

PORUTO

実際に遭遇した怖い実装や構築における脆弱性について

自己紹介

- 名前:箱崎 篤樹
- セキュリティエンジニア(オフENSEシブセキュリティ、Red Hat)
- 資格:OSCE3, HTB(CWES, CWEE, CPTS), CRTOなど
- 実績: HackTheBox: Omnisientランク(全完達成者)
- お仕事(株式会社ポルト)
 - 脆弱性診断(Web、API、プラットフォーム)
 - ペネトレーションテスト、Red team
- 他社での活動
 - 0Dayエクスプロイトの研究開発など
 - Synack SRT



自己紹介

- 名前:箱崎 篤樹
- セキュリティエンジニア(オフENSEシブセキュリティ、Red Hat)
- 資格:OSCE3, HTB(CWES, CWEE, CPTS), CRT0など
- 実績: HackTheBox: Omnisientランク(全完達成者)
- お仕事(株式会社ポルト)
 - 脆弱性診断(Web、API、プラットフォーム)
↑今日は脆弱性診断についてお話させていただきます。
 - ペネトレーションテスト、Red team
- 他社での活動
 - 0Dayエクスプロイトの研究開発など
 - Synack SRT



会社情報

脆弱性診断を中心にセキュリティ対策を支援するセキュリティベンダー

- ✓ 情報漏洩につながる脆弱性の診断をはじめ、ユーザ企業のセキュリティ方針策定などにも携わった深い専門知識と経験を持ったメンバーにより、皆様の安全なサービス運営やセキュリティ対策を支援します

会社概要

商号: 株式会社ポルト

設立: 2019年4月

代表: 山崎 康平

資本金: 1,120万円

所在地: 〒101-0062
東京都千代田区神田駿河台2-3-11
ヒューリック御茶ノ水3階

事業内容: セキュリティ診断

- ・ 情報セキュリティコンサルティング
- ・ セキュリティシステムの企画設計、開発および運用
- ・ 情報セキュリティサービス

基準審査登録番号: [021-0019-20](#)

本日、お話をさせていただく内容について

- 実際に遭遇した事例をお話させていただきますが、特定のリスクや機密保持の観点から、顧客情報は使用しておりません。
- 根本的な脆弱性の説明としては変わりませんが、パラメータ名などを変更するなどをしてお客様やシステムが特定されないようにしております。

世の中のシステムって本当に、
またはどれくらい脆弱性があるの？



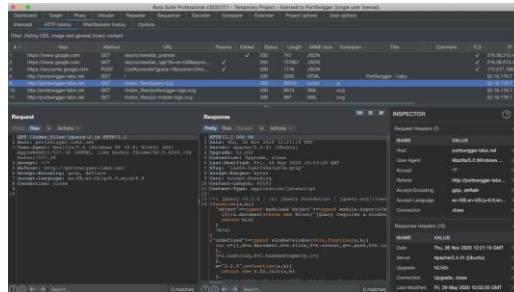
ユーザから送信されたパラメータを検
証せずそのままデータベースへ登録し
ていることなどもある



診断方法について

- 診断ではBurp Suiteというプロキシツールを使用して診断作業を行います
- Burp Suiteは自動スキャン機能があり、ツールによる診断も可能
- Burp Suiteで発生通信を見ながら、値を改竄していくことによって脆弱性を探す
- フォームなどで選択をすることによってフロントエンドで選択肢がないものについても中間プロキシを使用することによって、**意図しないパラメータの送信が可能**

能



脆弱性診断をしていて多いと感じる脆弱性

1. 認証認可の不備(リスクレベルHigh)

1. 他ユーザのデータの参照や改竄

2. ビジネスロジックに対する不備(リスクレベルHigh)

1. 課金機能が無課金で使用する。
2. 本来、一般ユーザが登録できないデータの登録によるサービスの利用

3. クロスサイトスクリプティング(リスクレベルMiddle)

- システムがどのように機能して、どのようなパラメータがシステムにどう作用するのかを理解していないと発見できないような脆弱性
- つまり、スキャナーなどのツール診断では見つかりにくい脆弱性

本日の内容

本当に怖い

クライアントサイド処理



このログイン処理のコードを知っていますか？

```
<script>
function authenticateUser(username,
password) {
    var accounts = apiService.sql(
        "SELECT * FROM users"
    );

    for (var i = 0; i < accounts.length;
i++) {
        var account = accounts[i];
        if (account.username === username
&&
            account.password === password)
        {
            return true;
        }
    }
    if ("true" === "true") {
        return false;
    }
}
```

```
$('#login').click(function() {
    var username = $("#username").val();
    var password = $("#password").val();

    var authenticate =
authenticateUser(username, password);

    if (authenticate === true) {
        $.cookie('loggedin', 'yes', {
expires: 1 });
    } else if (authenticate === false) {
        $("#error_message").show();
    }
});
</script>
```

このログイン処理のコードを知っていますか？

```
<script>
function authenticateUser(username,
password) {
```

```
    var accounts = apiService.sql(
```

```
        "SELECT * FROM "
```

```
    );
```

```
    for (i++;) {
```

```
        &&
```

```
        return true;
```

```
    }
```

```
    if ("true" === "true") {
        return false;
    }
```

```
}
```

```
$('#login').click(function() {
    var username = $("#username").val();
    .val();
```

```
rd);
```

```
, {
```

```
else) {
```

```
    $("#error_message").show();
```

```
    }
```

```
});
</script>
```

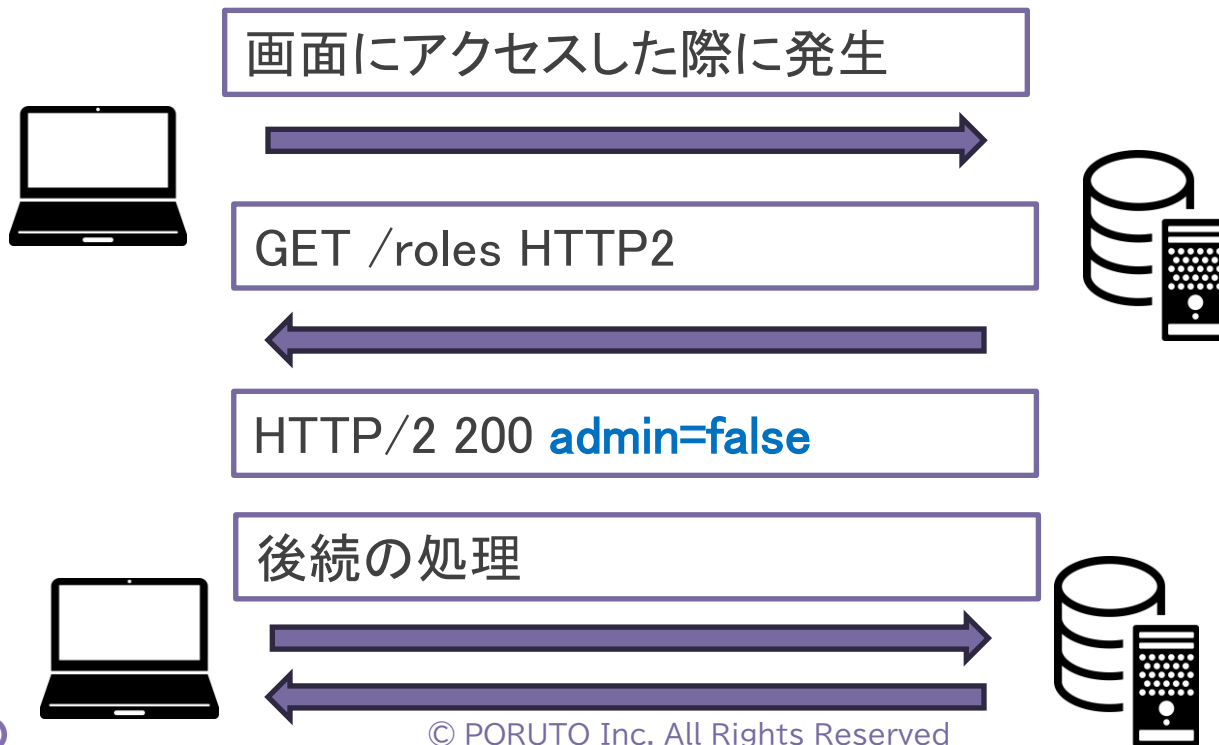
実はこのログイン処理のコード
は世界最悪のログイン処理コード
として有名なコードです。

実際にあった事例を ご紹介



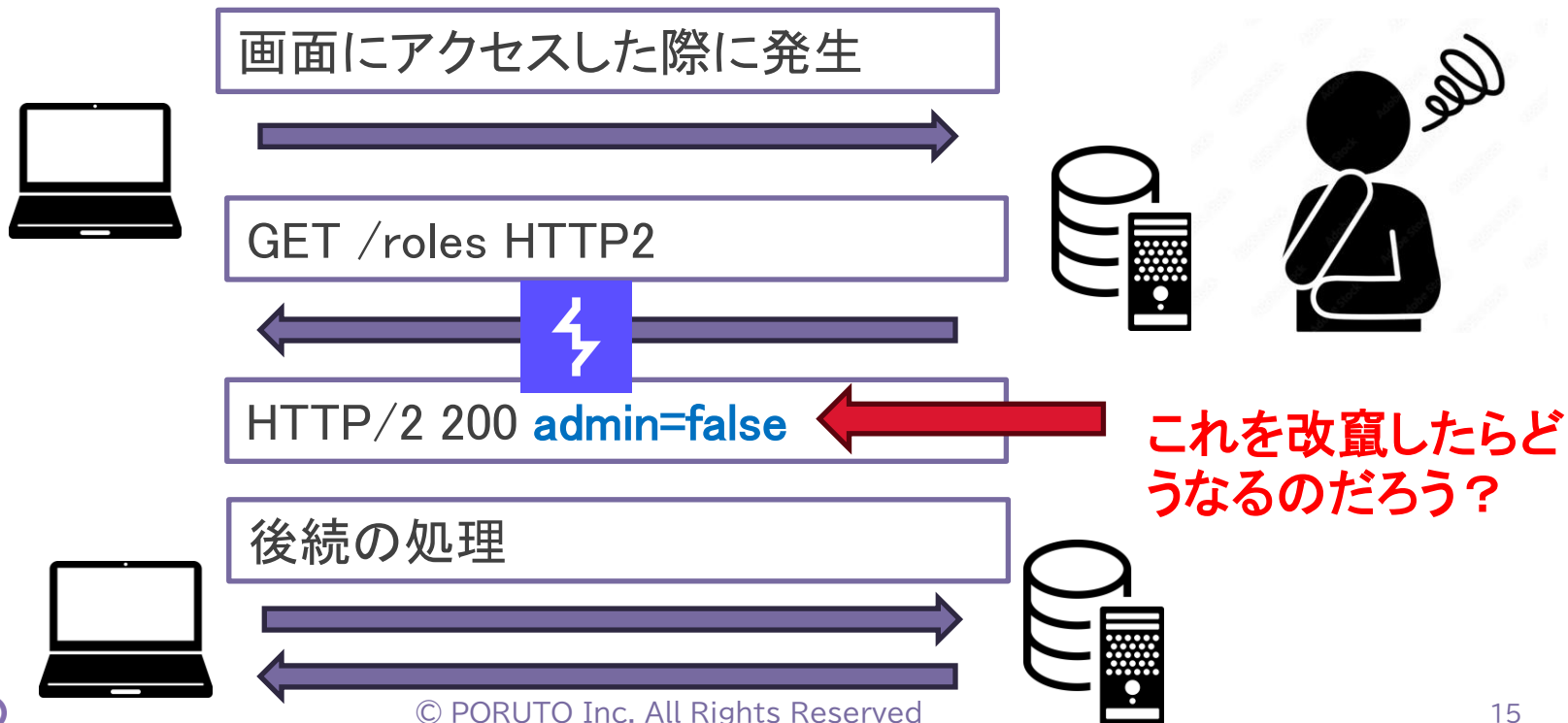
クライアントサイド処理 事例1

SaaSを使用してクライアントサイドで処理がされていた事例



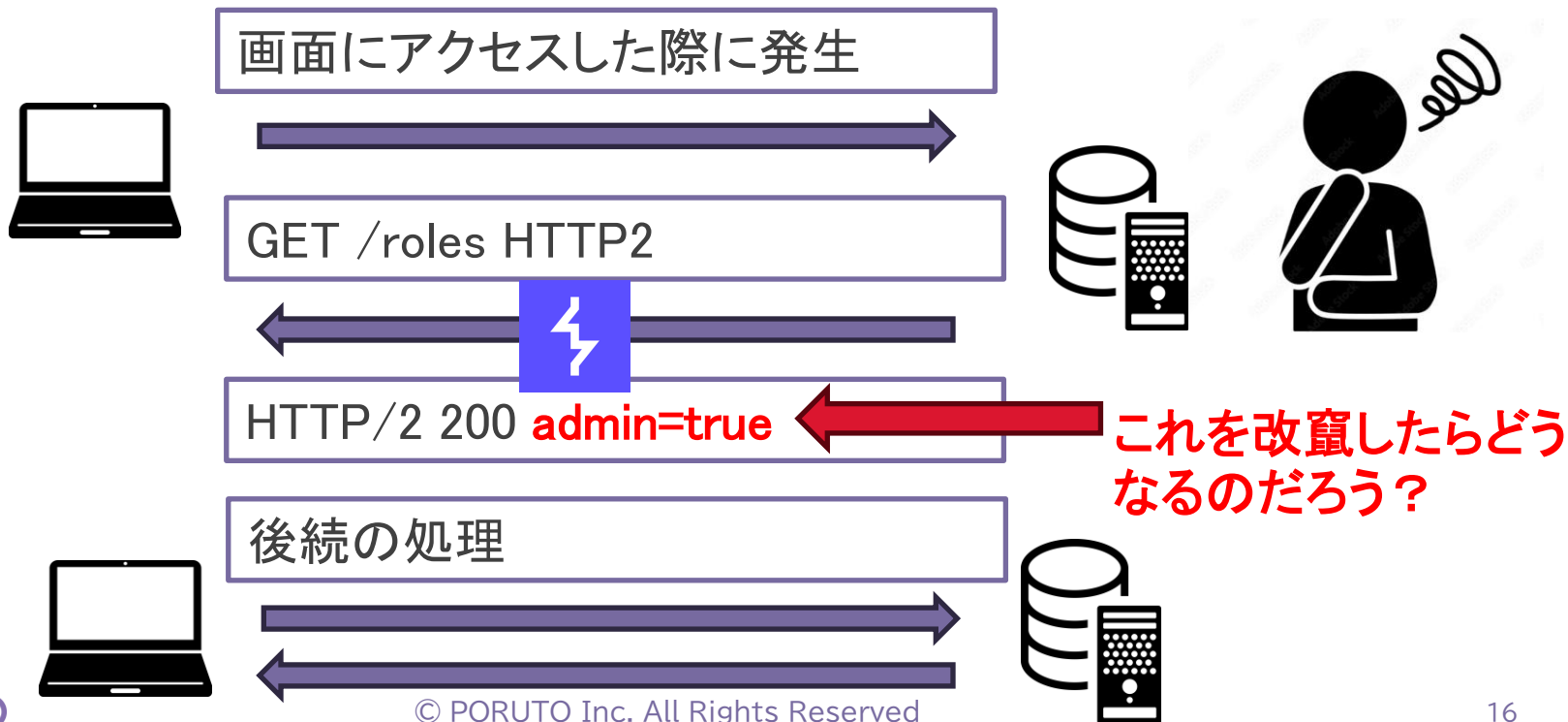
クライアントサイド処理 事例1

SaaSを使用してクライアントサイドで処理がされていた事例



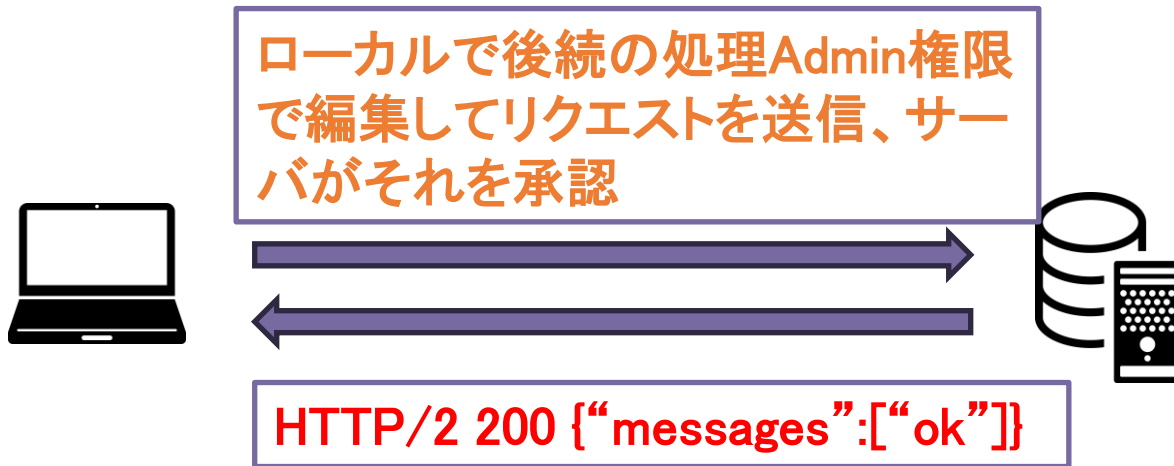
クライアントサイド処理 事例1

SaaSを使用してクライアントサイドで処理がされていた事例



クライアントサイド処理 事例1

SaaSを使用してクライアントサイドで処理がされていた事例

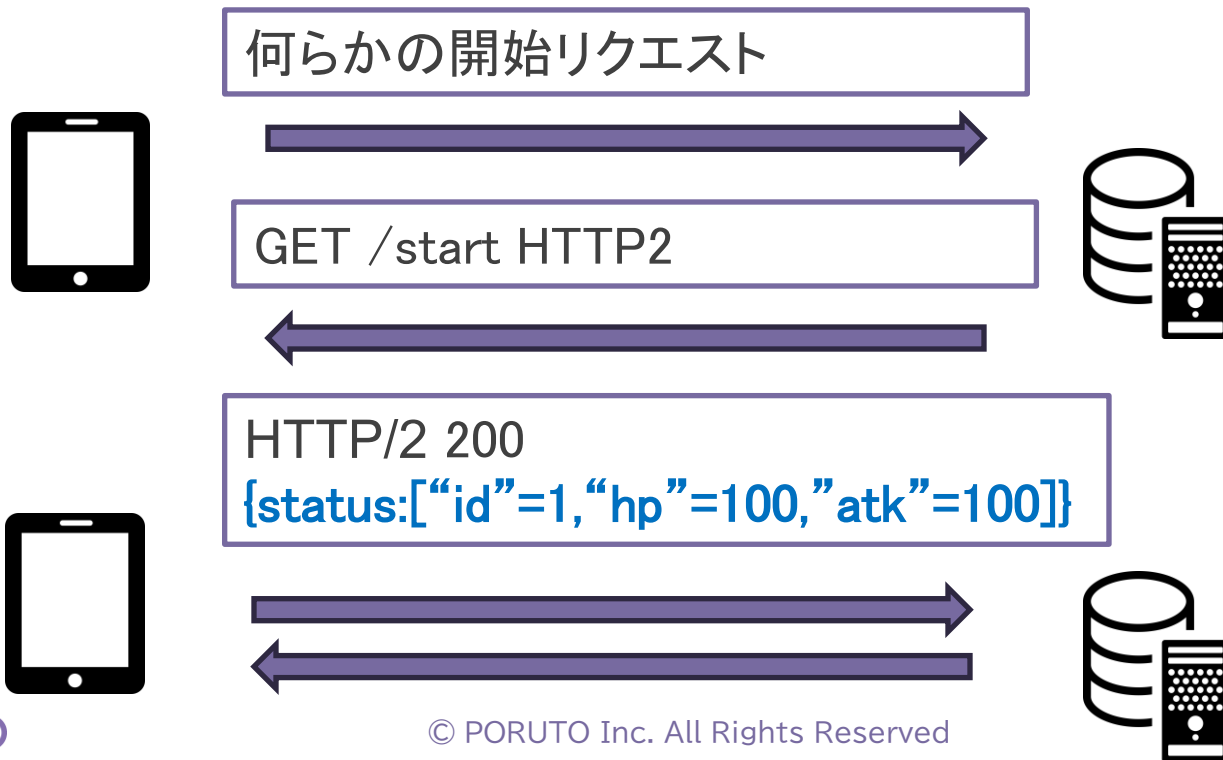


権限以上の操作が可能



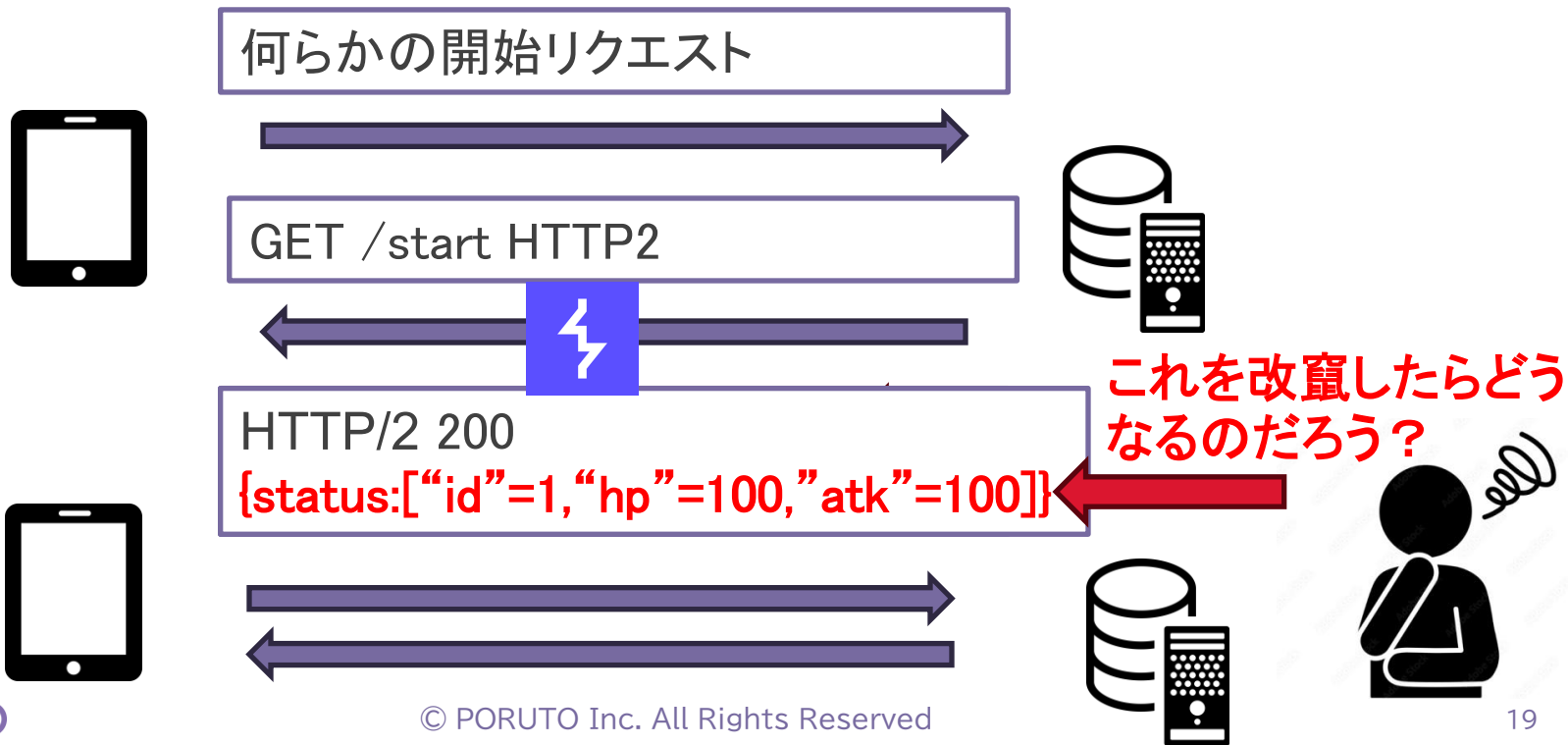
クライアントサイド処理 事例2

ゲームアプリでクライアントサイドで処理をしていた事例



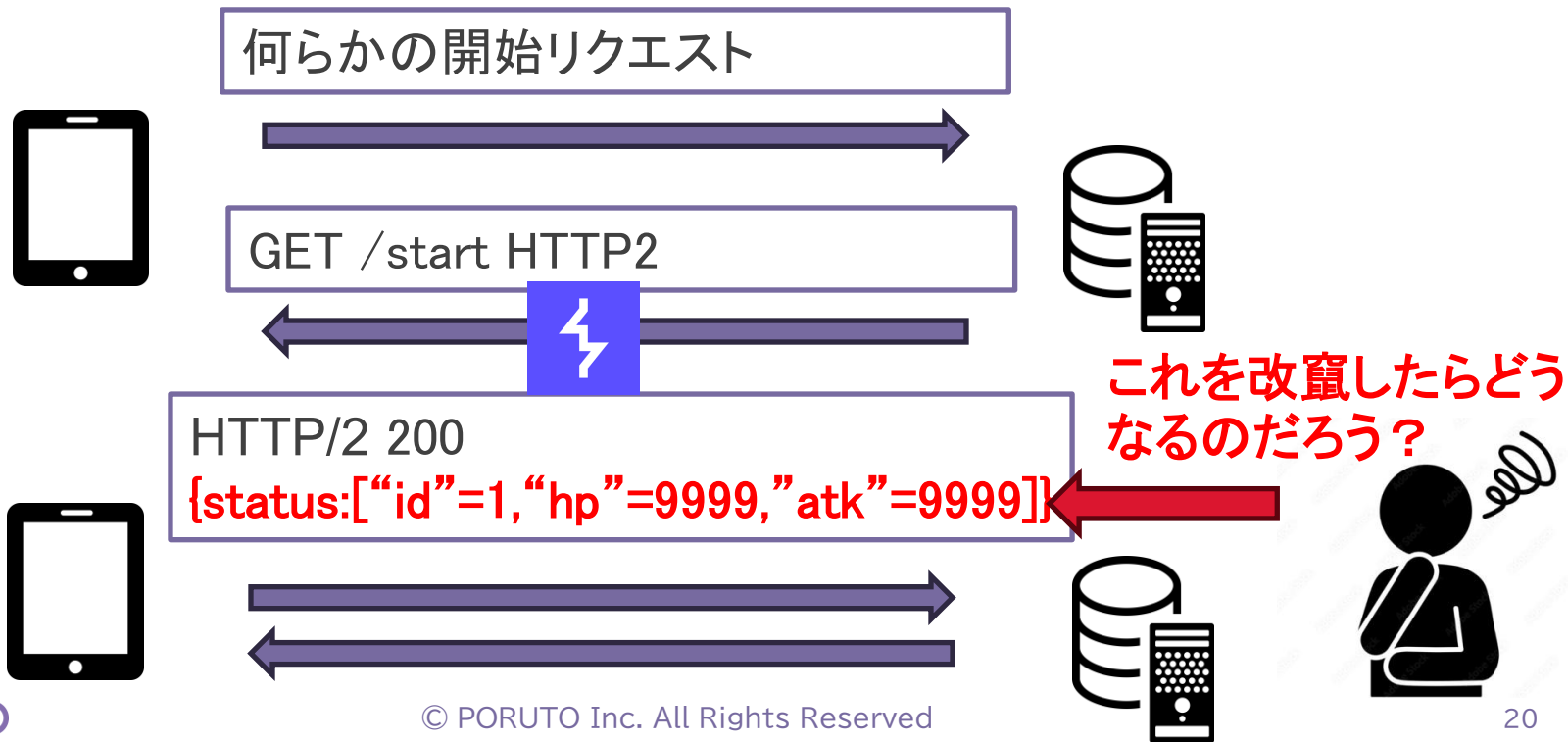
クライアントサイド処理 事例2

ゲームアプリでクライアントサイドで処理をしていた事例



クライアントサイド処理 事例2

ゲームアプリでクライアントサイドで処理をしていた事例



クライアントサイド処理 事例2

ゲームアプリでクライアントサイドで処理をしていた事例

改竄した値で、ゲームが進行してクリアしたリクエストが送信され、それをサーバが承認する。



```
HTTP/2 200  
{“messages”:[“ok”]}
```

プレイヤーの改竄が可能



被害にあわないために

- ユーザが送信する値は改竄が可能です。
 - ユーザが正常値での改竄を行った際にサーバー側でその入力値を信頼すると、悪用につながる可能性があります。
- 悪用の被害にあわないためには、ユーザからの入力をサーバーサイドで厳格にチェックすることを推奨します。



クライアントサイド処理 以外の脆弱性もたくさんある

仮説と検証

- システムの構成を見ながら、各機能におけるパラメータがどの機能に作用してどのように使用されるかなどを見ながら仮説を立てて検証をしていきます。
- では、実際に仮説通り動いてしまった時(ビジネスロジックの悪用など)
→ **WTF..??**
- セキュリティ業務をやっていて一番楽しい瞬間



まとめ



攻撃者に悪用され被害にあわないため
にセキュアコーディングや
脆弱性診断等を行ったりしながら
安全なシステム運用をしましょう



ご清聴
ありがとうございました