FORMATION GRATUITE / NIVEAU INTERMÉDIAIRE

Hacking éthique Linux

Manuel professionnel et syllabus



Visuel pédagogique original. Aucun reportage ou environnement tiers.

16 chapitres / 90 h de travail indicatives / TP corrigés / 16 QCM / Examen de 40 questions

Unité de formation Mashaah LLC, Delaware. Identité fournie par Bouda. Support original en français, daté du 6 octobre 2026. Mise en page justifiée le 7 octobre 2026. Formation et diplôme interne gratuits.

Formation en ligne : leçons, vidéos, exercices et QCM sur votre compte Vise la Réussite. Ce PDF est un support complémentaire facultatif. Les ateliers se réalisent dans votre laboratoire local autorisé.

Sommaire cliquable

Syllabus et organisation 3
Construire un plan d’audit et des critères de couverture 4
Versionner, isoler et rendre un atelier reproductible 6
Modéliser les menaces et les frontières de confiance 9
Diagnostiquer les échanges HTTP et examiner l’exposition 11
Étudier la vie d’une session et les protections CSRF 13
Vérifier l’isolation des organisations et les propriétés d’API 17
Suivre les entrées, les requêtes SQL et les sorties 22
Mener une revue de code et automatiser les contre-tests 24
Inventorier les composants et vérifier un avis de sécurité 26
Corréler les événements et préserver les preuves minimales 28
Linux : permissions, ACL et contexte d’exécution 30
Linux : unités, sockets et politique de service 32
Linux : rétention, sources et corrélation 34
Linux : correctifs, confinement et restauration 36
Corriger, vérifier et préparer un changement réversible 38
Projet intermédiaire : audit argumenté d’Atelier Aurore 40
Dossier pratique et commandes 42
Lecture du code fourni 49
Corrigés des QCM de chapitre 50
Sources et provenance 53

Syllabus et organisation

Parcours gratuit : 16 chapitres, 90 h de travail indicatives, 16 QCM de cinq questions à réussir à 4/5 avant l'examen de 40 questions, réussi à 32/40. Diplôme de réussite interne VLR, non surveillé, sans reconnaissance universitaire revendiquée. Projet autoévalué. Le laboratoire et les captures ont été exécutés sur macOS ; les commandes natives Windows et Linux sont documentées sans exécution native revendiquée.

Volume de travail indicatif : lectures, TP, révisions et projet ; il est distinct de la durée des vidéos. Le tronc commun est partagé entre les trois OS.

Prérequis recommandés

• Avoir étudié les bases sur Linux ou posséder des connaissances équivalentes : réseau, terminal, permissions et rapport.

• Disposer de Python 3.10 ou ultérieur, d'un navigateur récent et d'un dossier de TP personnel.

• Travailler exclusivement sur le laboratoire local et les données fictives distribuées.

Compétences visées

• Établir une couverture et un modèle de menace

• Vérifier les sessions, le CSRF et les contrôles d'accès entre organisations

• Étudier les propriétés d'API et le SQL paramétré

• Relier revue de code, tests et faux positifs

• Distinguer inventaire, avis et condition d'exploitation

• Interpréter permissions, exposition et journaux sur Linux

• Produire une recette réversible et un rapport argumenté

Volume de travail et évaluations

Chapitres 1 à 15 : 1,5 h de lecture, 2,5 h de TP et 1 h de révision par chapitre, soit 75 h. Chapitre 16 : 12 h de projet et 3 h de préparation finale, soit 15 h. Total indicatif : 90 h. Les vidéos ont leur durée propre. Le tronc commun est partagé entre les trois parcours ; le changement d'OS ne représente pas trois travaux complets distincts.

Sur le site, cinq questions concluent chaque chapitre : réussite à quatre bonnes réponses. Les seize QCM validés ouvrent le questionnaire final de 40 questions, cinq domaines de huit questions, réussi à 32/40. Le diplôme interne VLR atteste ce parcours non surveillé. Le projet utilise une grille d'autoévaluation distincte. Aucun jury, ECTS, agrément universitaire ou correction humaine du rapport n'est revendiqué.

Articulation des niveaux

Les bases préparent ce parcours. Le niveau avancé est également disponible en ligne avec son support complémentaire. Aucune équivalence professionnelle ou universitaire n'est revendiquée.

Parcours débutant recommandé: https://viselareussite.com/formations/hacking-ethique-bases-linux/

CHAPITRE 01 / 5 h indicatives

Construire un plan d’audit et des critères de couverture

Objectifs : Définir une mission multi-contrôles ; Transformer un besoin en tests ; Annoncer les exclusions

Le niveau intermédiaire commence par un changement de méthode : vous ne vérifiez plus une commande isolée, mais un ensemble de contrôles reliés à un besoin. La cible reste une application fictive et locale. Atelier Aurore représente deux organisations, A et B, avec trois identités distribuées : Alice et Claire appartiennent à A ; Bob appartient à B. Alice et Bob sont lecteurs, Claire est gestionnaire. Ces rôles donnent un vocabulaire pour raisonner, sans constituer un système d’identité de production.

Un audit utile annonce sa couverture. Vous pouvez étudier les accès aux documents, les modifications de profil, la gestion des sessions et la construction SQL, sans prétendre avoir évalué toute la sécurité de Python, du navigateur et de l’OS. Distinguez « testé », « étudié sans exécution », « hors périmètre » et « non applicable ». Une fonctionnalité absente n’est pas nécessairement une faiblesse ; une fonctionnalité présente n’est pas nécessairement conforme.

De l’objectif au cas de test

Écrivez d’abord la règle métier : « une identité de A ne lit pas un document de B ». Définissez le contrôle serveur attendu, les identités disponibles et la preuve minimale. Le cas Alice vers 202 doit être refusé sur la route corrigée. Alice vers 101 doit rester possible. Bob vers 202 vérifie que la ressource n’est pas bloquée pour son organisation légitime. Le lecteur doit retrouver ce raisonnement dans le plan, le résultat et le rapport.

Une fiche de test précise un identifiant, une hypothèse, une méthode, une condition préalable, un résultat attendu et une condition d’arrêt. Ajoutez la version du logiciel distribué et le budget de requêtes. Les essais de charge, la recherche de systèmes tiers et l’usage de comptes personnels sont exclus. Le navigateur ne doit pas suivre une destination extérieure pour étendre un exercice. L’autorisation porte uniquement sur les programmes et données de ce dossier.

Couverture et limites

Construisez une matrice entre contrôles et scénarios, plutôt qu’une liste de logiciels. Une ligne sans scénario est un objectif encore non vérifié. Un scénario sans contrôle identifié risque de produire un résultat difficile à interpréter. Le projet final doit montrer pourquoi chaque essai a été choisi et ce qu’un résultat négatif laisse inconnu. Les cas supplémentaires sont préparés avant exécution, avec la même limite locale.

La qualité d’une mission ne se mesure pas au nombre de constats. Un faux positif écarté, un contrôle confirmé et une limite correctement annoncée ont une valeur. N’inventez pas de score de criticité ni de conséquence financière pour un laboratoire fictif. Le niveau avancé approfondira des contextes plus complexes ; ce niveau construit d’abord une démarche reproductible et argumentée.

TP guidé

Méthode pas à pas

Remplissez mandat.md. Listez les deux documents, trois identités, routes de lecture et routes de profil. Créez six hypothèses, dont deux sur la conservation d’un usage légitime. Associez chaque hypothèse à une preuve et à une condition d’arrêt. Faites relire votre plan par vous-même avant de lancer application.py.

Exercice à réaliser

Produisez une matrice contrôle/scénario/couverture et justifiez deux exclusions. Expliquez pourquoi la route de contre-exemple peut être testée ici mais pas un service similaire trouvé sur Internet.

Cas entièrement fictif

Le responsable fictif demande « valider la séparation entre A et B » et propose aussi de tester son fournisseur d'email. Ce fournisseur n'apparaît pas dans le mandat pédagogique.

Corrigé raisonné

La séparation locale est étudiée avec les identités distribuées. Le fournisseur reste exclu. Le rapport précise qu'aucune messagerie n'a été testée. Les cas autorisés et refusés sont annoncés avant la collecte ; un refus systématique n'est pas considéré comme une correction fonctionnelle.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Quel cas doit accompagner Alice/202 refusé ?

A. Alice/101 autorisé

B. Un scan du fournisseur

C. Toutes les routes arrêtées

2. Une route absente du plan est découverte. Que faire ?

A. Vérifier sa couverture avant l'essai

B. La supprimer du rapport

C. La tester sans limite

3. Une fonctionnalité n'a pas été testée. Quel état retenir ?

A. Vulnérable par défaut

B. Non testée dans la couverture annoncée

C. Sécurisée par défaut

4. Quel résultat justifie un constat ?

A. Une observation liée à une hypothèse

B. Un objectif sans résultat

C. Le nombre de logiciels installés

5. Un fournisseur d'email est hors mandat. Que faire ?

A. L'exclure et l'annoncer

B. Le tester depuis le Wi-Fi

C. Tester ses comptes fictivement

CHAPITRE 02 / 5 h indicatives

Versionner, isoler et rendre un atelier reproductible

Objectifs : Lire le code avant exécution ; Relever les versions ; Préparer un retour à l'état initial

Le dossier intermédiaire utilise Python et sa bibliothèque standard. application.py démarre un service HTTP sur 127.0.0.1:8877 ; atelier.html propose des actions limitées à ce service. verifier.py démarre son propre serveur temporaire sur un port local choisi par le système pour exécuter ses tests, puis le ferme. Aucun de ces programmes ne découvre de réseau distant. Ne modifiez pas leur adresse d'écoute pour rendre le laboratoire accessible au Wi-Fi.

Le service de démonstration conserve sessions, profils et journal en mémoire. Un arrêt supprime cet état ; le redémarrage recrée les données initiales depuis le code. Le programme vérifie un en-tête Host local explicite et limite les corps JSON, mais cela n'en fait pas un serveur de production. Les protections volontairement omises sur des routes nommées vulnerable sont des objets d'étude, pas des recommandations de déploiement.

Préparation raisonnée

Lisez les imports, l'adresse de liaison, les routes et les fichiers consultés. Distinguez ce qui est téléchargé de ce qui est exécuté. Une archive pédagogique doit être conservée avec son origine et son empreinte si elle est fournie. Relevez l'OS, l'architecture et les versions Python et SQLite observées. Votre commande de lancement peut être py -3 sous Windows ou python3 sous Linux et macOS, selon l'installation officielle.

Créez un dossier réservé au parcours et séparez les résultats des fichiers de référence. Le programme de tests écrit resultats-http.json dans le dossier courant : lancez-le dans votre copie du laboratoire. Ne le lancez pas depuis un dossier contenant un fichier personnel portant le même nom. La destruction de l'état en mémoire ne remplace pas une sauvegarde de vos notes, captures et rapports.

Réseau et machine virtuelle

La boucle locale permet de réaliser les ateliers applicatifs sans machine cible séparée. Les exercices propres à l'OS qui modifient une configuration doivent rester dans une VM ou dans une copie de fichiers vous appartenant. Une VM en NAT peut communiquer avec l'extérieur ; un instantané n'est pas une sauvegarde indépendante. Les futures architectures multi-machines devront expliciter les segments, sorties, services et modes de restauration.

En cas de conflit sur 8877, identifiez d'abord le propriétaire. Ne terminez pas un processus inconnu et ne choisissez pas un autre port au hasard sans adapter le mandat et vérifier tous les scripts. Le serveur de tests utilise un port temporaire autorisé précisément pour éviter de remplacer un service existant. La procédure d'arrêt et la disparition de l'écoute font partie de votre preuve d'isolation.

TP guidé

Télécharger le laboratoire complet

Décompressez l'archive, puis lisez GUIDE-TP.md. Les scripts, le guide, les commandes OS, les modèles de mandat et rapport et la grille du projet sont fournis. Les fichiers inventaire-fictif.json et configuration-os.json sont des scénarios étiquetés, pas des relevés réels. Le service écoute uniquement sur 127.0.0.1:8877.

Méthode pas à pas

Copiez le dossier, lisez application.py et verifier.py, relevez les versions et démarrez le serveur avec votre interpréteur. Ouvrez l’adresse exacte. Arrêtez avec Ctrl+C, constatez qu’une nouvelle demande ne répond plus, puis exécutez verifier.py. Archivez la sortie réelle et resultats-http.json.

Exercice à réaliser

Rédigez une fiche de reproduction utilisable par un autre apprenant. Expliquez la différence entre le port 8877 de l’interface et le port temporaire des tests, ainsi que les données perdues au redémarrage.

Cas entièrement fictif

Une personne présente une capture de la page ouverte avant l’arrêt comme preuve que le serveur répond encore. Une autre reproduit réellement une nouvelle requête après arrêt.

Corrigé raisonné

Seule la nouvelle requête renseigne l’état du service à cet instant. La page déjà chargée peut rester visible. Le dossier distingue versions de référence, état en mémoire et preuves sauvegardées. Aucun résultat attendu n’est recopié comme observation exécutée.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Pourquoi distinguer 8877 et le port temporaire ?

A. Ils servent à deux contextes de lancement

B. Ils identifient deux attaques

C. Ils exigent deux comptes payants

2. Qu’est-ce qui disparaît au redémarrage ?

A. Les sessions et profils en mémoire

B. Vos captures sauvegardées

C. La source Python

3. Une ancienne page reste visible après arrêt. Que conclure ?

A. Une nouvelle requête est nécessaire

B. Le pare-feu est désactivé

C. Le serveur tourne encore forcément

4. Quel fichier les tests écrivent-ils ?

A. Votre profil navigateur

B. Une base distante

C. resultats-http.json

5. Le port 8877 est occupé. Quel geste convient ?

A. Ouvrir un port du routeur

B. Arrêter tous les processus Python

C. Identifier son propriétaire

CHAPITRE 03 / 5 h indicatives

Modéliser les menaces et les frontières de confiance

Objectifs : Dessiner les flux ; Identifier les frontières ; Relier menace et contrôle

Un modèle de menace décrit ce que le système doit protéger, les personnes et programmes qui agissent, les chemins des données et les hypothèses de confiance. Il ne prédit pas chaque attaque et ne remplace pas un test. Pour Atelier Aurore, les actifs pédagogiques sont les documents 101 et 202, les profils, les sessions et les preuves. Le navigateur transmet des entrées ; le serveur décide des permissions ; les données d'organisation viennent de la référence serveur.

Dessinez les composants avant les menaces. Le client choisit une identité fictive dans le laboratoire, reçoit un cookie et effectue des requêtes. Le serveur associe le cookie à un état en mémoire, identifie l'organisation et contrôle la ressource. La fonction SQL utilise une base en mémoire distincte pour la démonstration. Les journaux conservent méthode, route, heure et statut, sans valeur de cookie ni jeton CSRF.

Une frontière est une décision à vérifier

Une donnée transmise par le client ne devient pas fiable parce qu'un formulaire semble correct. Le champ role illustre cette différence. La route de contre-exemple conserve les propriétés reçues dans un profil fictif ; la route correcte n'admet que displayName. Le rôle effectif demeure dans la référence USERS et n'est pas modifié par cette démonstration. Cette limite permet de montrer l'affectation indue sans prétendre à une élévation réelle de privilèges.

Une grille de menaces peut examiner usurpation, altération, contestation des actions, divulgation, indisponibilité et extension de droits. Ces catégories servent à poser des questions. Elles ne justifient pas l'exécution de méthodes non autorisées. Dans ce niveau, les essais de disponibilité restent des contrôles bornés de fonctionnement ; aucune saturation de service n'est nécessaire.

Vérifier les hypothèses du dessin

Pour chaque flèche, demandez quelle information traverse la frontière et qui peut la modifier. Pour chaque stockage, demandez quelles données sont conservées et comment elles sont supprimées. Pour chaque rôle, vérifiez l'existence d'un contrôle serveur plutôt qu'un bouton masqué. La trace d'une requête peut soutenir la reproductibilité, mais ne prouve pas l'identité physique de son auteur.

Le modèle évolue avec le constat. Si vous découvrez une route oubliée dans le code distribué, ajoutez-la à l'inventaire et vérifiez si le mandat la couvre avant tout essai. Une route absente de l'interface peut rester accessible au client. Le rapport doit rendre visibles les hypothèses révisées, pas masquer le premier dessin comme s'il avait été complet dès le départ.

TP guidé

Méthode pas à pas

Dessinez client, session serveur, table de référence USERS, documents et journal. Annotez cinq flux, leur source et les contrôles attendus. Préparez trois scénarios d'abus sur papier et choisissez les deux qui correspondent aux routes distribuées. Comparez le dessin aux fonctions context, authorize et profile.

Exercice à réaliser

Expliquez pourquoi la valeur role du profil de contre-exemple ne suffit pas à établir un changement du rôle effectif. Ajoutez un contrôle et un contre-test pour la frontière client/serveur.

Cas entièrement fictif

Alice transmet role=gestionnaire dans son profil de TP. Le résultat conserve ce champ, mais USERS indique toujours lecteur. Le responsable fictif interprète à tort cet affichage comme une prise de contrôle.

Corrigé raisonné

Le défaut confirmé concerne les propriétés de profil acceptées. Une élévation effective n'est pas démontrée. La correction limite les champs et garde le rôle sous contrôle serveur. Le modèle montre où la décision est réellement prise et le rapport conserve cette nuance.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Le profil accepte role=gestionnaire, mais USERS reste lecteur. Que prouve-t-on ?

A. Une élévation effective certaine

B. Une propriété indue conservée

C. Une compromission de Windows

2. Quelle donnée détermine l'organisation admise ?

A. Le nom du bouton

B. La référence serveur USERS

C. Une affirmation libre du client

3. À quoi sert une catégorie de menace ?

A. À préparer une question et un contrôle

B. À autoriser toute méthode

C. À garantir une vulnérabilité

4. Quelle flèche mérite une frontière de confiance ?

A. Une entrée client vers une décision serveur

B. Un titre vers sa police

C. Une décoration vers une couleur

5. Le journal ne contient pas les corps JSON. Quel effet ?

A. Il prouve toutes les propriétés envoyées

B. Il identifie physiquement l'auteur

C. La preuve doit combiner plusieurs sources

CHAPITRE 04 / 5 h indicatives

Diagnostiquer les échanges HTTP et examiner l’exposition

Objectifs : Relier transport et application ; Interpréter des en-têtes ; Borner une observation réseau

L’analyse intermédiaire relie plusieurs couches : liaison du serveur, établissement de la connexion, requête HTTP, contexte de session et décision applicative. Un refus 403 signifie que la requête a atteint une logique de traitement ; une erreur de connexion décrit une autre situation. Une même page de navigateur peut déclencher plusieurs requêtes et garder un contenu antérieur. Notez l’action précise et l’heure avant d’interpréter la capture.

Le laboratoire répond en JSON avec Content-Type et Cache-Control: no-store. Ce dernier exprime une instruction de cache pour ses réponses ; il ne constitue pas une permission et ne garantit pas qu’aucune copie n’a jamais été faite ailleurs. X-Content-Type-Options: nosniff réduit certaines interprétations de contenu par le navigateur. La page reçoit aussi une politique CSP pédagogique, qui admet des scripts intégrés pour l’interface locale ; cette configuration ne doit pas être copiée comme une politique universelle de production.

Observer seulement les flux utiles

L’onglet Réseau du navigateur suffit pour étudier méthode, route, statut et en-têtes de ces ateliers. Un proxy passif ou Wireshark peut être étudié en complément, sur la seule boucle locale et selon les droits disponibles. Une capture générale d’interface peut collecter des communications privées. Limitez la source et le filtre avant de commencer, et vérifiez le contenu conservé avant de partager une image.

Le cookie de session et le jeton CSRF sont des valeurs temporaires du laboratoire. Leur présence explique un mécanisme, mais leur contenu n’est pas nécessaire au rapport. Masquez la valeur dans une copie, sans la décrire comme un secret de production. Le serveur ne chiffre pas HTTP ; le choix est limité au service local fictif. TLS et la validation des certificats restent nécessaires dans une architecture publique adaptée, mais ne sont pas testés ici.

Distinguer liaison, filtre et route

Une écoute sur 127.0.0.1:8877 ne désigne pas une exposition sur toutes les interfaces. La liaison du programme et une règle de pare-feu sont deux observations différentes. Un filtre réseau ne corrige pas une route qui applique une mauvaise permission aux utilisateurs admis. Le serveur rejette aussi un Host inattendu : ce contrôle local réduit une ambiguïté de destination, sans remplacer une authentification.

Votre diagnostic suit les données disponibles. Commencez par le processus de TP, l’adresse et le port, puis l’action dans le navigateur. Si une route correcte et une route de contre-exemple diffèrent, relevez les deux résultats sans changer de réseau ni désactiver les protections du poste. Un essai ciblé vaut mieux qu’une modification générale dont les conséquences seraient difficiles à isoler.

TP guidé

Méthode pas à pas

Ouvrez Réseau, chargez health, créez la session Alice et comparez les deux routes vers 202. Relevez adresse, port, méthode, statut, type de contenu et politique de cache. Conservez uniquement les champs utiles, avec valeurs de jeton masquées. Comparez ensuite l’écoute avant et après votre arrêt du serveur.

Exercice à réaliser

Produisez une fiche distinguant erreur de connexion, 401 et 403. Expliquez pourquoi une CSP présente et un cookie HttpOnly ne prouvent pas la validité de la permission inter-organisation.

Cas entièrement fictif

Le responsable fictif voit no-store et conclut que les documents de B ne peuvent pas être divulgués à A. La route de contre-exemple renvoie pourtant 202 à Alice.

Corrigé raisonné

Le cache et l'autorisation répondent à des questions distinctes. La permission doit vérifier l'organisation de la ressource côté serveur. L'observation est limitée à ce laboratoire, pas à tous les caches ni aux flux de votre réseau personnel.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Un 403 diffère d'une connexion refusée parce que…

A. Il prouve la bonne configuration de tous les ports

B. La requête a atteint un traitement applicatif

C. Il prouve un chiffrement TLS

2. no-store garantit-il une permission inter-organisation ?

A. Non, ce sont deux contrôles distincts

B. Oui pour tous les documents

C. Oui si le port est local

3. Que masquer dans une capture partageable ?

A. La valeur du cookie temporaire

B. Le nom de la route utile

C. Le statut utile

4. La CSP locale admet des scripts intégrés. Que dire ?

A. C'est un choix pédagogique annoncé

B. Cela corrige SQL

C. C'est une politique universelle parfaite

5. Une écoute locale prouve…

A. L'absence de tout défaut de code

B. La validité de toutes les permissions

C. Une portée de liaison du service observé

CHAPITRE 05 / 5 h indicatives

Étudier la vie d’une session et les protections CSRF

Objectifs : Vérifier création et invalidation ; Distinguer cookie et jeton CSRF ; Tester une modification légitime

Le sélecteur d’identité du laboratoire ne demande pas de mot de passe. Il représente une identité fictive déjà admise pour concentrer l’exercice sur la vie d’une session. Le serveur crée un identifiant aléatoire et conserve en mémoire son utilisateur, un jeton CSRF et une heure d’expiration. Le cookie renvoie cet identifiant lors des demandes prévues. Aucune adresse email réelle, aucun code de connexion et aucun fournisseur payant ne sont utilisés.

Une session doit être distinguée de la permission et de la preuve. Le serveur retrouve un contexte ; il vérifie ensuite les règles de la route. Le journal d’essai indique qu’une requête a été traitée, mais ne démontre pas l’identité physique d’un apprenant. Une copie de cookie dans un rapport serait disproportionnée. Les valeurs peuvent être masquées tandis que leurs attributs et le comportement restent décrits.

Attributs et cycle de vie

Le cookie pédagogique porte HttpOnly et SameSite=Strict. HttpOnly limite la lecture du cookie par un script de page ; le navigateur peut encore l’envoyer lors d’une requête. SameSite participe au contrôle des envois selon le contexte du navigateur. Le cookie n’utilise pas Secure dans cet atelier HTTP local. Cette absence est un choix de démonstration annoncé, pas un exemple de déploiement public. L’état serveur expire après quinze minutes ; une nouvelle identité invalide l’ancienne session présentée et la déconnexion la supprime.

Un test d’invalidation doit réutiliser le même contexte ancien, puis observer son refus. Voir « déconnecté » dans l’interface ne suffit pas à établir l’invalidation serveur. Le programme verifier.py conserve le cookie temporaire pendant son test, demande la déconnexion puis contrôle qu’une lecture protégée répond 401. Il ne l’écrit pas dans les résultats partageables.

CSRF : session présente, intention à établir

La route correcte de modification attend l’origine locale et le jeton associé à la session. Sans jeton, elle répond 403. Avec le bon contexte et un champ admis, elle répond 200. Un jeton d’une autre session n’est pas une preuve d’intention pour celle-ci. Le TP compare ces cas via un client de test ; il ne prétend pas simuler une attaque complète depuis un site distant dans un navigateur.

Les contrôles sont complémentaires. Un script exécuté dans l’origine de l’application pourrait compromettre certaines garanties ; une protection CSRF n’élimine donc pas les risques XSS. L’objectif est de savoir quelle hypothèse un contrôle soutient, puis de vérifier un cas refusé et un cas légitime. La correction ne consiste pas à interdire toutes les modifications de profil.

TP guidé

Méthode pas à pas

Créez Alice, observez les attributs du cookie sans exporter sa valeur. Lancez la modification sans jeton puis la modification correcte. Exécutez le test automatisé d’origine et la déconnexion. Notez les statuts et vérifiez que le journal ne contient pas de jeton. Reprenez le code des conditions dans do_POST et context.

Exercice à réaliser

Rédigez quatre tests du cycle de session : création, usage, changement d’identité et invalidation. Distinguez les attributs du cookie, l’expiration serveur et le jeton CSRF dans vos conclusions.

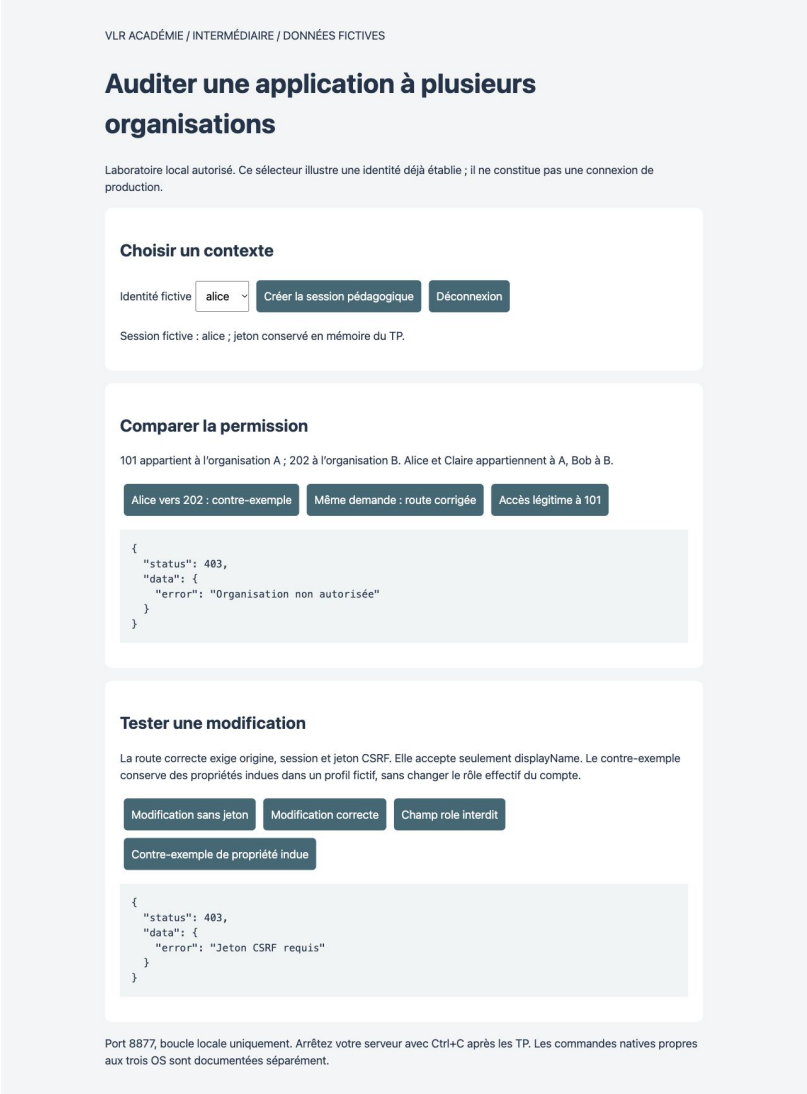
Cas entièrement fictif

L’interface locale affiche « modification refusée » sans jeton. Un apprenant affirme que toutes les attaques CSRF possibles sont donc éliminées, y compris sur une vraie application différente.

Corrigé raisonné

La preuve confirme les conditions des routes testées. Elle ne couvre ni une autre application, ni une origine compromise, ni toutes les configurations de navigateur. Le contre-test légitime reste nécessaire et le sélecteur n’est pas qualifié d’authentification de production.

Observer le laboratoire réel



Modification sans jeton sur la route corrigée : HTTP 403. Identités et données fictives. Aucune capture native Windows/Linux n’est revendiquée.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Quel contrôle vérifie la déconnexion serveur ?

A. Fermer seulement l’onglet

B. Réutiliser l’ancien contexte et constater 401

C. Lire seulement le message d’interface

2. HttpOnly signifie…

A. Le navigateur ne l’envoie jamais

B. Le serveur refuse tous les accès indus

C. Le cookie n’est pas lisible par le script de page

3. Pourquoi Secure est-il absent dans ce TP ?

A. TLS est inutile en production

B. Les permissions remplacent le transport

C. Le service est HTTP local et fictif

4. Sans jeton sur la route correcte de profil, quel statut attendre ?

A. Une ouverture de tous les documents

B. 403

C. 200

5. Un jeton CSRF élimine-t-il tous les risques XSS ?

A. Oui avec un nom long

B. Oui dans toute application

C. Non

CHAPITRE 06 / 5 h indicatives

Vérifier l’isolation des organisations et les propriétés d’API

Objectifs : Construire une matrice multi-identités ; Tester la propriété de ressource ; Distinguer propriété et rôle effectif

Une API peut accepter une identité valide tout en appliquant une permission incorrecte à la ressource demandée. Atelier Aurore distingue deux organisations et deux documents. Le contrôle corrigé compare l’organisation de l’identité, issue de USERS côté serveur, avec celle du document. Les documents sont volontairement peu nombreux pour permettre une matrice complète et une collecte minimale.

La route vulnerable montre une lecture inter-organisation sans la comparaison attendue. Alice reçoit 202 avec un statut 200. La route correcte renvoie 403 pour cette paire, et 200 pour Alice vers 101 ainsi que Bob vers 202. Ces résultats démontrent une séparation dans les cas distribués. Ils ne prouvent pas toutes les permissions d’une application réelle à plusieurs tenants, où rôles, délégations, exports et fonctions d’administration pourraient ajouter des chemins.

Ne pas confier l’organisation au client

Une application ne devrait pas décider de l’organisation admise sur la seule valeur d’un paramètre fourni par le client. Elle doit établir le contexte autorisé selon son modèle serveur. Des identifiants difficiles à deviner réduisent parfois la découverte, mais ne constituent pas la permission. Le test doit rester borné aux références 101 et 202, sans énumérer un jeu de données inconnu.

Le rôle est une autre dimension. Dans le laboratoire, la lecture des documents dépend de l’organisation ; le rôle gestionnaire ne crée pas un droit transversal sur B. Claire peut donc servir de contre-test conceptuel : un rôle plus élevé dans A ne devrait pas franchir la frontière d’organisation. La règle doit être formulée avant le résultat, et non inventée pour justifier une réponse observée.

Affectation indue de propriétés

La route correcte de profil admet uniquement displayName. Ajouter role provoque un refus 400, même avec une session et un jeton valides. La route de contre-exemple conserve la propriété indue dans un profil fictif. Le rôle effectif dans USERS reste inchangé ; la démonstration n’est donc pas une élévation effective de droits. Le rapport doit distinguer défaut de contrat d’entrée, état stocké et décision d’autorisation.

Préparez aussi les cas d’erreur : objet JSON requis, nom trop long, champ inattendu et session absente. Une limite de longueur participe au contrat, mais ne corrige pas une permission manquante. Un statut 400 peut être le comportement volontaire de la route corrigée. Les messages d’erreur doivent être interprétés avec la règle, pas notés comme des failles simplement parce qu’ils sont visibles.

TP guidé

Méthode pas à pas

Écrivez les attentes pour Alice, Bob, Claire et visiteur vers les deux documents. Exécutez les cas couverts par verifier.py puis complétez Claire via l’interface. Comparez le profil correct et le contre-exemple avec role. Vérifiez la référence USERS avant de conclure à un changement de rôle.

Exercice à réaliser

Produisez deux constats distincts : lecture inter-organisation et acceptation d’un champ indu. Pour chacun, proposez correction et contre-test, en indiquant exactement ce qui est démontré.

Cas entièrement fictif

Claire est gestionnaire de A. Une équipe fictive souhaite qu’elle lise tous les documents de B parce qu’elle possède un rôle de gestion. Cette extension n’est pas dans la règle du TP.

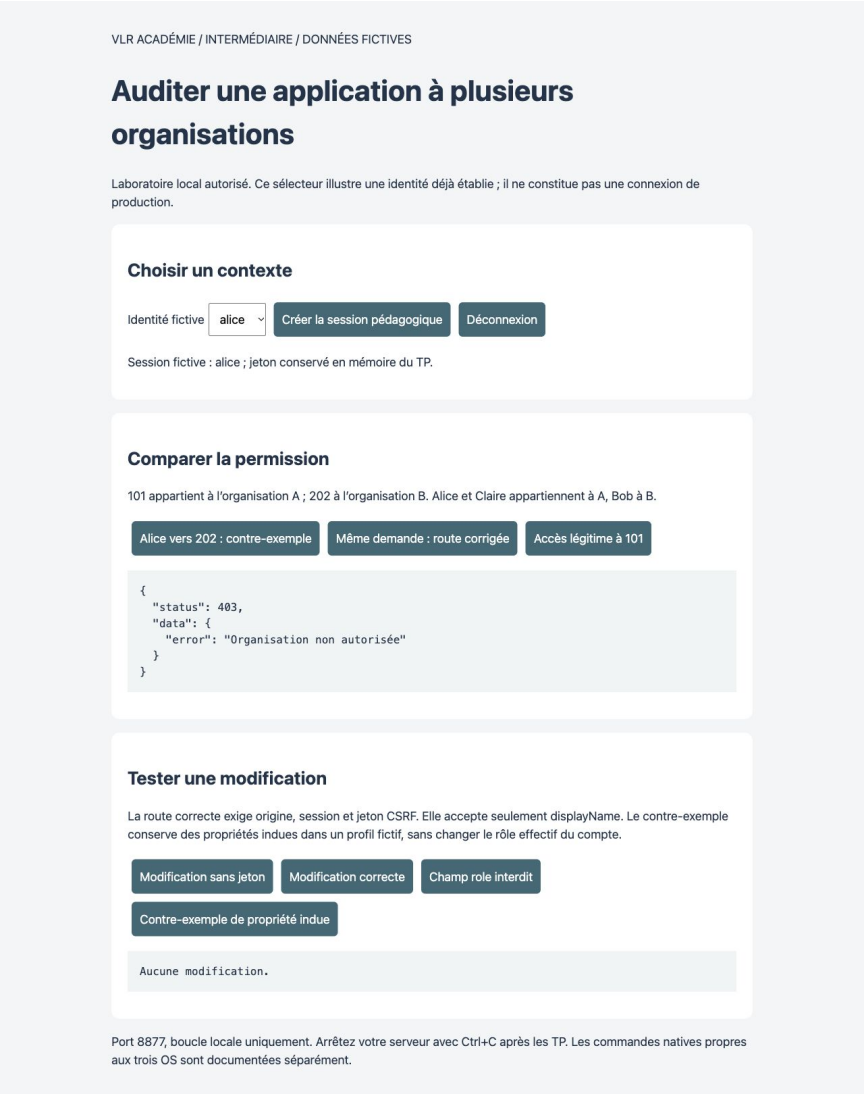
Corrigé raisonné

Le rôle dans A n’accorde pas automatiquement un droit sur B. L’organisation constitue une frontière séparée. La correction du profil limite les champs ; elle ne doit pas être présentée comme la correction unique de tous les accès API.

Observer le laboratoire réel



Capture réelle, macOS, 6 octobre 2026 : Alice lit le document fictif 202 de B sur le contre-exemple, HTTP 200.



Même demande sur la route corrigée : HTTP 403. Les accès légitimes sont vérifiés séparément dans la suite de tests.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Claire est gestionnaire de A. Peut-elle lire 202 dans la règle du TP ?

A. Oui si l’identifiant est court

B. Non, le document appartient à B

C. Oui grâce au rôle seul

2. Quel champ est admis dans le profil corrigé ?

A. role

B. displayName

C. tenant

3. role est envoyé avec session et jeton valides. Quel résultat attendre ?

A. Un rôle effectif élevé

B. 400 sur le profil corrigé

C. Un accès global à B

4. Un identifiant difficile à deviner remplace-t-il une permission ?

A. Oui pour une API

B. Non

C. Oui avec un cookie

5. Quel constat doit être distingué de la lecture inter-organisation ?

A. L’acceptation d’un champ de profil indu

B. La couleur du bouton

C. Le titre de la page

CHAPITRE 07 / 5 h indicatives

Suivre les entrées, les requêtes SQL et les sorties

Objectifs : Identifier une frontière d'interprétation ; Préserver une entrée légitime ; Adapter un contrôle à son contexte

Une entrée traverse souvent plusieurs opérations : lecture HTTP, conversion, validation, construction de requête, accès à une ressource et affichage. Un audit de code repère la source et les destinations, puis cherche la décision à chaque frontière. Le laboratoire SQL conserve seulement deux produits fictifs dans une base en mémoire. Il ne se connecte à aucune base de production et disparaît à la fin de chaque appel.

La fonction search comporte une construction concaténée de contre-exemple et une version paramétrée. Avec Lampe, les deux peuvent sembler correctes. Avec Bureau d'étude, l'apostrophe révèle le défaut de construction dans la version fragile ; la version paramétrée conserve la recherche légitime. Une entrée prévue de démonstration modifie la logique de la concaténation et retourne les deux produits, tandis que la recherche paramétrée la traite comme une valeur sans correspondance.

Un correctif qui respecte les données

Le paramètre SQL sépare la valeur de la structure de la requête. Retirer les apostrophes aurait dégradé un nom légitime sans constituer une stratégie générale. La validation peut limiter type, longueur et format quand cela correspond au contrat métier ; elle ne devrait pas effacer arbitrairement le sens d'une donnée. Le contre-test comprend donc les noms ordinaires, l'apostrophe légitime et le contre-exemple prévu.

L'encodage de sortie répond à une autre frontière. Pour un texte HTML, l'encodage empêche que les caractères de la donnée soient traités comme une structure HTML dans ce contexte. Une règle adaptée au texte d'un paragraphe ne couvre pas automatiquement une URL, un attribut ou un script. Ce cours reconnaît ces différences et demande une analyse de code, sans distribuer une collection de charges à lancer sur des sites extérieurs.

Chemins, fichiers et commandes

Lorsqu'une entrée désigne un chemin, le programme doit définir le répertoire autorisé et traiter la résolution réelle. Une simple suppression d'un motif textuel n'établit pas toujours cette frontière. Lorsqu'un programme appelle un interpréteur, une chaîne construite à partir d'une entrée peut changer la signification de la commande. Le laboratoire n'appelle aucun shell à partir d'une requête ; cette absence est un choix de réduction de surface à relever dans l'inventaire.

Pour chaque famille, indiquez ce que le code distribué permet effectivement d'essayer. Les démonstrations exécutables de ce module sont SQL et validation du profil. Les autres frontières font l'objet d'une revue conceptuelle. Une explication de prévention ne doit pas être présentée comme une exploitation réussie ou comme une couverture dynamique du système entier.

TP guidé

Méthode pas à pas

Exécutez les tests SQL de verifier.py. Relisez search, suivez term jusqu'à execute et comparez les deux appels. Relevez les trois entrées et leurs résultats. Ajoutez dans votre copie un cas légitime vide ou sans correspondance, puis expliquez son comportement. Documentez la correction sans modifier une cible extérieure.

Exercice à réaliser

Rédigez une fiche de flux pour displayName et une pour term. Distinguez validation, paramétrage SQL, permission et encodage de sortie. Justifiez pourquoi le nom avec apostrophe doit rester utilisable.

Cas entièrement fictif

Une équipe fictive remplace toutes les apostrophes et annonce que l'application est sécurisée. Elle ne teste plus les noms légitimes et ne relit pas le point de construction SQL.

Corrigé raisonné

La requête doit recevoir une valeur paramétrée. La suppression globale d'un caractère ne remplace pas cette séparation et peut altérer les données. Les usages légitimes et le contre-exemple sont vérifiés, avec la limite d'une base fictive en mémoire.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Pourquoi garder le nom avec apostrophe dans la recette ?

A. C'est toujours une attaque

B. C'est un usage légitime à préserver

C. Il faut supprimer tous les signes

2. Que sépare le paramètre SQL ?

A. Le cookie et la signature

B. La valeur et la structure de requête

C. Le réseau et le pare-feu

3. Un encodage de paragraphe couvre-t-il automatiquement un script ?

A. Oui universellement

B. Oui si le texte est court

C. Non, les contextes diffèrent

4. Une permission correcte remplace-t-elle la validation JSON ?

A. Oui pour un gestionnaire

B. Non

C. Oui sur localhost

5. Quel essai est réellement fourni ?

A. Un test de site tiers

B. Une récupération de mots de passe

C. SQL sur une base fictive en mémoire

CHAPITRE 08 / 5 h indicatives

Mener une revue de code et automatiser les contre-tests

Objectifs : Relier un constat à une fonction ; Choisir des tests significatifs ; Interpréter un résultat sans excès

La revue de code examine les décisions que la seule interface ne montre pas. Dans application.py, context retrouve la session, authorize compare les organisations, profile limite les champs et search illustre les deux constructions SQL. Chaque fonction possède une question différente. Lire le fichier du début à la fin sans annoncer ces questions produit souvent des remarques dispersées et difficiles à transformer en correction.

Commencez par une carte des entrées : chemins HTTP, en-têtes, cookie et corps JSON. Suivez ensuite les contrôles jusqu'à la réponse. Le nom vulnerable rend les contre-exemples explicites, mais le nom d'une fonction dans une vraie application ne serait pas une preuve de qualité. Une décision doit être lue dans son code et validée dans les conditions pertinentes. Le programme de tests apporte des exemples exécutables, pas une couverture absolue.

Trois familles de tests

Un test unitaire examine une décision isolée, par exemple l'organisation d'Alice face au document 202. Un test d'intégration HTTP examine la requête, la session, la route et la réponse ensemble. Un test de non-régression vérifie que la correction garde les usages admis. Dans verifier.py, ces dimensions se complètent : autorisation locale, parcours HTTP, SQL et restauration d'un fichier temporaire.

Le résultat attendu doit être indépendant de la ligne de code qu'il prétend vérifier. Pour l'accès Alice/202, la règle métier fournit 403 ; on ne la déduit pas de ce que la fonction renvoie aujourd'hui. Un test qui exige seulement « aucune exception » pourrait manquer une divulgation répondant 200. Un test qui attend seulement 403 pourrait aussi manquer une divulgation dans un corps incohérent : examinez la réponse utile dans votre dossier.

Résultat, échec et visibilité

Un échec de test peut signaler une régression ou un problème d'environnement. L'adresse, la version et la préparation servent à les distinguer. Ne remplacez pas la sortie en échec par celle d'un autre poste. Un test non exécuté n'est pas réussi. Si vous modifiez une copie du laboratoire, conservez la différence de code et le résultat avant/après, avec une description du changement voulu.

L'automatisation doit rester proportionnée. Le test HTTP démarre un serveur temporaire local, envoie un nombre limité de demandes et le ferme. Il ne demande pas d'URL arbitraire, de balayage réseau ou d'identifiant réel. Une analyse automatique peut relever un motif, mais elle ne remplace pas l'explication de son applicabilité. La preuve d'un contrôle reste contextualisée dans le rapport.

TP guidé

Méthode pas à pas

Exécutez verifier.py et conservez la sortie ainsi que resultats-http.json. Associez quatre assertions à leurs règles métier. Dans une copie, retirez la comparaison d'organisation de la route corrigée, constatez l'échec attendu, restaurez le fichier et relancez. Cette modification reste dans votre copie pédagogique et n'est jamais publiée.

Exercice à réaliser

Produisez une différence avant/après et un test de non-régression pour l'accès légitime. Expliquez ce que les quatre tests groupés couvrent, combien de scénarios HTTP sont enregistrés et quelles familles ne sont pas testées.

Cas entièrement fictif

Le rapport fictif dit « tous les tests passent donc il n'existe aucune vulnérabilité ». Pourtant, le service conserve volontairement des routes de contre-exemple et aucune évaluation complète de production n'est prévue.

Corrigé raisonné

La réussite confirme les assertions écrites dans cet environnement. Les routes vulnérables sont des objets pédagogiques annoncés, et la suite ne mesure pas toute la sécurité du système. Le rapport présente couverture, limites et résultats effectifs.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. D'où vient le résultat attendu d'un test d'autorisation ?

A. Du nom de la fonction

B. Du résultat actuel seulement

C. De la règle métier

2. Quel test examine plusieurs couches ensemble ?

A. La couleur d'un schéma

B. Un parcours HTTP avec session

C. Une empreinte seule

3. Tous les tests passent. Que conclure ?

A. Tous les OS sont validés

B. Toute vulnérabilité est impossible

C. Les assertions ont été vérifiées dans cet environnement

4. Quelle modification pédagogique reste admissible ?

A. Les permissions globales du poste

B. Une copie locale avec comparaison avant/après

C. Le site public sans mandat

5. Un outil relève un motif suspect. Quelle étape manque ?

A. La publication immédiate d'une faille

B. L'analyse de son contexte et de son applicabilité

C. Une conclusion financière

CHAPITRE 09 / 5 h indicatives

Inventorier les composants et vérifier un avis de sécurité

Objectifs : Décrire les composants réellement présents ; Vérifier l’applicabilité ; Préparer une mise à jour contrôlée

Un inventaire logiciel utile associe composant, version, origine, fonction, propriétaire et exposition. Une liste de noms sans contexte ne permet pas de prioriser. Le laboratoire utilise la bibliothèque standard Python et SQLite embarqué selon l’environnement de l’interpréteur. Relevez leurs versions réellement observées. Ne leur attribuez pas un avis de sécurité à partir d’un souvenir ou d’un numéro de version approximatif.

Un avis fournisseur décrit un produit, des versions, des conditions et des versions corrigées ou mesures de réduction. Un identifiant CVE facilite le rapprochement, tandis qu’une famille CWE décrit un type de faiblesse. L’applicabilité doit tenir compte de l’édition, des composants activés, de la configuration et des correctifs rétroportés. La présence d’une chaîne dans une bannière ne suffit pas à établir ce contexte.

Un jeu documentaire clairement fictif

Le dossier fournit inventaire-fictif.json avec un composant nommé vlr-demo-renderer. L’identifiant VLR-DEMO-01 est une référence pédagogique inventée, pas une CVE. Le scénario indique une fonctionnalité optionnelle et une version corrigée fictive pour apprendre le raisonnement. Le devoir consiste à classer une alerte applicable, une alerte non applicable et une information insuffisante, sans fabriquer un fait sur un vrai fournisseur.

La description doit garder cette provenance visible. Un exemple fictif prépare la lecture d’un avis réel, mais ne doit pas être cité publiquement comme une vulnérabilité du marché. Pour un exercice documentaire facultatif sur un vrai produit, consultez une source primaire récente et conservez l’URL, la date, les versions et les conditions. Aucun essai contre une installation extérieure n’est autorisé.

Risque et changement

Une sévérité technique décrit des propriétés selon un cadre et une version. Elle ne fixe pas seule la priorité métier, le délai de correction ou l’impact financier. Distinguez exposition, usage du composant, protections et conséquence plausible. Une version corrigée doit ensuite être validée avec les fonctions légitimes et les contre-tests. La sauvegarde et le retour arrière se préparent avant la modification.

Un inventaire peut également inclure les licences, les dépendances indirectes et la provenance des fichiers lorsque ces informations sont disponibles. Une empreinte aide à comparer un artefact téléchargé avec une référence fiable, mais ne résout pas seule la confiance envers la source. Le niveau avancé approfondira la chaîne logicielle ; ici, l’objectif est un dossier d’applicabilité honnête et reproductible.

TP guidé

Méthode pas à pas

Relevez les versions de Python et SQLite avec les outils du dossier. Lisez inventaire-fictif.json, construisez trois fiches d’applicabilité et séparez le jeu fictif de vos versions réelles. Préparez un plan de mise à jour sur papier, avec contrôle fonctionnel et retour à l’état initial.

Exercice à réaliser

Classez les trois lignes du jeu documentaire et expliquez l'information manquante. Pour une ligne, indiquez pourquoi un numéro de version seul ne suffit pas et quels contrôles accompagneraient un correctif.

Cas entièrement fictif

Le responsable fictif voit VLR-DEMO-01 et souhaite le publier comme une CVE récente. Il propose une mise à jour sans tester les fonctions utilisées par A et B.

Corrigé raisonné

VLR-DEMO-01 reste une référence pédagogique explicitement fictive. La mise à jour exige un contexte d'applicabilité, un changement défini et des tests fonctionnels. Aucun avis réel, montant ou partenariat ne sont inventés pour rendre le dossier impressionnant.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. VLR-DEMO-01 est…

A. Une référence fictive du TP

B. Un avis de Microsoft

C. Une CVE réelle confirmée

2. Une bannière seule suffit-elle à établir l'applicabilité ?

A. Oui si son texte est précis

B. Non

C. Oui pour toute version

3. Quel élément aide à expliquer un rétroportage ?

A. Le seul nom du navigateur

B. L'avis et la politique du fournisseur

C. Le nombre de captures

4. CVSS fournit principalement…

A. Un impact financier garanti

B. Une sévérité technique selon son cadre

C. Un délai contractuel universel

5. Quel test suit une mise à jour proposée ?

A. La recette des fonctions utilisées et le retour arrière

B. Un changement sans preuve

C. Un export de secrets

CHAPITRE 10 / 5 h indicatives

Corréler les événements et préserver les preuves minimales

Objectifs : Séparer les sources ; Construire une chronologie ; Vérifier la minimisation

Une chronologie d'audit relie des observations réellement disponibles, sans transformer une succession d'heures en causalité certaine. Le laboratoire conserve une liste en mémoire avec heure, méthode, chemin et statut. La capture d'écran montre une action dans l'interface ; le résultat JSON de tests décrit les attentes et observations du client ; le journal serveur décrit son traitement. Ces sources se complètent et peuvent avoir des limites différentes.

Le journal n'enregistre ni le cookie ni le jeton CSRF, et ne conserve pas les corps JSON. Cette minimisation limite les données collectées, mais réduit aussi ce que l'on peut conclure sur certains paramètres. Pour analyser un refus, combinez le plan de test, le contexte fictif choisi et le statut, sans prétendre lire une identité physique dans le journal. Les valeurs nécessaires peuvent rester dans un dossier privé expurgé avant partage.

Heure, ordre et incertitude

L'horodatage de l'application vient de son processus. Les captures et les notes peuvent utiliser un autre affichage de fuseau. Convertissez les heures de façon explicite et indiquez la précision. Deux événements proches ne prouvent pas qu'ils ont le même auteur ou la même cause. Un identifiant de test est souvent plus utile qu'une grande quantité de lignes sans association.

La route /api/journal montre les quarante événements les plus récents du laboratoire à une session fictive admise. Cette visibilité est un choix pédagogique annoncé, pas une politique recommandée pour tous les utilisateurs d'une application réelle. Une perte d'événements après arrêt est attendue puisque le stockage est en mémoire. Le rapport doit annoncer cette rétention plutôt que conclure à une disparition malveillante.

Empreinte et copies partageables

Une preuve originale est conservée avec son origine, son nom, l'heure de collecte et son empreinte. Une copie recadrée ou expurgée porte une mention de transformation. Une empreinte identique vérifie la cohérence des octets comparés, pas l'exactitude du commentaire ni la synchronisation de l'horloge. Ne réutilisez pas une capture du manuel comme si elle provenait de votre essai.

Les journaux natifs de Windows, Linux et macOS sont étudiés dans les modules propres à l'OS. Ils ne sont pas supposés contenir toutes les actions HTTP du laboratoire. Une absence de trace peut provenir d'un filtre, d'une rétention, d'une source différente ou d'une configuration. Le débutant apprenait à relever une ligne ; l'intermédiaire apprend à associer plusieurs sources en conservant leurs limites.

TP guidé

Méthode pas à pas

Après la session Alice, exécutez trois lectures et deux modifications prévues, puis relevez le journal local. Associez les cinq opérations à vos résultats et convertissez les heures dans une référence annoncée. Contrôlez l'absence de jeton dans le fichier partageable. Calculez son empreinte sur votre propre copie.

Exercice à réaliser

Produisez une chronologie de cinq lignes et une note d'incertitude. Expliquez ce que le journal démontre, ce que le client apporte et ce qui n'est pas attribuable à une personne réelle.

Cas entièrement fictif

Un rapport fictif interprète deux refus proches comme une attaque automatisée. Le plan montre pourtant deux actions manuelles prévues dans le TP et aucune preuve sur un acteur extérieur.

Corrigé raisonné

La chronologie décrit les actions du laboratoire et leur association disponible. Elle n'invente pas d'acteur ni d'intention. Les sources sont nommées séparément, la rétention en mémoire est indiquée et les jetons restent hors de la restitution.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. La trace HTTP conserve quoi dans le TP ?

A. Tous les cookies

B. Heure, méthode, route et statut

C. Tous les corps JSON

2. Deux événements proches prouvent-ils une cause commune ?

A. Non, une association doit être étayée

B. Oui si UTC est utilisé

C. Oui toujours

3. Le journal est vide après arrêt. Quelle explication annoncée existe ?

A. Le stockage est en mémoire

B. Une intrusion certaine

C. La suppression de toutes les captures

4. Que documenter pour une copie expurgée ?

A. La transformation et son origine

B. Une exécution inventée

C. La clé de session supprimée en clair

5. Quelle source connaît les attentes de verifier.py ?

A. Le journal natif universel

B. Le seul cookie

C. Le client de tests

CHAPITRE 11 / 5 h indicatives

Linux : permissions, ACL et contexte d’exécution

Objectifs : Lire les identités effectives ; Interpréter droits et ACL ; Limiter une modification à une copie

Linux distingue notamment les identités de processus, les groupes et les permissions de fichiers. Les bits classiques ne décrivent pas toujours l’ensemble des contrôles : des ACL ou un mécanisme obligatoire peuvent compléter la décision. L’observation doit nommer la distribution, les outils disponibles et le contexte. Un résultat sur une distribution n’est pas transposé automatiquement à toutes les autres.

Le rôle lecteur du laboratoire appartient à son modèle applicatif, pas à /etc/group. La fonction authorize compare les organisations de référence. Un processus qui peut lire un fichier local peut néanmoins le transmettre au mauvais client si sa logique HTTP est incorrecte. Un changement de chmod ne traite pas cette frontière applicative.

Lecture dans votre dossier

Les commandes id, ls -l et sha256sum décrivent identités, métadonnées et contenu. getfacl peut apporter les ACL lorsqu’il est installé et pertinent. La liste des outils ne constitue pas une obligation d’installer un paquet si le contexte ne l’autorise pas. Documentez l’absence et réalisez la partie disponible. Les commandes de la fiche sont à exécuter sur preuve.txt dans le dossier dédié.

Un dossier utilise les bits différemment d’un fichier : la traversée du chemin compte pour l’accès. Une permission 600 sur votre copie permet lecture et écriture au propriétaire dans le modèle classique ; elle ne chiffre pas et ne garantit pas l’absence d’accès administratif. N’appliquez aucune modification récursive au système ou au dossier personnel pour ce devoir.

Cas limite et explication

Une ACL visible doit être interprétée avec ses entrées et son contexte, sans résumer tous les droits à un nom. Une lecture refusée peut venir d’une frontière autre que les bits ordinaires. Le rapport décrit ce qu’il a observé et les contrôles qu’il n’a pas lus. Pour un changement réel, il faudrait un mandat et une procédure de retour arrière adaptés.

Méthode pas à pas

Dans votre copie du TP, relevez id, les permissions de preuve.txt et son empreinte. Examinez une ACL si l’outil existe. Comparez les observations avant/après un changement limité à copie-preuve.txt. Reliez chaque sortie à sa question sans l’utiliser comme preuve de permission HTTP.

Exercice à réaliser

Expliquez pourquoi 640, une ACL et un contexte de confinement ne se résument pas au même contrôle. Faites une note sur les différences entre fichier et dossier.

Cas entièrement fictif

La copie a des droits restrictifs, mais la route de contre-exemple transmet toujours le document de B à Alice. Le responsable fictif estime que Linux ignore ses permissions.

Corrigé raisonné

Le processus agit dans ses droits locaux ; le défaut concerne la décision applicative envers le client. La correction d'organisation reste nécessaire. Les essais natifs Linux sont à réaliser par l'apprenant et ne sont pas inventés dans les captures du manuel.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Un chmod corrige-t-il seul la permission HTTP ?

A. Oui avec 600

B. Oui

C. Non

2. getfacl apporte…

A. Une identité navigateur

B. Le score final

C. Des informations ACL si disponible

3. Le bit x d'un dossier participe à…

A. Une session HTTP

B. La traversée du chemin

C. Son chiffrement

4. 600 garantit-il l'absence de tout accès administratif ?

A. Oui

B. Non

C. Oui sur toute distribution

5. Quel changement convient au TP ?

A. Une copie dédiée

B. Un compte réel extérieur

C. Tout /etc récursivement

CHAPITRE 12 / 5 h indicatives

Linux : unités, sockets et politique de service

Objectifs : Distinguer unité et processus ; Lire une écoute ; Étudier le moindre privilège

Une unité gérée par systemd possède une configuration et un cycle de vie ; un processus Python lancé dans le terminal n’est pas automatiquement cette unité. Certaines distributions ou conteneurs n’utilisent pas systemd. L’intermédiaire doit reconnaître cette différence avant d’interpréter l’absence de systemctl ou une unité introuvable comme un problème de sécurité.

ss -ltnp 'sport = :8877' limite la lecture aux sockets TCP en écoute sur le port du TP. Les informations de processus dépendent des droits et du contexte. L’absence de nom de programme ne donne pas une raison automatique d’utiliser sudo. Vérifiez le PID de votre propre lancement, la liaison et l’heure, puis notez la limite restante.

Configuration fictive, observation réelle

configuration-os.json fournit un service Linux fictif avec utilisateur, adresse et méthode de restauration. Le fichier n’est pas une unité à installer. Il permet de proposer un compte minimal, une liaison locale et une méthode de validation. Les options de confinement systemd ou d’autres mécanismes doivent être étudiées dans la documentation de la version avant toute expérimentation en VM.

Un pare-feu, une liaison d’écoute et un contrôle applicatif sont trois décisions distinctes. Une liaison locale limite le service à la machine ; elle n’authentifie pas Alice et ne protège pas la référence de document à elle seule. Un service configuré avec un compte trop privilégié peut accroître les conséquences d’un défaut, sans que chaque processus de ce compte soit malveillant.

Préparer une vérification de service

Le dossier d’observation comprend processus, socket et réponse health. Après arrêt du serveur dans son terminal, comparez la disparition de l’écoute et une nouvelle requête. Une unité fictive dans le fichier n’est pas utilisée pour attribuer un résultat à systemd. L’atelier privilégie des preuves limitées et une lecture explicable plutôt qu’un inventaire global de la machine.

Méthode pas à pas

Relevez le seul port 8877 avant et après l’arrêt. Dans le scénario Linux, identifiez trois paramètres à contrôler et écrivez une recette. Si vous disposez d’une VM utilisant systemd, étudiez seulement votre unité de laboratoire autorisée, sans en créer sur le poste principal.

Exercice à réaliser

Comparez processus manuel, unité gérée et configuration documentaire. Expliquez comment une différence entre intention et socket réel pourrait être investiguée sans fermer un service inconnu.

Cas entièrement fictif

Un fichier documentaire annonce une unité VLR qui n’existe pas sur la distribution de l’apprenant. Celui-ci souhaite l’installer avec des privilèges administrateur pour rendre le rapport identique.

Corrigé raisonné

Le scénario fictif n'a pas à être installé. Le rapport distingue étude et observation. L'écoute du programme Python suffit pour le TP obligatoire ; un changement de service requiert un environnement et un mandat propres.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. systemctl est-il disponible dans tout environnement Linux ?

A. Oui

B. Oui dans tout conteneur

C. Non

2. ss avec sport = :8877 limite…

A. La lecture au port du TP

B. Le score CVSS

C. Tous les droits Linux

3. Un processus manuel est-il automatiquement une unité ?

A. Oui

B. Non

C. Oui si local

4. La configuration fictive doit-elle être installée ?

A. Oui pour valider le QCM

B. Non

C. Oui toujours

5. Une liaison locale garantit-elle la logique applicative ?

A. Non

B. Oui si Python est à jour

C. Oui

CHAPITRE 13 / 5 h indicatives

Linux : rétention, sources et corrélation

Objectifs : Décrire une source native ; Maîtriser un filtre ; Expliquer une absence de trace

Les distributions Linux peuvent collecter les traces par différents mécanismes. Sur un environnement qui le permet, journalctl consulte des événements associés à systemd ; d'autres sources peuvent être des fichiers applicatifs. La disponibilité et la rétention dépendent de la configuration. Le laboratoire Python distribué conserve sa propre trace en mémoire, ce qui ne constitue pas une preuve d'intégration au journal système.

Une commande de filtre doit préciser source, période et unité lorsqu'elle existe. Ne conservez pas un export de tous les journaux pour démontrer quelques actions locales. Une lecture à visibilité limitée peut être suffisante pour annoncer une limite. L'élévation de droits n'est pas un prérequis silencieux ; elle demande une justification dans le mandat de la VM.

Ordre et couverture

Associez le résultat client, la trace serveur et la source native disponible avec leurs heures. La rotation ou le stockage volatile peut retirer un historique ; la configuration peut ne pas collecter une action. Une absence ne prouve pas qu'aucune requête n'a eu lieu. La comparaison doit garder les événements réellement disponibles et ne pas combler les vides par des lignes imaginées.

Le journal HTTP ne contient pas les corps de profil. Il ne peut donc pas, seul, expliquer la propriété role envoyée. Le plan de test et le résultat du client fournissent cette information sur les données fictives. Le rapport distingue la source qui observe chaque élément, au lieu d'attribuer au serveur ou au système une précision qu'ils ne conservent pas.

Minimisation et conservation

Une preuve partageable garde les informations utiles et retire les valeurs privées. L'empreinte soutient une comparaison d'octets ; la provenance et la note de transformation soutiennent l'interprétation. Une chronologie avec une incertitude annoncée est plus fiable qu'un récit affirmatif construit à partir d'associations fragiles.

Méthode pas à pas

Produisez une chronologie des cinq opérations applicatives prévues. Si votre VM utilise un journal approprié, consultez une période limitée avec les exemples documentés de commandes-os.md. Marquez la couverture native disponible et vérifiez que les exports ne contiennent ni jeton ni données personnelles.

Exercice à réaliser

Expliquez le stockage volatile, la rotation et la différence entre trace client et trace serveur. Ajoutez à votre rapport une absence justifiée sans l'assimiler à une absence d'action.

Cas entièrement fictif

Le laboratoire redémarre et sa trace en mémoire est vide. Une équipe fictive annonce une suppression de preuves par un intrus.

Corrigé raisonné

Le cycle de vie annoncé explique la perte de cet état. Les captures et résultats sauvegardés restent des sources distinctes. Aucune intention hostile n’est déduite sans preuve ; la limite de rétention est documentée.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Le journal de TP est conservé…

A. Dans un service tiers

B. En mémoire du serveur

C. Universellement dans journalctl

2. Que peut expliquer une absence ?

A. Aucune action certaine

B. Une suppression hostile certaine

C. Filtre, rétention ou autre source

3. Le journal serveur contient-il le corps profile ?

A. Oui toujours

B. Oui après une copie

C. Non

4. Quelle preuve connaît les attentes ?

A. Le cookie seulement

B. Une décoration

C. Le résultat du client

5. Une chronologie doit annoncer…

A. Un auteur inventé

B. Sources et limites

C. Un événement déduit sans preuve

CHAPITRE 14 / 5 h indicatives

Linux : correctifs, confinement et restauration

Objectifs : Vérifier le contexte de distribution ; Distinguer les couches de protection ; Préparer un retour arrière

Une distribution maintient ses paquets et peut intégrer des correctifs sans reprendre un numéro amont complet. Le relevé de version doit donc être rapproché des avis et de la politique du fournisseur. Une alerte de bannière seule n'établit pas l'applicabilité. Le jeu inventaire-fictif.json reste clairement pédagogique et n'est pas une CVE à attribuer à votre système.

Les permissions classiques, le confinement d'un programme et le filtrage réseau répondent à des frontières différentes. AppArmor et SELinux ne sont pas des noms interchangeables ni des fonctions actives sur toutes les distributions. Une sortie vide ou un outil absent doit être interprété dans le contexte. Ne désactivez pas un mécanisme pour supprimer un message sans diagnostic.

Plan de changement

Décrivez distribution, versions, origine des paquets, fonction du composant, changement proposé, vérification et retour arrière. Dans une VM dédiée, un essai doit conserver son état de référence. Sur le poste principal, les travaux obligatoires se limitent aux lectures et à la restauration d'une copie du fichier pédagogique. Aucun changement d'administration global n'est requis.

La restauration d'un fichier peut être vérifiée par son contenu et son empreinte. Elle ne prouve pas la restauration de tous les états du système, du réseau ou d'une base. Une mise à jour pourrait aussi changer les dépendances ou nécessiter une migration. Le niveau avancé approfondira ces scénarios ; ici, indiquez ce qui est restauré et ce qui reste hors essai.

Conservation des usages

Une protection nouvelle peut réduire les capacités d'une application, mais elle doit préserver les actions admises. La recette locale vérifie toujours Alice/101 et Bob/202 en plus des refus. Une règle de pare-feu ne corrige pas le champ de profil indu ni la construction SQL. Le rapport justifie la mesure par son mécanisme, sans annonce globale de système « sécurisé ».

Méthode pas à pas

Étudiez le scénario Linux, préparez une grille mesure/bénéfice/limite/contrôle, puis réalisez la restauration du fichier de TP dans un dossier séparé. Relevez le gestionnaire et les fonctions disponibles de votre VM lorsque vous en disposez, sans inventer de résultat natif.

Exercice à réaliser

Rédigez un plan de correctif et expliquez le risque de conclure à partir d'une version amont seule. Distinguez dans votre recette permission locale, confinement et logique applicative.

Cas entièrement fictif

Une alerte documentaire cite une version différente de celle affichée par le paquet maintenu. L'équipe fictive souhaite publier immédiatement une vulnérabilité applicable sans lire l'avis fournisseur.

Corrigé raisonné

L’applicabilité reste à confirmer avec versions, configuration et rétroportages. Le jeu fictif n’est pas une alerte réelle. Les changements restent réversibles et limités à l’environnement autorisé.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Un rétroportage peut modifier…

A. Le rôle HTTP automatiquement

B. Le certificat

C. L’applicabilité d’un avis

2. AppArmor et SELinux sont-ils actifs partout ?

A. Non

B. Oui si le shell est bash

C. Oui

3. Avant changement en VM, que conserver ?

A. Les secrets dans le rapport

B. Une capture d’un autre OS

C. État de référence et recette

4. Restaurer preuve.txt prouve…

A. Tout le système restauré

B. Toute la base de l’entreprise

C. L’état du fichier comparé

5. Une bannière justifie-t-elle une faille publiée sans autre vérification ?

A. Non

B. Oui

C. Oui si la version est précise

CHAPITRE 15 / 5 h indicatives

Corriger, vérifier et préparer un changement réversible

Objectifs : Traiter la cause ; Préserver les usages légitimes ; Écrire un plan de déploiement fictif

Une correction utile répond à la cause du constat. Pour la lecture inter-organisation, la cause est l'absence de comparaison de contexte et de ressource sur la route de contre-exemple. Pour le profil, la cause est l'acceptation de propriétés qui ne font pas partie du contrat. Pour SQL, la cause est le mélange entre valeur et structure de requête. Un pare-feu plus restrictif ne traite pas à lui seul ces trois décisions applicatives.

Le laboratoire conserve les contre-exemples pour l'apprentissage et propose des fonctions correctes à comparer. Votre projet peut modifier une copie pour retirer les routes de contre-exemple et garder seulement les comportements admis. Cette copie reste locale. Ne publiez pas le serveur pédagogique sur une plateforme réelle, même après ces changements : des garanties de production, de maintenance et d'identité resteraient à concevoir.

Construire une recette

La recette associe chaque correction à un cas refusé et à un cas légitime. Alice/202 doit être refusé ; Alice/101 et Bob/202 doivent fonctionner. role doit être rejeté dans le profil, mais un nom valide doit être admis. Le terme avec apostrophe doit rester recherchable. Un test qui confirme seulement que la faille ne répond plus peut manquer une rupture fonctionnelle ou une correction placée sur la mauvaise route.

Conservez la version initiale, la différence de code, les résultats avant/après et le motif. Une correction de quelques lignes peut changer une politique d'accès ; sa petite taille ne réduit pas le besoin de vérification. Documentez aussi les protections que vous n'avez pas modifiées. Les limites d'un programme local en mémoire doivent apparaître dans la conclusion.

Changement et retour arrière

Un plan de changement fictif précise le responsable, la fenêtre, les prérequis, la sauvegarde, la validation et le critère de retour arrière. Dans le laboratoire, vous pouvez restaurer une copie de fichier et son empreinte. Dans une vraie application, le retour arrière peut aussi concerner des données et leurs migrations ; un fichier de code restauré ne garantit pas la cohérence de tous les états. Ce point sera développé au niveau avancé.

La priorité se justifie par l'applicabilité, l'exposition et l'effet, sans promettre une réduction chiffrée non mesurée. Un rapport doit permettre au propriétaire de décider, puis au lecteur de vérifier le résultat. La restitution inclut les points corrigés, ceux étudiés et ceux restant hors couverture. La formulation « sécurisé » sans périmètre est trop large pour ce projet.

TP guidé

Méthode pas à pas

Dans une copie, remplacez les chemins de contre-exemple par les décisions corrigées prévues. Écrivez une recette de huit cas et adaptez les attentes concernées. Exécutez les tests, conservez la différence et restaurez la référence initiale pour comparer l'empreinte. N'écrasez pas votre dossier de preuves.

Exercice à réaliser

Rédigez un plan de changement fictif d'une page et une fiche de validation. Définissez deux critères d'arrêt et un critère de retour arrière qui ne dépend pas seulement d'un statut HTTP.

Cas entièrement fictif

Après correction, toutes les lectures répondent 403. L'équipe fictive souhaite annoncer que la fuite inter-organisation est réglée et fermer la mission immédiatement.

Corrigé raisonné

La lecture indue est empêchée, mais l'usage légitime est cassé. La recette échoue et exige une correction de la politique, pas une validation globale. Le rapport distingue sécurité attendue, fonctionnement conservé et limites de l'environnement.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l'annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Toutes les lectures passent à 403 après correction. Que faire ?

A. Reprendre la correction car les usages légitimes échouent

B. Valider immédiatement

C. Supprimer les contre-tests

2. Quel correctif traite la lecture inter-organisation ?

A. Masquer seulement le bouton

B. Changer seulement la couleur

C. Contrôler le contexte contre la ressource

3. Une copie de code restaurée garantit-elle tous les états d'une application réelle ?

A. Oui si le fichier est court

B. Non

C. Oui avec le même nom

4. Quel élément précède un changement ?

A. La recette et le retour arrière préparés

B. Une annonce de succès

C. La suppression des preuves initiales

5. Le serveur de TP corrigé doit-il être déployé publiquement ?

A. Oui pour obtenir le diplôme

B. Non, il reste pédagogique

C. Oui automatiquement

CHAPITRE 16 / 15 h indicatives

Projet intermédiaire : audit argumenté d’Atelier Aurore

Objectifs : Assembler un dossier reproductible ; Défendre les conclusions ; Préparer la suite avancée

Le projet rassemble les compétences du parcours en une mission locale complète. Vous examinez la séparation entre organisations, les sessions, la modification du profil, le traitement SQL et la conservation des preuves. Quatre chapitres propres à votre OS complètent l’observation des identités, services, journaux et protections. Vous ne devez pas transformer ces observations locales en validation complète du système d’exploitation.

Le dossier contient un mandat, une fiche d’environnement, un schéma de flux, un inventaire, un plan de tests, des résultats et un rapport. Trois constats doivent être expliqués : lecture inter-organisation, propriété de profil indue et construction SQL fragile. Pour le profil, dites explicitement que le rôle effectif est resté inchangé. Ajoutez un faux positif écarté et une limite de couverture justifiée, sans inventer un quatrième défaut.

Livrables et soutenance personnelle

Chaque constat indique actif, condition préalable, étapes bornées, résultat réel, conséquence limitée, correction et contre-test. Les captures viennent de votre propre application locale. Les fichiers de résultat distinguent données de référence fictives et opérations réellement exécutées. Un README décrit la reproduction sans valeur de jeton. Les preuves partageables sont expurgées et portent leur origine.

Préparez une présentation orale de cinq minutes, même si vous la faites seul. Expliquez une frontière de confiance, un accès légitime conservé et une hypothèse que vos données ne permettent pas de confirmer. La soutenance personnelle et la grille du dossier constituent un entraînement ; la plateforme ne revendique pas de jury ni de correction humaine de vos fichiers. Le QCM valide des connaissances séparément de cette production pratique.

Certification et prochaine étape

Seize QCM de chapitre comportent chacun cinq questions et exigent quatre réponses correctes. Après leur réussite, le QCM final comporte quarante questions et demande trente-deux bonnes réponses. Le diplôme interne VLR est gratuit et porte le parcours correspondant à l’OS. Il ne constitue pas une reconnaissance universitaire ou une attestation supervisée de toutes vos compétences pratiques.

Le niveau avancé sera construit ensuite. Sa préparation exige de savoir interpréter une limite, relire du code, concevoir une recette et expliquer des décisions au propriétaire. Les scénarios multi-machines, investigations plus complexes et évaluations spécialisées devront recevoir des laboratoires et prérequis propres. Cette étape intermédiaire ne promet pas leur contenu comme s’il était déjà disponible.

TP guidé

Méthode pas à pas

Utilisez rapport-modele.md et grille-projet.md. Assemblez trois constats, un faux positif, une chronologie, une différence de code et une recette. Relisez les faits, la langue, les références et les preuves. Faites une reprise complète depuis une copie propre, puis passez les QCM du parcours sur VLR.

Exercice à réaliser

Livrez votre dossier privé avec une synthèse d’une page et des annexes. Expliquez trois acquis transférables et deux compétences à approfondir avant une formation avancée. Aucune preuve ne doit être copiée d’une autre personne.

Cas entièrement fictif

Le projet fictif prétend démontrer une compromission complète à partir d’un champ role accepté dans le profil. Le test montre pourtant que la référence serveur a conservé lecteur.

Corrigé raisonné

La conclusion est resserrée au défaut démontré. Les trois constats ont une preuve réelle, un correctif pertinent et des usages légitimes conservés. La maîtrise pratique est autoévaluée ; le diplôme est déclenché par le circuit de QCM annoncé, sans résultat inventé.

QCM de chapitre : entraînement

Répondez avant de consulter les corrigés de l’annexe. Pour valider la progression, passez ce QCM sur le site. Cinq questions ; seuil quatre bonnes réponses.

1. Combien de QCM de chapitre faut-il réussir ici ?

A. Douze

B. Seize

C. Un seul

2. Quel seuil final est annoncé ?

A. 20 sur 40

B. 31 sur 40

C. 32 bonnes réponses sur 40

3. La plateforme corrige-t-elle humainement le dossier ?

A. Non, le projet est autoévalué

B. Oui via l’OS

C. Oui via un jury annoncé

4. Quel constat ne doit pas être inventé ?

A. La lecture fictive réellement observée

B. Le champ de profil réellement accepté

C. Une élévation effective lorsque USERS reste inchangé

5. Les niveaux avancés sont-ils livrés par ce parcours ?

A. Oui après un seul TP

B. Oui dans le même diplôme

C. Non, ils seront développés ensuite

Dossier pratique et commandes

Télécharger les scripts, données fictives et modèles du TP:
https://viselareussite.com/images/academy/hacking-ethique-intermediaire-laboratoire/2026.10.6.1/laboratoire.zip

Atelier Aurore : guide des travaux pratiques

Version 2026.10.6.1. Données fictives. Formation gratuite VLR, niveau intermédiaire.

Préparer sa copie

Décompressez le dossier dans un répertoire de travail personnel. Python 3.10 ou ultérieur et un navigateur récent suffisent ; aucune dépendance externe n'est requise. Lisez application.py, atelier.html et verifier.py avant le lancement. Ne lancez pas ces scripts dans un dossier de production. Les fichiers de preuves appartiennent à votre copie de TP.

Windows : ouvrez PowerShell dans ce dossier puis py -3 application.py. Linux ou macOS : python3 application.py. Si cette commande n'existe pas, consultez la documentation officielle de Python pour votre OS. N'installez aucun outil depuis un lien non vérifié.

Ouvrez exactement http://127.0.0.1:8877. Le serveur reste sur la boucle locale. Ne changez pas l'adresse en 0.0.0.0. Si le port est pris, identifiez le processus ; ne terminez pas un service inconnu. Ctrl+C arrête votre serveur et supprime les sessions, profils et traces conservés en mémoire.

Lire le modèle

Alice et Claire appartiennent à l'organisation A ; Bob à B. Le document 101 appartient à A, 202 à B. Le sélecteur distribue une identité de démonstration : il ne vérifie aucune identité réelle. La route corrigée autorise l'organisation du document ; elle ne confère pas des droits globaux à un gestionnaire.

La route de contre-exemple accepte une lecture inter-organisation. Le profil de contre-exemple peut conserver role=gestionnaire sans modifier le rôle effectif dans USERS. Cette limite doit rester visible dans votre constat. La base SQL ne contient que deux produits fictifs en mémoire.

Séquence de référence

1. Remplir mandat.md et relever les versions.

2. Dessiner les flux navigateur, session, USERS, documents, journal et base SQL.

3. Créer une session Alice. Observer Alice/202 avec le contre-exemple, puis avec la route corrigée. Conserver les statuts réellement obtenus.

4. Vérifier Alice/101 et Bob/202 sur la route corrigée. Vérifier Bob/101 refusé. Ne confondre ni un refus universel ni un changement d'interface avec une permission correcte.

5. Tester la modification sans jeton, la modification correcte et le champ role interdit. Comparer la propriété indue à la référence serveur.

6. Déconnecter le contexte. Les tests automatisés vérifient aussi l'invalidation de l'ancien cookie.

7. Arrêter votre serveur. Lancer py -3 verifier.py sur Windows ou python3 verifier.py sur Linux/macOS. Les quatre tests utilisent un serveur temporaire sur un port local disponible ; ils le ferment à la fin. resultats-http.json contient des observations fictives sans cookie ni jeton.

8. Étudier les fonctions authorize, profile et search. Sur une copie, introduire une seule modification réversible et utiliser les tests pour voir ce qu'ils détectent. Comparer aux critères fonctionnels avant de conclure.

9. Consulter les fiches natives de votre OS. Distinguer les commandes réellement exécutées de celles étudiées dans la documentation.

10. Rédiger le rapport : trois constats étayés, un faux positif écarté, les usages légitimes conservés et les limites de couverture.

Répartition des ateliers

Chapitres 1-3 : mandat, reproductibilité, modèle de menace. Chapitres 4-7 : couches HTTP, sessions/CSRF, organisations/propriétés, SQL/entrées. Chapitres 8-10 : revue de code, dépendances fictives, corrélation et preuves. Chapitres 11-14 : permissions, exposition, journaux et recette propres à l'OS. Chapitres 15-16 : remediation et projet argumenté.

Les quinze premiers chapitres représentent chacun un travail indicatif de cinq heures : lecture 1,5 h, TP 2,5 h, révision 1 h. Le projet final représente 15 h, dont 12 h de dossier et 3 h de préparation finale. Ce volume de 90 h est indicatif et distinct de la durée réelle des vidéos. Une partie du tronc commun est partagée entre les trois parcours ; l'avoir étudiée ne représente pas trois travaux différents.

Évaluations

Chaque chapitre propose cinq questions sur le site : au moins quatre bonnes réponses. La plateforme ouvre l'examen après validation des seize QCM. Examen : 40 questions, cinq domaines de huit questions, réussite à 32/40. Le rapport utilise grille-projet.md en autoévaluation. Les QCM sont non surveillés. Le diplôme est interne VLR ; aucune reconnaissance universitaire ni certification tierce n'est revendiquée. Le certificat et l'accès sont gratuits.

Preuves et limites

Les captures fournies montrent de vrais essais du laboratoire sur macOS, avec données fictives. Les commandes natives Windows et Linux sont documentées, sans exécution native revendiquée. Les tokens, emails réels, journaux privés et mots de passe n'ont pas leur place dans un dossier partagé. Le serveur pédagogique ne doit pas être mis sur Internet. Le niveau avancé sera distinct ; ce dossier ne promet pas son ouverture immédiate.

Fiches natives : lecture et observation

Les commandes ci-dessous concernent votre dossier et votre serveur de TP. Windows et Linux : étudiées dans les documentations officielles, sans essai natif revendiqué. macOS : relevés propres au TP exécutés lors de la production. Les résultats varient selon version, installation et permissions ; un outil absent se signale, il ne se remplace pas par une suppression de protection.

Windows / PowerShell

Depuis le dossier du TP :

```
$PSVersionTable.PSVersion
Get-FileHash -LiteralPath .\preuve.txt -Algorithm SHA256
Get-Acl -LiteralPath .\preuve.txt | Format-List
Get-NetTCPConnection -LocalPort 8877 -State Listen
```

La dernière commande peut ne rien retourner si le serveur est arrêté. Relevez OwningProcess, puis Get-Process -Id ID_OBSERVE en remplaçant uniquement l'identifiant observé. Get-WinEvent exige une source présente et les droits correspondants ; ne supposez pas un journal d'application Python enregistré. Get-AuthenticodeSignature n'établit pas la sécurité globale d'un script. Ne changez pas la stratégie d'exécution globale pour ce TP Python.

Linux

Depuis le dossier du TP :

```
id
ls -l preuve.txt
sha256sum preuve.txt
ss -ltn 'sport = :8877'
```

getfacl preuve.txt est optionnel si disponible. Une socket locale n'indique pas à elle seule les droits applicatifs. systemctl et journalctl ne sont pertinents que sur un environnement compatible systemd ; le lancement direct de Python n'installe pas une unité. N'appliquez pas chmod récursivement ni une désactivation globale de confinement.

macOS

Depuis le dossier du TP :

```
id
ls -le preuve.txt
stat -f '%Sp %Su:%Sg %N' preuve.txt
shasum -a 256 preuve.txt
lsof -nP -iTCP:8877 -sTCP:LISTEN
```

Le port doit appartenir à votre Python et être lié à 127.0.0.1. Le journal mémoire de l'atelier n'est pas une preuve équivalente au journal unifié macOS. Console et log peuvent être étudiés sur une source de votre propre TP, en évitant des logs privés. Le TP ne requiert aucune installation launchd, modification de SIP ou Gatekeeper. codesign/spctl concernent un artefact connu et ne valent pas verdict universel.

Avant et après un changement

Copiez uniquement preuve.txt dans votre dossier de recette. Mesurez son empreinte, modifiez votre copie, comparez le contenu, puis restaurez la copie et vérifiez son empreinte initiale. Ne corrigez pas une ACL système. Les commandes sont des observations à interpréter, pas des configurations de production à appliquer.

Mandat pédagogique à compléter

Organisation fictive : Atelier Aurore. Version : 2026.10.6.1.

Périmètre : votre copie des scripts, données distribuées et boucle locale. Interface : 127.0.0.1:8877. Suite automatisée : port temporaire local. Identités : Alice, Bob, Claire, exclusivement fictives. Exclusions : réseau tiers, comptes personnels, messagerie, scans distants, charge, persistance, modification des protections de l'OS. Arrêt : destination hors boucle locale, donnée inattendue, port occupé par un service inconnu, effets dépassant la copie du TP.

À remplir : objectif, version Python, OS, date, budget de requêtes, contrôles, hypothèses, usages légitimes, critères de réussite, limites et emplacement des preuves.

Matrice : identifiant du contrôle | scénario | précondition | attendu | observation réelle | état de couverture | preuve. Une observation attendue ne remplace jamais un résultat réellement exécuté.

Rapport d'audit pédagogique

Résumé : besoin, périmètre local, observations, correction proposée et limites. Contexte : versions, identités fictives, organisations, état du serveur et dossier distribué. Couverture : contrôles testés, étudiés sans exécution, non applicables ou hors périmètre.

Fiche d'un constat

Identifiant et titre. Règle attendue. Route ou fonction. Préconditions. Méthode bornée. Résultat réel avec date. Preuve minimale. Conséquence démontrée dans le modèle fictif. Cause dans le code. Correction proposée. Contre-test positif et négatif. Limites.

Faux positif écarté

Hypothèse initiale, vérification contradictoire, résultat, raison de l'exclusion. Le rôle conservé dans un profil ne prouve pas un rôle effectif modifié.

Recette et retour arrière

Critères, ordre des vérifications, empreintes avant/après des fichiers du TP, résultat des tests et conservation des usages légitimes. Le changement ne porte que sur votre copie.

Annexes

Matrice, modèle de menace, résultats HTTP sans jetons, commandes propres à l'OS réellement exécutées, captures utiles, inventaire et avis fictif explicitement étiquetés. Aucune preuve réelle d'une autre personne.

Autoévaluation du projet intermédiaire : 100 points

Périmètre, exclusions et critères : 15 points. Modèle de menace et couverture argumentée : 15 points. Trois constats reproductibles avec preuves et causes : 25 points. Un faux positif réellement écarté : 10 points. Corrections et contre-tests préservant les usages légitimes : 20 points. Clarté du rapport, provenance et limites OS : 10 points. Plan de retour arrière sur la copie du TP : 5 points.

Attribuez chaque point à un élément retrouvé dans votre dossier. Une affirmation sans preuve n'est pas un résultat. Les constatations doivent rester distinctes : lecture entre organisations, propriétés de profil indues, construction SQL dans la base de TP. Ne prétendez ni à une élévation de rôle effective ni à une compromission de tiers. Cette grille ne donne pas lieu à une correction humaine. Elle n'est pas le score du QCM certificatif. Un dossier incomplet doit être repris avant de considérer la pratique comme acquise.

Lecture du code fourni

Ces extraits sont tirés du laboratoire distribué. Les fonctions complètes, le serveur et les tests se trouvent dans l’archive ZIP. Lisez les préconditions et utilisez uniquement votre copie locale.

Fonction authorize

```
def authorize(user,doc):return user['tenant']==doc['tenant']
```

Fonction profile

```
def profile(values,corrected=True):
 if corrected and set(values)-{'displayName'}:raise ValueError('Champs non admis')
 if not isinstance(values.get('displayName',''),str) or len(values.get('displayName',
''))>60:raise ValueError('Nom incorrect')
 return {'displayName':values.get('displayName','')} if corrected else values.copy()
```

Corrigés des QCM de chapitre

Ces corrigés servent à la révision. Le questionnaire final est passé dans la plateforme après les seize validations de chapitre. Le score du projet ne remplace pas le QCM final.

Chapitre 01

1. Réponse A : Alice/101 autorisé. Un contrôle conserve les usages légitimes.
2. Réponse A : Vérifier sa couverture avant l'essai. Une découverte ne crée pas une autorisation.
3. Réponse B : Non testée dans la couverture annoncée. Une absence de test est une limite.
4. Réponse A : Une observation liée à une hypothèse. Le constat doit être démontré.
5. Réponse A : L'exclure et l'annoncer. La mission locale ne couvre aucun tiers.

Chapitre 02

1. Réponse A : Ils servent à deux contextes de lancement. L'interface et la suite de tests ont des serveurs distincts.
2. Réponse A : Les sessions et profils en mémoire. Le programme ne persiste pas ces états.
3. Réponse A : Une nouvelle requête est nécessaire. Un contenu déjà chargé ne mesure pas l'état actuel.
4. Réponse C : resultats-http.json. Le résultat doit être créé dans la copie du TP.
5. Réponse C : Identifier son propriétaire. Un service inconnu doit être préservé.

Chapitre 03

1. Réponse B : Une propriété indue conservée. Le champ stocké et le rôle de décision sont distincts.
2. Réponse B : La référence serveur USERS. Le contexte autorisé doit être établi côté serveur.
3. Réponse A : À préparer une question et un contrôle. Une grille est un outil de raisonnement.
4. Réponse A : Une entrée client vers une décision serveur. Une entrée peut influencer la décision.
5. Réponse C : La preuve doit combiner plusieurs sources. La minimisation limite aussi la précision disponible.

Chapitre 04

1. Réponse B : La requête a atteint un traitement applicatif. Les observations concernent des couches différentes.
2. Réponse A : Non, ce sont deux contrôles distincts. Le cache ne décide pas de l'autorisation.
3. Réponse A : La valeur du cookie temporaire. La valeur du jeton n'aide pas à prouver le comportement.
4. Réponse A : C'est un choix pédagogique annoncé. La politique doit être adaptée à l'application réelle.
5. Réponse C : Une portée de liaison du service observé. La liaison ne remplace pas les contrôles applicatifs.

Chapitre 05

1. Réponse B : Réutiliser l'ancien contexte et constater 401. L'état serveur doit être invalidé.
2. Réponse C : Le cookie n'est pas lisible par le script de page. L'attribut a une portée spécifique.
3. Réponse C : Le service est HTTP local et fictif. Le choix local ne se transpose pas au public.
4. Réponse B : 403. La condition CSRF de ce TP doit être satisfaite.
5. Réponse C : Non. Les frontières et garanties ne sont pas identiques.

Chapitre 06

1. Réponse B : Non, le document appartient à B. Le rôle ne supprime pas la frontière d'organisation.
2. Réponse B : displayName. Le contrat n'admet que le nom affiché.

3. Réponse B : 400 sur le profil corrigé. Le contrat de propriété reste contrôlé.

4. Réponse B : Non. La décision d'accès reste nécessaire.

5. Réponse A : L'acceptation d'un champ de profil indu. Les causes et corrections sont différentes.

Chapitre 07

1. Réponse B : C'est un usage légitime à préserver. La correction doit respecter les données attendues.

2. Réponse B : La valeur et la structure de requête. La valeur ne doit pas modifier la syntaxe SQL.

3. Réponse C : Non, les contextes diffèrent. L'encodage dépend de la destination.

4. Réponse B : Non. Le contrat d'entrée et le droit sont distincts.

5. Réponse C : SQL sur une base fictive en mémoire. Le programme distribué délimite sa démonstration.

Chapitre 08

1. Réponse C : De la règle métier. Le test doit être indépendant du défaut observé.

2. Réponse B : Un parcours HTTP avec session. Le parcours relie requête, contexte et route.

3. Réponse C : Les assertions ont été vérifiées dans cet environnement. La couverture reste limitée aux assertions.

4. Réponse B : Une copie locale avec comparaison avant/après. La copie distribuée constitue le périmètre de TP.

5. Réponse B : L'analyse de son contexte et de son applicabilité. Un signal automatique n'est pas un constat complet.

Chapitre 09

1. Réponse A : Une référence fictive du TP. Sa provenance fictive doit rester explicite.

2. Réponse B : Non. Configuration et correctifs peuvent modifier le contexte.

3. Réponse B : L'avis et la politique du fournisseur. Une version maintenue peut intégrer des correctifs.

4. Réponse B : Une sévérité technique selon son cadre. Le risque métier exige son contexte.

5. Réponse A : La recette des fonctions utilisées et le retour arrière. La correction doit conserver les usages admis.

Chapitre 10

1. Réponse B : Heure, méthode, route et statut. Le journal est volontairement minimal.

2. Réponse A : Non, une association doit être étayée. La proximité temporelle ne suffit pas.

3. Réponse A : Le stockage est en mémoire. Le cycle de vie explique la rétention.

4. Réponse A : La transformation et son origine. L'interprétation doit rester traçable.

5. Réponse C : Le client de tests. Le résultat client et la trace serveur ont des rôles distincts.

Chapitre 11

1. Réponse C : Non. Les droits locaux et la décision applicative doivent être distingués.

2. Réponse C : Des informations ACL si disponible. Les droits locaux et la décision applicative doivent être distingués.

3. Réponse B : La traversée du chemin. Les droits locaux et la décision applicative doivent être distingués.

4. Réponse B : Non. Les droits locaux et la décision applicative doivent être distingués.

5. Réponse A : Une copie dédiée. Les droits locaux et la décision applicative doivent être distingués.

Chapitre 12

1. Réponse C : Non. Le contrôle porte sur le laboratoire réellement observé, avec sa limite.

2. Réponse A : La lecture au port du TP. Le contrôle porte sur le laboratoire réellement observé, avec sa limite.

3. Réponse B : Non. Le contrôle porte sur le laboratoire réellement observé, avec sa limite.

4. Réponse B : Non. Le contrôle porte sur le laboratoire réellement observé, avec sa limite.

5. Réponse A : Non. Le contrôle porte sur le laboratoire réellement observé, avec sa limite.

Chapitre 13

1. Réponse B : En mémoire du serveur. Les sources et leurs couvertures sont distinctes ; une supposition n’est pas une preuve.

2. Réponse C : Filtre, rétention ou autre source. Les sources et leurs couvertures sont distinctes ; une supposition n’est pas une preuve.

3. Réponse C : Non. Les sources et leurs couvertures sont distinctes ; une supposition n’est pas une preuve.

4. Réponse C : Le résultat du client. Les sources et leurs couvertures sont distinctes ; une supposition n’est pas une preuve.

5. Réponse B : Sources et limites. Les sources et leurs couvertures sont distinctes ; une supposition n’est pas une preuve.

Chapitre 14

1. Réponse C : L’applicabilité d’un avis. La mesure doit être adaptée au risque et garder une recette bornée.

2. Réponse A : Non. La mesure doit être adaptée au risque et garder une recette bornée.

3. Réponse C : État de référence et recette. La mesure doit être adaptée au risque et garder une recette bornée.

4. Réponse C : L’état du fichier comparé. La mesure doit être adaptée au risque et garder une recette bornée.

5. Réponse A : Non. La mesure doit être adaptée au risque et garder une recette bornée.

Chapitre 15

1. Réponse A : Reprendre la correction car les usages légitimes échouent. Un blocage général ne conserve pas le fonctionnement.

2. Réponse C : Contrôler le contexte contre la ressource. La permission doit être appliquée au bon endroit.

3. Réponse B : Non. Les données et migrations peuvent avoir un autre état.

4. Réponse A : La recette et le retour arrière préparés. La validation doit être définie avant l’opération.

5. Réponse B : Non, il reste pédagogique. Des garanties de production restent hors du modèle.

Chapitre 16

1. Réponse B : Seize. Le parcours intermédiaire comporte seize chapitres.

2. Réponse C : 32 bonnes réponses sur 40. Le seuil de réussite est 80 %.

3. Réponse A : Non, le projet est autoévalué. Le diplôme repose sur le circuit de QCM décrit.

4. Réponse C : Une élévation effective lorsque USERS reste inchangé. La conclusion doit suivre la preuve.

5. Réponse C : Non, ils seront développés ensuite. Les programmes futurs restent distincts.

Sources et provenance

Les leçons et scénarios sont originaux. Les références ci-dessous soutiennent les concepts et les observations documentaires. Les avis VLR-DEMO-01 et le composant vlr-demo-renderer sont entièrement fictifs. Les sources ont été consultées le 6 octobre 2026. Les commandes natives Windows et Linux restent documentaires ; le laboratoire Python a été testé sur macOS.

OWASP : modèle de menace: https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html

OWASP : autorisation: https://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html

OWASP : sessions: https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html

OWASP : prévention CSRF: https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

Python : paramètres SQLite: https://docs.python.org/3/library/sqlite3.html

Microsoft : Get-Acl: https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.security/get-acl

Microsoft : Get-NetTCPConnection: https://learn.microsoft.com/en-us/powershell/module/nettcpip/get-nettcpconnection

Linux : ss: https://man7.org/linux/man-pages/man8/ss.8.html

Apple : sécurité de la plateforme: https://support.apple.com/guide/security/welcome/web

Cours sur Vise la Réussite: https://viselareussite.com/formations/hacking-ethique-intermediaire-linux/

Vidéos : supports originaux avec narration synthétique française ; deux voix au choix et sous-titres avec découpage estimé. Captures : interface locale réelle, données fictives, macOS. Schémas et couvertures : créations originales sans carte.