intdash Server 構築手順書
（インテグレーションパートナー様向け）

intdash リリース 2022Q2R1

第 1 版（2022 年 8 月）

aptpod

Rev. 95765d1

目次

01 はじめに

重要:

- このドキュメントに記載されている仕様は予告なく変更される場合があります。このドキュメントは情報提供を目的としたものであり、仕様を保証するものではありません。
- 説明で使用している画面は一例です。ご使用の環境やアプリケーションのバージョンによって、表示や手順が一部異なる場合があります。

注釈: このドキュメントに記載されている会社名、サービス名、製品名等は、一般に、各社の登録商標または商標です。本文および図表中には、「™」、「®」は明記していません。

1.1 このドキュメントで構築するサーバーの構成

このドキュメントでは、intdash サーバーを PoC（Proof of Concept）向けの小規模構成で構築する手順を説明します。intdash サーバーは API サーバーと DB サーバーの 2 インスタンスで構成されます。

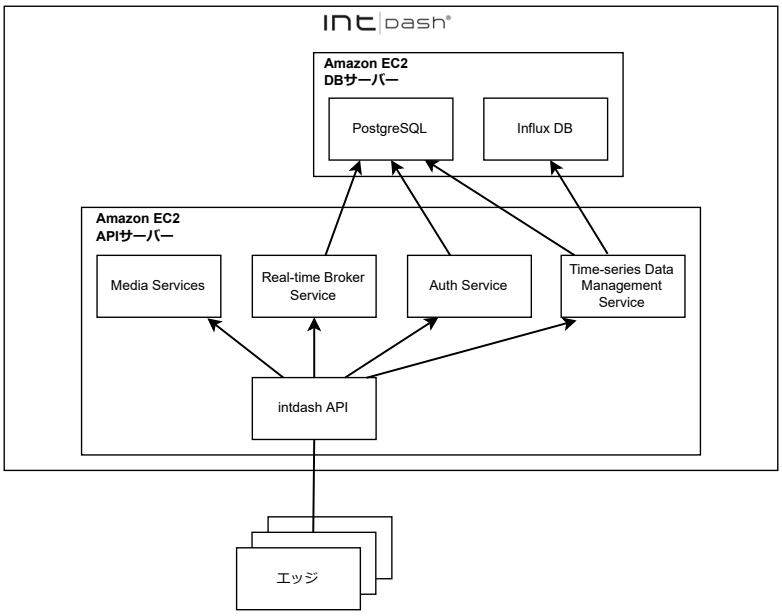


図 1　このドキュメントで構築する intdash サーバーの構成

1.2 対象読者

このドキュメントは、インテグレーションパートナー様において intdash サーバーの構築および管理を行う方を対象に書かれています。

このドキュメントを使って intdash 環境を構築する方は、ネットワーク管理やサーバー管理のほか、IaaS(AWS)でのインスタンス構築についての基礎知識を持っていることを前提とします。

1.3 前提事項

intdash サーバーを構築し、設定を行うには以下のソフトウェアが必要です。

- SSH 接続が可能なターミナルソフトウェア
- ウェブブラウザー Google Chrome

1.4 このドキュメントの構成

次の章以降は、以下のような構成になっています。

intdash サーバーを作成する (p. 5)
Amazon EC2 インスタンスに intdash サーバーをインストールします。SSH 接続によるターミナル操作が中心です。

intdash サーバーの設定を行う (p. 9)
起動した intdash サーバーにおいて認証やアカウント関連の設定を行います。SSH 接続によるターミナル操作と、設定用の専用ウェブアプリケーションでの操作です。

動作を確認する (p. 25)
intdash サーバーが正常に起動できたことを確認するため、簡単な動作確認を行います。リアルタイムデータを実際に送信して表示します。

付録：intdash を構成するサービスと設定ファイル (p. 30)
intdash アプリケーションに関する技術的な補足事項を記載しています。

02 intdash サーバーを作成する

Amazon EC2 を使って intdash サーバーを作成します。

2.1 準備

AWS 上に以下を準備してください。

- AWS アカウント
- intdash サーバー用のネットワーク環境
  - VPC
  - サブネット
    - サブネットを 2 つ作成することを推奨します。
      - パブリックサブネット（API サーバー用）
      - プライベートサブネット（DB サーバー用）
  - ルーティングテーブル
  - セキュリティグループ
    - パブリックのインバウンド通信は以下を許可します。
      - HTTP(TCP 80 番ポート)
      - HTTPS(TCP 443 番ポート)
    - メンテナンス用途として、作業端末から以下の SSH 接続を行えるようにご利用の環境に応じて設定してください。
      - SSH(TCP 22 番ポート)
    - その他の通信を制限する場合は、使用されているポートについての情報を 付録：intdash を構成するサービスと設定ファイル (p. 30) で確認してください。
- Elastic IP（外部に公開する場合のみ必要）
- インスタンス（API サーバー用と DB サーバー用に計 2 つ）
  - インスタンスタイプ：m5.large 以上
  - オペレーティングシステム：Amazon Linux 2
  - アーキテクチャ：x86_64
  - ストレージ：
    - ルートデバイス：20GB 以上
    - 追加ボリューム：10GB 以上
      - /data にマウント
      - ボリュームサイズは保存する計測データの量に応じて調整
- intdash yum リポジトリの認証情報
  - 未取得の場合は当社インテグレーションパートナー窓口にお問合せください。

2.2 DB サーバーを作成する

SSH を使って DB サーバー用の EC2 インスタンスに接続し、以下の OSS(Open Source Software) をインストールします。

- **リレーショナルデータベース PostgreSQL** (https://www.postgresql.org/)
- **時系列データベース InfluxDB** (https://www.influxdata.com/)

1. PostgreSQL をインストールします。

```
$ sudo amazon-linux-extras enable postgresql11=stable
$ sudo yum -y install postgresql postgresql-server
```

2. PostgreSQL の data ディレクトリとして、追加ボリューム内のディレクトリを設定します。

```
$ sudo mkdir -p /data/pgsql/11/data
$ sudo chown postgres:postgres /data/pgsql/11/data
$ sudo chmod 700 /data/pgsql/11/data
$ sudo sed -i \
-e 's/Environment=PGDATA=\/var\/lib\/pgsql\/data/Environment=PGDATA=\/data\/pgsql\/11\/data/' \
/usr/lib/systemd/system/postgresql.service
```

3. PostgreSQL の初期化とサービスの起動を行います。

```
$ sudo postgresql-setup --initdb
$ sudo systemctl enable postgresql.service
$ sudo systemctl start postgresql.service
```

4. PostgreSQL のユーザーとデータベースを作成します。

```
$ sudo -u postgres psql -d template1
template1=# CREATE ROLE "intdash" WITH LOGIN ENCRYPTED PASSWORD 'POSTGRESQL-INTDASH-USER-PASSWORD';
template1=# CREATE DATABASE "intdash-api" OWNER "intdash";
template1=# CREATE DATABASE "intdash-micro-auth" OWNER "intdash";
template1=# CREATE DATABASE "intdash-micro-measurement" OWNER "intdash";
template1=# CREATE DATABASE "intdash-micro-config" OWNER "intdash";
template1=# CREATE DATABASE "intdash-micro-broker" OWNER "intdash";
template1=# CREATE DATABASE "intdash-micro-media" OWNER "intdash";
```

POSTGRESQL-INTDASH-USER-PASSWORD
　PostgreSQL データベース接続ユーザーのパスワード。任意の値を設定してください。

5. PostgreSQL の外部接続設定を行います。

```
$ sudo cp -p /data/pgsql/11/data/postgresql.conf /data/pgsql/11/data/postgresql.conf.orig
$ sudo sed -i -e "s/#listen_addresses = 'localhost'/listen_addresses = '\*'/" \
/data/pgsql/11/data/postgresql.conf
$ sudo cp -p /data/pgsql/11/data/pg_hba.conf /data/pgsql/11/data/pg_hba.conf.orig
```

/data/pgsql/11/data/pg_hba.conf の末尾を以下のように変更します。

```
$ sudo vi /data/pgsql/11/data/pg_hba.conf
# TYPE  DATABASE        USER            ADDRESS                 METHOD

local   all             all                                     trust
host    all             all             127.0.0.1/32            trust
host    all             all             ::1/128                 trust
host    all             all             0.0.0.0/0               md5
host    all             all             ::/0                    md5
```

設定を反映させます。

```
$ sudo systemctl restart postgresql.service
```

6. InfluxDB をインストールします。

```
$ sudo yum -y install https://dl.influxdata.com/influxdb/releases/influxdb-1.8.10.x86_64.rpm
```

7. InfluxDB の data ディレクトリとして、追加ボリューム内のディレクトリを設定します。

```
$ sudo mkdir -p /data/influxdb
$ sudo chown influxdb:influxdb /data/influxdb
$ sudo chmod 700 /data/influxdb
$ sudo cp -p /etc/influxdb/influxdb.conf /etc/influxdb/influxdb.conf.orig
$ sudo sed -i -e 's/dir = "\/var\/lib\/influxdb\/meta"/dir = "\/data\/influxdb\/meta"/' \
/etc/influxdb/influxdb.conf
$ sudo sed -i -e 's/dir = "\/var\/lib\/influxdb\/data"/dir = "\/data\/influxdb\/data"/' \
/etc/influxdb/influxdb.conf
$ sudo sed -i -e 's/wal-dir = "\/var\/lib\/influxdb\/wal"/wal-dir = "\/data\/influxdb\/wal"/' \
/etc/influxdb/influxdb.conf
```

8. InfluxDB の初期設定とサービスの起動を行います。

```
$ sudo sed -i -e 's/# index-version = "inmem"/index-version = "tsi1"/' /etc/influxdb/influxdb.conf
$ sudo sed -i -e 's/# auth-enabled = false/auth-enabled = true/' /etc/influxdb/influxdb.conf
$ systemctl enable influxdb.service
$ systemctl start influxdb.service
```

9. InfluxDB のユーザーとデータベースを作成します。

```
$ influx
> CREATE USER intdash WITH PASSWORD 'INFLUXDB-INTDASH-USER-PASSWORD' WITH ALL PRIVILEGES;
> auth
username: intdash
password: INFLUXDB-INTDASH-USER-PASSWORD
> CREATE DATABASE "intdash" WITH SHARD DURATION 1h;
```

INFLUXDB-INTDASH-USER-PASSWORD

InfluxDB データベース接続ユーザーのパスワード。任意の値を設定してください。

2.3 API サーバーを作成する

SSH を使って API サーバー用の EC2 インスタンスに接続し、intdash サーバーの各独自アプリケーションと以下の OSS(Open Source Software) をインストールします。

- **ウェブサーバー nginx** (https://nginx.org/en/)
- **エッジルーター Traefik** (https://doc.traefik.io/traefik/)

1. 利用するリポジトリの有効化を行います。

```
$ sudo amazon-linux-extras enable nginx1=stable
$ sudo curl -sL https://rpm.nodesource.com/setup_14.x | sudo bash -
```

以下のように intdash リポジトリの設定ファイルを作成します。

```
$ sudo vi /etc/yum.repos.d/intdash.repo
[intdash]
name=intdash packages
baseurl=https://private-repository.aptpod.jp/intdash/202206/linux/rpm
username=INTDASH-REPOSITORY-USERNAME
password=INTDASH-REPOSITORY-PASSWORD
enabled=0
gpgcheck=0

[intdash-testing]
name=intdash packages - testing
baseurl=https://private-repository.aptpod.jp/intdash/202206/testing/linux/rpm
username=INTDASH-REPOSITORY-USERNAME
password=INTDASH-REPOSITORY-PASSWORD
enabled=0
gpgcheck=0
```

INTDASH-REPOSITORY-USERNAME
: intdash yum リポジトリ用のユーザー ID

INTDASH-REPOSITORY-PASSWORD
: intdash yum リポジトリ用のパスワード

2. 必要なソフトウェアをインストールします。

```
$ sudo yum -y --enablerepo=intdash,intdash-testing install \
intdash-api intdash-ffmpeg intdash-micro-auth intdash-micro-broker \
intdash-micro-config intdash-micro-measurement intdash-micro-media \
intdash-nginx intdash-web-admin intdash-web-edges intdash-web-email \
intdash-web-me intdash-web-measurements intdash-web-media intdash-web-oauth2 \
intdash-web-password intdash-web-shared-assets intdash-web-signin \
intdash-web-widget-app-links vm2m-assets vm2m-dataviz vm2m-dataviz-backend \
vm2m-dataviz-plugins-core
$ sudo yum -y install httpd-tools
```

以上で intdash サーバーが作成できました。intdash サーバーの設定を行う (p. 9) に進んでください。

03 intdash サーバーの設定を行う

intdash サーバーを作成する (p. 5) の手順に従って intdash サーバーの準備ができたら、設定を行います。

3.1 ドメイン名とサーバー証明書を設定する

このドキュメントでは、例として以下のドメイン名と証明書ファイルを使用します。

項目	この説明で使用する例
intdash 用のドメイン名	intdash.example.com
第三者認証局により発行された SSL サーバー証明書 (サーバー証明書と中間証明書が連結されたもの)	/etc/pki/intdash/certs/intdash.pem
SSL サーバー秘密鍵	/etc/pki/intdash/private/intdash.pem

注釈: intdash 用ドメイン名は、組織で所有・管理しているドメイン名に対しサブドメインとして払い出す方法が一般的です。その場合、例えば example.com を所有、管理している場合は、intdash 用のドメイン名は intdash.example.com のようにします。
詳細については、組織のドメイン名管理者にご相談ください。

注釈:

- サーバー証明書は、SSL/TLS 証明書とも呼ばれ、クライアントとサーバーの間の接続情報を保護し、第三者からの読み取りや変更を防ぐものです。
- セキュリティ上の理由から、自己署名証明書ではなく第三者認証局により発行された証明書を使用してください。自己署名証明書を使用すると、クライアントからの接続時にサーバー証明書の検証に失敗します。サーバー証明書の検証をスキップした場合、通信相手の認証をしないことになり、中間者攻撃等による盗聴・改ざんを許すことになります。
- 証明書を発行する認証局の例としては、Cybertrust、DigiCert、Let's Encrypt 等があります。Let's Encrypt では、無料で証明書の発行を受けることができます。

3.1.1 ウェブサーバーの設定を行う

ウェブサーバー nginx の設定ファイルで、使用するドメイン名とサーバー証明書を設定します。

1. SSH で API サーバーに接続し、設定ファイルのディレクトリに移動します。

```
# cd /etc/nginx/conf.d
```

2. デフォルトの設定ファイルをバックアップします。

```
# cp -p intdash.conf intdash.conf.org
```

3. テキストエディターで設定ファイルを開きます。

```
# vi intdash.conf
```

4. ポート 80（HTTP）とポート 443（HTTPS）の [server_name] にドメイン名を設定します。

```
...

server {
    listen       80;
    listen       [::]:80;
    server_name  intdash.example.com;   <--
    access_log   off;

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen       443 ssl http2;
    listen       [::]:443 ssl http2;
    server_name  intdash.example.com;   <--
    root         /usr/share/nginx/html;

    ssl_certificate     /etc/pki/intdash/certs/intdash.pem;
    ssl_certificate_key /etc/pki/intdash/private/intdash.pem;

...
```

5. 同じ設定ファイル内でポート 443 の [ssl_certificate] にサーバー証明書、[ssl_certificate_key] にサーバー証明書の秘密鍵を指定します。その後設定ファイルを閉じます。

```
...

server {
    listen       80;
    listen       [::]:80;
```

(次のページに続く)

(前のページからの続き)

```
    server_name  intdash.example.com;
    access_log   off;

    location / {
        return 301 https://$host$request_uri;
    }
}

server {
    listen       443 ssl http2;
    listen       [::]:443 ssl http2;
    server_name  intdash.example.com;
    root         /usr/share/nginx/html;

    ssl_certificate     /etc/pki/intdash/certs/intdash.pem;        <--
    ssl_certificate_key /etc/pki/intdash/private/intdash.pem;      <--

...
```

6. 設定を反映させるため、以下のコマンドで nginx を起動します。

```
# systemctl start nginx
```

7. 起動後、nginx が正常に起動していることを確認します。

```
# systemctl status nginx
```

正常に動作していれば `Active: active (running)` と表示されます。

3.1.2 エッジルーター Traefik の設定を行う

intdash の API サービスや、マイクロサービスへのプロキシを担う Traefik の設定を行います。

1. 前の手順に引き続き SSH で API サーバーに接続し、設定ファイルのディレクトリに移動します。

```
# cd /etc/intdash/traefik.d
```

2. デフォルトの設定ファイルをバックアップします。

```
# cp -p middleware.toml middleware.toml.org
```

3. テキストエディターで設定ファイルを開きます。

```
# vi middleware.toml
```

4. 以下の個所にドメイン名を入力します。その後、設定ファイルを閉じます。

```
[http]
  [http.middlewares]
```

(次のページに続く)

(前のページからの続き)

```
  ...

  [http.middlewares.cors]
    [http.middlewares.cors.headers]

      ...

      accessControlAllowOriginList = ["https://intdash.example.com", "http://intdash.example.com",
↪"wss://intdash.example.com:443/api/v1/ws/measurements"]   <--

      ...
```

3.1.3 認証処理を行う intdash-micro-auth の設定を行う

intdash で認証処理を行うマイクロサービス（intdash-micro-auth）の設定を行います。

1. 前の手順に引き続き SSH で API サーバーに接続し、設定ファイルのディレクトリに移動します。

```
# cd /etc/intdash
```

2. デフォルトの設定ファイルをバックアップします。

```
# cp -p authd.conf authd.conf.org
```

3. テキストエディターで設定ファイルを開きます。

```
# vi authd.conf
```

4. 以下の個所にドメイン名を入力します。その後、設定ファイルを閉じます。

   - [email] セクション
     - verification-page-uri

```
[email]
  from = "noreply@example.com"
  from-display-name = "VM2M"
  reply-to = ""
  limit-per-user = 1
  verification-expiration-period = "3h"
  verification-page-uri = "https://intdash.example.com/email/activate"   <--
  ...
```

   - [oauth2] セクション
     - issuer
     - sign-in-page-uri
     - change-password-page-uri

```
[oauth2]
  issuer = "https://intdash.example.com"   <--
  access-token-expiration-period = "1h"
  refresh-token-expiration-days = 30
  sign-in-page-uri = "https://intdash.example.com/signin/"   <--
  change-password-page-uri = "https://intdash.example.com/oauth2/authorization/api/password-change"  <--
  ...
```

- [oauth2.default-client] サブセクション
  - password-recovery-redirect-base-url
  - web-redirect-base-urls

```
[oauth2.default-client]
  password-recovery-redirect-base-url = "https://intdash.example.com"   <--
  web-redirect-base-urls = [
    "http://localhost:8080",
    "https://localhost:8080",
    "https://intdash.example.com",   <--
  ]
  ...
```

3.1.4 intdash-micro-auth JWT(JSON Web Token) の秘密鍵を固定化する

intdash-micro-auth サービスの設定ファイル `/etc/intdash/authd.conf` で、認証用の JWT 秘密鍵や RSA 秘密鍵を設定します。

> **注釈:** ここで固定値を設定しない場合、intdash-micro-auth サービスが再起動されるたびに、認証用の JWT 秘密鍵や RSA 秘密鍵が自動生成されます。その場合、サービスを再起動するたびに、以下のようなことが起こります。
>
> - API トークンが無効になる
> - ユーザーのログイン状態がリセットされる
> - メールアドレス確認用のメールに含まれるリンクが無効になる
>
> そのため、固定の値を設定することを推奨します。

- [keys.oauth2-rsa.private-key] サブセクション
  - `key`：PEM 形式の RSA 秘密鍵を設定します。

```
[keys]
  [keys.oauth2-rsa]
    [keys.oauth2-rsa.private-key]
      driver = "static"
      [keys.oauth2-rsa.private-key.static]
        key = "-----BEGIN RSA PRIVATE KEY-----\nXXXXXX...XXXXXX\n-----END RSA PRIVATE KEY-----"   <--
```

- [keys.oauth2-hmac], [keys.api-token], [keys.password-recovery], [keys.email-verification] サブセクション
  - `hash-secret`：ランダムで生成した文字列を設定します。

```
[keys.oauth2-hmac]
  hash-secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"   <--
  rotated-hash-secrets = [""]
[keys.api-token]
  hash-secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"   <--
  rotated-hash-secrets = [""]
[keys.password-recovery]
  hash-secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"   <--
  rotated-hash-secrets = [""]
[keys.email-verification]
  hash-secret = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"   <--
  rotated-hash-secrets = [""]
```

この設定後は intdash-micro-auth サービスを再起動してください。また、この設定変更のデータベースへの反映は不要です。

注釈: [keys.oauth2-rsa.private-key] サブセクションでは、`driver = "file"` を指定することで、PEM 形式の文字列を直接書き込む代わりに RSA 秘密鍵のファイルパスを指定して読み込ませることも可能です。秘密鍵ファイルは intdash ユーザーで動作する intdash-micro-auth サービスから読み込まれます。intdash ユーザーからの読み込みができるように、ファイルのパーミッション、オーナー、グループを設定してください。

```
[keys]
  [keys.oauth2-rsa]
    [keys.oauth2-rsa.private-key]
      driver = "file"
      [keys.oauth2-rsa.private-key.file]
        path = "/etc/intdash/authd.keys/privkey.pem"   <--
```

3.1.5 intdash-web-oauth2 の設定を行う

intdash の Oauth2 認証におけるウェブ UI クライアントの認証を担う UI サービス（intdash-web-oauth2）の設定を行います。

1. 前の手順に引き続き SSH で intdash サーバーに接続し、設定ファイルのディレクトリに移動します。

   ```
   # cd /etc/sysconfig
   ```

2. デフォルトの設定ファイルをバックアップします。

   ```
   # cp -p intdash-web-oauth2 intdash-web-oauth2.org
   ```

3. テキストエディターで設定ファイルを開きます。

   ```
   # vi intdash-web-oauth2
   ```

4. `AUTHORIZATION_HOST` と `CLIENT_HOST` にドメイン名を入力します。その後、設定ファイルを閉じます。

```
HOST="127.0.0.1"
PORT="13003"

API_HTTP_URL="http://127.0.0.1:8080"
AUTHORIZATION_HOST="https://intdash.example.com"   <--
CLIENT_HOST="https://intdash.example.com"          <--
CLIENT_ID="533bc9ea_authorization_code.aptpod.co.jp"
RETURN_TO_URL="/"
ALLOW_LOGGING="false"
```

3.2 DB サーバーに設定を反映する

API サーバーに設定した情報を基に DB サーバーのマイグレーションを行います。

3.2.1 API サーバーに DB サーバーの接続情報を設定する

intdash の各マイクロサービスに、作成した DB サーバーの接続情報を設定します。

1. 前の手順に引き続き SSH で API サーバーに接続し、設定ファイルのディレクトリに移動します。

```
# cd /etc/intdash
```

2. これまでの手順でバックアップを行っていないデフォルトの設定ファイルをバックアップします。

```
# cp -p brokerd.conf brokerd.conf.org
# cp -p configd.conf configd.conf.org
# cp -p intdashd.conf intdashd.conf.org
# cp -p measurementd.conf measurementd.conf.org
# cp -p media.toml media.toml.org
```

3. テキストエディターで以下の設定ファイルをそれぞれ開き、手順 4. のように編集します。

```
# vi auth.toml
# vi brokerd.conf
# vi configd.conf
# vi intdashd.conf
# vi measurementd.conf
# vi media.toml
```

4. 以下の個所に DB サーバーの PostgreSQL の接続情報を入力します。その後、設定ファイルを閉じます。
   - [attrdb] セクション

```
password = "POSTGRESQL-INTDASH-USER-PASSWORD"
address = "DB-SERVER-ADDRESS:5432"
```

POSTGRESQL-INTDASH-USER-PASSWORD
: PostgreSQL データベース接続ユーザーのパスワード

DB-SERVER-ADDRESS

DB サーバーのアドレス（ドメイン名または IP アドレス）

5. テキストエディターで以下の設定ファイルを開きます。

```
# vi measurementd.conf
```

6. 以下の個所に DB サーバーの InfluxDB の接続情報を入力します。その後、設定ファイルを閉じます。

   - [*.influx] セクション全て

```
address = "http://DB-SERVER-ADDRESS:8086"   <--
db-name = "intdash"
username = "intdash"
password = "INFLUXDB-INTDASH-USER-PASSWORD"   <--
```

INFLUXDB-INTDASH-USER-PASSWORD

InfluxDB データベース接続ユーザーのパスワード

DB-SERVER-ADDRESS

DB サーバーのアドレス（ドメイン名または IP アドレス）

7. data ファイルの出力先を追加ボリュームにするために、テキストエディターで以下の設定ファイルを開きます。

```
# vi measurementd.conf
# vi mediad.toml
```

8. 以下のようにそれぞれパス情報を書き換えます。

   - measurementd.conf
     - [upload] セクション

```
[upload]
  [upload.storage]
    driver = "file"
    [upload.storage.file]
      directory = "/data/intdash/upload"   <--
      remove-interval-seconds = 604800
```

   - mediad.toml
     - [MediaFilmed セクション

```
[MediaFilmed]
  ffmpeg = "/opt/intdash-ffmpeg/bin/ffmpeg"
  ffmpeg-log-level = "info"
  ffmpeg-threads = 0
  dummy-file-path = "/usr/share/intdash-micro-media/assets/dummy.ts"
  [MediaFilmed.NatsStreaming]
    url = "nats://127.0.0.1:4222"
    username = "intdash"
```

(次のページに続く)

(前のページからの続き)

```
    password = "intdash"
    client-id-prefix = "media"
  [MediaFilmed.Storage]
    type = "local"
    [MediaFilmed.Storage.Local]
      base-dir = "/data/intdash/media/hls"   <--
```

9. 出力先のフォルダを作成しておきます。

```
# mkdir -p /data/intdash/upload
# chown intdash:intdash /data/intdash/upload
# chmod 700 /data/intdash/upload
# mkdir -p /data/intdash/media/hls
# chown intdash:intdash /data/intdash/media/hls
# chmod 700 /data/intdash/media/hls
```

3.2.2 API サーバーから DB サーバーにデータベースのマイグレーションを行う

intdash の各マイクロサービス用のデータベースのマイグレーションを行います。

1. 前の手順に引き続き SSH で API サーバーに接続し、データベースのマイグレーションコマンドを実行します。

```
# sudo -u intdash intdashd db migrate -c /etc/intdash/intdashd.conf
# sudo -u intdash authd db migrate -c /etc/intdash/authd.conf -p INTDASH-USER-PASSWORD
# sudo -u intdash measurementd db migrate -c /etc/intdash/measurementd.conf
# sudo -u intdash configd db migrate -c /etc/intdash/configd.conf
# sudo -u intdash brokerd db migrate -c /etc/intdash/brokerd.conf
# sudo -u intdash mediad db migrate -c /etc/intdash/mediad.toml
```

INTDASH-USER-PASSWORD

管理者ユーザー（ユーザー名「intdash」）のパスワード。任意の値を設定してください。

2. SSH で DB サーバーに接続し、以下のコマンドを実行して 2 つの sql ファイルを用意します。

```
$ curl -O https://docs.intdash.jp/manual/intdash-server-setup/ed1/ja/config.sql
$ curl -O https://docs.intdash.jp/manual/intdash-server-setup/ed1/ja/app-config.sql
```

3. 以下のコマンドを実行して設定を追加します。

```
$ psql -U intdash -d intdash-api -1 -f config.sql
$ psql -U intdash -d intdash-micro-config -1 -f app-config.sql
```

3.3 intdash サービスを起動する

API サーバーで各 intdash サービスを起動します。

3.3.1 各 intdash サービスの起動

intdash の各マイクロサービスを起動します。

1. SSH で API サーバーに接続し、各サービスを有効化するために以下のコマンドを実行します。

```
# systemctl enable nginx.service
# systemctl enable intdash-api-auth.service
# systemctl enable intdash-api-gateway.service
# systemctl enable intdash-api.service
# systemctl enable intdash-micro-auth.service
# systemctl enable intdash-micro-broker.service
# systemctl enable intdash-micro-config.service
# systemctl enable intdash-micro-measurement.service
# systemctl enable intdash-micro-media.service
# systemctl enable intdash-web-admin.service
# systemctl enable intdash-web-edges.service
# systemctl enable intdash-web-me.service
# systemctl enable intdash-web-measurements.service
# systemctl enable intdash-web-media.service
# systemctl enable intdash-web-oauth2-redirector.service
# systemctl enable intdash-web-oauth2.service
# systemctl enable intdash-web-signin-redirector.service
# systemctl enable intdash-web-widget-app-links.service
# systemctl enable vm2m-dataviz-backend.service
```

2. 各サービスを起動するために以下のコマンドを実行します。

```
# systemctl start intdash-api-auth.service
# systemctl start intdash-api-gateway.service
# systemctl start intdash-api.service
# systemctl start intdash-micro-auth.service
# systemctl start intdash-micro-broker.service
# systemctl start intdash-micro-config.service
# systemctl start intdash-micro-measurement.service
# systemctl start intdash-micro-media.service
# systemctl start intdash-web-admin.service
# systemctl start intdash-web-edges.service
# systemctl start intdash-web-me.service
# systemctl start intdash-web-measurements.service
# systemctl start intdash-web-media.service
# systemctl start intdash-web-oauth2-redirector.service
# systemctl start intdash-web-oauth2.service
# systemctl start intdash-web-signin-redirector.service
# systemctl start intdash-web-widget-app-links.service
```

(次のページに続く)

(前のページからの続き)

```
# systemctl start vm2m-dataviz-backend.service
```

3. 起動後、各サービスが正常に起動していることを確認します。

```
# systemctl status nginx.service
# systemctl status intdash-api-auth.service
# systemctl status intdash-api-gateway.service
# systemctl status intdash-api.service
# systemctl status intdash-micro-auth.service
# systemctl status intdash-micro-broker.service
# systemctl status intdash-micro-config.service
# systemctl status intdash-micro-measurement.service
# systemctl status intdash-micro-media.service
# systemctl status intdash-web-admin.service
# systemctl status intdash-web-edges.service
# systemctl status intdash-web-me.service
# systemctl status intdash-web-measurements.service
# systemctl status intdash-web-media.service
# systemctl status intdash-web-oauth2-redirector.service
# systemctl status intdash-web-oauth2.service
# systemctl status intdash-web-signin-redirector.service
# systemctl status intdash-web-widget-app-links.service
# systemctl status vm2m-dataviz-backend.service
```

正常に動作していれば Active: active (running) と表示されます。

3.4 管理者ユーザーのアカウントを設定する

初期状態では、「intdash」という名前の管理者ユーザーアカウントが存在します。本セクションではこのアカウントのメールアドレスを設定します。あわせて、メールサーバーの設定も行います。

3.4.1 メールサーバーと通知メール送信元を設定する

パスワードを忘れた場合の認証の際などに、intdash サーバーが使用者にメールを送信することがあります。その際に使用する送信元のメールアドレスと SMTP サーバーを設定します。

1. SSH で intdash サーバーに接続し、設定ファイルのディレクトリに移動します。

```
# cd /etc/intdash
```

2. テキストエディターで設定ファイルを開きます。

```
# vi authd.conf
```

4. [email] セクションに SMTP サーバーの情報と送信メールの情報を入力します。SMTP サーバーの情報はお使いの環境に合わせて設定してください。その後、設定ファイルを閉じます。

   - [email] セクション

    - from: 送信元として使用されるメールアドレス
    - reply-to: Reply-to ヘッダーに使用されるメールアドレス（設定は必須ではありません）
  - [email.smtp] サブセクション
    - ご使用の環境の SMTP サーバーの情報

```
[email]
  from = "noreply@example.com"
  from-display-name = "VM2M"
  reply-to = ""

  ...

  [email.smtp]
    address = "127.0.0.1:25"
    hostname = ""
    enable-tls = false
    enable-starttls-auto = false
    insecure-tls = false
    authentication = ""
    username = ""
    password = ""
```

5. 設定を反映させるため、以下のコマンドで intdash-micro-auth サービスを再起動します。

```
# systemctl restart intdash-micro-auth
```

6. 再起動後、intdash-micro-auth が正常に起動していることを確認します。

```
# systemctl status intdash-micro-auth
```

   正常に動作していれば `Active: active (running)` と表示されます。

3.4.2 管理者ユーザーのメールアドレスを設定する

1. ウェブブラウザーで `https://intdash.example.com/users/me/` にアクセスします。
2. サインイン画面が表示されたら、管理者ユーザー名 `intdash` と、さきほど設定したパスワード (INTDASH-USER-PASSWORD) でサインインします。

   ユーザー `intdash` のマイページが表示されます。

図 2　マイページ

3. [メールアドレス] をクリックして、管理者ユーザーのメールアドレスを入力し、[変更を保存] をクリックします。

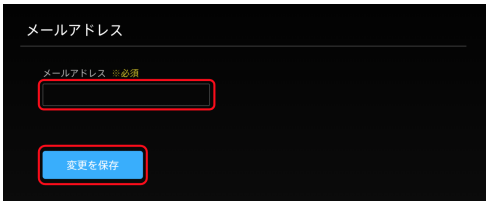


図 3　メールアドレスを設定

4. 以下のメッセージが表示されます。メールアドレスが正しいことを確認し、[閉じる] をクリックします。

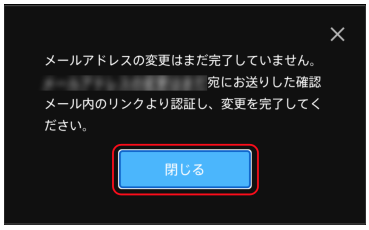
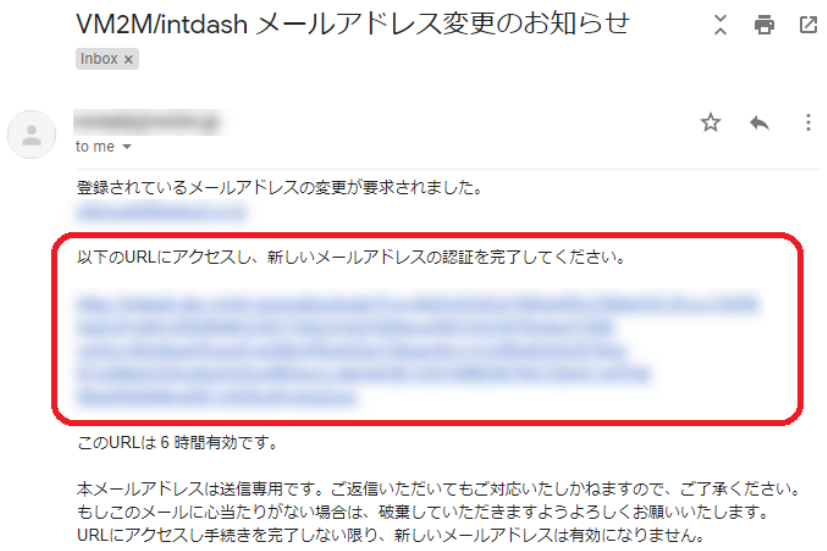


図 4　メールアドレスを確認

メールアドレス宛に確認メールが送信されます。

5. お使いのメールクライアントで確認メールを開きます。

メールに記載されているアクティベーション用 URL にアクセスします。



6. 再度、マイページの [メールアドレス] を開き、以下のように [認証済み] となっていれば設定は完了です。

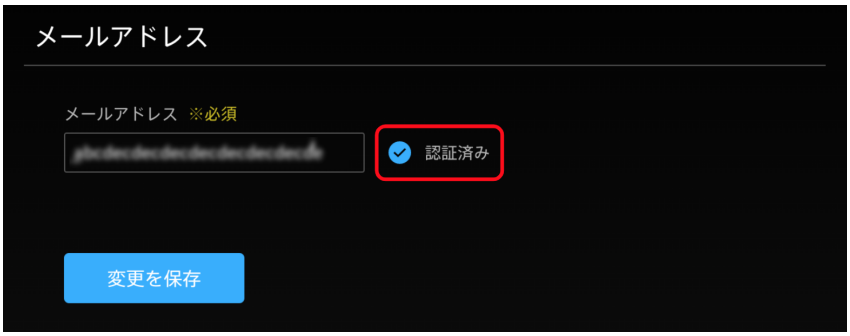


図 5　メールアドレス認証済み

3.5 Google Maps の API キーを設定する（Google Maps を使用する場合のみ）

Data Visualizer で Google Maps ビジュアルパーツを使用するためには、Google Maps の API キーを intdash サーバーに設定しておく必要があります。

注釈:

- Data Visualizer では、Open Street Map を使用することもできます。Open Street Map を使用する場合、ここで API キーを設定する必要はありません。
- Google Maps の詳細に関しては Google Maps 公式サイト を参照してください。

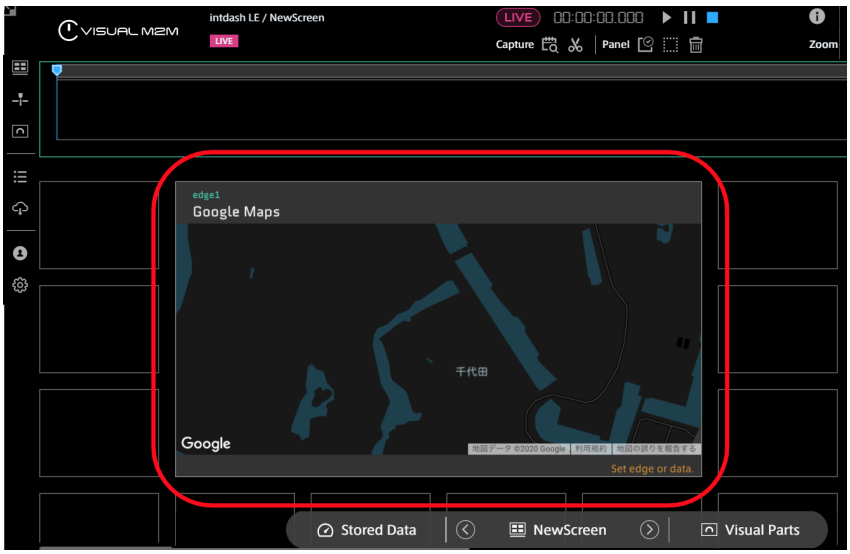


図 6　Google Maps ビジュアルパーツ

Google Maps の API キーの設定は、intdash サーバーの設定用 API に管理者ユーザーとしてアクセスして行います。

1. SSH で intdash サーバーに接続します。
2. 下記のコマンドで、管理者ユーザー intdash のパスワードをシェル変数に設定します。

```
# read -s PASSWORD
<-- Enter the password and press <Enter>.
```

3. 下記のコマンドで、intdash にサインインします。

```
# curl -X POST http://127.0.0.1:8080/api/auth/oauth2/token \
-H "Content-Type: application/x-www-form-urlencoded" \
-d "username"="intdash" \
-d "password"="${PASSWORD}" \
-d "grant_type"="password" \
-d "client_id"="99dcf67c_default.aptpod.co.jp"
```

4. 下記のコマンドで、前の手順でのレスポンスの JSON に含まれる `access_token` の値をシェル変数に設定します。

```
# read -s ACCESS_TOKEN
<-- Enter the access token and press <Enter>.
```

5. 下記のコマンドで、Google Maps API キーを設定します。

   お持ちの Google Maps API キーを、`googleMapsApiKey` の値として入力してください。

   以下では、`-d '{"content" ...` の行は表示の都合で折り返して表示されていますが、実際はこの行内に改行は不要です。

```
# curl -X PUT http://127.0.0.1:8080/api/v1/configs/vm2m-2nd-variables \
-H "Content-Type: application/json" \
```

(次のページに続く)

(前のページからの続き)

```
-H "Authorization: Bearer ${ACCESS_TOKEN}" \
-d '{"content":"{\"googleMapsApiKey\":\"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\",\
↪"googleMapsApiClientID\":\"\"}"}'
```

注釈: API キーではなく Client ID をお持ちの場合は、下記のように `googleMapsApiClientID` の値として入力してください。

```
-d '{"content":"{\"googleMapsApiKey\":\"\",\"googleMapsApiClientID\":\
↪"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX\"}"}'
```

以上で設定は完了です。動作を確認する (p. 25) に進んでください。

04 動作を確認する

本章では、簡易的な動作確認として以下を行います。

- 動作確認用に新しくユーザー（ user1 ）とエッジ（ edge1 ）を作成する。
- iPhone で intdash Motion アプリケーションを起動し、edge1 として動画を撮影しながら intdash サーバーに送信する。
- PC で Data Visualizer アプリケーションを使って動画を確認する。
- Data Visualizer で過去のデータを再生する。

4.1 ユーザーアカウントを作成する

管理者ユーザーとして Admin Console にログインし、ユーザーアカウント（user1）を作成します。

1. ウェブブラウザーで Admin Console `https://intdash.example.com/admin/users` にアクセスします。
2. サインイン画面が表示されたら、管理者ユーザー名 `intdash` と、さきほど設定したパスワードでサインインします。
3. [ユーザーを作成] をクリックし必要事項を入力して、[作成] をクリックします。
   - 名前: `user1`
   - ロール: `member`（一般ユーザー）
4. ユーザーが作成され、一時パスワードが作成されます。一時パスワードをメモしておきます。

4.2 エッジアカウントを作成する

引き続き Admin Console で、エッジアカウントを作成します。

1. [エッジ一覧] で、[エッジを作成] をクリックし、必要事項を入力して、[作成] をクリックします。
   - 名前: `edge1`

   UUID とクライアントシークレットが表示されますが、動作確認ではこれらは必要ありません。
2. 右上の  をクリックし、ログアウトします。以降の手順では新しいユーザー `user1` を使用します。

4.3 ダッシュボードを準備する

1. Chrome ブラウザーで Visual M2M Data Visualizer `https://intdash.example.com/vm2m/` にアクセスし、ユーザー `user1` としてログインします。

   初めてログインする場合、パスワード再設定する必要があります。画面の指示に従ってパスワードを再設定してください。
2. [Links]>[Download DAT File] から、動画データのデータ設定ファイル（DAT ファイル）をダウンロードします。

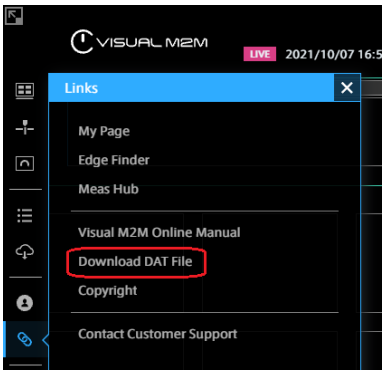


図 7　DAT ファイルダウンロードページを開く

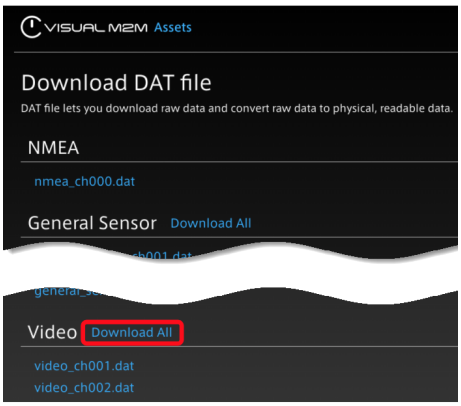


図 8　動画用の DAT ファイルをダウンロード

3. [Data Settings]>[Import] をクリックし、ダウンロードしたデータ設定ファイルをインポートします。

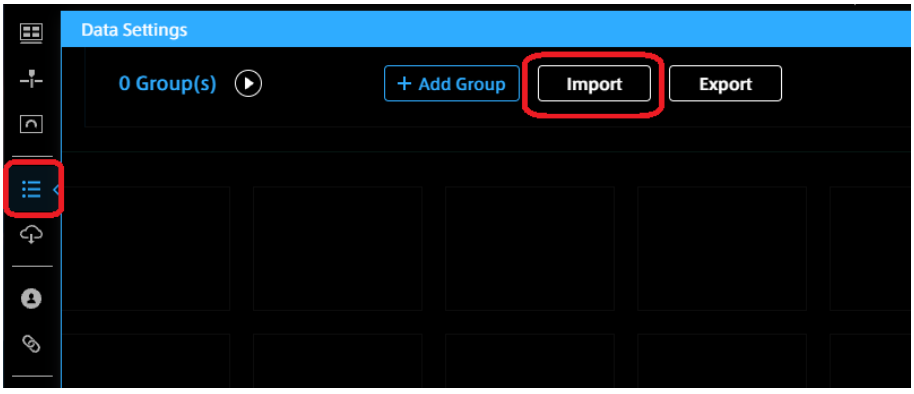


図 9　DAT ファイルのインポート

インポートが成功すると、以下のように「Video」というグループが作成されます。

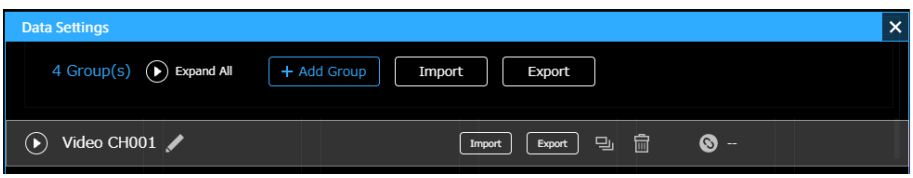


図 10　Video グループ

4. ダッシュボード上にパネルを作成します。

図 11　パネルを作成

5. パネルをクリックし、データ送信元として edge1 を選択し、Video:ch_001 をバインドします。

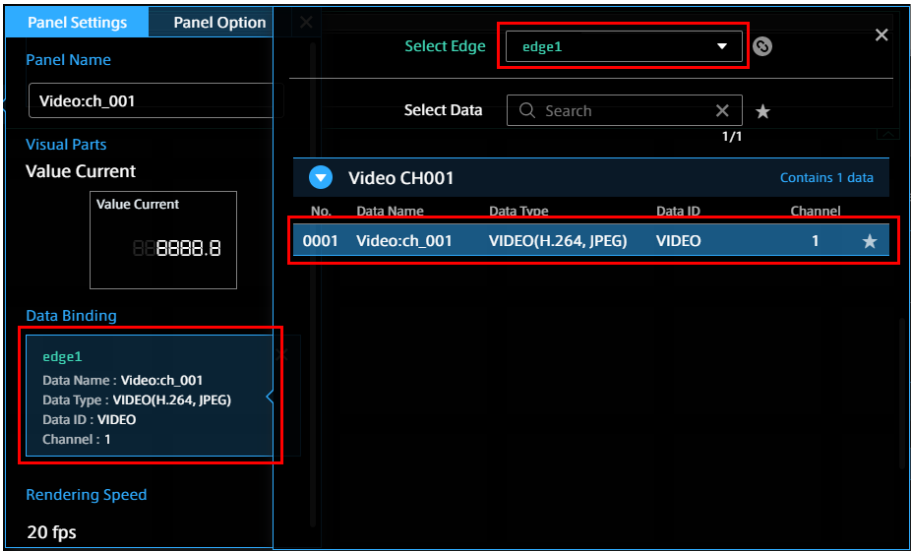


図 12　データ送信元を選択し、動画データをバインド

6. ビジュアルパーツは、Video Player を選択します。

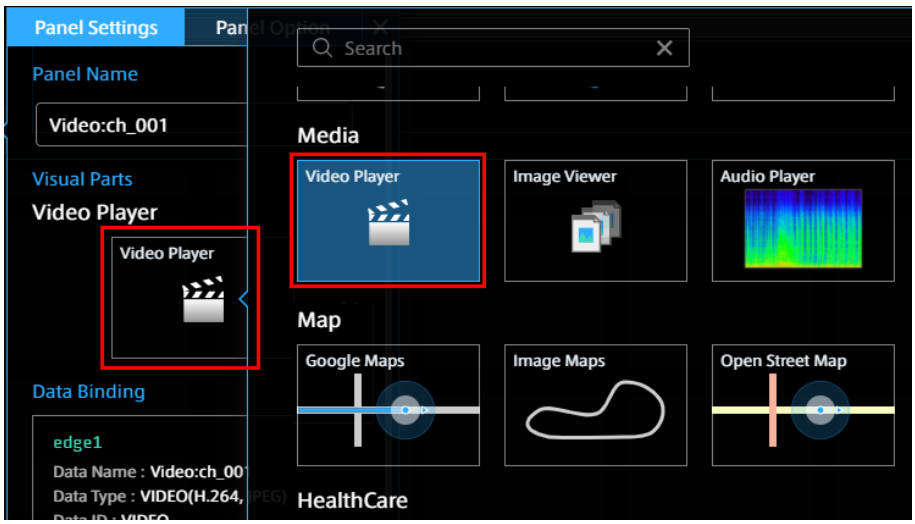


図 13　Video Player を選択

4.4 iPhone からリアルタイム動画を送信する

iPhone で intdash Motion アプリケーションを起動し、動画を送信します。

1. iPhone で intdash Motion を起動し、ログインします。

   ログインには以下の情報を使用します。

   - URL: intdash サーバーの URL `https://intdash.example.com`
   - ユーザー名: ユーザーアカウントを作成する (p. 25) で作成した `user1`
   - パスワード: user1 に設定したパスワード

   > **注釈:** ログインは、エッジとしてではなくユーザー（user1）として行います。

2. [Settings] を開いて、以下のように設定します。

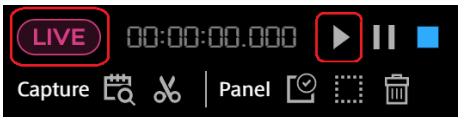
   - [Send Data As]: `edge1`
   - [Video]: オン
     - [Stream to Server]: オン
     - [Save to Server]: オン
     - [Channel]: 1
     - [Codec and Options]: H.264

3. Main 画面に戻り、▶ をタップして、撮影と送信を開始します。

4.5 Data Visualizer で表示を確認する

1. Data Visualizer で LIVE 再生モードにして、[Play] をクリックします。

   intdash Motion で撮影している動画がリアルタイムに表示されます。



2. 動作確認ができたら、intdash Motion で■をタップし、計測を終了します。
3. Data Visualizer で、[Stored Data]>[Measurements] をクリックすると、過去に取得したデータが表示されます。上の手順で撮影したデータが表示されているはずですので、それをクリックして再生します。

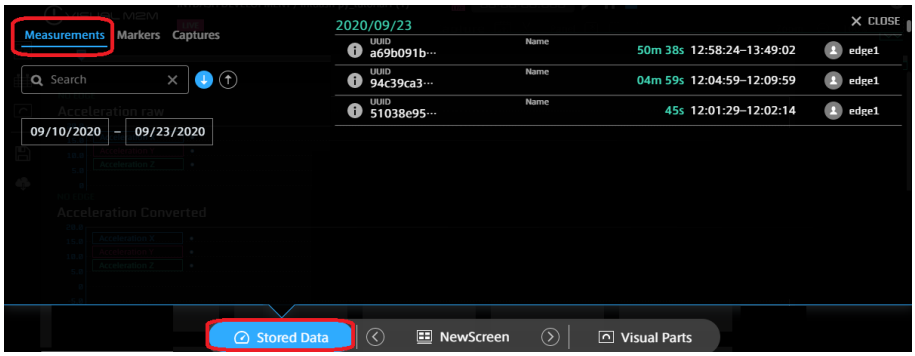


図 14　過去に取得したデータのリストから再生

以上で動作確認は終了です。

05 付録：intdash を構成するサービスと設定ファイル

intdash を構成する各サービスの設定ファイル、サービスが使用するポート、用途は以下の通りです。

5.1 独自アプリケーション

intdash を構成している独自アプリケーションは以下の通りです。

- intdash-api
  - 設定ファイル: `/etc/intdash/intdashd.conf`
  - サービスポート: 8097/tcp、8179/tcp
  - 用途: intdash の API サービス。各種マイクロサービスへのプロキシも担う。
- intdash-micro-auth
  - 設定ファイル: `/etc/intdash/authd.conf`
  - サービスポート: 8094/tcp
  - 用途: 認証を担うマイクロサービス
- intdash-micro-broker
  - 設定ファイル: `/etc/intdash/brokerd.conf`
  - サービスポート: 8180/tcp、8178/tcp
  - 用途: 計測データのリアルタイム処理を担うマイクロサービス
- intdash-micro-measurement
  - 設定ファイル: `/etc/intdash/measurementd.conf`
  - サービスポート: 8095/tcp
  - 用途: 計測データの保存や過去計測の読み込みを担うマイクロサービス
- intdash-micro-config
  - 設定ファイル: `/etc/intdash/configd.conf`
  - サービスポート: 8096/tcp
  - 用途: スマートフォンアプリの設定を管理するマイクロサービス
- intdash-micro-media
  - 設定ファイル: `/etc/intdash/mediad.toml`
  - サービスポート: 8085/tcp
  - 用途: H.264 動画の計測データを取り扱うサービス
- intdash-web-oauth2-redirector
  - 設定ファイル: `/etc/sysconfig/intdash-web-oauth2-redirector`
  - サービスポート: UNIX domain socket
    - `/var/run/intdash/intdash-web-oauth2-redirector.sock`
  - 用途: Oauth2 認証クライアントアプリ用のリダイレクト URL を生成する。

- **intdash-web-signin-redirector**
  - 設定ファイル: `/etc/sysconfig/intdash-web-signin-redirector`
  - サービスポート: UNIX domain socket
    - `/var/run/intdash/intdash-web-signin-redirector.sock`
  - 用途: 認証系におけるサインイン機能のリダイレクト URL を生成する。
- **intdash-web-me**
  - 設定ファイル: `/etc/sysconfig/intdash-web-me`
  - サービスポート: 13000/tcp
  - 用途: ログインしているユーザー（自分自身）についての情報を表示するウェブアプリケーション「My Page」
- **intdash-web-edges**
  - 設定ファイル: `/etc/sysconfig/intdash-web-edges`
  - サービスポート: 13001/tcp
  - 用途: エッジの情報表示のためのウェブアプリケーション「Edge Finder」
- **intdash-web-oauth2**
  - 設定ファイル: `/etc/sysconfig/intdash-web-oauth2`
  - サービスポート: 13003/tcp
  - 用途: Oauth2 認証においてクライアント側の処理を行うサービス
- **intdash-web-measurements**
  - 設定ファイル: `/etc/sysconfig/intdash-web-measurements`
  - サービスポート: 13004/tcp
  - 用途: 計測管理のためのウェブアプリケーション「Meas Hub」
- **intdash-web-widget-app-links**
  - 設定ファイル: `/etc/sysconfig/intdash-web-widget-app-links`
  - サービスポート: 13005/tcp
  - 用途: 他のウェブアプリケーションへのリンクの生成（アプリケーションアイコンによるリンク）を担うサービス
- **intdash-web-admin**
  - 設定ファイル: `/etc/sysconfig/intdash-web-admin`
  - サービスポート: 13006/tcp
  - 用途: ユーザーとエッジ管理のためのウェブアプリケーション「Admin Console」
- **vm2m-dataviz-backend**
  - 設定ファイル: `/etc/sysconfig/vm2m-dataviz-backend`
  - サービスポート: UNIX domain socket
  - 用途: 計測の可視化ウェブアプリケーションの設定等を保持するサービス

5.2 オープンソースアプリケーション

intdash では以下の OSS(Open Source Software) を利用しています。

- **ウェブサーバー nginx** (https://nginx.org/en/)
  - 設定ファイル: `/etc/nginx` 以下（intdash ウェブサーバーの設定: `conf.d/intdash.conf` 、各アプリケーションの設定: `intdash.d` 以下)
  - サービスポート: 80/tcp、443/tcp
- **エッジルーター Traefik** (https://doc.traefik.io/traefik/)
  - 設定ファイル: `/etc/intdash` 以下（メインの設定: `traefik.toml` 、各アプリケーションの設定: `traefik.d` 以下)
  - サービスポート: 8080/tcp、18080/tcp
- **リレーショナルデータベース PostgreSQL** (https://www.postgresql.org/)
  - 設定ファイル: `/data/pgsql/11/data` 以下
  - サービスポート: 5432/tcp
- **時系列データベース InfluxDB** (https://www.influxdata.com/)
  - 設定ファイル: `/etc/influxdb/influxdb.conf`
  - サービスポート: 8086/tcp