

Server web de laborator cu restaurare verificată

Fișă publică de proiect · exerciții și criterii de acceptare

Pregătești un server Ubuntu pentru o aplicație demonstrativă. Parcurgi SSH, firewall, web, bază, TLS și backup fără a modifica un server activ pentru exerciții.

Livrabile

- Utilizator administrativ
- Config web
- Bază cu drepturi minime
- TLS
- Restaurare de test

Criterii de acceptare

- Accesul SSH este păstrat
- Document root este public
- Backup-ul poate fi restaurat

Primul acces fără a pierde sesiunea de recuperare

La configurarea unui server nou, păstrează o sesiune funcțională până când noul utilizator și cheia lui sunt testate. Creează contul cu drepturi potrivite și verifică sudo numai dacă este necesar. Nu dezactivează accesul existent înainte de a confirma alternativa și consola de recuperare. Comenzile care modifică utilizatori trebuie executate numai pe serverul tău de laborator și cu nume alese explicit.

Exemplu de lucru

Plan Ubuntu: verifică hostname și whoami; creează utilizatorul de administrare; configurează cheia publică; deschide a doua sesiune; testează accesul și drepturile; confirmă mecanismul de recuperare; abia apoi restrânge accesul vechi.

Dovadă: login separat reușit, utilizator corect, sudo conform rolului.

Interpretarea rezultatului

O sesiune deja deschisă poate continua să funcționeze chiar dacă loginul nou este stricat. De aceea testul trebuie făcut printr-o conexiune separată. Nu confunda dreptul de administrare cu utilizatorul procesului aplicației; acesta ar trebui să aibă un rol mai restrâns.

Exercițiu de consolidare

Scrie un plan de revenire dacă noul utilizator nu poate autentifica după schimbarea configurației SSH.

1. Primul login și utilizatorii

Scrie un plan de revenire dacă noul utilizator nu poate autentifica după schimbarea configurației SSH.

Verificare orientativă

O sesiune deja deschisă poate continua să funcționeze chiar dacă loginul nou este stricat. De aceea testul trebuie făcut printr-o conexiune separată. Nu confunda dreptul de administrare cu utilizatorul procesului aplicației; acesta ar trebui să aibă un rol mai restrâns.

De evitat

Nu elimina singura cale de administrare activă.

Hardening SSH cu validare înainte de reload

O configurație mai strictă este utilă numai dacă păstrează accesul legitim și recuperarea. Modifică fișierul sau fragmentul potrivit distribuției, verifică sintaxa și efectele configurației incluse. Nu presupune că o linie din fișierul principal câștigă față de toate fragmentele. Păstrează sesiunea curentă și testează o conexiune nouă după reload. Evită schimbarea simultană a portului, firewallului și metodelor de autentificare fără pași separați.

Exemplu de lucru

Comenzi de observare/validare Ubuntu:

```
sudo sshd -t
```

```
sudo sshd -T
```

```
systemctl status ssh --no-pager
```

Plan: copie configurație → editare minimă → sshd -t → reload conform unității → conexiune nouă.

Reguli de evaluat: autentificare pe cheie; acces root; utilizatori permiși; recuperare.

Interpretarea rezultatului

sshd -t verifică sintaxa, nu dovedește că firewallul permite conexiunea. sshd -T arată configurația efectivă în contextul disponibil; opțiunile Match pot necesita verificare specifică. Nu închide sesiunea veche până când testul separat nu reușește.

Exercițiu de consolidare

Identifică un caz în care sintaxa este validă, dar noul login este refuzat, și descrie verificarea necesară.

2. SSH pe chei și hardening

Identifică un caz în care sintaxa este validă, dar noul login este refuzat, și descrie verificarea necesară.

Verificare orientativă

sshd -t verifică sintaxa, nu dovedește că firewallul permite conexiunea. sshd -T arată configurația efectivă în contextul disponibil; opțiunile Match pot necesita verificare specifică. Nu închide sesiunea veche până când testul separat nu reușește.

De evitat

Dezactivarea parolei înainte de testarea cheii poate bloca accesul.

Firewall configurat înainte de activare

Înainte de activarea unui firewall, identifică portul SSH real și serviciile care trebuie expuse. Permite accesul de administrare și păstrează consola de recuperare. Nu expune baza de date public doar fiindcă aplicația o folosește local. Fail2ban poate limita unele încercări repetitive, dar nu înlocuiește cheile, actualizările sau politica de acces. Comenzile de modificare se aplică numai într-un server de laborator autorizat.

Exemplu de lucru

Observare: `sudo ufw status verbose; ss -lnt`.
Plan: port SSH confirmat → regulă de acces potrivită → HTTP/HTTPS dacă sunt necesare → activare controlată → test din a doua sesiune.
Fail2ban: verifică jailurile active și logul; nu presupune că instalarea activează toate protecțiile dorite.

Interpretarea rezultatului

O regulă prea largă poate anula intenția unei politici restrictive. Verifică și IPv6 dacă este activ. Pentru servicii interne, ascultarea pe 127.0.0.1 sau pe o rețea privată este o componentă distinctă de firewall. Documentează fiecare port și scopul lui.

Exercițiu de consolidare

Construiește un tabel port-serviciu-public/privat și justifică fiecare expunere pentru o aplicație Laravel.

3. Firewall și fail2ban

Construiește un tabel port-serviciu-public/privat și justifică fiecare expunere pentru o aplicație Laravel.

Verificare orientativă

O regulă prea largă poate anula intenția unei politici restrictive. Verifică și IPv6 dacă este activ. Pentru servicii interne, ascultarea pe 127.0.0.1 sau pe o rețea privată este o componentă distinctă de firewall. Documentează fiecare port și scopul lui.

De evitat

Un firewall configurat doar pentru o familie IP lasă reguli incomplete.

Nginx trimite PHP către PHP-FPM, nu către browser

Nginx servește resursele și transmite execuția PHP către FPM. Document root pentru Laravel trebuie să fie public, astfel încât .env și codul aplicației să nu devină fișiere web. Calea socketului FPM depinde de versiunea instalată și trebuie verificată. Un 502 poate indica un upstream indisponibil, iar un 404 poate proveni din rutare sau cale; nu le trata identic. Verifică sintaxa înainte de reload.

Exemplu de lucru

Fragment orientativ Laravel:

```
root /cale/proiect/public;
index index.php;
location / { try_files $uri $uri/ /index.php?$query_string; }
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    fastcgi_pass unix:/run/php/php8.3-fpm.sock;
}
```

Validare: `sudo nginx -t`
Socketul este exemplu și trebuie confirmat pe server.

Interpretarea rezultatului

Fragmentul nu este un virtual host complet și nu înlocuiește configurația TLS și restricțiile aplicației. Testează că `/.env` și directoarele private nu sunt accesibile. O pagină PHP simplă reușită nu dovedește că toate rutele Laravel funcționează.

Exercițiu de consolidare

Simulează în laborator o cale de socket greșită și identifică mesajul relevant din log fără a publica date sensibile.

4. Nginx și PHP-FPM

Simulează în laborator o cale de socket greșită și identifică mesajul relevant din log fără a publica date sensibile.

Verificare orientativă

Fragmentul nu este un virtual host complet și nu înlocuiește configurația TLS și restricțiile aplicației. Testează că `/.env` și directoarele private nu sunt accesibile. O pagină PHP simplă reușită nu dovedește că toate rutele Laravel funcționează.

De evitat

Expunerea rădăcinii proiectului poate divulga fișiere interne.

Utilizator MySQL cu drepturi pentru o singură aplicație

Aplicația nu trebuie să se conecteze ca root DB. Creează o bază și un utilizator cu drepturi limitate la schema necesară, iar accesul de administrare rămâne separat. Nu pune parola în istoricul comenzilor sau într-un repository. Pentru migrații, unele drepturi de schemă pot fi necesare; poți separa contul de deploy de cel de rulare după arhitectură. Backupul trebuie testat prin restaurare, nu doar prin existența fișierului.

Exemplu de lucru

Plan: schema atelier_demo; cont aplicație local; drepturi necesare; secret în configurație privată; test conexiune; migrații controlate; backup și restaurare într-o schemă separată.
Verificări: `SELECT DATABASE(); SELECT CURRENT_USER(); SHOW GRANTS;`
Nu expune portul DB public dacă aplicația și DB sunt pe aceeași gazdă.

Interpretarea rezultatului

`CURRENT_USER` arată contul folosit pentru drepturi. O conexiune reușită nu dovedește că accesul este minim. Verifică operațiile permise și cele refuzate într-un mediu de test. Nu publica `SHOW GRANTS` dacă include informații sensibile ale infrastructurii.

Exercițiu de consolidare

Definește drepturile necesare unui proces care doar citește rapoarte și compară-le cu cele ale procesului de migrare.

5. MySQL

Definește drepturile necesare unui proces care doar citește rapoarte și compară-le cu cele ale procesului de migrare.

Verificare orientativă

CURRENT_USER arată contul folosit pentru drepturi. O conexiune reușită nu dovedește că accesul este minim. Verifică operațiile permise și cele refuzate într-un mediu de test. Nu publica SHOW GRANTS dacă include informații sensibile ale infrastructurii.

De evitat

Drepturile globale cresc impactul unei breșe.

TLS cu reînnoire verificată

Certificatul trebuie să corespundă domeniului și să fie livrat de virtual hostul corect. Înaintea emiterii, verifică DNS, porturile și metoda de validare. Reînnoirea automată trebuie testată conform clientului folosit, nu presupusă după prima emiterie. Nu amesteca emiterea certificatului cu schimbări mari de aplicație. Pentru un domeniu real, folosește documentația clientului și a furnizorului DNS, nu comenzi copiate pentru alt server.

Exemplu de lucru

Checklist: DNS corect; server accesibil pentru metoda de validare; certificat pentru numele exacte; redirect HTTP-HTTPS; resurse fără mixed content; programare reînnoire; test reînnoire; alertă expirare prin mecanism autorizat.
Verificare browser: domeniu, lanț, expirare, erori de consolă.

Interpretarea rezultatului

Un lacăt în browser nu dovedește securitatea aplicației, doar o parte a transportului. Dacă un asset este cerut prin HTTP, pagina poate avea mixed content. Verifică și subdomeniile folosite de resurse, fără a presupune că certificatul principal le acoperă.

Exercițiu de consolidare

Describe ce verifici dacă emiterea reușește, dar browserul vede certificatul altui domeniu.

6. SSL cu Let's Encrypt

Describe ce verifici dacă emiterea reușește, dar browserul vede certificatul altui domeniu.

Verificare orientativă

Un lacăt în browser nu dovedește securitatea aplicației, doar o parte a transportului. Dacă un asset este cerut prin HTTP, pagina poate avea mixed content. Verifică și subdomeniile folosite de resurse, fără a presupune că certificatul principal le acoperă.

De evitat

HTTPS valid azi nu dovedește că reînnoirea automată funcționează.

Backup cu dovadă de restaurare

O copie de siguranță trebuie să includă datele necesare, configurația recuperabilă și versiunea aplicației. Păstrează-o în afara directorului public, cu acces limitat și o politică de retenție. O arhivă creată cu succes poate fi incompletă sau incompatibilă. Restaurarea într-o destinație separată verifică mai mult decât lista fișierelor. Definește câtă pierdere de date și cât timp de recuperare sunt acceptabile pentru proiect.

Exemplu de lucru

Plan laborator: export DB → arhivă fișiere necesare → checksum → copiere sigură → restaurare în alt director/DB → verificare rânduri și fișiere → test aplicație.
Registru: dată, versiune, dimensiune, hash, destinație, rezultat restaurare.
Exclude din public: backupuri, .env, dumpuri și chei.

Interpretarea rezultatului

Hashul detectează modificarea fișierului, nu dovedește completitudinea logică a datelor. O restaurare care pornește aplicația, dar pierde atașamentele, nu este completă. Păstrează cheile de decriptare recuperabile printr-un mecanism separat și controlat.

Exercițiu de consolidare

Construiește un test de restaurare cu un document și o înregistrare cunoscute, fără a folosi date personale reale.

7. Backup automat

Construiește un test de restaurare cu un document și o înregistrare cunoscute, fără a folosi date personale reale.

Verificare orientativă

Hashul detectează modificarea fișierului, nu dovedește completitudinea logică a datelor. O restaurare care pornește aplicația, dar pierde atașamentele, nu este completă. Păstrează cheile de decriptare recuperabile printr-un mecanism separat și controlat.

De evitat

Un fișier backup care nu a fost restaurat nu dovedește recuperabilitatea.

Atelier final: Server de laborator și restaurare

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

Aplicație de test cu director public separat, un fișier de configurare privat și o bază de date fictivă. Backupul include fișierele și un export al bazei.

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

Documentează utilizatorul serviciului, document root, accesul SSH și procedura de backup. Restaurează într-o destinație de test și verifică o pagină care citește date.

3. Variație și verificare — 15 minute

Modifică o înregistrare în copia restaurată și verifică izolarea față de sursă. Notează timpul de recuperare și orice pas manual necesar.

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

Fișierul privat nu este servit prin HTTP. Restaurarea recuperează datele așteptate; simpla existență a arhivei nu este suficientă. Accesul administrativ rămâne disponibil după schimbări.

Ce predai

- Utilizator administrativ
- Config web
- Bază cu drepturi minime
- TLS
- Restaurare de test

Criterii de acceptare

- Accesul SSH este păstrat
- Document root este public
- Backup-ul poate fi restaurat

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

- Ubuntu Server documentation — Canonical
- nginx documentation — nginx

<https://documentation.ubuntu.com/server/>

<https://nginx.org/en/docs/>