visa	06/02/2020 15:57:33	b1281e7c-b3d7-412a-bc72-cba2a53cecea	410000XXXXXX5000	200.00	036	Test Merchant	Y (ARes: C, RReq: Y)
Visa	06/02/2020 15:57:31	d7e61090-1b7f40fd-aedd-95cdb49ce1c2	410000XXXXXX0100	1.00	036	Test Merchant	Y (ARes: Y)

取引を検索

このセクションでは、特定の取引について、データベースから検索できます。取引に関する情報を入力することで、取引をフィルタリングできます。フィールドには以下が含まれます：

- From** - 指定した日付以降の取引のみが含まれます
- To** - 指定した日付以前の取引のみが含まれます
- 3DS Server Transaction ID** - 指定した取引IDの取引のみが含まれます

- **Account number** - カード会員のカード番号による取引のみが含まれます (PANまたはトークンによって表される場合があります)
- **Minimum purchase amount** - 指定した金額を超える取引のみが含まれます
- **Maximum purchase amount** - 指定した金額未満の取引のみが含まれます
- **Merchant name** - 指定した加盟店によって処理された取引のみが含まれます。加盟店名のすべてまたは一部を使用できます。
- **Provider(s)** - 指定した1つまたは複数の国際ブランドによって処理された取引のみが含まれます
- **Status** - 指定した結果ステータスの取引のみが含まれます (例: Transaction Successful の場合は"Y"など)
- **Purchase currency** - 指定した通貨で実行された取引のみが含まれます。これは通貨コードで定義されます
- **Message category** - PA (決済) またはNPA (非決済) のいずれかの取引のみが含まれます

目的のフィルタを設定し、**Search**をクリックすると、**Transaction List**の下に結果が表示されます。**Clear**をクリックすると、フィールドがリセットされます。

取引のリスト

取引リストには、すべての取引、または上記の検索パラメータのいずれかを選択した場合はフィルタリングされたリストが表示されます。

表示される取引の詳細は以下のとおりです：

- **Provider** - 取引に使用された国際ブランド
- **Date** - 取引が処理された日時
- **3DS Server Transaction ID** - 特定の取引の3DSサーバー取引ID
- **Account number** - 取引に使用されたカード番号の全桁。PANまたはトークンで表される場合があります。

重要

認証API v1で実行された取引においては、カード番号(PAN)の全桁が暗号化されデータベースに保存されます。認証API v2で実行された取引においては、カード番号(PAN)はトランケート形式で保存されます(上6桁と下4桁のみが保存され、他全桁は削除されます)。なので、使用するAPIバージョンによって、検索結果には入力された**Account number**と一部だけマッチする取引が表示される場合があります。取引を正確に特定する必要がある場合、他の絞り込み条件とあわせて検索することを推奨します。

- **Amount** - 取引の購入金額
- **Currency** - 取引が処理された通貨コード
- **Merchant name** - 取引を処理している加盟店の名前
- **Status** - 取引に対して返された認証結果。以下のフォーマットが使用されます：
 - **Frictionless (i.e. no challenge)** - "最終ステータス (ARes: AResステータス)"、例: "Y (ARes: Y)"
 - **Challenge** - "最終ステータス (ARes: AResステータス", RReq: RReqステータス)、例: "N (ARes: C, RReq: N)"

取引行を選択すると、Transaction detailsセクションの下に詳細が表示されます。

取引の詳細

取引詳細には、取引リストで選択した取引の詳細が表示されます。

Transaction details ^

<p>3DS Server Transaction ID bb1414a4-da5e-44c8-b82d-828e00a1e564</p> <p>Merchant ID 123456789012345</p> <p>Account number 410000XXXXX5000</p> <p>ThreeDS Requestor name 3dsclient.local.visa</p> <p>ARes status C</p> <p>Status reason</p> <p>Device channel BRW (Browser-based Authentication)</p> <p>Authentication value AAEBBWGCUwJgAAAABIJTAAAAA=</p> <p>ECI 05</p> <p>Message category PA (Payment)</p> <p>Recurring frequency</p> <p>Recurring expiry date</p> <p>SDK transaction ID</p> <p>ACS transaction ID b7b4a86b-91f2-4aa4-8862-b2f889749f42</p> <p>Error code</p> <p>Error component</p> <p>Message version 2.1.0</p>	<p>Purchase date (dd/mm/yyyy) 06/02/2020 15:57:41</p> <p>Merchant name Test Merchant</p> <p>Amount 200.00</p> <p>ThreeDS Requestor ID 123456789.visa</p> <p>RReq status Y</p> <p>Acquirer BIN 40001</p> <p>Pay token indicator</p> <p>Authentication type Static</p> <p>Merchant category code 3200</p> <p>Card expiry date</p> <p>Purchase currency 036</p> <p>Challenge mandated N</p> <p>DS transaction ID 4cbce215-d3c6-437f-b1d7-cd48ade61cd7</p> <p>ThreeDS Requestor challenge indicator</p> <p>Error detail</p> <p>Error description <div style="border: 1px solid #ccc; height: 60px; width: 100%;"></div></p>
--	---

Requests / Responses

^

表示される取引の詳細は以下のとおりです：

- **3DS Server Transaction ID** - 単一の取引を特定するために3DSサーバーによって割り当てられた、世界で1つだけの取引識別子です
- **Purchase date** - 購入の日時
- **Merchant ID** - アクワイアラーが割り当てた加盟店識別子。これは、3DSリクエスターに代わって送信されるオーソリゼーションリクエストで使用されるものと同じ値である必要があります
- **Merchant name** - アクワイアラーまたは決済システムによって割り当てられた加盟店名
- **Account number** - 決済取引のオーソリゼーションリクエストに使用されるカード番号。PANまたはトークンで表される場合があります。カード番号の上6桁と下4桁は表示され、残り桁は全て"X"と表示されます。
- **Amount** - 取引の購入金額
- **ThreeDS Requestor name** - DSが割り当てた3DSリクエスター名。各DSが、各3DSリクエスターに個別に一意の名前を提供します。
- **ThreeDS Requestor ID** - DSが割り当てた3DSリクエスター識別子。各DSが、各3DSリクエスターに個別に一意の識別子を提供します。

- **ARes/RReq status** - 取引が認証済みの取引またはアカウント確認のどちらに該当するかを示します。考えられる値は以下のとおりです：
 - **Authenticated (Y)** - 認証確認が成功しました
 - **Not Authenticated (N)** - 認証/アカウントが確認されませんでした。取引が拒否されました
 - **Unauthenticated (U)** - 認証/アカウント確認を実行できませんでした。技術的またはその他の問題
 - **Attempted (A)** - 試行の処理が実行されました。認証/検証されませんでした。認証/検証が試行されたことの証明が提供されます
 - **Challenge Required** - CReq/CResを使用した追加認証が必要です
 - **Rejected (R)** - 認証/アカウント確認が拒否されました。イシューアは取引/検証を拒否し、オーソリゼーションを試行しないように要求しています
- **Status reason** - Transaction Statusフィールドが指定された値を持つ理由に関する情報を提供します。決済チャネルについては、Transaction StatusフィールドがN、U、またはRの場合に必須です。非決済チャネルについては、DSで定義されている場合に条件付き必須です。可能な値は以下のとおりです：
 - **01** - カード認証が失敗しました
 - **02** - 不明なデバイス
 - **03** - サポートされていないデバイス
 - **04** - 認証頻度制限を超えました
 - **05** - 期限切れのカード
 - **06** - カード番号が無効です
 - **07** - 取引が無効です
 - **08** - カードレコードがありません
 - **09** - セキュリティ障害
 - **10** - 盗難されたカード
 - **11** - 詐欺の疑い
 - **12** - カード会員に取引が許可されていません
 - **13** - カード会員がサービスに登録されていません
 - **14** - ACSで取引がタイムアウトしました
 - **15** - 信用：低

- **16** - 信用：中
- **17** - 信用：高
- **18** - 信用：最高
- **19** - ACS最大チャレンジ数を超過しました
- **20** - 非決済取引はサポートされていません
- **21** - 3RI取引はサポートされていません
- **22-79** - EMVCoが今後使用するため予約されています（EMVCoによって定義されるまで値は無効です）
- **80-99** - DSが使用するため予約されています
- **Acquirer BIN** - アクワイアラーのアクワイアリング機関の識別コード
- **Device channel** - 取引の発生元のチャンネルを示します。可能な値は以下のとおりです：
 - App-based (01-APP)
 - Browser-based (02-BRW)
 - 3DS Requestor Initiated (03-3RI)
- **Pay token indicator** - 値**True**は、取引がACSによって受信される前にトークン化解除されたことを示します。このデータ要素は、トークン化解除が発生した3-Dセキュアドメインに存在するシステム（すなわち3DSサーバーまたはDS）によって入力されます。存在する場合、このフィールドに対して有効なレスポンスはブール値**true**のみです。
- **Authentication value** - サポートされている各DSのACS登録の一環として提供される決済システム固有の値。Authentication Valueは、オーソリゼーション中やクリアリング中など、認証の証明を提供するのにも使用できます。この値は7日間保存され、その後はマスクされます。
- **Authentication type** - チャレンジが必要な場合にイシューアがカード会員のチャレンジに使用した認証方法のタイプを示します。AResメッセージ内にある場合もありますし、RReqメッセージ内ではACSによって使用されたものを示します。可能な値は以下のとおりです：
 - **01** - 静的
 - **02** - 動的
 - **03** - OOB
 - **04-79** - EMVCoが今後使用するため予約されています（EMVCoによって定義されるまで値は無効です）
 - **80-99** - DSが使用するため予約されています

- **ECI** - 国際ブランドのカード会員認証のステータスを識別するトランザクションの電子商取引インジケータ値
- **Merchant category code** - 加盟店の事業、製品、またはサービスの種類を説明するDS固有のコード
- **Message category** - 取引のカテゴリを識別します。可能な値は以下のとおりです：
 - **01** - PA（決済）
 - **02** - NPA（非決済）
 - **03-79** - EMVCoが今後使用するため予約されています（EMVCoによって定義されるまで値は無効です）
 - **80-99** - DSが使用するため予約されています
- **Card expiry date** - カード会員によって3DSリクエストに提供されたPANまたはトークンの有効期限（YYMM形式）
- **Recurring frequency** - 定期取引の場合、オーソリゼーション間の最低日数を示します
- **Purchase currency** - 購入金額を表現する通貨コード
- **Recurring expiry date** - 定期取引の場合、それ以降オーソリゼーションが実行されない日付（YYYYMMDD形式）
- **Challenge mandated** - 局所的/地域的な義務またはその他の不確定要素によって、取引のオーソリゼーションにチャレンジが必要かどうかを示します。可能な値は以下のとおりです：
 - **Y** - チャレンジが必須
 - **N** - チャレンジが必須ではない
- **SDK transaction ID** - 単一の取引を特定するために3DS SDKによって割り当てられた、世界で1つだけの取引識別子です
- **DS transaction ID** - 単一の取引を特定するためにDSによって割り当てられた、世界で1つだけの取引識別子です
- **ACS transaction ID** - 単一の取引を特定するためにACSによって割り当てられた、世界で1つだけの取引識別子です
- **ThreeDS requestor challenge indicator** - この取引について加盟店によってチャレンジがリクエストされているかどうかを示します。例：
 - 01-PAの場合、3DSリクエストは取引に懸念を持ち、チャレンジを要求できます。
 - 02-NPAの場合、新しいカードをウォレットに追加するときにチャレンジが必要な場合があります。

- 局所的/地域的な義務またはその他の不確定要素。
- 可能な値は以下のとおりです：
 - **01** - 指定なし
 - **02** - チャレンジリクエストなし
 - **03** - チャレンジがリクエストされました：3DSリクエスト指定
 - **04** - チャレンジがリクエストされました：義務
 - **05-79** - EMVCoが今後使用するため予約されています（EMVCoによって定義されるまで値は無効です）
 - **80-99** - DSが使用するため予約されています
 - 注：要素が指定されない場合、想定されるアクションは、ACSが01 =指定なしと解釈することです。
- **Error code** - 利用可能な場合、メッセージで識別された問題のタイプを示すコード。
- **Error detail** - 利用可能な場合、メッセージで識別された問題に関する追加の詳細。
- **Error component** - 利用可能な場合、エラーを識別した3-Dセキュア・コンポーネントを示すコード。可能な値は以下のとおりです：
 - **C** - 3DS SDK
 - **S** - 3DSサーバー
 - **D** - DS
 - **A** - ACS
- **Error description** - 利用可能な場合、メッセージで識別された問題を説明するテキスト。
- **Message version** - 3DSサーバーによって使用されるプロトコル・バージョン識別子。AReqメッセージに設定されます。メッセージバージョン番号は、3DS取引中に変更されません。

取引メッセージ

取引の3DS2メッセージを表示するには、Transactionsセクションの下部の [Requests / Responses](#) をクリックします。

Message type AReq

Time stamp 06/02/2020 15:57:43

Message content

```
{
  "notificationURL": "https://testlab.3dsecure.cloud/api/v1/brw/challenge/notification",
  "messageCategory": "01",
  "purchaseDate": "20200206045741",
  "threeDSRequestorRefNumber": "3DS_LOA_SER_GPPL_020100_00075",
  "merchantCountryCode": "840",
  "browserColorDepth": "24",
  "browserJavaEnabled": false,
  "mcc": "3200",
  "purchaseCurrency": "036",
  "browserAcceptHeader": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8",
  "browserTZ": "0",
  "merchantName": "Test Merchant",
  "threeDSRequestorAuthenticationInd": "01",
  "threeDSRequestorURL": "http://gpayments.com",
  "acctNumber": "41000XXXXXX5000",
  "browserScreenHeight": "768",
  "messageType": "AReq",
  "threeDSRequestorName": "3dsclient.local.visa",
  "acquirerMerchantID": "123456789012345",
  "messageVersion": "2.1.0",
  "browserLanguage": "en-US",
  "threeDSRequestorID": "123456789.visa",
  "acquirerBIN": "40001",
  "deviceChannel": "02",
  "browserIP": "XXXXXXXXXXXX",
  "browserUserAgent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.75 Safari/537.36",
  "purchaseAmount": "20000",
  "purchaseExponent": "2",
  "threeDSRequestorOperatorID": "AS_TEST_LAB_OPER_00001",
  "threeDSCompInd": "U",
  "threeDSRequestorURL": "https://testlab.3dsecure.cloud:14443/api/v1/ds/result/request",
  "browserScreenWidth": "1024",
  "threeDSRequestorTransID": "bb1414a4-da5e-44c0-b82d-028e00a1e564"
}
```

- **Message type** - AReq、AReq、RReq、RReq、3DSメッセージタイプ。
- **Time stamp** - メッセージが送受信された日時。
- **Message content** - 送受信された生のJSONメッセージコンテンツ。

ノード管理

ActiveServerは、**単一ノード**または**複数ノード**セットアップのどちらでも実行できます。これらのノードの詳細は、**Deployment**ページの**Nodes**タブで確認できます。このページは、**システム全体**のログレベル設定とは異なる設定が必要な場合に、特定のノードのログレベルを設定できる場所でもあります。

Info

複数ノード・セットアップのガイドは、今後のドキュメントバージョンで提供されます。

ユーザーアクセス

ユーザーがノード設定を表示および編集するには、**System admin**ロールが必要です。

The screenshot shows the 'Deployment' page with the 'Nodes' tab selected. A table lists one node: 'Node c22c22c2' with host name 'rnnrning-pppppp-as-888cfdc888-m18t8' and status 'ONLINE'. Below the table, the details for 'Node c22c22c2' are shown, including Node name, Node ID, Operating system, Physical memory size, Cores, Java version, CPU architecture, and IP address. The logging section shows 'Use system default' is 'Enabled' and 'Log level' is 'INFO'. A 'Save' button is at the bottom.

Node name	Host name	Status
Node c22c22c2	rnnrning-pppppp-as-888cfdc888-m18t8	ONLINE

Node c22c22c2

Details

Node name	Node c22c22c2	Node ID	c22c22c2-4cc7-4fc2-859c-3c3cc3f3c0c3
Operating system	Linux	Java version	1.8.0_201
Physical memory size	16636354560	CPU architecture	amd64
Cores	1	IP address	100.99.11.119

Logging

Use system default Enabled Log level **INFO**

Save

ノード詳細

詳細を表示するには、**Node list**からノードを選択します。

- **Node name** - ステータスの追跡を維持するためにノードに指定できる編集可能な名前。
- **Node ID** - データベース内のノードのシステム生成UUID。
- **Operating system** - ノードのサーバーに使用されているオペレーティング・システム。
- **Java version** - ノードのサーバーに使用されているJavaバージョン。
- **Physical memory size** - ノードのサーバーに搭載されている物理メモリの合計量。
- **CPU architecture** - ノードのサーバーに搭載されているCPUアーキテクチャ・チップのタイプ
- **Cores** - ノードのサーバーに搭載されている論理コア数。
- **IP address** - ノードがホストされているIPアドレス。

ログ

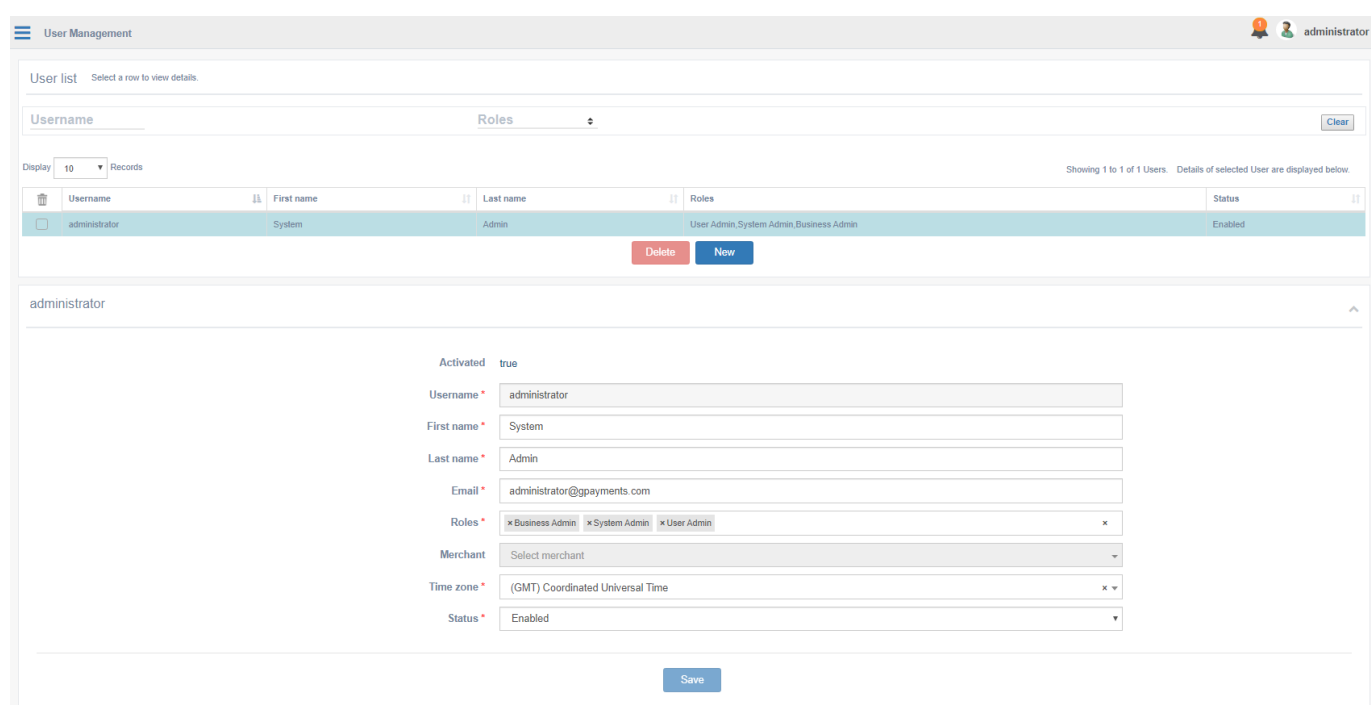
特定のノードに異なるログレベルが必要な場合は、以下の設定を変更できます。

- **Use system default** - *Enabled*の場合、ノードはシステム全体のログレベル設定を使用します。*Disabled*の場合、ノードのログ・ベルを手動で変更できます。
- **Log level** - **Use system default**が*Disabled*の場合に編集でき、**ログレベル**を変更できます。

ユーザーの管理

ユーザーは、管理インターフェイスへのアクセス権を提供するために作成されます。ユーザー**ロール**は、特定の業務上の責任に関するタスクを完了するため、適切なアクセス権を提供するためにユーザーに割り当てられます。**ロール**の詳細については、[ロールと権限ガイド](#)を参照してください。

ユーザーの**作成**、**表示**、**編集**、**削除**プロセスは以下のとおりです。



The screenshot displays the 'User Management' interface. At the top, there is a 'User list' section with a search bar for 'Username' and a dropdown for 'Roles'. Below this is a table with columns for 'Username', 'First name', 'Last name', 'Roles', and 'Status'. The table contains one entry for the 'administrator' user. Below the table are 'Delete' and 'New' buttons. The 'New' button is highlighted, leading to the 'administrator' user details page. This page shows various fields for user information: 'Activated' (true), 'Username' (administrator), 'First name' (System), 'Last name' (Admin), 'Email' (administrator@gpayments.com), 'Roles' (Business Admin, System Admin, User Admin), 'Merchant' (Select merchant), 'Time zone' ((GMT) Coordinated Universal Time), and 'Status' (Enabled). A 'Save' button is located at the bottom of the details page.

ユーザーの作成

ユーザーを作成するには、まず管理インターフェイスの**User Management**ページに移動し、**New**ボタンを選択します。

以下に記載のフィールドを使用して、**New user**画面のフィールドを入力します。

重要

以下の**Role/s**および**Merchant**フィールドは、システムへの適切なアクセス権を割り当てるために重要です。

新しいユーザーを作成する前に、**ロールと権限ガイド**を確認することをお勧めします。

- **Username** - 管理インターフェイスへのログインに使用される、ユーザーに割り当てられた一意の値。必須
- **First name** - ユーザーの名前。必須
- **Last name** - ユーザーの名字。必須
- **Email** - ユーザーの有効な電子メールアドレス。このアドレスは、パスワードリセットやシステム通知などの電子メール通知を送信するのに使用されます。必須
- **Roles** - ユーザーに割り当てることができる、ロールの複数の選択ボックス。必須
- **Merchant** - 割り当てられたユーザーロールによって、単一の加盟店の**スコープ**がユーザーに付与された場合、すでに管理用に作成された加盟店をユーザーに割り当てることができます。割り当てられたユーザーロールによって、すべての加盟店のスコープがユーザーに付与された場合や、加盟店なしのスコープが付与された場合、このフィールドは入力する必要がなく、選択できません。必須
- **Time zone** - 管理インターフェイス上の日時の表示のデフォルト・タイムゾーン。必須
- **Status** - ユーザーのシステムステータス：
 - **Enabled** - ユーザーは管理インターフェイスの機能にアクセスできます。
 - **Disabled** - ユーザーは管理インターフェイスの機能にアクセスできません。

ユーザーアクセス

ユーザーがユーザーを作成するには、**User admin**ロールが必要です。

ユーザー詳細の表示

ユーザーの詳細は、管理インターフェイスの**User Management**ページからユーザーのリストから選択することでアクセスできます。

表示されるユーザー数は、以下のフィールドを使用してフィルタリングすることで制限できます。

- **Username** - ユーザー名の全体または一部。
- **Role(s)** - 単一システム・ロールのドロップダウン選択。

結果の表には、[上記](#)の**Username**、**First name**、**Last name**、**Roles**、**Status**が表示されます。

ユーザーアクセス

ユーザーがユーザー詳細を表示するには、**User admin**ロールが必要です。

ユーザー詳細の編集

ユーザーを編集するには、プロフィールを[表示](#)したり、利用可能な[フィールド](#)を編集したりします。

重要

常に**User Admin**ロールのユーザーがシステムに1人は存在する必要があります。ユーザー詳細の編集によってこのチェックが失敗すると、エラーが発生します。

ユーザーアクセス


ユーザーがユーザー詳細を編集するには、**User admin**ロールが必要です。

ユーザーの削除

ユーザーを削除するには、まず管理インターフェイスの**User Management**ページに移動します。リストを[フィルタリング](#)してユーザーを探し、検索結果の表でユーザー名に隣接する**delete** **チェックボックス**を選択します。**Delete**ボタンを選択し、ダイアログボックスで確認します。

 **重要**

常に**User Admin**ロールのユーザーがシステムに少なくとも1人は存在する必要があります。ユーザーの削除によってこのチェックが失敗すると、エラーが発生します。

 **ユーザーアクセス**

ユーザーがユーザーを削除するには、**User admin**ロールが必要です。

監査ログ

Audit logsでは、監査のため、管理者のアクティビティの包括的なログにアクセスできます。これには、設定、加盟店、ユーザー、および展開情報への変更に関するログが含まれています。

ユーザーアクセス

ユーザーが監査ログを閲覧するには、**System admin**ロールが必要です。

監査ログの検索

デフォルトでは、**Audit log list**に最も新しい監査ログエントリが表示されます。以下を使用すると、リストをフィルタリングできます。

- **From** - この日付以降のエントリのみが表示されます。
- **To** - この日付以前のエントリのみが表示されます。
- **User** - 監査対象のエントリを実行したユーザーのユーザー名を完全または部分一致検索します。
- **Revision type** - データベース・エンティティで実行された操作のタイプ。
 - **Addition** - データベース・エンティティに追加された新規レコード。
 - **Modification** - データベース・エンティティで変更された既存のレコード。
 - **Deletion** - データベース・エンティティから削除された既存のレコード。
- **Entity name** - 監査の影響を受けるデータベースのテーブル。

Searchボタンを選択すると、関連する**Audit logs**が表示されます。**Clear**ボタンを使用すると、検索フィールドがリセットされます。

監査ログ・リスト

Audit log listには、検索条件によって返されたログエントリーの数を含む**Audit logs**の表が、以下の情報と共に表示されます。

- **Entity name** - 監査の影響を受けるデータベースのテーブル。
- **Revision type** - データベース・エンティティで実行された操作のタイプ。
 - **Addition** - データベース・エンティティに追加された新規レコード。
 - **Modification** - データベース・エンティティで変更された既存のレコード。
 - **Deletion** - データベース・エンティティから削除された既存のレコード。
- **Revision date** - 監査ログの日時（dd/mm/yyyy形式）。
- **User** - 監査対象のエントリーを実行したユーザーのユーザー名。
- **IP** - ユーザーのIPアドレス。

Audit log listからログエントリーを選択すると、変更された**項目**と**新しい/古い値**が**Audit log details**に表示されます。

システム設定

Settingsでは、ActiveServerインスタンスのシステム設定を構成できます。**Settings**には以下の3つのタブがあります：

- ・ システム
- ・ セキュリティー
- ・ ActiveMerchantから移行

システム

システムには2つのセクションがあります。

サーバーURL

- ・ **External URL** - 認証コールバックおよび製品のアクティブ化に使用される、外部からアクセス可能なURL。URLには、`https://"あなたのActiveServerドメイン名": "サーバーポート番号"` の形式で入力できます。 `https://paymentgateway.com:8443` 等。URLの **サーバーポート番号** は、**ActiveServer**設定ファイルで設定された（使用されているプロトコルに応じて） `as.server.http.port` または `as.server.https.port` の値です。

警告

外部URLを更新すると、各国際ブランドのDirectory Server設定の**3DSサーバーURL**が更新されます。いずれかの国際ブランドの**3DSサーバーURL**の値が空の場合、新しい**外部URL**値と**HTTPSリスニングポート**を足した値が自動的に設定されます。**3DSサーバーURL**値が既に設定されている場合、変更は行われません。

この自動更新は、これらのURLの形式は一般的に同じであるため、セットアッププロセスを支援するための機能です。アーキテクチャーのセットアップによって**3DSサーバーURL**に個別のURLが割り当てられている場合、取引を実行する前にこの設定を更新する必要があります。

🔥 注釈

ロードバランサーがセットアップされていた場合、このURLは上記のURLパターンとは異なる場合があります。外部URL用のロードバランサーが上記のサーバーポートにリクエストを転送していることを確認してください。

例：URL `https://paymentgateway.com` がサーバーコールバック用に設定され、`https://admin.paymentgateway.com` が管理UIインターフェースリクエスト用に設定されている場合、`https://paymentgateway.com` が外部URLに使用されます。

- **API URL** - 認証APIおよび管理API呼び出しの受信に使用されるオプションのURL。このURLのドメイン名は、API (x.509) の認証用のクライアント証明書生成にも使用されます。入力されない場合、デフォルトで**ActiveServer**はクライアント証明書の生成に外部URLのドメイン名を使用します。このURLは外部からアクセス可能である必要はありません。URLの形式は**外部URL**と同じで、**ポート番号**を指定できます。
- **Admin URL** - Eメールを利用したユーザーのアクティブ化とパスワードのリセット手順に使用されるオプションのURL。この値は、**ActiveServer**の設定ファイルに**Admin port**が有効と設定される場合のみ入力できます。Admin portが有効と設定され、この値が空白の場合は、localhostドメインと指定されたAdmin portが使用されます。ですがこの場合、リモートホストからドメインへアクセスする際に問題が発生します。Admin portが無効と設定される場合、**External URL**に入力されたドメイン名が使用されます。このURLは外部からアクセスできなくても支障はありません。

🔥 重要

上記のURLの場合、システム機能が正しく実行されることを確認するためにURL妥当性確認が実行されます。妥当性確認では、URLにパスまたはクエリ文字列が含まれていないことを確認します。

例： `https://domainname<:port>` は妥当性確認に成功しますが、 `https://domainname<:port>/path?queryString` は失敗します。

ログ

- **Log level** - コンソール出力およびシステムログの詳細度。利用可能な値(右に行くほど詳細度が上がります) : **ERROR > INFO > DEBUG**.

セキュリティ

- **Session timeout (読み取り専用)** - 有効期限が切れ、ユーザーにログイン認証情報の再入力を要求するまでの、ログイン・セッションが有効な間隔。デフォルトのセッションは900秒(15分)に設定されています。この設定を変更する場合は `application-prod.properties` に以下の行を追加してインスタンスを再起動させて下さい。

```
as.settings.session-timeout={単位 秒}
```

例： セッションのタイムアウト時間を1800秒(30分)に設定したい場合は

```
as.settings.session-timeout=1800
```

 と追加して下さい。

重要

値は300 ~ 3600秒までの整数が設定可能です。

- **Session failed attempts** - セッション・ロック時間で指定された期間ログインが一時的に無効化されるまでの失敗ログイン試行回数。一定時間経過後、正しい認証情報を指定することでセッションを再確立できます（単位：試行回数）。
- **Session lock time** - 失敗ログイン試行回数を超えた場合にユーザーがロックされる間隔（単位：分）
- **Password expiry period** - 新しいパスワードの作成が要求されるまでの、パスワードが有効な日数（単位：日数）
- **Password history check** - 特定のパスワードを再度利用できるようになるまで一意のパスワードの使用が要求される数（単位：一意のパスワード数）
- **Force two factor login** - サーバー上の**すべての**ユーザーに対して2要素認証を **enable**（有効化） **disable**（無効化）します。ActiveServerでは、ユーザーに2要素認証を提供するのにGoogle認証システムを使用しています。この設定が**有効化**されると、アカウントで2要素認証がまだセットアップされていないユーザーは、システム機能を使用する前に、次回ログイン時に2要素認証をセットアップすることが強制されます。Google認証システムのセットアップの手順が画面上に表示されます。

鍵ローテーション

Rotate keyを選択すると、現在の暗号化キーの作成日が表示され、ユーザーが使用される鍵をローテーションできます。

HSM

この機能を使用すると、変更された場合にユーザーがHSM PINを更新できます。

- **Full file name and path of PKCS#11 library** - この値は `application-prod.properties` から読み取られ、`application-prod.properties` ファイルを更新し、サーバーを再起動することでのみ変更できます。
- **Slot number of HSM** - この値は `application-prod.properties` から読み取られ、`application-prod.properties` ファイルを更新し、サーバーを再起動することでのみ変更できます。
- **HSM PIN** - 新しいHSM PINを入力できます。

Test HSM connectionボタンを選択すると、入力した**HSM PIN**を使用したHSMへの接続が試行されます。テストが成功すると、システムによって"HSM connection successful"というメッセージが表示されます。失敗した場合は"Invalid HSM Pin"と表示されます。

Updateボタンを選択すると、**HSM PIN**の値でデータベースが更新されます。**更新後はサーバーの再起動が必要です。**

警告

HSM PINテストの結果にかかわらず、HSM PINは更新されます。これは、必要に応じて、HSM PINが変更される前に、**ActiveServer**データベースを更新できるようにするためです。誤ったHSM PINを使用すると取引が失敗するため、システムを更新する前に正しいPINが入力されていることを確認してください。

Version 1.0.4


この機能はバージョン1.0.4リリースで追加されました。

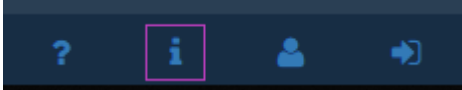
ActiveMerchantから移行

ActiveMerchant Migrationタブでは、**ビジネス管理者**ユーザーがGPayments**ActiveMerchant** (3DS1 MPI) から加盟店とアクワイアラーをインポートして、3DS1から3DS2への移行をを可能にします。

移行機能の詳細については、[ActiveMerchantからの移行ガイド](#)をご確認下さい。

ActiveServer情報を見る

About ActiveServerページは、ページの左下の  アイコンからアクセスできます。




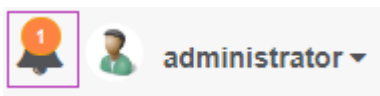
About ActiveServerには、ソフトウェア展開に関する基本情報が表示されます。これには以下が含まれます。

- **ActiveServer version** - 現在インストールされているActiveServerのソフトウェアバージョン。
- **OS name** - 使用しているオペレーティング・システム。
- **OS version** - 使用しているオペレーティング・システムのバージョン。
- **Database name** - 使用しているデータベース。
- **Database version** - 使用しているデータベースのバージョン。
- **Java edition and version** - インストールされているJavaのエディションとバージョン。
- **Node count** - このActiveServerのインスタンスに展開されているノードの数。
- **Supported 3DS version** - サポートされている3Dセキュアバージョン。
- **3DS Server reference number** - このActiveServerのインスタンスの一意の参照番号。
- **EULA** - [View]ボタンを選択すると、PDFビューアーでEULAが開き、PDFをダウンロードまたは印刷する追加オプションが表示されます。

通知

システム通知は、重要なイベントに注意が必要なときにユーザーに対して表示されます。すべてのユーザーには、パスワードの有効期限がもうすぐ切れるなど、アカウントベースの通知に関連する通知が送信されます。**System admins**には、ライセンスや使用状況のアップロードの問題など、サーバーイベントに関する通知も表示されます。

通知は、 アイコンを選択することで管理インターフェイスの右上に表示されます。



システム通知

システム通知は**System admins**に対して表示され、監視していない場合、システム実行手順に悪影響を及ぼす可能性があります。以下の表は、発生する可能性があるシステム通知とその解決方法について示しています。

通知	シナリオ	通知メッセージ	解決策
ライセンスなし	ソフトウェアが最初に初期化されたときに、システムにライセンスがなくなり、製品がアクティブ化されるまでこの通知が表示されます。	ライセンス警告 ：このインスタンスはアクティブ化されていません。 Deployment > Activation status ページで Product Activation Key (PAK) を追加してください。PAKはGPaymentsの MyAccount activation ページにあります。	製品を正しくライセンス認証するため、ユーザーはライセンスガイドに従う必要があります。
ライセンスの問題：警告	GPaymentsのライセンス・サーバーは、現在のインスタンスに関連付けられたアカウントで決済が未決であることを示しています。これは指定された期間内に予告なく無効化されます。	ライセンス警告 ：アカウントの期限が切れているため、このインスタンスはy日で無効化されます。詳細については、 GPayments support にお問い合わせください。	ユーザーが請求の問題を解決するには、適宜アカウント・チームに連絡し、GPaymentsサポートに問い合わせる必要があります。

通知	シナリオ	通知メッセージ	解決策
ライセンスの問題：停止	GPaymentsのライセンス・サーバーは、現在のインスタンスに関連付けられたアカウントで決済が未済であることを示しています。現在のインスタンスは無効化されています。	ライセンス警告： アカウントの期限が切れているため、このインスタンスは無効化されています。詳細については、 GPayments support にお問い合わせください。	ユーザーが請求の問題を解決するには、適宜アカウント・チームに連絡し、GPaymentsサポートに問い合わせる必要があります。
アップロードの問題：警告	ActiveServerインスタンスが一定期間GPaymentsライセンス・サーバーへのアップロードに失敗する場合、システムは警告プロセスを開始し、サーバーが無効化されるまで、ユーザーにエラーを修正するための期間を60日間提供します。	ライセンス警告： GPaymentsライセンス・サーバーに対する正常な認証の使用がx日間報告されていないため、このインスタンスはy日で無効化されます。詳細については、 GPayments support にお問い合わせください。	ユーザーは使用状況のアップロードが失敗する理由を調査するか、問題解決のための支援についてGPaymentsサポートに問い合わせる必要があります。
アップロードの問題：停止	ActiveServerインスタンスが60日間GPaymentsライセンス・サーバーへのアップロードに失敗した場合、サーバーは無効化されます。	ライセンス警告： GPaymentsライセンス・サーバーに対する正常な認証の使用が60日間報告されていないため、このインスタンスは無効化されました。詳細については、 GPayments support にお問い合わせください。	ユーザーは使用状況のアップロードが失敗する理由を調査するか、問題解決のための支援についてGPaymentsサポートに問い合わせる必要があります。

ユーザー通知

ユーザー通知は、イベントがアカウントに影響を与える場合にユーザーに対して表示されます。以下の表は、発生する可能性があるユーザー通知とその解決方法について示しています。

通知	シナリオ	通知メッセージ	解決策
パスワード期限切れ	ユーザーのパスワードの有効期限が切れるまで、あと7日しか残っていません。	user のパスワードは expiry date に期限切れになります。	ユーザーは User profile > Change password 画面からパスワードを更新する必要があります。

ユーザープロフィール

User profileでは、ユーザーが自身のアカウントに関する詳細を編集したり、パスワードを変更したりすることができます。

User profileにアクセスするには、管理インターフェイスの左下の**Profile**アイコンを選択します。 **edit profile**と**change password**の2つのセクションがあります。



備考

すべてのユーザーは自分のユーザープロフィールを管理できます。

プロフィールを編集

ユーザーのアカウント詳細には以下が含まれます：

- **Username (required)** - 管理インターフェイスにログインするのに使用される名前。
- **First name (required)** - ユーザーの名前。
- **Last name (required)** - ユーザーの名字。
- **Email (required)** - ユーザーの完全修飾電子メールアドレス。このアドレスは、パスワードリセットやその他のシステム通知を送信するのに使用されます。
- **Time zone** - 管理ダッシュボード上において表示される日時のローカル・タイムゾーン。
- **Two factor login status (required)** - 現在のユーザーについて2FAの**enable**（有効）または**disable**（無効）を切り替えます。無効化された状態から有効に切り替えると、ユーザーは、画面に表示される手順に従ってGoogle認証システムを使用して2FAをセットアップするように要求されます。
- **Enabled** - ユーザーはログイン時にこのアカウントに関連付けられた認証コードを入力するよう求められます。そうしないとログインが失敗します。
- **Disabled** - 2FAが無効化され、ログイン時に2FAがなくなります。

🔥 重要

インスタンスで**強制2要素認証**が有効な場合、ユーザーは2FAを無効化できなくなります。

管理APIクライアント証明書

- **Download** - 管理APIリクエストに使用されるユーザーのクライアント証明書のダウンロードを許可します。この機能の詳細については、[APIドキュメント概要](#)を参照してください。
- **Revoke** - セキュリティが侵害された場合、現在のクライアント証明書を失効させ、新しいIDで証明書を再発行します。

⚠️ 警告

Revokeした場合過去に発行されたクライアント証明書は失効され、新しいクライアント証明書をダウンロードするまではAPIリクエストができなくなります。

認証APIマスタークライアント証明書

ActiveServerは、[ここで説明されている](#)ように認証APIではX.509認証を使用してユーザーを認証します。決済代行会社等が加盟店に代わって**ActiveServer**に接続する必要がある場合、すべての加盟店ユーザーの複数のクライアント証明書の保存を避けたい場合に重宝されます。**認証APIマスタークライアント証明書**を**マーチャントトークン**と組み合わせて使用して、加盟店に代わって**ビジネス管理者ユーザー**を認証できます。マーチャントトークンでマスター証明書を使用する方法については、[こちら](#)を参照してください。

🔥 重要

ビジネス管理者のみこの証明書を管理できます。すべての加盟店に代わって認証に使用できるため注意が必要です。安全な場所に保管しましょう。

- **Download** - 認証APIリクエストに使用されるユーザーのクライアント証明書のダウンロードを許可します。この機能の詳細については、[APIドキュメント概要](#)を参照してください。
- **Revoke** - セキュリティが侵害された場合、現在のクライアント証明書を失効させ、新しいIDで証明書を再発行します。

 警告


Revokeした場合過去に発行されたクライアント証明書は失効され、新しいクライアント証明書をダウンロードするまではAPIリクエストができなくなります。

CA証明書

- **Download** - 管理APIにリクエストを送信する際に必要なサーバーCA証明書をダウンロードします。この機能の詳細については、[APIドキュメント概要](#)を参照してください。

 Tip

管理APIクライアント証明書とCA証明書の機能は[アクティブ化](#)されたインスタンスでのみご利用可能です。

 バージョン1.0.3

証明書のダウンロードはバージョン1.0.3で追加されました。

パスワード変更

ユーザーは以下のフィールドを入力することでパスワードを変更できます。

- **Current password (required)** - ユーザーの現在のパスワード。正しくない場合は、パスワードの変更が失敗します。
- **New password (required)** - ユーザーの新しいパスワード。[パスワード履歴チェックルール](#)を確認する必要があります。
 - *Requirements* - 8~100文字で、少なくとも1つの文字と1つの数字を含む必要があります。
- **Confirm new password** - ユーザーの新しいパスワード。**New password**フィールドと一致する必要があります。

APIドキュメントまとめ

はじめに

APIドキュメントの読み方

GPaymentsのActiveServer APIのドキュメントには、各APIの許可されるエンドポイントおよびリクエストHTTPメソッドが含まれています。たとえば、`/api/v1/auth/3ri` はエンドポイントURLであり、許可されるHTTPメソッドはPOSTです。

POST	<code>/api/v1/auth/3ri</code>	Perform the authentication process for a 3DS Requestor Initiated environment.	🔒
GET	<code>/api/v1/auth/app/result</code>	Request the result of an authentication challenge in an App-based environment.	🔒
POST	<code>/api/v1/auth/app/{messageCategory}</code>	Perform the authentication process for an App-based environment.	🔒
POST	<code>/api/v1/auth/brw</code>	Perform the authentication process for a Browser-based environment.	🔒
POST	<code>/api/v1/auth/brw/init/{messageCategory}</code>	Initialise authentication and get 3DS Method Data	🔒
GET	<code>/api/v1/auth/brw/result</code>	Request the result of an authentication challenge in a Browser-based environment.	🔒

エンドポイントの詳細を確認するには、エンドポイント・ボックスのいずれかをクリックし、詳細を表示します。以下のスクリーンショットのように表示されます。このスクリーンショットは、APIドキュメントの読み方を説明しています。

POST `/api/v1/auth/brw/init/{messageCategory}` Initialise authentication and get 3DS Method Data

Parameters Try it out

Name	Description
initAuthRequestBRW * required (body)	initAuthRequestBRW

Switch to view example
JSON request
type of field

Field name

* means this field is required

Field description

Example value

Conditions of field

```

InitAuthRequestBRW {
  acctID string
    examples: ActiveServer_3DS_Test_Account_000000001
    Additional information about the account optionally provided by the 3DS Requestor.
    Length: Maximum 64 characters
  Optional
  AcctInfo > {...}
  acctNumber* string
    example: 7654310430720050
    minLength: 13
    maxLength: 19
    Account number that will be used in the authorisation request for payment transactions. May be represented by PAN, token.
    Length: 13-19 characters
    Required
  acctType string
    example: 03
    Indicates the type of account. For example, for a multi-account card product. (Required in some markets (for example, for Merchants in Brazil). Otherwise, it is optional.)
    Length: 2 characters
    Values accepted:
    01 = Not Applicable
    02 = Credit
    03 = Debit
    Conditional
  Enum:
  > Array [ 3 ]
  acquirerMerchantId string
  
```

認証API

認証APIを使用すると、加盟店と決済代行会社が3Dセキュア2フローをeコマースサイトに統合できます。すべての認証APIは `/api/v1/auth/...` で始まり、RESTfulの命名規則に従っています。利用可能な主な認証フローには3つあります。

- ・ **アプリベース** - 登録されたエージェント（加盟店、デジタル・ウォレットなど）によって指定されたアプリから発生したコンシューマー・デバイス上での取引中の認証。たとえば、コンシューマー・デバイス上の加盟店のアプリ内でチェックアウト・プロセス中に発生したeコマース取引などです。
- ・ **ブラウザベース** - ブラウザを使用してWebサイトから発生したコンシューマー・デバイスまたはコンピュータ上での取引中の認証。たとえば、Webサイト内でチェックアウト・プロセス中に発生したeコマース取引などです。
- ・ **3DS Requestor Initiated (3RI)** - 直接カード会員を介さないアカウント情報の確認。たとえば、アカウントがまだ有効であることを確認するサブスクリプションベースのeコマース加盟店などです。

最新の認証APIは[こちら](#)で確認できます。

項目条件の説明

各項目には下記のいずれかの条件が記載されています。

- ・ **必須** - この項目はリクエストを送信する際に必ず必要です。ActiveServerはこの項目の有無を確認し、妥当性の確認を行います。必須項目には*印が記載されています。
- ・ **条件付き** - 項目に記載された条件が満たされていた場合にはリクエストに定義する必要があります。項目が定義されていればActiveServerは妥当性の確認を行います。送信されたJSONにこの項目が定義されていなければActiveServerは妥当性の確認を行いません。
- ・ **オプション** - オプションでリクエストに定義できます。項目が定義されていればActiveServerは妥当性の確認を行います。送信されたJSONにこの項目が定義されていなければActiveServerは妥当性の確認を行いません。

最新の認証APIは[ここ](#)からアクセスできます。

認証API - 認証方法

すべての認証APIエンドポイントは、クライアントとサーバーの相互認証が必要なX.509認証によって保護されています。これは、ActiveServerによって発行された[クライアント証明書](#)を使用して実行されます。ActiveServerは2種類のクライアント証明書を提供します。各加盟店用の加

盟店クライアント証明書、またはビジネス管理者ユーザー向けの認証APIマスタークライアント証明書です。認証APIの呼び出しを行う手順は次のとおりです。

1. このガイドに従ってインスタンスがアクティブ化されていることを確認します。
2. クライアント証明書をダウンロードする。
 - ・ ダッシュボードの加盟店ページから認証APIのクライアント証明書をダウンロードします。この証明書は .p12 形式です。
 - ・ 認証APIマスタークライアント証明書の場合、管理UIインターフェースのユーザープロフィールページからダウンロードします。
3. ユーザープロフィールページからCA証明書チェーンをダウンロードします。CA証明書は、最初の証明書がサーバーCAで、次にGPayments中間CA、最後にGPaymentsルートCAとなっている .pem 形式です。

備考

GPayments 3DSリクエスターリクエスターデモコードを利用している場合、CA証明書は既にプロジェクトに含まれています。

4. 統合ガイドまとめに従って、認証APIを呼び出すために初めから3DSリクエスターをセットアップできます。手順2で選択したクライアント証明書方法の使用に関する具体的な詳細については、デモ3DSリクエスターの設定を参照してください。

警告

インスタンスがアクティブ化されていない場合、DOWNLOAD ボタンは表示されません。

認証APIリファレンス

チャンネル	処理	エンドポイント (API v1)	エンドポイント (API v2)
BRW	InitAuth	/api/v1/auth/brw/init/ {messageCategory}	/api/v2/auth/brw/init
	Auth	/api/v1/auth/brw	authUrl from initAuth response
	Result	/api/v1/auth/brw/result	/api/v2/auth/brw/result

チャンネル	処理	エンドポイント (API v1)	エンドポイント (API v2)
	ChallengeStatus	/api/v1/auth/challenge/status	/api/v2/auth/challenge/status
APP	Auth	/api/v1/auth/app/{messageCategory}	/api/v2/auth/app
	Result	/api/v1/auth/app/result	/api/v2/auth/app/result
3RI		/api/v1/auth/3ri/{messageCategory}	/api/v2/auth/3ri

各API仕様についての詳細は[APIドキュメント](#)から確認できます。

管理API

管理APIは加盟店の管理などの管理タスクを実行するのに利用できます。すべての管理APIは `/api/v1/admin/...` で始まり、RESTfulの命名規則に従っています。たとえば、[こちら](#)で説明されている `creating merchant` は、POST APIエンドポイント `/api/v1/admin/merchants` を呼ぶことで実行できます。

最新の管理APIは[こちら](#)で確認できます。

管理API - 認証方法

管理APIへの認証は認証APIと同様に[クライアント証明書](#)を利用して実行されます。

- この[ガイド](#)に従ってインスタンスがアクティブ化されていることを確認します。
- ダッシュボードの[ユーザープロフィールページ](#)から管理APIの[クライアント証明書](#)をダウンロードします。この証明書は `.p12` 形式です。
- 同ページから[CA証明書チェーン](#)証明書をダウンロードします。CA証明書は、最初の証明書がサーバーCAで、次にGPayments中間CA、最後にGPaymentsルートCAとなっている `.pem` 形式です。
- HTTPクライアント・リクエスト内に含めることで、API認証に対してダウンロードしたクライアント証明書とCAバンドルを使用します。この方法については使用しているHTTPクライアントを参照してください。テスト方法についてはAPIクイック・スタートに従ってください。

管理APIクイック・スタート

cURLを使用してAPIを試すには、**openssl**を使用し、以下の `curl` コマンドを使用して、X.509証明書をPEM形式に変換する必要があります。

1. <https://www.openssl.org/>から**openssl**コマンドラインをダウンロードしてインストールします。
2. 上記の**管理API認証方法**セクションに従って、クライアント証明書とCA証明書をダウンロードします。
3. **OpenSSL**などを使用して `.p12` から `.pem` 形式のクライアント証明書と秘密鍵を取得する。
4. AS管理APIエントリーポイントを使用して、cURLリクエストを実行します。この場合、`/api/v1/admin/merchants` から加盟店のリストが取得されます。

```
curl -k https://your.server.address:7443/api/v1/admin/merchants --cacert cacerts.pem --cert crt.pem -v --key key.pem
```

5. 以下のようなcURLレスポンスが表示されるはずですが。

```
<HTTP/1.1 200 OK
<Expires: 0
<Cache-Control: no-cache, no-store, max-age=0, must-revalidate
<X-XSS-Protection: 1; mode=block
<Pragma: no-cache
<X-Frame-Options: DENY
<Date: Mon, 27 May 2019 09:51:59 GMT
<X-Total-Count: 1
<Connection: keep-alive
<X-Content-Type-Options: nosniff
<Strict-Transport-Security: max-age=31536000 ; includeSubDomains
<Transfer-Encoding: chunked
<Content-Type: application/json;charset=UTF-8
<Link: </api/v1/admin/merchants?page=0&size=20>; rel="last",</api/v1/admin/
merchants?page=0&size=20>; rel="first"
<
* Connection #0 to host your.server.address left intact
[{"bins":"40001 (Test
Acquirer)","merchantId":"123456789012345","name":"Test
Merchant","status":"ENABLED","merId":"48f3b030-ed1d-4f7f-
aa70-85cda288e0cb"}]%
```

OpenSSLを使用してP12からPEM形式に変更する方法

curlなどのHTTPクライアントを使用する際にクライアント証明書が.pem形式である必要があります。以下のコマンドを使用することで.p12形式のクライアント証明書から秘密鍵とクライアント証明書を.pem形式で出力することが出来ます。

1. ターミナルを開きます。
2. 以下のコマンドを使用して.p12ファイルから秘密鍵を抽出します。要求された場合はパスワードを入力します。

```
openssl pkcs12 -in YOUR_P12_FILE_NAME.p12 -nocerts -out key.pem
```

3. 以下のコマンドを使用して.p12ファイルから証明書を抽出します。要求された場合はパスワードを入力します。

```
openssl pkcs12 -in YOUR_P12_FILE_NAME.p12 -clcerts -nokeys -out crt.pem
```

4. 秘密鍵からパスワードを消す

```
openssl rsa -in key.pem -out key_no_pass.pem
```

エラーコード

このセクションでは、ActiveServerにエラーが発生する場合のエラーコードについて詳しく説明します。

認証APIエラーコードの紹介

ActiveServerには6つのカテゴリーのエラーコードがあります：

1. 3DSエラーコード

EMVCo Core Protocol Specificationsによって定義されているエラーコードです。これらのエラーコードは**3DSサーバー (ActiveServer)**、**DS**、**ACS**もしくは**3DS SDK**によって返却されることがあります。エラーを探知したコンポーネントはエラー・レスポンスを返し、そのJSONレスポンス内の `errorComponent` フィールドを自己へ入力します（例：DSがエラーを探知した場合は、DSは `errorComponent` フィールドに `D` を入力します）。もしエラーはActiveServer以外のコンポーネントで探知した場合、ActiveServerは同様のJSONエラーを**3DSリクエスター**へ返します。 `errorMessageType`、`errorDetail` および `errorDescription` のフィールドによってエラー内容を解釈できます。各フィールドに関する説明は `ApiResponse` をご参照ください。

2. 取引エラーコード

ActiveServerが定義する取引エラーコードです。これらのエラーは3DSサーバーが探知するため、`errorComponent` のフィールドは常に `S` に設定されます。

3. 汎用エラーコード

3DSエラーコードもしくは**取引エラーコード**に含まれないエラーコードは全て**汎用エラーコード**として返却されます。また、管理APIからも返却されます。認証APIの説明については**認証API説明**をご参照ください。

4. セキュリティ・エラーコード

5. ユーザー・エラーコード

6. セットアップ・エラーコード

また、

- **3DSエラーコード、取引エラーコード、汎用エラーコードのみ**認証API(`/api/v2/auth/***`)から返却されます。

- ・ **セキュリティ・エラーコード、ユーザー・エラーコード、セットアップ・エラーコード**は認証APIから返却されません。
- ・ 各エラーコードは、以下テーブルに記載する関連のHTTPステータスコードで返却されます。
- ・ **説明欄**にはエラーが返却されるシナリオ例が挙げられます。また、一部のエラーコードに関しては推奨のエラー解決方法が記載されています。

注意

認証API v2では返却されませんのタグの付いたエラーコードは、`/api/v2/auth/***` においてレスポンスとして返却されません。

3DSエラーコード (101 ~ 405)

コード	名前	HTTPステータスコード	説明
101	MESSAGE_RECEIVED_INVALID	400	受信したメッセージが無効です。メッセージはAReq、ARes、CReq、CRes、PReq、PRes、RReq、RResのいずれではありません。 例) 3DSサーバーが送信したAReqに対して、DSからAResもしくはError以外のメッセージが返却された場合、など。

コード	名前	HTTPステータスコード	説明
102	MESSAGE_VERSION_NUMBER_NOT_SUPPORTED	400	サポートされていないメッセージバージョン番号です。受信したメッセージバージョン番号は受信するコンポーネントにとって無効です。例) DSが送信する messageVersion フィールドの値が無効な値の場合、もしくはACSが対応しない値の場合、など。
103	SENT_MESSAGES_LIMIT_EXCEEDED	500	送信済みメッセージが上限を超えました。DSへ送信するPReq件数の上限を超えました。 認証API v2では返却されません (PReqはActiveServerとDSの間の内部処理であり、認証フロー外のため)。

コード	名前	HTTPステータスコード	説明
201	REQUIRED_DATA_ELEMENT_MISSING	400	<p>仕様に必須と指定されたメッセージ項目が入力されていません。このエラーはリクエストに必須とされたフィールドが欠けている場合に返却されます。例)</p> <p><code>/api/v2/auth/brw</code> へのAPIコールに <code>messageCategory</code> フィールドが入力されていない場合、など。もし <code>errorMessageType</code> フィールドは <code>AReq</code> もしくは入力されていない、かつ <code>errorComponent</code> フィールドは <code>s</code> の場合、3DSリクエスターから送信されたリクエストは認証APIに必須と指定されたフィールドを入力していません。その場合、<code>errorDetail</code> に記載されたフィールドは正しくリクエストに入力されているかをご確認ください。</p>
202	CRITICAL_MESSAGE_EXTENSION_NOT_RECOGNISED	400	<p>重要なメッセージ拡張が存在しません。</p> <p><code>messageExtension</code> に識別子が欠けている場合、DSもしくはACSからこのエラーコードが返却されます。</p>

コード	名前	HTTPステータスコード	説明
203	FORMAT_OF_ONE_OR_MORE_DATA_ELEMENTS_IS_INVALID_ACCORDING_TO_THE_SPECIFICATION	400	<p>情報項目が正しい形式ではないか、仕様に無効と定義された形式で入力されています。フィールドに入力された情報が正しい形式ではない場合にこのエラーコードが返却されます。例)</p> <p><code>/api/v2/auth/brw</code> のAPIコールの <code>purchaseAmount</code> フィールドに数字以外のデータが入力されている場合、など。もし <code>errorMessageType</code> フィールドは <code>AReq</code> もしくは入力されていない、かつ <code>errorComponent</code> フィールドは <code>S</code> の場合、3DSリクエスターから送信されたリクエストは認証APIに必須と指定されたフィールドを入力していません。その場合、<code>errorDetail</code> に記載されたフィールドは正しくリクエストに入力されているかをご確認ください。</p>
204	DUPLICATE_DATA_ELEMENT	400	<p>重複した情報項目が見つかりました。</p>
301	TRANSACTION_ID_NOT_RECOGNISED	400	<p>受信した取引IDは受信するコンポーネントにとって無効です。例) <code>/api/v2/auth/brw</code> では3DSリクエスターが <code>threeDSServerTransID</code> を生成するが、これは <code>/api/v2/auth/brw/init</code> で返却される <code>threeDSServerTransID</code> とは異なります。</p>

コード	名前	HTTPステータスコード	説明
302	DATA_DECRYPTION_FAILURE	500	技術上の理由などにより、受信側がデータ復号に失敗しました。SDK暗号化データの復号に失敗した場合にDSから返却されます。
303	ACCESS_DENIED_INVALID_ENDPOINT	401	APIリクエストのエンドポイントが無効です。リクエストURLを確認してください。Reference numberが登録されたコンポーネントと一致しません（例：ACSからDSへ送信した acsReferenceNumber は無効です）。
304	ISO_CODE_INVALID	400	ISOテーブルにより、国名もしくは貨幣の値のISOコードは無効です。
305	TRANSACTION_DATA_NOT_VALID	400	取引データが無効です。エラー詳細を確認して、取引データが無効の理由をご参照ください。
306	MERCHANT_CATEGORY_CODE_MCC_NOT_VALID_FOR_PAYMENT_SYSTEM	400	加盟店カテゴリー・コード(MCC)が無効です。無効の加盟店カテゴリー・コード(MCC)がAReqに入力されると、DSからこのエラーコードが返却されます。
307	SERIAL_NUMBER_NOT_VALID	500	シリアル番号が無効です。 認証API v2では返却されません （PReqはActiveServerとDSの間の内部処理であり、認証フロー外のため）。

コード	名前	HTTPステータスコード	説明
402	TRANSACTION_TIMED_OUT	408	取引がタイムアウトしました。このエラーコードはActiveServerからDSへリクエストを送信する際に取引がタイムアウトする場合に返却されます。例) AReqをDSへ送信する場合、など。
403	TRANSIENT_SYSTEM_FAILURE	500	システムが短時間故障しました。例) 処理速度の遅いバックエンド・システムによる障害、など。
404	PERMANENT_SYSTEM_FAILURE	500	システムが恒久的に故障しました。例) クリティカル・データベースへアクセス不能、など。DS設定がActiveServerに正しく設定されていない場合に返却されます (例: DSへ接続するためのクライアント証明書が正しくインストールされていない、など)。
405	SYSTEM_CONNECTION_FAILURE	500	システムへ接続できませんでした。例) 送信側が受信側と接続を確立できない場合、など。

取引エラーコード (1001 ~ 1027)

コード	名前	HTTPステータスコード	説明
1001	DIRECTORY_SERVER_NOT_FOUND	500	指定されたPANに属する国際ブランドのディレクトリー・サーバーが見つかりませんでした。管理UIにあるデフォルトURLフィールドが空の場合に返却されます。管理UIのデフォルトURLが正しく設定されていることをご確認ください。
1002	ERROR_SAVE_TRANSACTION	500	取引の保存中にエラーが発生しました。データベースへ取引情報を保存する際に失敗する場に返却されます。
1003	ERROR_SAVE_TRANSACTION_MESSAGE	500	取引メッセージの保存中にエラーが発生しました。 認証API v2では返却されません。 ただし生のメッセージ（例：生のAReq JSONメッセージ）を保存する際にエラーが発生する場合、引は失敗しません。
1004	UNHANDLED_EXCEPTION	500	取引中に未処理の例外が発生しました。エラー詳細を確認するか、エラーログを報告してください。
1005	PAN_NOT_PARTICIPATING	400	カード番号(PAN)は加入していません。 認証v2では返却されません。
1009	MERCHANT_INTERFACE_DISABLED	400	加盟店のインターフェイスは無効と設定されています。 認証API v2では返却されません。 併りに MERCHANT_ID_THREEDS_REQUESTOR_ID_INVALID (1026) が返却されます。
1011	INVALID_LICENSE	403	使用しているActiveServerライセンスは無効です。速やかにGPayments社へお問い合わせください。

コード	名前	HTTPステータスコード	説明
1013	INVALID_TRANSACTION_ID	400	3DSサーバーの取引IDを認識できません。リクエスト内の <code>threeDSSTransID</code> が無効の場合に返却されます。
1014	INVALID_REQUESTOR_TRANSACTION_ID	400	3DSリクエスターの取引IDを認識できません。 <code>threeDSRequestorTransID</code> がUUID形式でない場合に返却されます。
1015	THREEDS_REQUESTOR_NOT_FOUND	400	3DSリクエスターIDもしくは加盟店IDが無効です。 認証API v2では返却されません 。クライアント証明書もしくは加盟店IDが無効の場合、 <code>MERCHANT_ID_THREEDS_REQUESTOR_ID_INVALID</code> (1026) が返却されます。
1016	MISSING_REQUIRED_ELEMENT	400	必要な項目が見つかりません。認証APIで必要となっているフィールドが入力されていない場合に返却されます。
1018	ELEMENT_NOT_DEFINED	400	情報項目は仕様に定義されていません。 認証API v2では返却されません 。
1019	PROTOCOL_OLD	500	プロトコルのバージョンが古いです。 認証API v2では返却されません 。
1020	ERROR_TRANSMISSION_DATA	500	データ通信中にエラーが発生しました。DSへ送信、もしくはDSへリクエストを送受信する際エラーが発生した場合に返却されます。ただし、タイムアウトによってエラーが発生した場合は <code>TRANSACTION_TIMED_OUT</code> (402) が返却されます。また、通信が確立されていない場合は <code>DIRECTORY_SERVER_NOT_AVAILABLE</code> (1001) が返却されます。

コード	名前	HTTPステータスコード	説明
1021	PRIOR_TRANS_ID_NOT_FOUND	400	カード会員の前回の取引IDはデータベースに 見つからないか、その形式が無効です。リク エスト内の <code>priorTransID</code> が無効のUUID形式で送 信する場合に返却されます。 <code>priorTransID</code> フ ールドにはカード会員の前回の取引の <code>threeDSServerTransID</code> を入力してください
1022	INVALID_FORMAT	400	複数のデータ項目は仕様に定義された形式に じていません。情報項目を無効の形式で送信 する場合に返却されます。例) <code>/api/v2/auth/brw</code> のリクエストで設定した <code>browserInfo</code> の値がActiveServerが収集した <code>browserInfo</code> の値と異なります。
1023	CARD_RANGE_IS_NOT_VALID	400	指定されたカード・レンジは無効です。 認証 API v2では返却されません。
1024	CACHE_UPDATE_IS_DISABLE	500	キャッシュ更新の設定は無効と設定されてい ず。 認証API v2では返却されません。
1025	CACHE_REFRESH_INTERVAL_IS_NOT_SET	500	キャッシュのリフレッシュ間隔が設定されて ません。 認証API v2では返却されません。
1026	MERCHANT_ID_THREEDS_ REQUESTOR_ID_INVALID	400	認証リクエストに無効の <code>merchantId</code> を送信 います。リクエスト内の <code>merchantId</code> は加盟 クライアント証明書の <code>merchantId</code> と一致す とをご確認ください。または マスターク ライアント証明書 を使用する場合、 <code>merchantToker</code> 一致することをご確認ください。マスターク ライアント証明書失効させる度に、必ずク ライアント証明書もしくはAPIリクエスト内の <code>merchantToken</code> を更新してください。
1027	UNSUPPORTED_API_VERSION	403	サポートされないAPIバージョンでリクエ ストを送信する場合に返却されます。例) API v はAWS KMS機能をサポートしません。

汎用エラーコード (2000 ~ 2009)

コード	名前	HTTPステータスコード	説明	認証API説明
2000	NOT_FOUND	404	リソースが見つかりません。	認証API v2では返却されません
2001	DUPLICATE_RECORD	409	レコードがすでに存在します。	認証API v2では返却されません
2002	VALIDATION_ERROR	400	入力が無効です。	リクエストが正しくJSONメッセージに生成されていない場合に返却されます。
2003	INVALID_REQUEST	400	リクエストが無効です。	認証API v2では返却されません
2004	CONCURRENCY_FAILURE	409	ノードの更新に失敗しました。	認証API v2では返却されません
2005	ACCESS_DENIED	401	アクセスが拒否されました。	エラー詳細を確認して、アクセスが拒否された理由をご参照ください。
2006	METHOD_NOT_SUPPORTED	405	リクエストHTTPメソッドがサポートされていません。	認証API v2では返却されません

コード	名前	HTTPステータスコード	説明	認証API説明
2007	INTERNAL_SERVER_ERROR	500	内部サーバーエラー。	ActiveServerに内部サーバーエラーが発生しています。設定、もしくはセットアップの問題による可能性があります。エラー詳細をご確認ください。
2008	DATA_INTEGRITY_VIOLATION_ERROR	400	指定された値は整合性制約に違反しています。整合性制約に違反する情報を入力した場合に返却されます。例) 一意の主キーはテーブルに入力されていません。	認証API v2では返却されません
2009	SESSION_TIMED_OUT	408	セッションがタイムアウトしました。	取引が既に終了した場合に返却されます。

セキュリティ・エラーコード (3001 ~ 3024)

コード	名前	HTTPステータスコード	説明
3001	JDK_NOT_SUPPORT_SHA224WITHRSA	500	使用しているJDKはRSAアルゴリズムでのSHA224に対応しません。
3002	NO_SUCH_ALGORITHM	500	そのようなアルゴリズムはありません。

コード	名前	HTTP ステータス コード	説明
3003	INVALID_CERT	400	証明書の公開鍵には対応する秘密鍵との互換性がありません。
3004	INVALID_CHAIN	400	CA証明書ストアで1つ以上の中間証明書が見つからないため、ActiveServerは完全な証明書チェーンを構築できません。再度試す前に、完全なチェーンを含む証明書をインストール/インポートするか、見つからない中間証明書をインストールする必要があります。
3005	NO_PRIVATE_KEY_FOUND	400	秘密鍵が見つかりませんでした。
3006	INVALID_CERTIFICATE_CONTENT	400	証明書のコンテンツが無効です。
3007	CERTIFICATE_IO_READ	400	証明書を読み取ることができませんでした。
3008	SUCH_PROVIDER_EXCEPTION	500	そのようなプロバイダー例外はありません。
3009	NO_KEY	400	このオブジェクトには既存のキーがないため、証明書をインストールできませんでした。
3010	CERTIFICATE_CHAIN_BAD_FORMAT	400	証明書チェーンの形式が無効です。
3011	MISMATCHED_PASSWORDS	400	パスワードフィールドが一致しません。
3012	IMPORT_CERTIFICATE	400	クライアント証明書をインストールしてください。
3013	IMPORT_NO_CERTIFICATE	400	エクスポートする証明書がありません。
3014	FAILED_TO_INITIALIZE	500	初期化に失敗しました。
3015	ENCRYPTION_FAIL	500	暗号化に失敗しました。

コード	名前	HTTPステータスコード	説明
3016	DECRYPTION_FAIL	500	復号化に失敗しました。
3017	INVALID_HSM_PROVIDER	500	ハードウェア暗号化用に指定されたプロバイダー名はサポートされていません。
3018	INVALID_PKCS11_CONFIG	500	PKCS11設定パスが無効です。
3019	FAILED_TO_INITIALIZE_PKCS11	500	PKCS11の初期化に失敗しました。
3020	IMPORT_FAIL	500	インポートに失敗しました。
3021	NOT_SUPPORTED_IBM_PROVIDER	500	SUNプロバイダーのみがサポートされています。
3022	UNABLE_TO_LOAD_KEYSTORE	500	キーストアのロードに失敗しました。
3023	UNABLE_TO_LOAD_CERTIFICATE	500	証明書のロードに失敗しました。
3024	INVALID_KEY_SIZE	500	キーサイズが無効です。

ユーザー・エラーコード (4000 ~ 4022)

コード	名前	HTTPステータスコード	説明
4000	DUPLICATE_EMAIL	400	電子メールはすでに使用されています。
4001	LAST_ADMIN_DELETE_NOT_ALLOWED	400	このアクションを実行するには、少なくともシステム管理者ユーザーである必要があります。
4002	ACCOUNT_IS_LOCKED	401	アカウントがロックされています。

コード	名前	HTTPステータスコード	説明
4003	ACCOUNT_IS_DISABLED	401	アカウントが無効です。
4004	ACCOUNT_WILL_BE_LOCKED	401	あと1回誤るとアカウントがロックされます。パスワードを忘れた場合は、「Lost your password」をクリックしてください。
4005	ACCOUNT_WAS_LOCKED	401	パスワードは1時間ロックされます。
4006	ACCOUNT_IS_INACTIVE	401	アカウントがアクティブ化されていません。
4007	PASSWORD_POLICY_MATCH	401	パスワードは最低8文字であり、少なくとも1つの文字と1つの数字が含まれている必要があります。
4008	LOGIN_ALREADY_IN_USE	401	ユーザー名はすでに使用されています。
4009	EMAIL_ALREADY_IN_USE	401	電子メールはすでに使用されています。
4010	INVALID_TOTP_CODE	400	totp認証コードが無効です。
4011	EMAIL_SENDING_FAILED	400	電子メールの送信に失敗しました。
4012	EMAIL_NOT_REGISTERED	400	電子メールが登録されていません。
4014	FAILED_TO_CREATE_ACCOUNT	500	アカウントの作成に失敗しました。
4015	TWO_FA_MANDATORY	400	2要素認証の使用は必須です。
4016	PASSWORD_EXPIRED	403	ユーザーのパスワードの有効期限が切れました。
4017	PASSWORD_EXPIRED_WARNING	403	ユーザーのパスワードが期限切れになります。

コード	名前	HTTPステータスコード	説明
4018	PASSWORD_HISTORY_MATCHED	403	パスワードが過去のパスワードと一致しました。
4019	INVALID_TOKEN	400	トークンが無効です。
4020	INVALID_HSM_PIN	400	HSM PINが無効です。
4021	INVALID_PASSWORD	400	パスワードが無効です。
4022	EMAIL_INVALID_ACTIVATION	403	アカウントアクティブ化コードが無効です。

セットアップ・エラーコード (5000)

コード	名前	HTTPステータスコード	説明
5000	SETUP_NOT_ALLOWED	500	セットアップが許可されていません。

用語集

このページでは、3Dセキュア2に関連する用語の一覧を提供しています。このドキュメントの他の場所で使用されていないものもありますが、内容を完全にするために記載しています。用語を調べる場合や、よくわからない単語に遭遇したときにこのページを参照してください。

Term	Acronym	Definition
3DS Client		EMV 3-Dセキュア・プロトコルを開始するための消費者と3DSリクエストターのやり取りを容易にする、ブラウザベースまたはモバイルアプリのオンラインショッピングサイトなどの消費者向けコンポーネント。
3DS Integrator		3DSリクエストター環境を容易化および統合し、オプションで加盟店とアクワイアラー間の統合を容易にするEMV 3-Dセキュア参加者。
3DSリクエストター		AReqメッセージとも呼ばれるEMV 3-Dセキュア認証リクエストのイニシエーター。たとえば、購入フロー内で認証をリクエストする加盟店やデジタル・ウォレットなどです。
3DSリクエストター App		3DS SDKを使用して、3-Dセキュア取引を処理できるコンシューマー・デバイス上のアプリ。3DSリクエストターアプリは、3DS SDKとの統合によって有効化されます。
3DSリクエストター Environment		これは、通常、3DSインテグレーターによって容易化される加盟店/アクワイアラーロードメインの、3DSリクエストター制御コンポーネントを指します。これらのコンポーネントには、3DSリクエストターアプリ、3DS SDK、3DSサーバーが含まれます。3DSリクエストター環境の実装は、3DSインテグレーターで定義されているとおりに変化します。
3DS Software Development Kit	3DS SDK	3-Dセキュア・ソフトウェア開発キット。3DSリクエストターアプリに組み込まれているコンポーネント。3DS SDKは、3DSサーバーの代わりに3-Dセキュアに関連する機能を実行します。
3DS Requestor Initiated	3RI	アカウントがまだ有効であることを確認するために、3DSリクエストターによって開始された3-Dセキュア取引。主なユースケースは、サブスクリプションユーザーの決済方法がまだ有効であることを確認するために、加盟店が非決済取引を実行することを要求する定期的な取引（TVサブスクリプション、公共料金決済など）です。

Term	Acronym	Definition
3DS Server	3DSS	オンライン取引を処理し、3DSリクエスターおよびディレクトリ・サーバー間の通信を容易にする3DSインテグレーターのサーバーまたはシステムを指します。
3-D Secure	3DS	3ドメイン・セキュア。バージョン2以降では、決済、非決済、およびアカウント確認カード取引の安全な処理が可能なeコマース認証プロトコル。
Access Control Server	ACS	イシューードメインで機能するコンポーネント。カード番号およびデバイスタイプで認証が利用可能かどうかを確認し、特定のカード会員を認証します。
Attempts		EMV 3DS仕様では、決済認証が利用できない場合に認証試行の証明が生成されるプロセスを示すのに使用されます。試行のサポートは各DSで判断されます。
Authentication		3-Dセキュアのコンテキストでは、eコマース取引を行った個人に決済カードを使用する資格があることを確認するプロセスです。
Authentication Request Message	AREq	認証プロセスを開始するために、DSを介して3DSサーバーからACSに送信されるEMV 3-Dセキュア・メッセージ。
Authentication Response Message	ARes	認証リクエスト・メッセージに対するレスポンスで、DSを介してACSから返されるEMV 3-Dセキュア・メッセージ。
Authentication Value	AV	オーソリゼーション処理中にオーソリゼーションシステムが認証結果の完全性を検証するための方法を提供する、ACSによって生成された暗号値。AVアルゴリズムは各決済システムによって定義されます。
Authorisation		イシューアまたはイシューアの代理のプロセッサーが決済の取引を承認するプロセス。
Authorisation System		決済システムがイシューアおよびアクワイアラーに対してオンライン金融処理、オーソリゼーション、清算、決算サービスを提供するシステムおよびサービス。
Bank Identification Number	BIN	発行金融機関を一意に識別する決済カードアカウント番号の最初の6桁。ISO 7812ではイシューア識別番号 (IIN) とも呼ばれます。
Base64		RFC 2045で定義される、認証値データ要素に適用されるエンコーディング。

Term	Acronym	Definition
Base64 URL		RFC 7515で定義される、3DSメソッドデータ、デバイス情報、CReq/CResメッセージに適用されるエンコーディング。
Card		カードは「EMV 3-Dセキュア・プロトコルおよびコア機能仕様」において、決済カードのアカウントと同義です。
Certificate Authority	CA	デジタル証明書を発行するエンティティ。
Cardholder		カードの発行対象の個人、またはそのカードの使用がオーソリゼーションされた個人。
Challenge		ACSが3DSクライアントと通信し、カード会員とのやり取りを通じて追加情報を取得するプロセス。
Challenge Flow		「EMV 3-Dセキュア・プロトコルおよびコア機能仕様」で定義される、カード会員の操作に関する3-Dセキュア・フロー。
Challenge Request Message	CReq	3DS SDKまたは3DSサーバーによって送信されるEMV 3-Dセキュア・メッセージ。認証プロセスをサポートするためにカード会員からACSに追加情報が送信されます。
Challenge Response Message	CRes	CReqメッセージに対するACSレスポンス。カード会員認証の結果を示したり、アプリベース・モデルの場合は、認証の完了にはさらなるカード会員操作が必要であることを示す信号を示したりすることができます。
Consumer Device		カード会員が認証や購入を含む決済アクティビティを実行するのに使用するスマートフォン、ノートPC、またはタブレットなど、カード会員によって使用されるデバイス。
Device Channel		取引の発生元のチャンネルを示します。以下のいずれか： <ul style="list-style-type: none"> ・ アプリベース (01-APP) ・ ブラウザベース (02-BRW) ・ 3DSリクエスター起因 (03-3RI)
Device Information		認証プロセスで使用される、コンシューマー・デバイスによって提供されるデータ。

Term	Acronym	Definition
Directory Server	DS	相互運用ドメインで機能するサーバーコンポーネント。以下を含むいくつかの機能を実行します：3DSサーバーの認証、3DSサーバーおよびACS間のメッセージのルーティング、3DSサーバー、3DS SDK、および3DSリクエスターの検証。
Directory Server Certificate Authority	DS CAまたはCA DS	相互運用ドメインで機能するコンポーネント。認証局（DS）が選択したデジタル証明書を生成し、3-Dセキュアに参加しているコンポーネントに分配します。通常、DSが接続されている決済システムがCAを操作します。
Directory Server ID		決済システムに対して一意の登録済みアプリケーション・プロバイダー識別子（RID）。RIDはISO 7816-5標準によって定義されています。
Electronic Commerce Indicator	ECI	カード会員を認証するための試行の結果を示す、ACSによって提供される決済システム固有の値。
Frictionless		カード会員の介入なく行われた場合に、認証プロセスを説明するのに使用されます。
Frictionless Flow		EMVCoコア仕様セクション2.5.1で定義される、カード会員が介入しない3-Dセキュア・フロー。
Message Authentication Code	MAC	第三者によるデータの改変および偽造から送信者および受信者を保護する対称（秘密鍵）暗号化方式。
Merchant		決済カードを使用して行われた決済を受理するためにアクワイアラーと契約しているエンティティ。加盟店は、カード番号を取得してカード会員のオンラインショッピング体験を管理し、その後決済認証を行う3DSサーバーに制御を移します。
Non-Payment Authentication	NPA	取引が付随しない3DS認証タイプ。アイデンティティ確認に使用されません。
One-Time Passcode	OTP	コンピューターシステムまたはその他のデジタル・デバイス上の1回ログイン・セッションまたは取引に対してのみ有効なパスコード。

Term	Acronym	Definition
Out-of-Band	OOB	3-Dセキュア・フロー外で並行して実行されるチャレンジ・アクティビティ。最終チャレンジ・リクエストは、ACSによってチェックされるデータの送信には使用されませんが、認証が完了したことをのみを通知します。ACS認証方式または実装は3-Dセキュア仕様では定義されません。
Preparation Request Message	PReq	DSカード範囲に対応するACSおよびDSプロトコルバージョン、およびオプションで3DSサーバーの内部ストレージ情報を更新するための3DSメソッドURLを要求する、3DSサーバーからDSに送信される3-Dセキュア・メッセージ。
Preparation Response Message	PRes	3DSサーバーの内部ストレージを更新できるようにするための、DSカード範囲、ACSおよびDSのアクティブ・プロトコル・バージョン、3DSメソッドURLを含むPReqメッセージに対するレスポンス。
Proof or authentication attempt		試行を参照。
Registered Application Provider Identifier	RID	決済システムに対して一意の登録済みアプリケーション・プロバイダー識別子 (RID)。RIDはISO 7816-5標準によって定義されており、ISO/IEC 7816-5登録局によって発行されます。RIDは5バイトです。
Results Request Message	RReq	3DSサーバーに認証取引の結果を送信するために、ACSからDSに送信されるメッセージ。
Results Response Message	RRes	結果リクエスト・メッセージの受信を確認するために、DSを介して3DSサーバーからACSに送信されるメッセージ。

ドキュメントバージョン

リリースノート

ActiveServer v1.3.2

[リリース日: 2020年3月30日]

変更	詳細
[#619] 改善	Directory Serverの設定と証明書ページの読み込みのパフォーマンスを最適化しました。
[#771] 改善	ロードバランシングで「X-forwarded」ヘッダーが使用されている場合に、ポートの確認プロセスをより柔軟に拡張しました。
[#774] 改善	認証の証明として提供される、データベースに保存されている「Authentication Value(CAVV)」の暗号化を追加しました。
[#781] 改善	管理APIの呼び出しに対するセキュリティを強化しました。
[#799] 改善	起動時にKMSの接続テストを追加しました。KMSが正しく初期化されていない場合にエラーになります。
[#801] 修正	API v2「enrol」リクエスト中にエラーを返す可能性があった問題を修正しました。API「enrol」リクエストのログ出力を追加しました。
その他	小規模なバグ修正、パフォーマンスとセキュリティの強化

ActiveServer v1.3.1

[リリース日: 2020年3月19日]

変更	詳細
[#468] 改善	セットアップウィザードのセキュリティを強化

変更	詳細
[#664] 改善	ローカルファイルへのログ出力を無効にする機能が追加されました。
[#718] 修正	購入金額を使用して取引レポートを検索するときに発生したエラーを修正しました。
[#735] 修正	認証API v2決済認証 (PA) でオプションの「purchaseCurrency」フィールドが提供されなかったときに発生していたエラーを修正しました。
[#736] 修正	3RIチャネル認証の認証API v2で「merchantName」の上書きに関する問題を修正しました。
[#747] 改善	「/api/v2/brw/result」および「/api/v2/app/result」API応答に「dsReferenceNumber」および「acsReferenceNumber」を追加しました
[#753] 改善	認証APIエラー応答で返される追加のHTTPステータスコードのサポートを追加しました
[#754] 改善	外部URL、API URL、管理URL、3DSサーバーURLのURLを入力した際の妥当性確認を追加しました。これにより、追加のパスやクエリ文字列を含めることができなくなります。
その他	小規模なバグ修正、パフォーマンスとセキュリティの強化

ActiveServer 1.3.0

[リリース日: 2020年2月7日]

変更	詳細
[#347] 改善	AWS KMSを暗号化タイプとして使用するためのサポートが追加されました
[#375] 修正	管理APIを介してログインページへの不正アクセスを許可する可能性があった問題を修正しました
[#589] 改善	認証API v1に新しいオプションフィールド (addrMatch) を追加し、ユーザーがカード会員の請求先住所と配送先住所が一致するかどうかを指定できるようになりました。

変更	詳細
[#621] 改善	/enrol APIを呼び出すときに最大19桁のPANを処理するためのPResキャッシュのサポートが追加されました。
[#637] 改善	潜在的な競合を防ぐためにキーストアの処理プロセスを改善しました。
[#639] 改善	PANストレージと暗号化キーへの変更を含む、認証APIv2を追加し、マスク化されていないPANは保存されなくなりました。v2の取引ではPANは常にマスク化され保存されます。
[#642] 改善	管理インターフェースでのHTTP 302リダイレクトのエラー処理の改善
[#644] 改善	カードレンジキャッシュのPReqメッセージプロセスのパフォーマンスを改善
[#652] 変更	認証の証明として提供された「authenticationValue」(CAVV)は7日間のみ保存され、その後マスクされます
[#656] 改善	ログファイルを収集するフォルダを指定するための構成を追加
[#657] 修正	「threeDSMethodData」が受信されない場合にトランザクション処理が停止する可能性がある問題を修正しました。
[#660] 改善	データベース接続プールのパフォーマンスの向上
[#679] 変更	システムのPANに最初の6桁と最後の4桁が表示され、残りの桁は切り捨てられてマスクされます。
[#682] 変更	s3.bucket-nameプロパティのパス設定をより柔軟に変更
その他	小規模なバグ修正、パフォーマンスとセキュリティの強化

ActiveServer v1.2.2

[リリース日: 21/11/19]

変更	詳細
[#167] 改善	3DSメソッド通知プロセスを改善
[#627] 変更	HTTPとHTTPSの両方に適用されるようにセキュリティヘッダーポリシーを変更
[#628] 修正	管理UIのコンテンツセキュリティポリシーヘッダーの問題を修正
[#629] 修正	フィールド authenticationType の妥当性チェックの問題を修正

ActiveServer v1.2.1

[リリース日: 15/11/19]

変更	詳細
[#584] 改善	サーバーリクエストのHTTPレスポンスステータスをノーマライズしました
[#586] 改善	ページセキュリティを強化するため、追加のHTTPヘッダーを追加しました
[#616] 修正	UAH(980)の通貨コード指数の問題を修正しました
[#617] 修正	アクワイアラーBINを使用した加盟店の検索時の問題を修正しました

ActiveServer v1.2.0.1

[リリース日: 04/11/19]

変更	詳細
[#610] 修正	Oracleデータベース初期化の問題を修正しました

ActiveServer v1.2.0

[リリース日: 01/11/19]

変更	詳細
[#293] 改善	EMVCoの条件付きフィールドEMV Payment Token Indicatorをサポートするため、payTokenIndをAuth APIに追加しました
[#351] 変更	今後は、加盟店を作成または編集する際には、一意のMerchant名とMerchant IDの組み合わせが必要となります
[#404] 修正	加盟店ユーザーがダッシュボードにアクセスできない問題を修正しました
[#494] 変更	EMVCoの仕様に従い、Base64urlエンコーディングからパディングを削除しました
[#542] 改善	ActiveMerchantからMerchantとAcquirerのプロファイルをインポートする際のサポートを追加しました
[#546] 変更	今後は、取引レポートの購入金額の表示および検索対象が、補助単位ではなく通貨単位となります
[#561] 改善	データベーステーブルのパフォーマンスのインデックス作成が改善されました
[#581] 改善	DS証明書が追加された時にインスタンスの再起動を促す警告ダイアログを追加しました
[#583] 改善	管理インターフェイスに個別にアクセスできるよう、新しいAdmin URL設定を追加しました
[#590] 改善	サーバー起動中のキーストア初期化プロセスを改善しました
[#599] 変更	Cache更新の間隔、Preparation Response (PRes)のタイムアウト、およびPreparation Response (PRes)のタイムアウトのグローバル設定を削除しました。これらの設定は今まで通り、DS設定ページで国際ブランドごとに管理できます
その他	小規模なバグ修正、パフォーマンスとセキュリティの強化

ActiveServer v1.1.4

[リリース日: 27/09/19]

変更	詳細
[#559] 改善	外部URLを更新すると、空の値がある場合、Directory Server設定のすべての3DSサーバーURLが自動的に更新されるようになりました
[#579] 変更	Mastercard自動コンプライアンステスト中に発生したデータベースインデックスエラーを修正
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.1.3

[リリース日: 20/09/19]

変更	詳細
[#573] 修正	特定のHSMのキー生成に関する問題を修正。
[#574] 改善	キーをローテーション際に確認ダイアログを追加しました。
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.1.2

[リリース日: 19/09/2019]

変更	詳細
[#383] 改善	ActiveServer EULAは、管理UIのページからアクセスするように変更し、リリースパッケージからは削除されました。
[#424] 変更	管理APIを介したAcquirer BINの管理では、システム内のAcquirerのUUIDではなくBINの文字列値を使用するようになりました。そのため、Acquirer管理APIエンドポイントは削除されました。管理UIは、既存のAcquirer BINを選択するか、値を入力できるようになりました
[#450] 変更	admin.portを設定すると、すべての管理インターフェースUI要求がそのポート番号に制限されるようになりました

変更	詳細
[#507] 改善	BRW、APP、3RIチャネルのAPI応答にdsTransIDとmessageVersionを追加しました
[#519] 改善	システム内の加盟店の代わりに認証に使用できる認証APIマスタークライアント証明書を追加しました
[#547] 改善	Directory Server証明書ページで既存の秘密鍵を上書きする可能性がある場合、ユーザーに追加の警告ダイアログを追加しました
[#548] 改善	新しいチャレンジステータスAPIエンドポイント (/api/v1/auth/challenge/status) が追加され、チャレンジ要求をキャンセルするときに3DSリクエスターがキャンセル理由をオプションで提供できるようになりました。
[#552] 改善	インストールウィザードのパフォーマンスを強化しました
[#555] 変更	ActiveServerのリスナーポートを内部利用のみに変更しました。
[#557] 変更	3DS SDKにのみに必要なタイムアウト設定なのでCResおよびACSメソッドタイムアウト設定を削除しました。
[#560] 変更	マーチャントの管理APIエンドポイントを変更し（証明書のエクスポート/失効、キーローテーション）、リクエストとレスポンスから未使用のパラメーターを削除しました。設定用の管理APIエンドポイントも削除されました
[#565] 修正	ユーザーがセッション失敗回数の上限を超えることができるという問題を修正しました。
[#569] 修正	Directory Serverの設定でPReq値が設定されていない場合、PReqが送信されない問題を修正しました。
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.1.1

[リリース日: 30/08/2019]

変更	詳細
[#509] 修正	3DSリクエストのサンプルコードv1.1をサポートするために、未完了のトランザクションをタイムアウトするための新しい監視エンドポイントが追加されました
[#537] 修正	認証APIにオプションのマーチャント名項目を追加して、マーチャントプロフィールのマーチャント名を上書きできるようにしました。
[#541] 改善	DB2およびPostgreSQLのapplication-prod.propertiesにサンプルデータベースコネクタ設定を追加しました
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.1.0

[Release Date: 16/08/2019]

変更	詳細
[#151] 修正	証明書に含まれる場合、クライアント/サーバー証明書のインストール中にCA証明書チェーンをインポートする機能を追加しました
[#152] 修正	DSのプロバイダーが別のPReqエンドポイントを必要とする場合、PReq URLを指定する機能が追加されました
[#371] 修正	管理インターフェイスのセッションタイムアウトが機能しないバグを修正しました。値の設定を構成プロパティから設定するようになりました
[#425] 変更	変更された値をより適切に表示するように監査ログレポートを変更しました
[#447] 修正	RReqおよびRResメッセージがTransaction Detailsページに表示されるようになりました
[#461] 修正	PostgreSQLデータベースのサポートを追加
[#483] 修正	デバッグログレベルの認証APIメッセージの時間指定ログを追加

変更	詳細
[#487] 修正	Mastercardコンプライアンステストを実地するとき3DSサーバーの参照番号を上書きする機能を追加しました
[#488] 修正	DS証明書ページを再設計して、CSRをより簡単に管理し、ボタンを合理化しました
[#493] 変更	デフォルトのテストマーチャントは、テスト目的で使用されるため、削除できなくなりました
[#497] 修正	DB2のデータベースのサポートが追加されました
[#499] 修正	3DSサーバーURLが設定されていれば、DSのCSRを生成する際にコモンネームが事前に入力されるようになりました
[#505] 変更	ブラウザ情報の収集または3DSメソッドがスキップされると、必須フィールドが欠落した実際のエラーメッセージが表示されるようになりました
[#508] 修正	トランザクションの詳細ページに表示されるECI値を追加しました
[#516] 変更	ログインページのエラーメッセージを変更して、セキュリティを改善しました
[#520] 変更	moment.jsファイルを、外部CDNからではなくローカルのファイルからロードするように変更しました
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.0.5

[リリース日: 04/07/2019]

変更点	詳細
[#322] 修正	日付時刻がユーザーのローカルタイムゾーンで表示されない点を修正しました。(User Profileページで設定されたタイムゾーン)

変更点	詳細
[#378] 改善	Merchantページの詳細からCA証明書をダウンロードする機能を追加しました。
[#401] 変更	新規にインストールした場合のシステムキーストアの名前を「as_sys_randomUUID.jks」に変更しました。
[#402] 修正	"3DS Server Transaction ID"、"Min purchase amount"、"Max purchase amount"を取引のフィルターに設定した場合に表示される結果が違う問題を修正しました。
[#412] 修正	ユーザーがパスワードを間違えた回数が上限を超えた場合にロックされる問題を修正しました。
[#422] 修正	Directory Servers > Settings > HTTPS callback portにおいて使用されているポートと違うポートが表示される問題を修正しました。
[#428] 変更	/api/v1/auth/3ri 認証APIリクエストにおいて{messageCategory}を追加しました。
[#433] 変更	.htmlの接尾辞を全てのページから削除しました。
[#446] 改善	Merchantページの詳細でエラーが発生した際のエラーメッセージを改善しました。
[#448] 改善	Directory Server証明書のインポートのロジックとエラー処理を改善しました。
[#449] 改善	Directory Server > Settings > 3DS Server URL (前External URL), Directory Server > Settings > HTTP listening port (前HTTPS callback port), Settings > 3DS2 > API URL (前Auth API URL)システムラベルの読みやすさを改善しました。
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.0.4

[リリース日: 31/05/2019]

変更点	詳細
[#386] 修正	HSMが使用された時にアクティブ化がエラーになる問題を修正しました。
[#390] 改善	Settings > SecurityページからHSM PINを変更する機能を追加しました。
[#380] 改善	Amazon Aurora MySQL 5.7のサポートを追加しました。

ActiveServer v1.0.3

[リリース日：2019/05/27]

変更点	詳細
[#376] 変更	結果の列挙を 00 または 01 の値で提供するように enro1 APIレスポンスを更新しました。
[#379] 修正	ダッシュボード履歴データが表示されない場合があるという問題を修正しました。
[#380] 修正	アクセス時にDS enum値が古い加盟店でエラーが表示されるという問題を修正しました。

ActiveServer v1.0.2

[リリース日：2019/05/24]

変更点	詳細
データベース・サポート	MSSQL Server 2017のサポートを追加しました。
[#301] 改善	.x509認証を使用するように管理APIエンドポイントを更新しました。
[#349] 変更	ログファイルの形式をas.dd-mm-yyyy.logからas.yyyy-mm-dd.logに変更し、ベースのログフォルダに保存されるように変更しました。
[#356] 変更	application-prod.propertiesのDSポートのデフォルト値が9600の範囲になるように変更しました。
[#368] 修正	enro1 APIが内部サーバーエラーを返すという問題を修正しました。

変更点	詳細
[#373] 改善	User ProfileページにAPIリクエストで使用されるCA証明書のダウンロードを追加しました。

ActiveServer v1.0.1

[リリース日 : 2019/05/17]

変更点	詳細
[#326] 修正	一部のブラウザでサイドメニューの読み込みが遅くなるという問題を修正しました。
[#327] 修正	Oracle DBの使用時の互換性の問題を修正しました。
[#328] 変更	AuthResponseApp APIにacsReferenceNumberを追加しました。
その他	小規模なバグ修正、パフォーマンス、セキュリティの強化

ActiveServer v1.0.0

[リリース日 : 2019/05/09]

変更点	詳細
リリース	初期リリース

法的通知

機密保持に関する声明

GPaymentsはこの文書に含まれる機密情報と知的財産に対するすべての権利を留保します。この文書には、GPaymentsの事業、商業、財務、または技術活動に関する情報が含まれる場合があります。第三者に対するこの情報の開示はGPaymentsに大きな不利益をもたらすため、この情報は受領者の単独使用を意図しています。この文書のいかなる部分も、書面による事前の許可なしに、いかなる形式または方法による複製、検索システムへの保存、または転送が禁じられています。この情報は、受領者との既存の秘密保持契約に基づいて提供されます。

著作権に関する声明

この文書の著作権© 2003-2019はGPayments Pty Ltdのものであり、すべての権利が留保されます。GPayments Pty Ltdの著作物を複製または使用する権限は、この文書またはその他の文書における著作物の可用性によって暗示されるものではありません。

この文書で使用されているすべてのサードパーティ製品およびサービス名およびロゴは、それぞれの所有者の商号、サービスマーク、商標、または登録商標です。

この文書のスクリーンショットで使用されている企業、組織、製品、個人、およびイベントの例は架空のものです。実際の企業、組織、製品、個人、またはイベントとの関連は意図しておらず、推測されるべきではありません。

免責事項

GPayments Pty Ltdは、この文書に含まれる製品、プロトコル、または標準のいずれについても説明しておらず、説明を意図するものではありません。GPayments Pty Ltdは、いかなる目的のためにも、この情報の内容、完全性、正確性、または適切性を保証するものではありません。情報は明示的または暗黙的な保証なしに「現状のまま」提供され、予告なしで変更されることがあります。GPayments Pty Ltdは、商品適格性および特定の目的への適合性に関するすべての暗黙的な保証、および侵害に対する保証を含む、この情報に関するすべての保証を否認します。この文書に含まれる製品、プロトコル、または標準に関して、GPayments Pty Ltdが行った決定および/または声明は信頼されるべきではありません。

責任

いかなる場合においても、GPayments Pty Ltdは、GPaymentsがそのような損害の可能性について知らされていた場合であっても、契約上の行為、過失、またはその他の不正な行為において、この情報またはここに記載されている製品、プロトコル、標準の使用または使用不能に起因するか、それらと関連して発生するかにかかわらず、いかなる特別、付随的、間接的、または結果として起こる損害（事業利益の喪失、事業の中断、事業情報の喪失、またはその他の金銭的損失に対する損害を含みますが、これらに限定されません）についても責任を負いません。