CORRIGÉ NON OFFICIEL

BTS Services Informatiques aux Organisations

E6 - Cybersécurité des services informatiques (spécifique à l'option)

Session 2026

Proposition de corrigé rédigée par FormaV pour s'entraîner. Ce n'est pas le corrigé officiel de l'examen et elle ne remplace pas les éléments de correction officiels.

Ce document a été mis en ligne par l'organisme FormaV®.

Toute reproduction, représentation ou diffusion, même partielle, sans autorisation préalable, est strictement interdite.

Pour consulter tous nos sujets d'annales et corrigés du BTS SIO, veuillez visiter :
coursbtssio.fr/annales-sujets-et-corriges

BTS Services informatiques aux organisations – Option B : Solutions logicielles et applications métiers

Session 2026

U7 – Cybersécurité des services informatiques

PROPOSITION DE CORRIGÉ

Proposition de corrigé établie par FormaV à partir du sujet, non officielle. Elle ne préjuge pas des consignes nationales de correction données aux jurys.

DOSSIER A – ÉVOLUTIONS DE L'INTRANET DE L'AGENCE NDV EN MATIÈRE DE SÉCURITÉ — 42 points

Mission A1 – Évolution de la gestion des journaux d'activité (logs) du SGBD de l'intranet

Question A1.1

Compléter le code de la fonction print_logs() pour aussi tenir compte du filtre par nature. Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés.

Ajout après la ligne 16 (déclaration du filtre par nature, sur le modèle de la date) :

```
$nature = null;
if (isset($_POST['type_de_logs']) AND $_POST['type_de_logs'] != '') {
    $nature = $_POST['type_de_logs'];
}
```

Ligne 17 (requête modifiée) :

```
$query = "SELECT * FROM t_logs
    WHERE (:date IS NULL OR DATE(log_date) = :date)
    AND (:nature IS NULL OR log_nature = :nature)
    ORDER BY log_date ASC";
```

Ajout après la ligne 20 (liaison du paramètre) :

```
$stmt->bindParam(':nature', $nature, PDO::PARAM_STR);
```

Question A1.2

a) Indiquer les éléments dans les journaux qui confirment que l'utilisateur 'ndv_64784' peut se connecter depuis une autre adresse IP que le serveur local et avec plus de droits que nécessaire.

Document A1a, lignes 6 à 14 :

- lignes 6 et 7 (16:28:07 et 16:30:11) : deux tentatives refusées du compte admin depuis 172.17.12.45, puis connexion réussie avec ndv_64784 depuis la même adresse à la ligne 8 ;
- ligne 8 (18:10:36) : « CONNEXION [ndv_64784]@[172.17.12.45] ndv_64784@172.17.12.45 on ndv using SSL/TLS » : connexion depuis une adresse IP distante et non depuis localhost ;
- ligne 9 (18:11:14) : « CREATE USER 'test'@'%' IDENTIFIED WITH 'mysql_native_password' AS '' » exécutée par ndv_64784 (compte accessible depuis n'importe quelle adresse et sans mot de

passe) : création d'un compte, opération qui excède les droits de création, lecture, mise à jour et suppression sur les données ;

- ligne 10 (18:11:27) : « GRANT ALL ON *.* TO 'test'@'%' » : attribution de tous les privilèges sur toutes les bases du serveur ;
- lignes 12 à 14 : le compte test se connecte depuis 172.17.12.45, exécute SHOW TABLES puis « DROP TABLE Contrat », conséquence directe de l'excès de droits.

b) Éléments dans les habilitations (document A3) :

- ligne ndv_64784, hôte « % » : connexion possible depuis n'importe quelle adresse IP, et non uniquement depuis localhost ;
- privilèges « ALL PRIVILEGES » sur la base ndv, alors que le besoin se limite à SELECT, INSERT, UPDATE et DELETE ;
- option « Grant » à « Oui » : l'utilisateur peut transmettre ses droits à d'autres comptes, ce que confirme la création du compte test dans les journaux.

c) Requêtes de correction (exécutées par un compte administrateur) :

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'ndv_64784'@'%';
RENAME USER 'ndv_64784'@'%' TO 'ndv_64784'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE ON ndv.* TO 'ndv_64784'@'localhost';
```

Complément attendu : suppression du compte créé le 19 novembre 2024.

```
DROP USER 'test'@'%';
```

Mission A2 – Renforcement de la protection des données personnelles de l'intranet

Question A2.1

a) Rappeler les critères qui caractérisent une donnée à caractère personnel et une donnée sensible au sens de la CNIL. b) Lister, s'il y en a, les données à caractère personnel et les données sensibles contenues dans les tables de la base de données de l'intranet.

a) Une donnée à caractère personnel est toute information se rapportant à une personne physique identifiée ou identifiable, directement (nom, prénom) ou indirectement (identifiant, numéro, adresse électronique, numéro de téléphone, etc.).

Une donnée sensible est une donnée révélant l'origine raciale ou ethnique, les opinions politiques, les convictions religieuses ou philosophiques, l'appartenance syndicale, une donnée génétique, une donnée biométrique aux fins d'identifier une personne, une donnée concernant la santé, la vie sexuelle ou l'orientation sexuelle d'une personne. Son traitement est en principe interdit, sauf exceptions.

b) Données à caractère personnel (document A7) :

- ContactExterne : nom, prenom, adresseMel, telephone ;
- Collaborateur : matricule, nom, prenom, adresseMel, telephone ;
- OrganisationCliente : nomPrenomDirigeant, et adresseMel et telephone lorsqu'ils concernent une personne physique ;
- Contrat : idReferent et idResponsable, qui relient un contrat à une personne physique.

Données sensibles : aucune n'est présente dans les tables du schéma.

Question A2.2

Indiquer si le chiffrement des données à caractère personnel relève d'une obligation légale ou d'une recommandation (bonne pratique).

Le chiffrement des données à caractère personnel relève d'une recommandation (bonne pratique) et non d'une obligation précise. L'article 32 du RGPD impose une sécurité du traitement appropriée au risque et cite le chiffrement parmi les mesures possibles (« selon les besoins »). Le chiffrement est donc un moyen que le responsable du traitement retient après analyse des risques, sans être imposé en tant que tel.

Question A2.3

Proposer un autre scénario de risque, impliquant un acteur interne dont vous préciserez le rôle, contre lequel le chiffrement des adresses de courrier électronique stockées dans la base serait une protection.

Scénario : un collaborateur de l'agence NDV, par exemple un développeur ou un stagiaire ayant un accès en lecture à la base pour la maintenance, exporte la table ContactExterne et revend les adresses à un tiers, ou les utilise pour envoyer des courriels frauduleux à des clients.

Protection apportée : les adresses étant chiffrées avec AES, l'acteur qui n'a accès qu'à la base ne peut pas les lire sans la clé de chiffrement, qui est conservée dans la configuration de l'application.

Question A2.4

Écrire les requêtes permettant d'adapter la base de données et de chiffrer les adresses de courrier électronique des contacts externes actuellement stockées dans la base de données.

Sauvegarder la table ContactExterne avant la modification.

```
ALTER TABLE ContactExterne MODIFY adresseMel VARBINARY(200);
UPDATE ContactExterne SET adresseMel = AES_ENCRYPT(adresseMel, 'lhh4&abt2!');
```

La modification de type doit précéder le chiffrement. Le résultat de AES_ENCRYPT est complété jusqu'à un multiple de 16 octets : la taille de 200 octets suffit pour des adresses de 191 caractères au plus.

Question A2.5

Corriger la requête préparée dans la fonction getContact() pour rétablir l'affichage de l'adresse de courrier électronique en clair.

Ligne 4 (requête corrigée) :

```
$sql = "SELECT ce.nom, ce.prenom,
    AES_DECRYPT(ce.adresseMel, :key) AS adresseMel, ce.telephone
    FROM Contrat c
    JOIN ContactExterne ce ON c.idReferent = ce.id
    WHERE c.id = :id";
```

Ajout après la ligne 6 (liaison de la clé) :

```
$stmt->bindValue(':key', $_ENV['AES_key'], PDO::PARAM_STR);
```

Mission A3 – Évolution de l'application de gestion des contrats pour sécuriser les échanges

Question A3.1

Réaliser les modifications nécessaires du schéma de la base ndv afin d'intégrer les nouveaux besoins. Seuls les éléments du schéma existant concernés par l'évolution seront repris, dans le formalisme de votre choix.

Ajout d'une table de référence des rôles :

RoleGenerique (id, libelle)
Clé primaire : id

Ajout d'une table d'affectation, qui associe pour chaque contrat et chaque rôle une personne de l'agence et son homologue chez le client :

Participation (idContrat, idRole, matricule, idContactExterne)
Clé primaire : (idContrat, idRole)
Clés étrangères : idContrat en référence à id de Contrat ; idRole en référence à id de RoleGenerique ; matricule en référence à matricule de Collaborateur ; idContactExterne en référence à id de ContactExterne

Contrainte à traduire par un déclencheur ou un contrôle applicatif : le contact idContactExterne doit appartenir à l'organisation cliente du contrat, soit ContactExterne.idOrganisation égal à Contrat.idClient.

La même personne peut figurer dans plusieurs lignes de Participation pour un même contrat, puisque la clé primaire porte sur le couple (idContrat, idRole).

Relation Contrat modifiée :

Contrat (id, reference, label, dateCreation, idClient)
Clé primaire : id
Clé étrangère : idClient en référence à id de OrganisationCliente

Les attributs idReferent et idResponsable sont supprimés : le chef de projet de chaque partie est désormais enregistré dans Participation avec le rôle « Chef de projet ».

Cette suppression impose d'adapter les requêtes existantes, par exemple getContact(), qui devra rechercher le contact via Participation avec le rôle « Chef de projet ». Une variante acceptée consiste à conserver idReferent et idResponsable dans Contrat et à ne gérer dans Participation que les rôles supplémentaires.

DOSSIER B – REPRISE ET SÉCURISATION DU SITE INTERNET DU CLIENT ARCHITECT.60 — 38 points

Mission B1 – Affichage des documents clients sur leur espace dédié

Question B1.1

a) Écrire chaque requête exécutée par le serveur de base de données suite à l'injection des charges utiles payload1 et payload2. b) Indiquer les affichages attendus dans le navigateur lors des injections des charges utiles payload1 et payload2. c) Expliquer pourquoi il n'est pas nécessaire que l'utilisateur 'sbegard' existe pour que ces deux charges utiles fonctionnent.

a) payload1 : la partie située après # est en commentaire (dont la vérification du mot de passe).

```
WHERE username ='sbegard' OR 1=1 #' AND password ='hash du mot de passe saisi'
```

La requête réellement exécutée se termine donc par : WHERE username ='sbegard' OR 1=1

payload2 : la partie située après # est également en commentaire.

```
WHERE username ='sbegard' AND 1=0
UNION SELECT null, username, password, null, null FROM T_Users #' AND password ='...'
```

b) payload1 : la condition OR 1=1 est vraie pour toutes les lignes. Le titre affiché est « Bienvenue sur votre espace sbegard' OR 1=1 # ! » (la valeur saisie est reprise telle quelle), puis un tableau contenant les trois documents de la table T_Docs, ceux de MonsieurT et ceux de LambertM : Croquis_rdc / Croquis qui présente la décoration du rdc / rdc.jpg ; Croquis_etage / Croquis qui présente la décoration de l'étage / etage.jpg ; Plan_maison / Plan total de la maison / maison.jpg, chacun avec un lien « Modifier ».

payload2 : la première partie ne retourne aucune ligne (1=0) ; seules les lignes de l'UNION sont renvoyées, dans l'ordre des colonnes id_user, doc_name, doc_description, doc_id, doc_file. La page affiche « Bienvenue sur votre espace sbegard' AND 1=0 UNION SELECT null, username, password, null, null FROM T_Users # ! » puis un tableau dont la colonne « Nom doc. » contient les identifiants (MonsieurT, LambertM) et la colonne « Description doc. » les mots de passe hachés en SHA-1 de la table T_Users (document B3).

Remarque : la colonne doc_id valant null, une lecture stricte de la ligne 19 conduirait à l'alerte « Identifiant et/ou mot de passe invalide ! ». L'énoncé présentant cette charge utile comme une preuve d'exploitation, on retient l'affichage des identifiants et des hachés. Il suffirait d'ailleurs de placer une valeur non nulle en quatrième position (UNION SELECT null, username, password, 1, null FROM T_Users #) pour franchir ce test.

c) La condition WHERE username = 'sbegard' est court-circuitée : avec OR 1=1, la condition est vraie pour toute ligne quelle que soit la valeur de username ; avec AND 1=0 suivi d'un UNION, c'est la seconde requête qui fournit les lignes, indépendamment du nom saisi. De plus, la vérification du mot de passe (AND password = ...) est placée après le caractère # et n'est jamais évaluée.

Question B1.2

a) Proposer une charge utile (payload) qui a pu être utilisée par les attaquants pour vider la table des documents. b) Indiquer quelle faille de sécurité dans le code du formulaire de connexion form_connexion.php a pu permettre aux attaquants de connaître le nom de la table visée par l'attaque.

a) Charge utile :

```
sbegard'; DELETE FROM T_Docs #
```

La fermeture du guillemet suivie du point-virgule termine la requête SELECT, puis une seconde requête DELETE est exécutée ; le reste est commenté par #. PDO avec MySQL accepte par défaut les requêtes multiples. Une variante équivalente est sbegard'; TRUNCATE TABLE T_Docs #.

b) Le commentaire HTML de la ligne 1 du formulaire form_connexion.php, « <!-- Accès à l'espace client et aux documents de la table T_Docs --> », est visible dans le code source de la page. Il expose le nom de la table : c'est une fuite d'information, liée à la mauvaise configuration et à l'exposition de détails d'implémentation côté client.

Question B1.3

a) Indiquer ce qui rend possible les injections SQL dans ce code. b) Modifier le code exécuté pour empêcher les injections SQL. (Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés).

a) Les variables $username et $password, issues directement de $_POST puis de $_SESSION, sont concaténées dans la chaîne SQL (lignes 12 à 15) sans requête préparée ni échappement. Seule la variable $username est exploitable, car $password est transformée par SHA1() à la ligne 7 avant d'être insérée dans la requête. Le caractère guillemet simple de la saisie modifie donc la structure de la requête. PDO en mode émulation autorise en outre les requêtes multiples.

b) Lignes 12 à 15 (requête paramétrée) :

```
$sql = "SELECT id_user, doc_name, doc_description, doc_id, doc_file
    FROM T_Users LEFT JOIN T_Docs ON T_Users.id_user = T_Docs.doc_id_user
    WHERE username = :username
    AND password = :password";
```

Ajout après la ligne 16 (liaison des paramètres) :

```
$select->bindValue(':username', $username, PDO::PARAM_STR);
$select->bindValue(':password', $password, PDO::PARAM_STR);
```

Ligne 19 (condition corrigée : parenthèse fermante ajoutée et clé entre guillemets) :

```
if ((count($lesDocuments) > 0) && ($lesDocuments[0]['doc_id'] != null)) {
```

Lignes 56 et 57 (mise à jour paramétrée, même risque d'injection) :

```
$sql = "UPDATE T_Docs SET doc_name = :docname, doc_description = :docdescription
    WHERE doc_id = :docid";
$updateDoc = $bdd->prepare($sql);
$updateDoc->bindValue(':docname', $docname, PDO::PARAM_STR);
$updateDoc->bindValue(':docdescription', $docdescription, PDO::PARAM_STR);
$updateDoc->bindValue(':docid', $docid, PDO::PARAM_INT);
$updateDoc->execute();
```

Mission B2 – Modification des documents clients sur leur espace dédié

Question B2.1

a) Identifier, dans le projet OWASP TOP Ten 2021, la faille de sécurité décelée. b) Modifier le code source du fichier espace_client.php pour corriger la faille de sécurité. (Indiquer seulement le code ajouté et/ou modifié en renseignant les numéros de ligne concernés).

a) A01:2021 – Contrôles d'accès défaillants : l'utilisateur peut agir sur une donnée (document d'un autre client) sans en avoir le droit. Il s'agit d'une référence directe non sécurisée à un objet (IDOR).

b) Ajout après la ligne 55 (contrôle d'appartenance du document au client connecté) :

```
$documentDuClient = false;
foreach ($lesDocuments as $document) {
    if ($document['doc_id'] == $docid) {
        $documentDuClient = true;
    }
}
```

Lignes 56 et 57 (exécution conditionnée) :

```
if ($documentDuClient) {
    $sql = "UPDATE T_Docs SET doc_name = :docname, doc_description = :docdescription
        WHERE doc_id = :docid";
    $updateDoc = $bdd->prepare($sql);
    $updateDoc->bindValue(':docname', $docname, PDO::PARAM_STR);
    $updateDoc->bindValue(':docdescription', $docdescription, PDO::PARAM_STR);
    $updateDoc->bindValue(':docid', $docid, PDO::PARAM_INT);
    $updateDoc->execute();
} else {
    echo '<script>alert("Modification non autorisée !");</script>';
}
```

Le tableau $lesDocuments ne contient que les documents de l'utilisateur connecté : toute valeur de docid absente de ce tableau est refusée.

Mission B3 – Création des comptes clients pour l'accès à leur espace dédié

Question B3.1

a) Identifier l'algorithme de hachage utilisé dans le processus de création des comptes clients et indiquer pourquoi il n'est pas efficace. b) Identifier, dans le projet OWASP TOP Ten 2021, la faille de sécurité décelée liée à cet algorithme de hachage.

a) L'algorithme utilisé est SHA-1 (ligne 10 de traitement_inscription.php : $newpasshash = SHA1($newpass);). Il n'est pas efficace pour stocker des mots de passe : c'est une fonction de hachage rapide, non conçue pour cet usage, sans sel ni coût de calcul. Il est de plus vulnérable aux collisions (document B5). Des attaques par force brute ou par tables précalculées permettent donc de retrouver rapidement les mots de passe.

b) A02:2021 – Défaillances cryptographiques : utilisation d'un algorithme de chiffrement ou de hachage obsolète, qui expose des données sensibles. La faille A07:2021 – Identification et authentification de mauvaise qualité peut également être citée, en lien avec l'usurpation des comptes.

Question B3.2

a) Modifier le code source lié à la création des comptes clients (fichier traitement_inscription.php) pour implémenter cette solution. (Indiquer seulement les lignes ajoutées et/ou modifiées.) b) Modifier le code source lié à l'affichage des documents du client dans le fichier espace_client.php pour prendre en compte ces changements. (Indiquer seulement les lignes ajoutées et/ou modifiées.)

a) Lignes 9 et 10 (remplacées) :

```
// On hache le mot de passe avec l'algorithme Argon2i
$newpasshash = password_hash($newpass, PASSWORD_ARGON2I);
```

La ligne 18 reste inchangée. La colonne password de T_Users, de type varchar(255) (document B3), peut déjà stocker le haché produit par Argon2i (environ 95 caractères) : aucune modification de la base n'est nécessaire.

b) Lignes 6 à 9 (remplacées) : vérification du mot de passe par password_verify, et non plus par comparaison de hachés dans la requête :

```
if (isset($_POST['login'])) {
    $sqlAuth = "SELECT password FROM T_Users WHERE username = :username";
    $auth = $bdd->prepare($sqlAuth);
    $auth->bindValue(':username', $_POST['username'], PDO::PARAM_STR);
    $auth->execute();
    $compte = $auth->fetch(PDO::FETCH_ASSOC);
    if ($compte && password_verify($_POST['password'], $compte['password'])) {
        $_SESSION['username'] = $_POST['username'];
    } else {
        $_SESSION['username'] = null;
    }
}
```

Lignes 10 et 11 : la ligne $password = $_SESSION['password']; est supprimée, car le mot de passe n'est plus stocké en session.

```
$username = $_SESSION['username'] ?? null;
```

Lignes 12 à 15 (le mot de passe n'est plus comparé dans la requête) :

```
$sql = "SELECT id_user, doc_name, doc_description, doc_id, doc_file
    FROM T_Users LEFT JOIN T_Docs ON T_Users.id_user = T_Docs.doc_id_user
    WHERE username = :username";
```

Ajout après la ligne 16 :

```
$select->bindValue(':username', $username, PDO::PARAM_STR);
```

Ligne 19 : inchangée par rapport à la correction de B1.3.

```
if ((count($lesDocuments) > 0) && ($lesDocuments[0]['doc_id'] != null)) {
```

Les comptes existants, dont le mot de passe est haché en SHA-1, ne sont pas vérifiables par password_verify : ils doivent être réinitialisés.

TABLEAU RÉCAPITULATIF DU BARÈME — 80 points

Dossier	Points
DOSSIER A – Évolutions de l'intranet de l'agence NDV en matière de sécurité	42
DOSSIER B – Reprise et sécurisation du site internet du client Architect.60	38
TOTAL	**80**

Continue à t'entraîner

Retrouve tous les sujets et corrigés du BTS SIO, classés par épreuve et par année :
coursbtssio.fr/annales-sujets-et-corriges

Pour réviser tout le programme avant l'examen, les Fiches de Révision du BTS SIO reprennent l'essentiel de chaque épreuve :
coursbtssio.fr/fiches-de-revision

Copyright © 2027 FormaV®. Tous droits réservés.

FormaV® est une marque déposée auprès de l'INPI sous le numéro 5227897.

Ce document a été élaboré par FormaV® avec le plus grand soin afin d'accompagner chaque apprenant vers la réussite de ses examens. Son contenu (textes, graphiques, méthodologies, tableaux, exercices, concepts, mises en forme) constitue une œuvre protégée par le droit d'auteur.

Toute copie, partage, reproduction, diffusion ou mise à disposition, même partielle, gratuite ou payante, est strictement interdite sans accord préalable et écrit de FormaV®, conformément aux articles L.111-1 et suivants du Code de la propriété intellectuelle. Dans une logique anti-plagiat, FormaV® se réserve le droit de vérifier toute utilisation illicite, y compris sur les plateformes en ligne ou sites tiers.

En utilisant ce document, vous vous engagez à respecter ces règles et à préserver l'intégrité du travail fourni. La consultation de ce document est strictement personnelle.

Merci de respecter le travail accompli afin de permettre la création continue de ressources pédagogiques fiables et accessibles.