FIDO Notary設定・デプロイ手順書

目次

[**1.** **FIDOサーバの設定** 2](#_Toc129275102)

[**1-1.RPサーバの初期登録** 2](#_Toc129275103)

[**1-2.DB接続** 3](#_Toc129275104)

[**2.** **RPサーバの設定** 4](#_Toc129275105)

[**2-1.DB接続先の設定** 4](#_Toc129275106)

[**2-2．RPサーバ設定** 4](#_Toc129275107)

[**2-3.署名の設定** 5](#_Toc129275108)

[**2-4.URLディレクトリの設定** 5](#_Toc129275109)

[**3.** **RPサーバプロジェクトのインポートとローカル環境での動作確認** 6](#_Toc129275110)

[**3-1.プロジェクトファイルのインポート** 6](#_Toc129275111)

[**3-2.ローカル環境での起動** 6](#_Toc129275112)

[**4.　RPサーバのEC2環境へのデプロイ** 7](#_Toc129275113)

[**4-1.warファイルの作成** 7](#_Toc129275114)

[**4-2.手動でのEC2サーバへ配置** 7](#_Toc129275115)

[**4-3.マクロでのEC2サーバへ配置** 8](#_Toc129275116)

[**4-3-1.マクロの設定** 8](#_Toc129275117)

[**4-3-2.マクロの実行** 8](#_Toc129275118)

# **FIDOサーバの設定**

FIDOサーバは「https://github.com/line/line-fido2-server」より取得したOSSをそのまま以下のディレクトリに配置しております

設定変更の際はディレクトリ内のファイルを編集・再起動を行ってください。

　/mnt/xvdb/move/line-fido2-server-main

## **1-1.RPサーバの初期登録**

以下のファイルにFIDOサーバ起動時にDBへ登録を行うSQL文が記載されているのでこちらに必要なRP名を追加することで起動時にDBへ登録されます

/line-fido2-server/server/src/main/resources/data.sql

insert into rp (id, name, description) values(‘RP名’, 'example1', 'example1');

※FIDOサーバの再起動を行うとDBが初期化されるため、認証情報は再登録が必要となります。再起動せずに追加を行う場合は「2-2.DB接続」よりDBへ接続し直接DBへInserを行ってください。

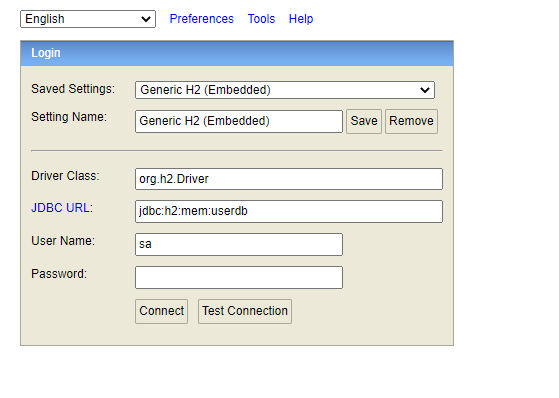
## **1-2.DB接続**

FIDOサーバのDBは以下のURLより接続してください。

※環境を移行した際は移行環境に合わせて赤字の部分を変更してください。

<http://54.202.220.116:8081/h2-console/>

ログインコンソール画面が表示されるので以下の内容でログインしてください



# **RPサーバの設定**

RPサーバはGradleで作成・アーカイブしていますので、eclipse等でGradleプロジェクトとしてインポートを行うとそのままSpringBootとしてローカル環境で実行可能となっております。

## **2-1.DB接続先の設定**

以下のファイルにDBパス,ユーザーID,パスワードを記載しておりますので環境に合わせて設定を行ってください。

/securing-web/src/main/resources/application.properties

spring.datasource.url=DBパス

spring.datasource.username=ユーザーID

spring.datasource.password=パスワード

## **2-2．RPサーバ設定**

以下ソースにFIDOサーバへ接続する際のRP\_IDとORIGINを定義しております。

RP\_IDとORIGINはサーバのドメインと一致している必要があります。

また、設定する名前はFIDOサーバに登録されているものである必要があります。

/securing-web/src/main/java/com/example/securingweb/service/LineFido2ServerServiceImpl.java

private static final String RP\_ID = RP\_ID;

private static final String ORIGIN = ORIGIN;

private static final String FIDO\_URI = FIDOサーバの接続先;

## **2-3.署名の設定**

以下のソースに署名の作成・検証時の内容を定義しております

/securing-web/src/main/java/com/example/securingweb/util/SignatureUtil.java

private static String aud = 署名の検証・作成時に設定するAUD;

private static String iss = 署名の検証・作成時に設定するISS;

private static String authnCtxFido = FIDO認証後の署名作成時に設定するAuthnCtx;

private static String chkAuthnCtx = 署名の検証時に設定するAuthnCtx;

private static String authnCtxPW = ID/PW認証時の署名作成時に設定するAuthnCtx;

private static long expireTime = 署名作成時に設定する有効期限

private static String url = 署名の検証時に使用する公開鍵取得時のURL

private static String urlBody =署名の検証時に使用する公開鍵取得時のBody

## **2-4.URLディレクトリの設定**

サーバURLにディレクトリ階層を設定している場合、Javascriptから呼び出すSpringのパスを設定してください。

const actionPath = ディレクトリ階層のパス（なしの場合は「“/”」のみを設定）;

# **RPサーバプロジェクトのインポートとローカル環境での動作確認**

作業手順はeclipseを使用した場合の手順となります

## **3-1.プロジェクトファイルのインポート**

1. 「securing-web.zip」を任意のディレクトリへ解答します
2. eclipseのメニューより「ファイル」→「インポート」→「既存のGradleプロジェクト」を選択します
3. プロジェクト・ルート・ディレクトリで1の場所を指定し「完了」を押下します
4. インポートしたプロジェクトを右クリック→「Gradle」→「Gradleプロジェクトのリフレッシュ」

## **3-2.ローカル環境での起動**

1. MYSQLが必要になりますので別途インストールを行い、以下のSQLでユーザー登録用テーブルを作成してください。

CREATE TABLE `users` (

`id` int NOT NULL AUTO\_INCREMENT,

`name` varchar(256) NOT NULL,

`password` varchar(256) NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=18 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

1. 「2.RPサーバの設定」を参考に各設定をローカル環境向けに設定してください。

※FIDOサーバの接続先についてはそのままEC2上にあるサーバを参照することも可能です。

1. プロジェクトを右クリック「実行」→「Spring Boot アプリケーション」で実行。
2. 正常に起動していればブラウザから以下のURLで認証画面を表示可能となっています。

<http://localhost:8080/login?url=>リダイレクトURL

# **4.　RPサーバのEC2環境へのデプロイ**

## **4-1.warファイルの作成**

1. プロジェクトを右クリック「実行」→「実行の構成」をクリック
2. 「Gradleタスク」をダブルクリックし新規作成
3. Gradleタスクにwarを追加、作業ディレクトリにワークスペースから本プロジェクトを選択
4. 「実行」をクリック
5. プロジェクトのワークスペース内の/build/libsにwarファイルが作成されます

※warファイルを再作成する際は既に作成されているwarファイルを削除してから作成してください

## **4-2.手動でのEC2サーバへ配置**

※TeraTermを使用した場合の手順となります

※手順は現在使用しているEC2サーバに対する内容となっております。

1. TeraTermでEC2サーバ（ホスト：54.202.220.116）へ接続..
2. メニューから「ファイル」→「SSH SCP」をクリック。
3. Fromに4-1で作成したwarファイルを設定、ToにEC2サーバ内の作業ディレクトリを設定し「Send」をクリック。
4. EC2サーバ内で以下のコマンドを実行

--移動

cd 3で配置したディレクトリ

--権限変更

chmod 644 securing-web.war

--rootに切り替え

sudo su –

--所有者をrootに変更

chown root.root securing-web.war

--ファイルを移動（上書き確認が行われるのでyを入力）

mv securing-web.war /opt/tomcat/webapps/securing-web.war

--tomcat再起動

sudo systemctl restart tomcat

1. 以下のコマンドで実行ログのモニタリングが可能です（Ctrl + cで終了）

tail -f /opt/tomcat/logs/catalina.out

## **4-3.マクロでのEC2サーバへ配置**

※TeraTermのマクロを使用することで4-2の操作を自動で行う事が出来ます。

### **4-3-1.マクロの設定**

1. securing-web\EC2デプロイマクロ\securing-web\_deploy.ttlをエディタで開く
2. 以下の部分を環境に合わせて設定

HOST = 配置先EC2サーバの接続先

USER\_ID = EC2サーバへのログインID

AUTH\_KEY = EC2サーバログイン時の認証鍵ファイルのパス

FILE\_TO = ファイル転送先のEC2サーバ内のパス

FILE\_FROM = 転送するwarファイルのパス

### **4-3-2.マクロの実行**

1. TeraTeamを起動
2. 接続ダイアログが表示されるので「キャンセル」を押下
3. メニューの「コントロール」→「マクロ」を選択
4. 4-3-1で設定を行ったttlファイルを選択
5. 4-2と同様のデプロイが行われ、完了後はログのモニタリング状態になります（Ctrl + cで終了）

※EC2サーバへログイン状態からマクロを再実行する場合は２１行目にある「wait '$' '#'」をコメントアウトすることで実行できます。