

Audit de securitate pentru un formular de test

Fișă publică de proiect · exerciții și criterii de acceptare

Modelezi riscurile, verifici input-ul, output-ul și sesiunile, apoi testezi accesul. Toate experimentele se fac pe sisteme proprii sau autorizate.

Livrabile

- Model de amenințări
- Checklist SQL
- Checklist XSS
- CSRF
- Sesiuni
- Headere

Criterii de acceptare

- Nu se concatenează input în SQL
- Textul este escape-uit
- Operațiunile private verifică accesul

Model de amenințări pentru un formular și fișiere private

Începe cu activele și granițele de încredere: datele utilizatorului, sesiunea, fișierele și serviciile externe. Pentru fiecare, întreabă cine poate trimite intrări și ce efect poate produce. Nu transforma securitatea într-o listă de headere bifate. Un fișier privat livrat dintr-un director public poate ocoli complet policy-ul controllerului. Prioritizează amenințările după impact și probabilitate în contextul aplicației, fără a inventa procente.

Exemplu de lucru

Activ;Amenințare;Control;Test
Cerere privată;acces alt user;policy;user B refuzat
Formular;date invalide;validare server;payload direct
PDF;URL public;storage privat;cale directă 404
Sesiune;furt sau fixare;configurație și regenerare;flux login/logout
Integrare;efect duplicat;idempotență;retry după succes

Interpretarea rezultatului

Un control trebuie legat de un test observabil. Un scan automat poate ajuta, dar nu demonstrează toate regulile de business. Revizuieste modelul când adaugi exporturi, roluri sau acțiuni AI, fiindcă acestea pot crea granițe noi.

Exercițiu de consolidare

Adaugă un export CSV și identifică două amenințări care nu erau acoperite doar de formular.

1. Modelul de amenințări

Adaugă un export CSV și identifică două amenințări care nu erau acoperite doar de formular.

Verificare orientativă

Un control trebuie legat de un test observabil. Un scan automat poate ajuta, dar nu demonstrează toate regulile de business. Revizuieste modelul când adaugi exporturi, roluri sau acțiuni AI, fiindcă acestea pot crea granițe noi.

De evitat

O listă generică de atacuri nu descrie riscul aplicației tale.

SQL injection prevenită prin separarea valorilor

Parametrizarea este controlul de bază pentru valorile SQL. Validarea poate limita domeniul, dar nu înlocuiește parametrul. Pentru identificatori precum coloana de sortare, folosește o allowlist; aceștia nu sunt valori obișnuite de parametru. Exemplele de securitate se rulează numai într-un laborator propriu, fără a testa site-uri străine. Escaparea HTML nu protejează interogările SQL.

Exemplu de lucru

```
Laravel:  
$email = $request->validate(['email' => ['required', 'email']])['email'];  
$user = User::where('email', $email)->first();  
Sortare: mapare newest-created_at desc, oldest-created_at asc.  
De evitat: concatenarea unei valori în DB::select cu SQL construit manual.  
Test: intrare cu ghilimele rămâne dată, fără schimbarea structurii.
```

Interpretarea rezultatului

Query builderul parametrizat nu autorizează accesul la toate rândurile. Aplică și scope-ul utilizatorului unde este necesar. Pentru SQL raw, verifică bindingurile concrete și nu presupune că orice metodă cu „raw” este sigură. Nu publica payloaduri împotriva unui serviciu fără autorizare.

Exercițiu de consolidare

Construiește un test local pentru o valoare cu apostrof și confirmă că nu provoacă eroare sau extinderea rezultatului.

2. Injection

Construiește un test local pentru o valoare cu apostrof și confirmă că nu provoacă eroare sau extinderea rezultatului.

Verificare orientativă

Query builderul parametrizat nu autorizează accesul la toate rândurile. Aplică și scope-ul utilizatorului unde este necesar. Pentru SQL raw, verifică bindingurile concrete și nu presupune că orice metodă cu „raw” este sigură. Nu publica payloaduri împotriva unui serviciu fără autorizare.

De evitat

Nu testa injection pe site-uri fără autorizare.

XSS controlat la ieșire, după context

Textul utilizatorului trebuie afișat fără interpretare ca markup. În Blade, `{{ }}` escapează pentru HTML; în JavaScript, `textContent` este potrivit pentru text simplu. Un URL, un atribut sau un script inline are alte reguli, deci nu folosi o singură funcție drept protecție universală. Dacă produsul acceptă HTML bogat, definește sanitizarea și lista de elemente permise. CSP este un strat suplimentar, nu scuză pentru inserarea necontrolată.

Exemplu de lucru

Blade sigur pentru text: `<p>{{ $message }}</p>`
JS sigur pentru text: `element.textContent = message;`
Intrare de laborator: `<img src=x onerror=alert(1)>`
Așteptat: text vizibil, fără execuție.
De revizuit: `{!! $message !!}` și `innerHTML` cu date neîncrezătoare.

Interpretarea rezultatului

Testul arată o clasă de problemă, nu dovedește securitatea tuturor contextelor. Verifică și atributele, linkurile și conținutul importat. Nu scoate escaparea doar pentru a „repara” afișarea unor taguri dintr-o sursă necontrolată.

Exercițiu de consolidare

Adaugă un câmp URL și explică de ce validarea schemei și escaparea atributului au scopuri diferite.

3. XSS

Adaugă un câmp URL și explică de ce validarea schemei și escaparea atributului au scopuri diferite.

Verificare orientativă

Testul arată o clasă de problemă, nu dovedește securitatea tuturor contextelor. Verifică și atributele, linkurile și conținutul importat. Nu scoate escaparea doar pentru a „repara” afișarea unor taguri dintr-o sursă necontrolată.

De evitat

Sanitizarea și escaping-ul nu sunt interschimbabile în orice context.

CSRF și metode HTTP pentru schimbări de stare

Un formular care modifică date și folosește autentificare pe cookie trebuie să aibă protecția CSRF potrivită aplicației. Nu muta o operație de ștergere pe GET ca să eviți tokenul. În Laravel web, folosește mecanismul frameworkului și testează cererea fără token în contextul real. CSRF nu înlocuiește autorizarea: un utilizator autentic poate tot încerca să modifice resursa altuia. API-urile cu alte mecanisme de autentificare au un model diferit.

Exemplu de lucru

Blade:
`<form method="POST" action="/cereri/42/status">`
 `@csrf`
 `<input type="hidden" name="status" value="cancelled">`
 `<button type="submit">Anulează cererea</button>`
`</form>`
Server: auth → CSRF în fluxul web → policy → validare → regulă de stare.

Interpretarea rezultatului

Tokenul nu trebuie publicat în loguri. Testele frameworkului pot dezactiva anumite middleware-uri, deci verificarea CSRF trebuie proiectată în contextul potrivit, fără concluzii false dintr-un test simplificat. Linkul GET poate afișa confirmarea, dar nu trebuie să execute anularea.

Exercițiu de consolidare

Describe cum testezi separat lipsa tokenului și lipsa dreptului asupra resursei.

4. CSRF

Describe cum testezi separat lipsa tokenului și lipsa dreptului asupra resursei.

Verificare orientativă

Tokenul nu trebuie publicat în loguri. Testele frameworkului pot dezactiva anumite middleware-uri, deci verificarea CSRF trebuie proiectată în contextul potrivit, fără concluzii false dintr-un test simplificat. Linkul GET poate afișa confirmarea, dar nu trebuie să execute anularea.

De evitat

CORS nu înlocuiește protecția CSRF.

Sesiune cu cookie și invalidare consecvente

Sesiunea trebuie regenerată după autentificare și invalidată conform politicii la logout. Cookie-ul are nevoie de attribute potrivite transportului și fluxului aplicației. HttpOnly reduce accesul JavaScript la cookie, dar nu elimină toate efectele XSS. SameSite poate reduce anumite cereri cross-site, dar nu înlocuiește toate controalele CSRF. Nu păstra parole sau secrete inutile în sesiune. Verifică și schimbarea parolei și revocarea accesului.

Exemplu de lucru

Matrice: login valid → id sesiune nou; logout → acces privat refuzat; reset parolă → politica sesiunilor respectată; cookie HTTPS → Secure în producție; JavaScript → fără citirea cookie-ului HttpOnly.

Date de sesiune: identificator user și stare minimă, fără parola în clar.

Interpretarea rezultatului

Un cookie setat corect nu repară o policy lipsă. Verifică timeouturile și comportamentul pe dispozitive multiple conform produsului. Nu folosi aceeași cheie de aplicație în proiecte independente doar pentru comoditate. Pentru incidente, planul de invalidare trebuie să fie posibil și documentat.

Exercițiu de consolidare

Definește ce se întâmplă cu sesiunile existente după schimbarea parolei și scrie un test pentru regula aleasă.

5. Autentificare și sesiuni

Definește ce se întâmplă cu sesiunile existente după schimbarea parolei și scrie un test pentru regula aleasă.

Verificare orientativă

Un cookie setat corect nu repară o policy lipsă. Verifică timeouturile și comportamentul pe dispozitive multiple conform produsului. Nu folosi aceeași cheie de aplicație în proiecte independente doar pentru comoditate. Pentru incidente, planul de invalidare trebuie să fie posibil și documentat.

De evitat

Ascunderea linkului de logout nu revocă o sesiune.

Headere și configurare verificate pe răspunsul efectiv

Headerele se verifică pe răspunsul livrat public, deoarece proxy-ul și aplicația pot adăuga sau suprascrie valori. CSP trebuie introdusă după inventarierea resurselor, eventual printr-o etapă de raportare. HSTS are consecințe asupra accesului HTTPS și subdomeniilor; nu activa includeSubDomains fără a verifica întregul domeniu. Dezactivează debugul public și protejează fișierele de configurație. Un scor de scanner nu este echivalent cu un audit complet.

Exemplu de lucru

Verificări: Content-Type corect; X-Content-Type-Options; politică de framing; CSP adaptată; cookie-uri; redirect HTTPS; lipsa .env și backupurilor publice.

Instrument: inspectează headerele în Network sau cu un client HTTP.

Test funcțional: pagina și integrările permise continuă să funcționeze după politică.

Interpretarea rezultatului

O CSP prea largă poate oferi puțină protecție, iar una prea strictă poate rupe aplicația. Documentează excepțiile și motivul lor. Pentru fișiere private, verifică și Cache-Control, nu doar headerele de securitate generale.

Exercițiu de consolidare

Identifică o resursă externă folosită de pagina de laborator și descrie cum o tratezi în politică fără wildcard universal.

6. Headere, HTTPS, configurări

Identifică o resursă externă folosită de pagina de laborator și descrie cum o tratezi în politică fără wildcard universal.

Verificare orientativă

O CSP prea largă poate oferi puțină protecție, iar una prea strictă poate rupe aplicația. Documentează excepțiile și motivul lor. Pentru fișiere private, verifică și Cache-Control, nu doar headerele de securitate generale.

De evitat

O politică copiată poate bloca plăți, fonturi sau scripturi necesare.

Atelier final: Formular cu trei suprafețe de risc

Timp de lucru estimat: 40 de minute. Folosește datele de laborator de mai jos și salvează rezultatul separat de orice proiect real.

Date de pornire

Laborator propriu: căutare în SQL, afișare de comentarii și actualizare de profil. Folosește doar date fictive și conturi create pentru test.

1. Pregătire — 5 minute

Extrage intrările și condițiile din enunț. Notează ce rezultat trebuie produs și ce informații nu sunt confirmate. Pregătește fișierele sau tabelul în care vei păstra dovada lucrului.

2. Construire — 15 minute

Urmărește traseul input-bază-HTML. Verifică parametrizarea SQL, escapingul în contextul HTML și protecția cererii de modificare. Adaugă un test de acces cu alt cont.

3. Variație și verificare — 15 minute

Alege o singură eroare intenționată, demonstrează-o local și remediaz-o. În raport include efectul, controlul și testul de regresie.

4. Autoevaluare — 5 minute

Compară rezultatul cu reperele de mai jos. Înregistrează cel puțin o eroare identificată sau un caz-limită verificat, plus corectura ori explicația comportamentului observat.

Repere de rezolvare

Inputul nu devine structură SQL, textul special nu este executat, iar profilul altui cont nu este modificat. Validarea singură nu înlocuiește autorizarea sau encodingul ieșirii.

Ce predai

- Model de amenințări
- Checklist SQL
- Checklist XSS
- CSRF
- Sesiuni
- Headere

Criterii de acceptare

- Nu se concatenează input în SQL
- Textul este escape-uit
- Operațiile private verifică accesul

Păstrează o versiune inițială și una revizuită, cu explicația diferențelor. Dacă un criteriu nu este îndeplinit, revino la lecția relevantă și repetă verificarea. Durata este orientativă; instalarea mediului și aprofundarea pot necesita timp suplimentar.

Registrul de lucru

Pentru fiecare etapă notează intrarea, rezultatul, dovada verificării, eroarea găsită și corectura. Nu introduce date personale sau secrete în fișa partajată.

Documentație și surse

- OWASP Top 10 — OWASP
- Learn web development — MDN

<https://owasp.org/projects/top-ten>

https://developer.mozilla.org/en-US/docs/Learn_web_development