4.7 クッキー出力にまつわる脆弱性

セッション変数は外部から変更できないのに対して、クッキー値はアプリ利用者によって、書き換えができます

	クッキー	セッション変数
使いやすさ	APIにより設定、取得	変数とほぼ同じように使える
配列やオブジェクトの格納	アプリ側で文字列に変換する必要あり	通常の変数と同様に使える場合が多い
サイズの制限	厳しい制限あり	実用上は無制限
利用者による格納情報の直接参照	容易	**不可能**
クッキー脆弱時の情報漏洩しやすさ	クッキー漏洩すればデータも漏洩	漏洩のしにくさを制御可能
利用者によるデータ改変	容易	**不可能**
第三者によるデータ改変	XSSやHTTPヘッダインジェクション などの脆弱性があれば可能	**不可能**
情報の寿命の制限	容易	セッションの寿命
異なるサーバとの情報共有	ドメインが同じであれば可能	基本的に不可能

クッキーのセキュア属性の不備

HTTPとHTTPSが混在するサイトで、セッションIDを保持するクッキーにセキュア属性をつけると、HTTPページでセッション管理機能が使えなくなるので、アプリが動かなくなります。

クッキーのセキュア属性不備に対する対策

①サイト全体を常時TLSにして、クッキーに secure属性 をつける

PHPの場合には、php.ini に以下を設定する

```
session.cookie_secure = On
```

Apache Tomcat の場合は、HTTPS接続された状況でリクエストされるときには、セッションIDのクッキーに対しては自動的にセッション属性をつけることができます

ASP.NETの場合には、web.configファイルに以下を設定します

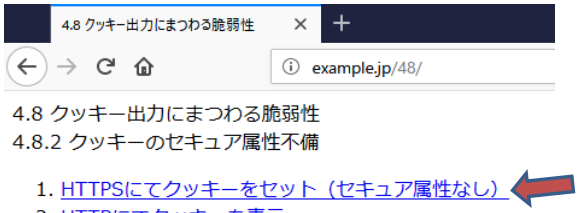
```
<configuration>
 <system.web>
   <httpcookies requireSSL="true" />
 </system.web>
</configuration>
```

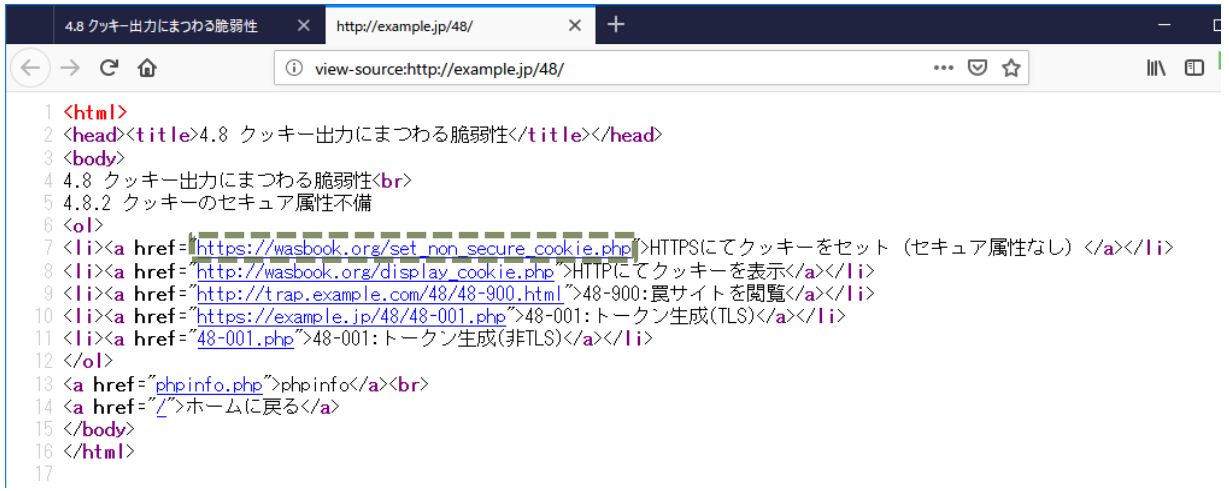
②セッションIDとは別にセキュア属性つきのクッキーとして、トークンを発行し、ページごとにトークンを確認する **48-001:トークン生成(TLS)**

4.8.2 クッキーのセキュア属性不備

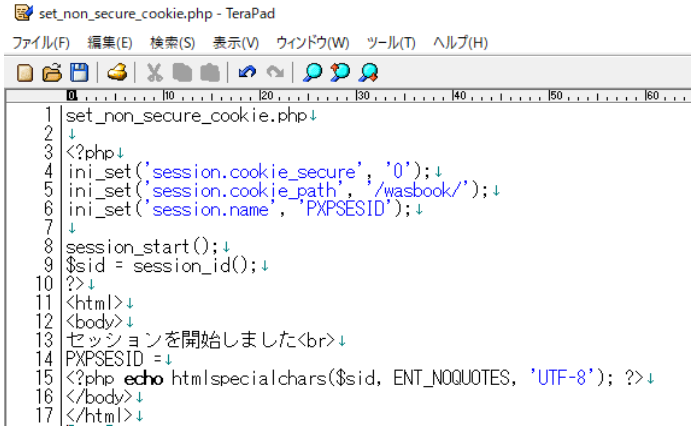
HTTPSにてクッキーをセット（セキュア属性なし）
HTTPにてクッキーを表示
48-900:罠サイトを閲覧

【ブラウザ】

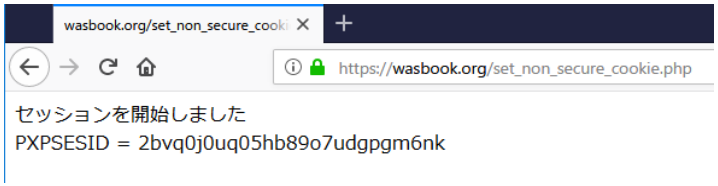


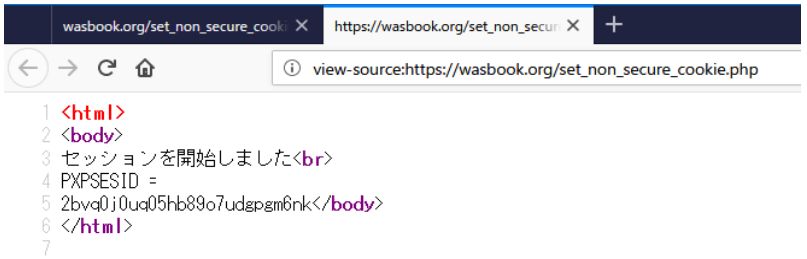


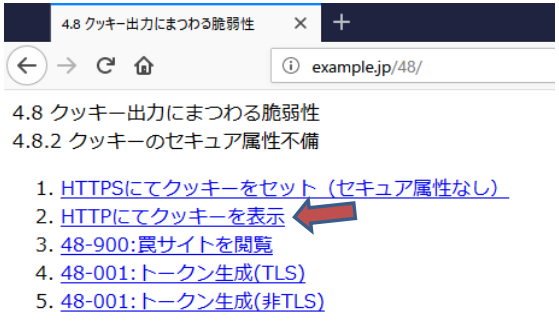
【httpsサーバ： wasbook.org/set_non_secure_cookie.php 】



【ブラウザ】


wasbook.org/set_non_secure_cook
https://wasbook.org/set_non_secure_cookie.php
セッションを開始しました
PXPSESID = 2bvq0j0uq05hb89o7udgpgm6nk

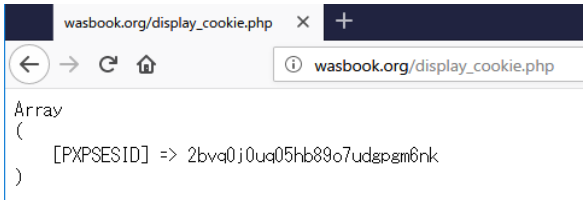

wasbook.org/set_non_secure_cook
https://wasbook.org/set_non_secur
view-source:https://wasbook.org/set_non_secure_cookie.php
<html>
<body>
セッションを開始しました
PXPSESID =
2bvq0j0uq05hb89o7udgpgm6nk</body>
</html>

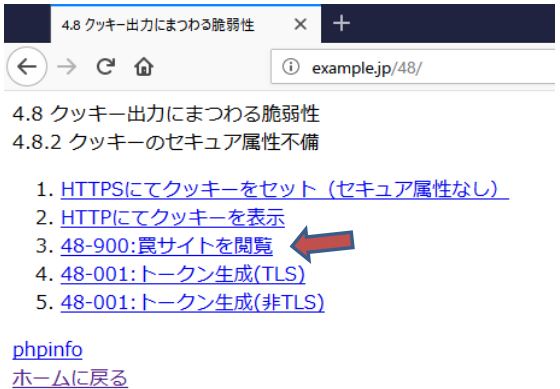

4.8 クッキー出力にまつわる脆弱性
example.jp/48/
4.8 クッキー出力にまつわる脆弱性
4.8.2 クッキーのセキュア属性不備
1. HTTPSにてクッキーをセット（セキュア属性なし）
2. HTTPにてクッキーを表示
3. 48-900:罠サイトを閲覧
4. 48-001:トークン生成(TLS)
5. 48-001:トークン生成(非TLS)
phpinfo
ホームに戻る

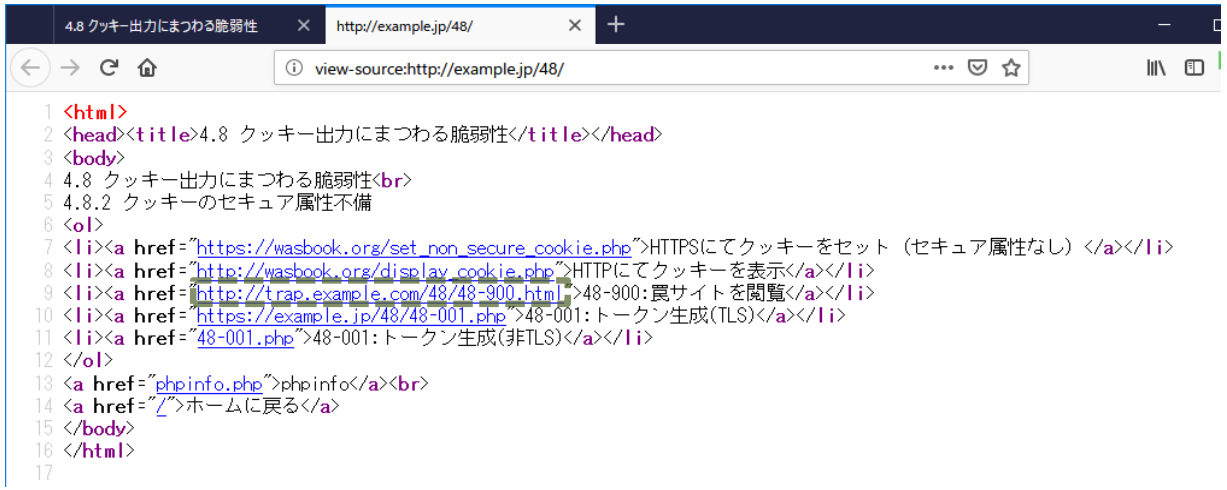

4.8 クッキー出力にまつわる脆弱性
http://example.jp/48/
view-source:http://example.jp/48/
<html>
<head><title>4.8 クッキー出力にまつわる脆弱性</title></head>
<body>
4.8 クッキー出力にまつわる脆弱性
4.8.2 クッキーのセキュア属性不備

HTTPSにてクッキーをセット（セキュア属性なし）
HTTPにてクッキーを表示
48-900:罠サイトを閲覧
48-001:トークン生成(TLS)
48-001:トークン生成(非TLS)

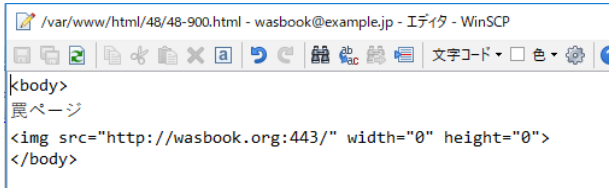
phpinfo
ホームに戻る
</body>
</html>


wasbook.org/display_cookie.php
wasbook.org/display_cookie.php
Array
(
[PXPSESID] => 2bvq0j0uq05hb89o7udgpgm6nk
)

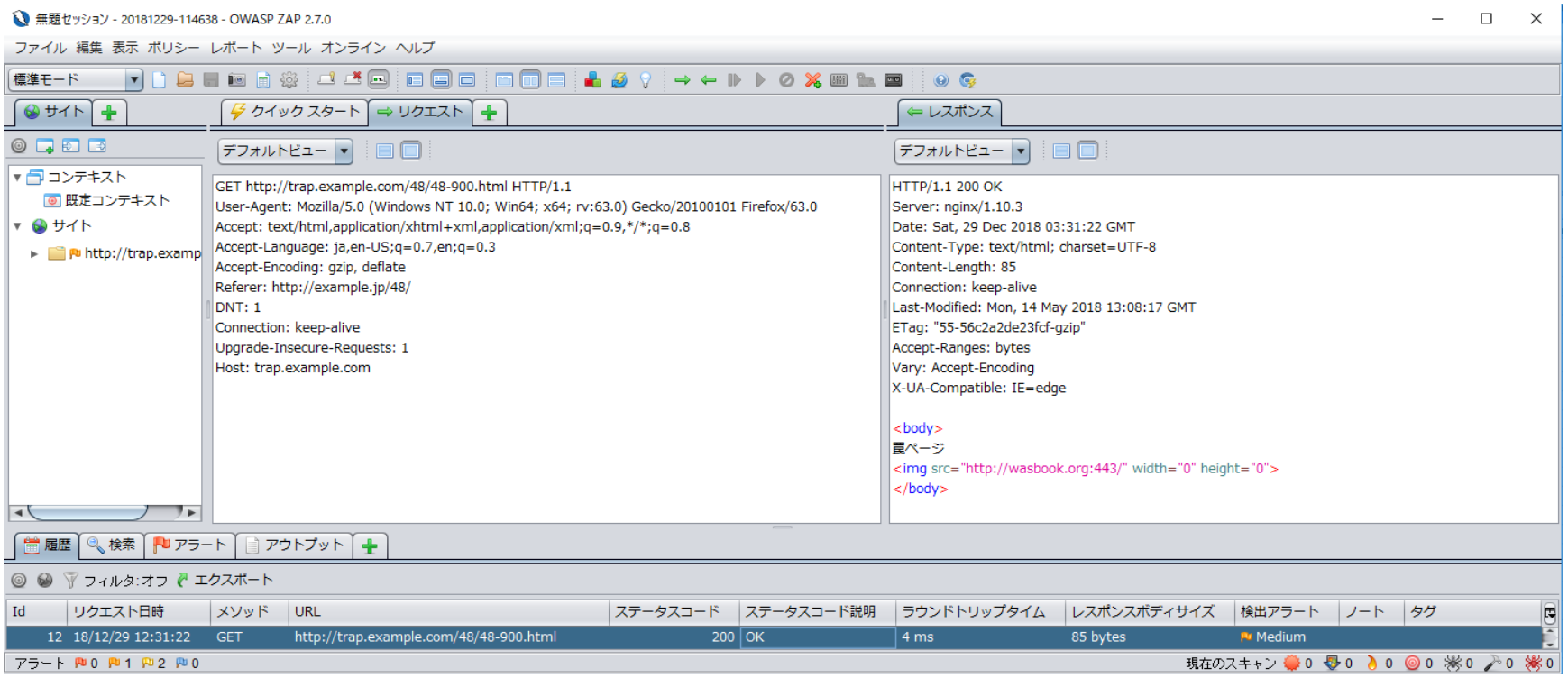




【サーバ： 48/48-900.html 】

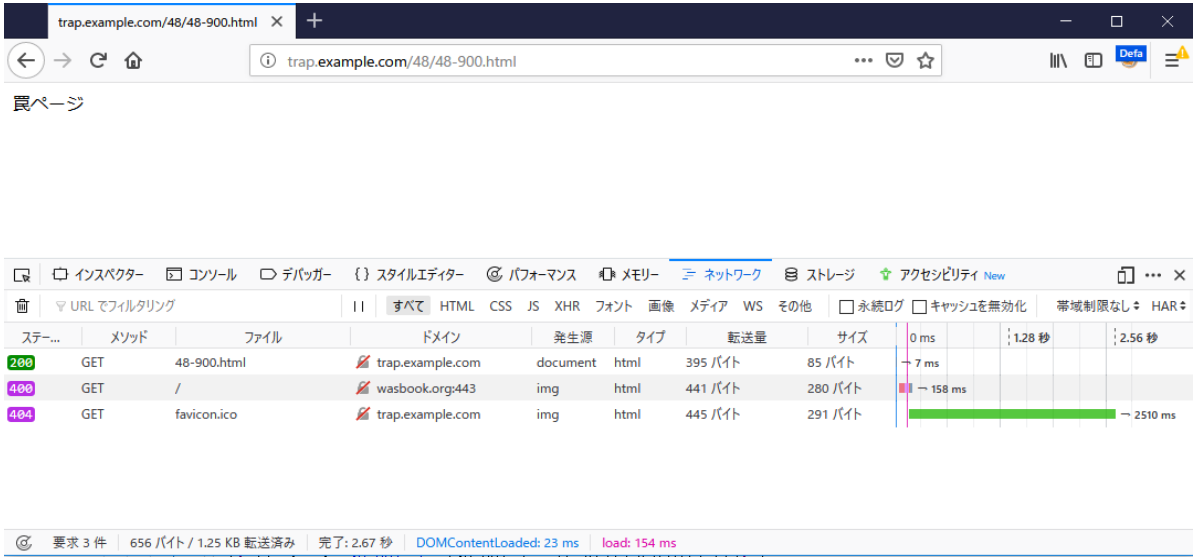


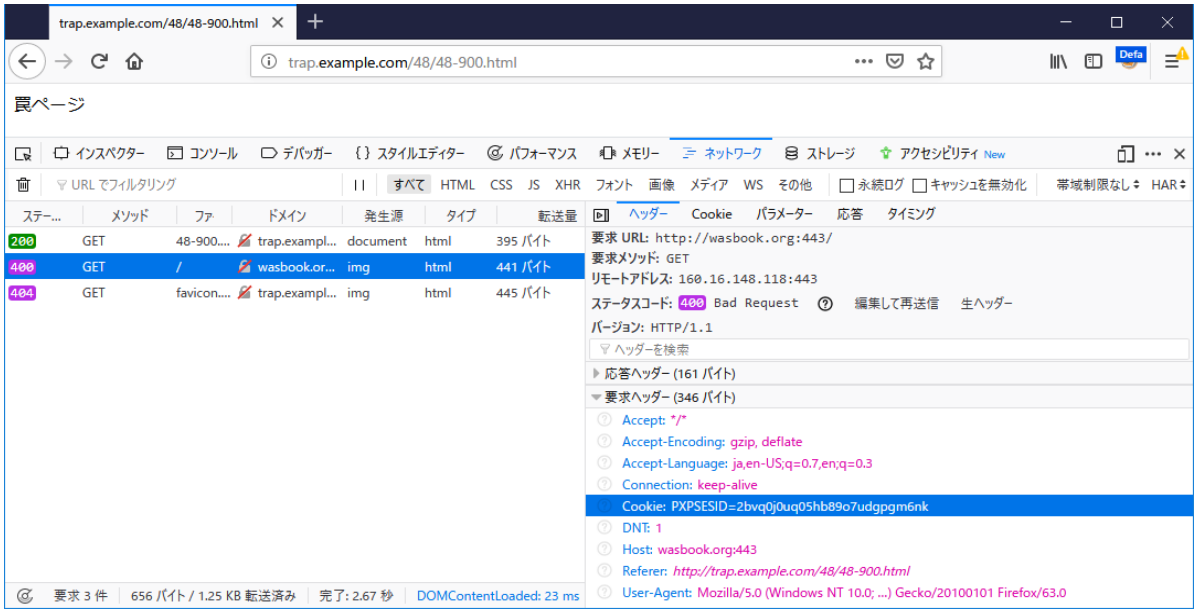
【ブラウザ→ 偽サーバ ： リクエスト trap.example.com/48/48-900.html → レスポンス 】



【ブラウザ】

「CTRL」+「SHIFT」+「E」 → 「ネットワーク」 → 「wasbook.org:443」を選択して、クッキーPXPSESIDを確認する

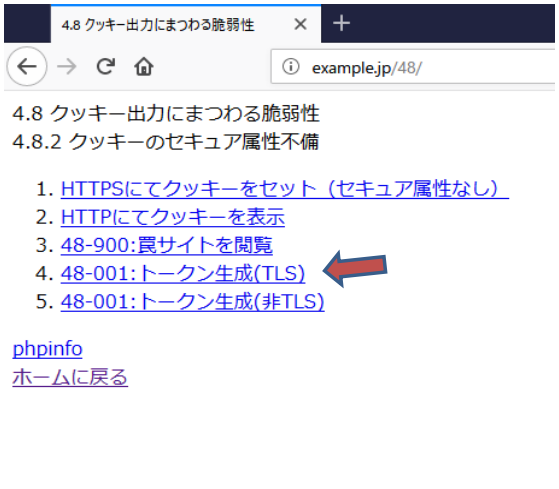




「wasbook.org:443」はHTTPS通信を待ち受けているので、
HTTPリクエストに対して、エラー400になっているが、
HTTPリクエスト自体は平文で流れ、ネットワーク上で漏洩する
危険性があることがわかる

48-001:トークン生成(TLS)

【ブラウザ】



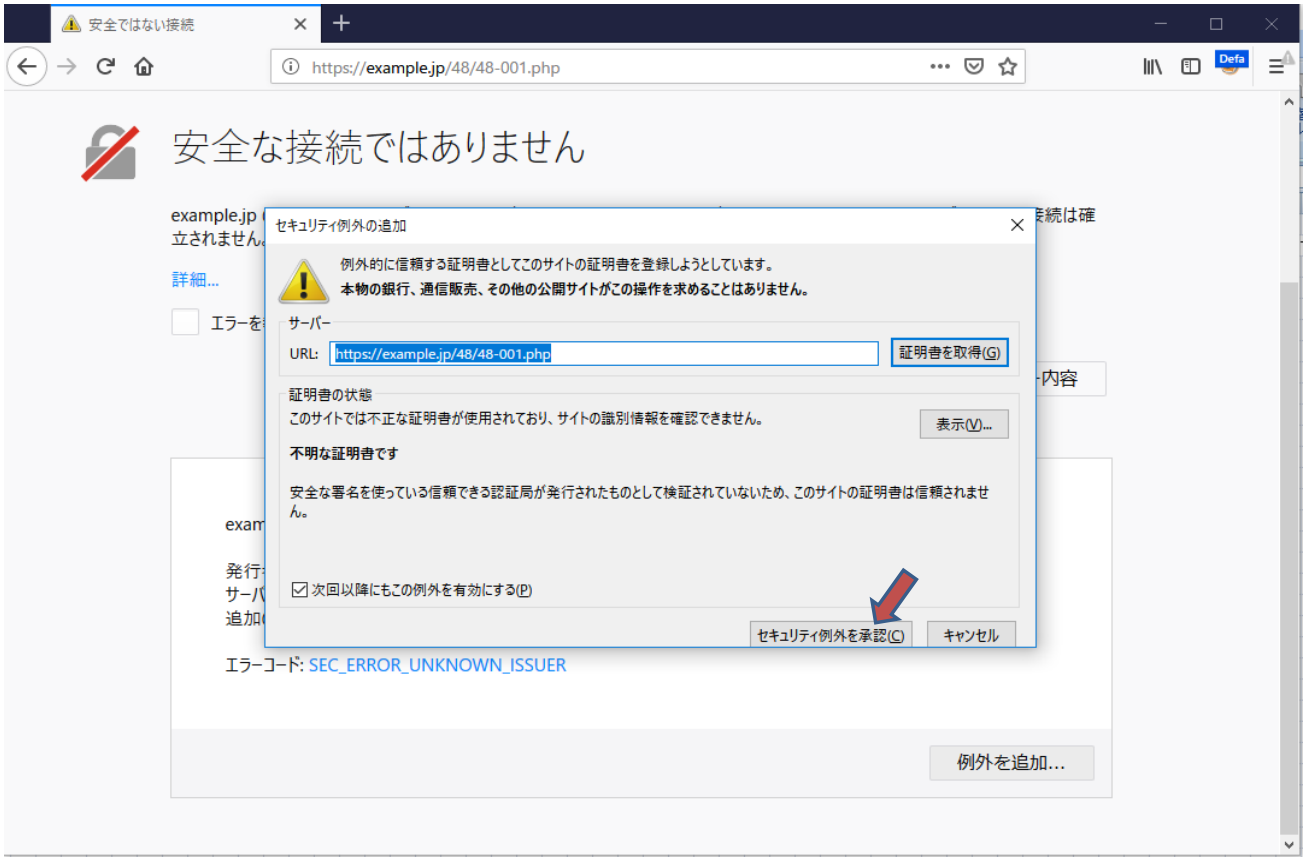
【サーバ： 48/48-001.php 】

【サーバ： 48/48-002.php 】

【ブラウザ】

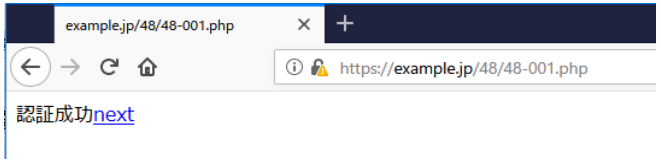
「エラー内容」をクリックすると、画面下部に証明書が信頼できない旨、表示される

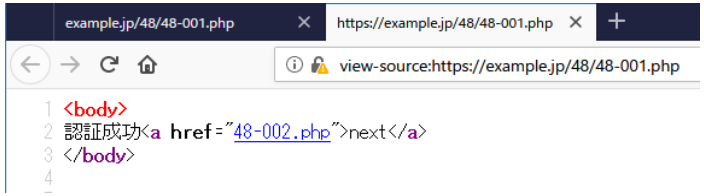
「例外を追加」をクリック

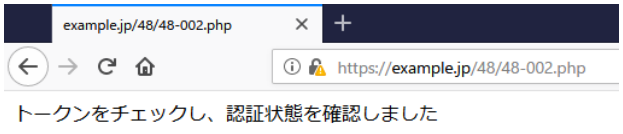


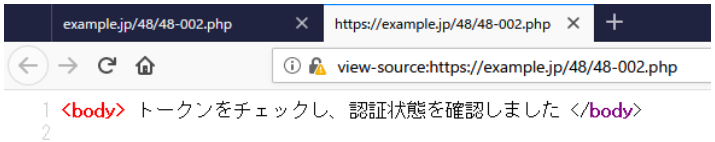
テスト環境として、自己署名証明書が設定されているので、
テストとして許容することにする。

「セキュリティ例外を承認」をクリック

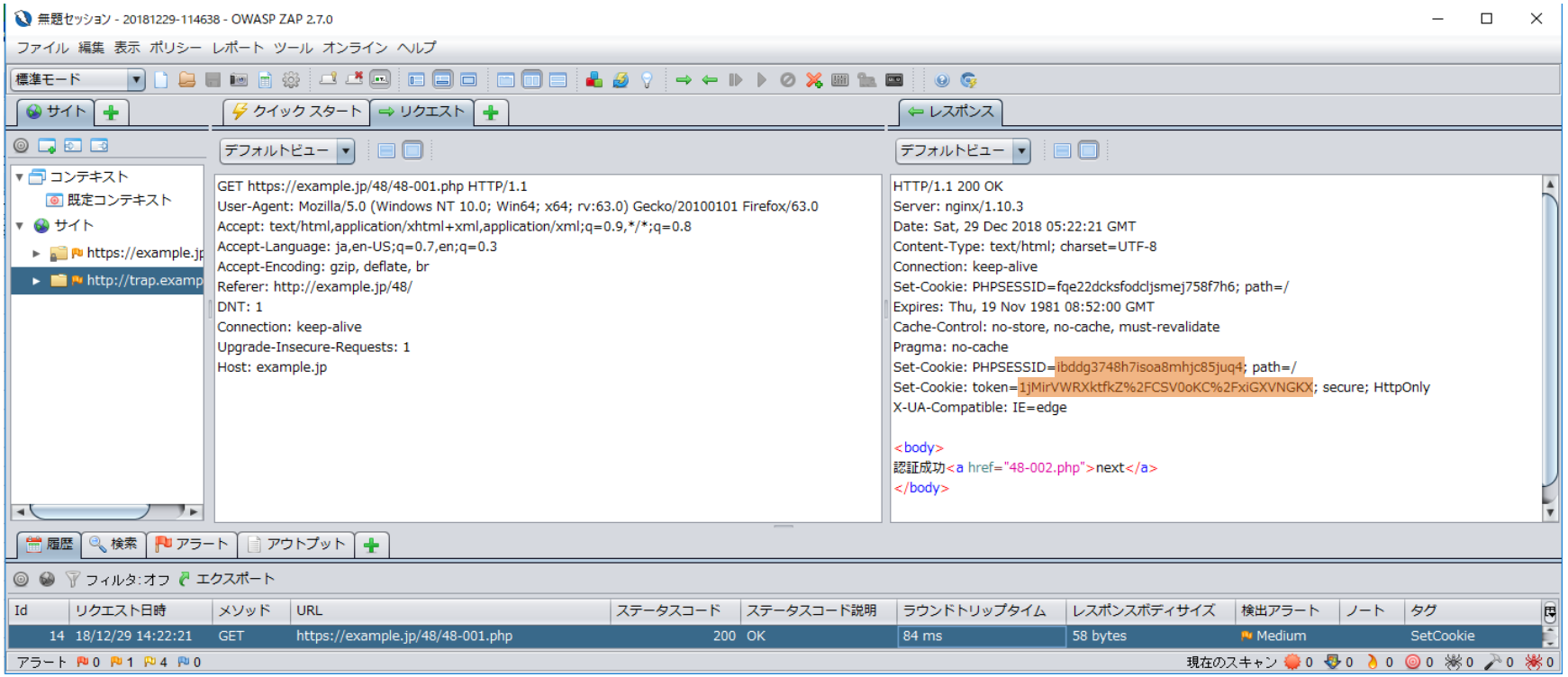




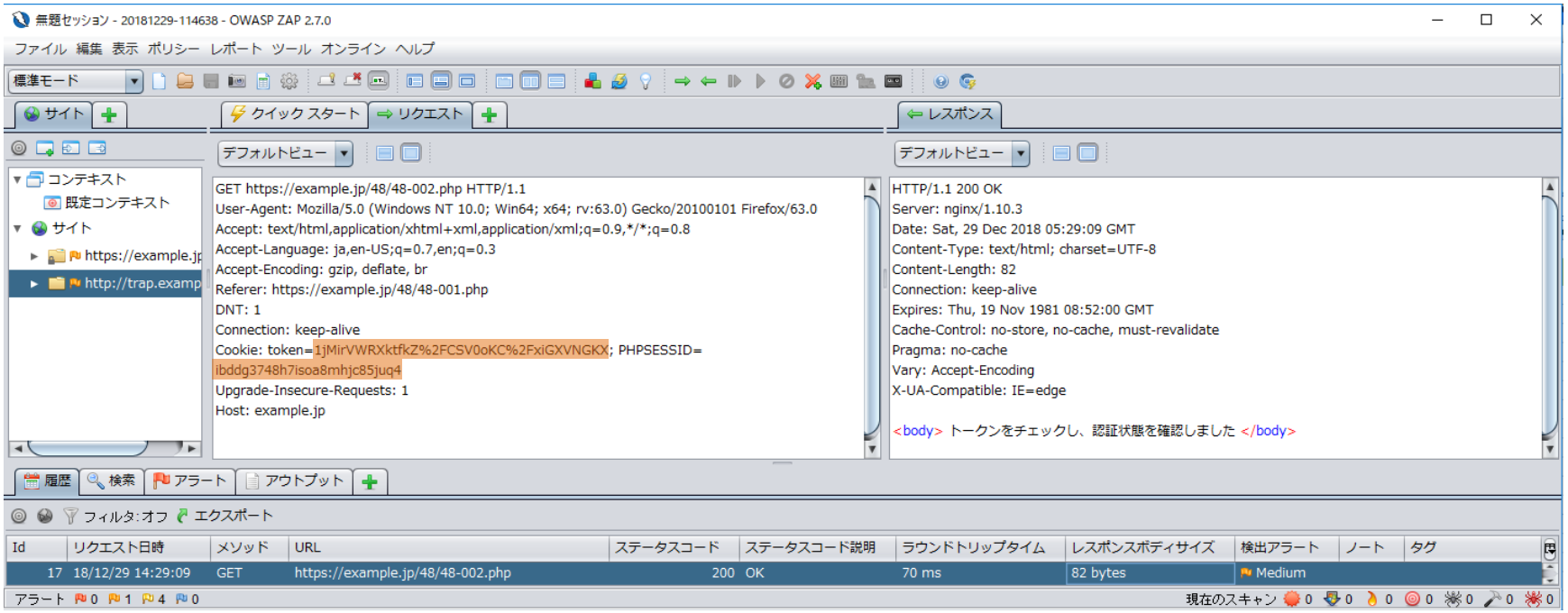




【ブラウザ→ サーバ ： リクエスト 48/48-001.php → レスポンス 】

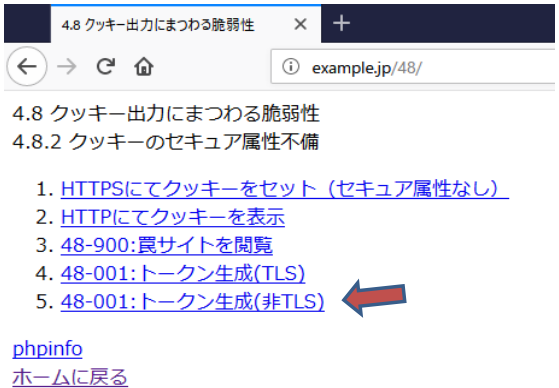


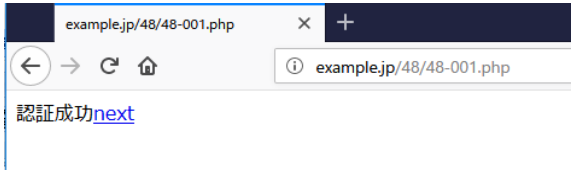
【ブラウザ→ サーバ ： リクエスト 48/48-002.php → レスポンス 】

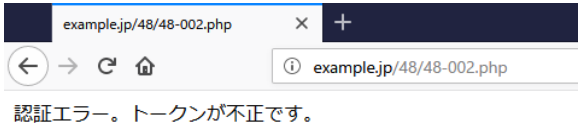


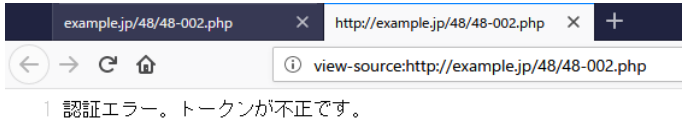
48-001:トークン生成(非TLS)

【ブラウザ】

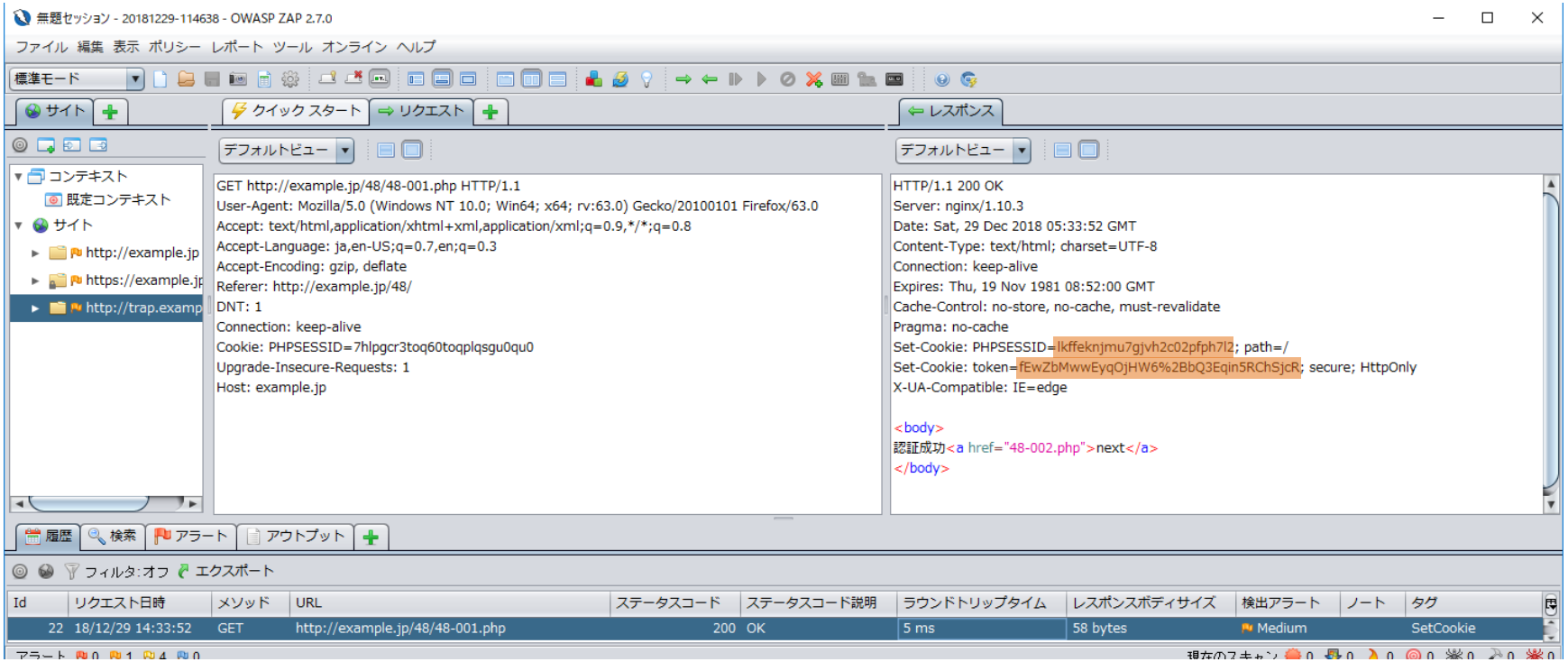








【ブラウザ→ サーバ ： リクエスト 48/48-001.php → レスポンス 】



【ブラウザ→ サーバ ： リクエスト 48/48-002.php → レスポンス 】

48-001.php で発行したトークンがセキュア属性付きクッキーだったので、HTTPS ではない 48-002.php でトークンが受け取れなかった。