Bloc de compétences 3 – Assurer la cybersécurité d'une solution applicative et de son développement option SLAM	**Année 2**
Finalité métier : Le métier de la cybersécurité consiste à répondre aux besoins de cybersécurité dans le développement de solutions applicatives d'une organisation notamment au regard du développement des menaces et attaques en provenance d'internet et des risques liés aux usages numériques. Les contextes de travail, ouverts et évolutifs, nécessitent de mener une veille informationnelle et technologique et de prendre en compte leurs aspects humains, technologiques, organisationnels, économiques et juridiques. Le métier implique de respecter la réglementation, les méthodes, les normes et standards qui prévalent dans la législation nationale et internationale ainsi que dans les recommandations des organismes gouvernementaux et professionnels.	**Contexte professionnel** : Vous travaillez pour le compte d'une société de conseil en technologies spécialisée dans la cybersécurité. Rattaché (e) au responsable de la sécurité des systèmes d'information (RSSI), vous êtes chargé(e) de la mise en œuvre de la cybersécurité dans le développement de solutions applicatives des différents clients. Vous serez chargée ou chargé de : - Participer à la vérification des éléments contribuant à la qualité d'un développement informatique - Prendre en compte la sécurité dans un projet de développement d'une solution applicative - Mettre en œuvre et vérifier la conformité d'une solution applicative et de son développement à un référentiel, une norme ou un standard de sécurité - Prévenir les attaques - Analyser les connexions (logs) - Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures

3.5 B. Assurer la cybersécurité d'une solution applicative et de son développement	**Année 2**
Votre mission : Vous participez aux activités de cybersécurité dans le développement de solutions applicatives de vos clients et vous êtes chargé.e de : - Évaluer les menaces d'une application et en estimer les risques associés - Contrer les menaces évaluées d'une application pour en réduire les	**Contexte professionnel** : Vous travaillez pour le compte d'une entreprise de services du numérique (ESN) qui apporte son expertise auprès de ses clients en matière de cybersécurité dans le développement de logiciels dès le stade de la conception de la solution applicative jusqu'à sa livraison.

vulnérabilités - Respecter les exigences de sécurité légales, réglementaires ou métier, ou de bonnes pratiques internes à l'organisation - Appliquer les mesures d'assurance sécurité dans toutes les phases du projet et du développement - Vérifier les bonnes pratiques de codage - Identifier les bugs de sécurité d'une application et les corriger	

Séquence **3.5 A2D**	**Sensibiliser les développeurs juniors à la sécurité des applications web**			
Durée totale en heures du scénario pour la séquence 9 h	Votre entreprise cliente a recruté de nouveaux développeurs juniors. Elle souhaite les sensibiliser aux mécanismes de différentes failles de sécurité des applications web afin qu'ils intègrent très tôt les bonnes pratiques de sécurité quand ils développent une application. Afin de répondre à ce besoin, vous préparez des tests d'intrusion sur une application web vulnérable que vous présenterez lors d'un webinaire. Vous vous appuyez sur le guide de tests proposés par l'OWASP (Open Web Application Security Project - https://owasp.org/www-project-top-ten/). Les sites https://www.exploit-db.com/ et www.xss-payloads.com peuvent être également source d'inspirations d'exploits sur des codes vulnérables. Les scénarios de tests de sécurité sont conçus en collaboration avec votre collègue chef de projet de l'équipe sécurité. Ces tests permettront aux développeurs de votre client de découvrir les failles de sécurité possibles, les moyens de les éviter et les outils pour les détecter. ***Note aux auteurs :*** *Cette séquence a pour objectif de consolider et d'approfondir les acquis de la séquence 3.4 B2 étudiée en 1ère année. Elle a été réalisée avec l'application vulnérable DVWA (Damn Vulnerable Web Application - http://www.dvwa.co.uk/). L'application a été clonée depuis son dépôt Github et installée sur une machine virtuelle. (Elle est également disponible sous forme d'image ISO, de machine VirtualBox/Vmware ou image Docker).* *La séquence peut être également réalisée avec la plateforme OWASP Mutillidae II ou metasploitable 2 en s'appuyant sur les productions OWASP rédigées par Patrice Dignan pour le réseau Certa. Toute autre application web comportant les failles de sécurité abordées peut également être utilisée.*			
	Compétences travaillées	**Savoirs associés**	**Indicateurs de performance**	**Prérequis / Transversalités**
	• Prévenir les attaques	Savoirs technologiques	• Le respect des bonnes pratiques de développement informatique est vérifié (les structures de	Prérequis : Bloc 3 – 1ère année – Séquence 3.4B2

	• Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures	• Développement informatique : méthodes, normes, standards et bonnes pratiques • Sécurité du développement d'application : gestion de projet, architectures logicielles, rôle des protocoles, authentification, habilitations et privilèges des utilisateurs, confidentialité des échanges, tests de sécurité, audit de code • Sécurité des applications *Web* : risques, menaces et protocoles	données sont normalisées, les accès aux données sont optimisés, le code est modulaire et robuste, les tests sont effectués) • Des tests de sécurité sont prévus et mis en œuvre • La communication écrite et orale est adaptée à l'interlocuteur.	

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h	Le chef de projet de l'équipe sécurité vous a fourni un dossier documentaire et la machine virtuelle de test (distribution Kali). Il vous demande de préparer et de tester l'environnement de formation. 1. Importer la machine virtuelle de test 2. Cloner l'application DVWA depuis son dépôt Git 3. Installer puis configurer l'application vulnérable fournie par votre RSSI 4. Vérifier les éléments de configuration recommandés	• Machine virtuelle de test[1] qui hébergera l'application web vulnérable, le service web et le serveur de base de données MySQL/MariaDB. • Dossier documentaire comportant la description des ressources utilisées, les éléments d'installation et de configuration recommandés[2], les accès aux différents services et une description de l'application. *Remarque* : *le lien suivant permettrait aux étudiants autonomes d'installer et de configurer l'application DVWA sur une machine kali Linux sans les précisions du dossier documentaire :*	• La machine virtuelle est importée dans l'environnement de test. • L'accès à la base de données de l'application est opérationnel. • L'application vulnérable est bien configurée et démarre correctement.

[1] Appelée dorénavant serveur web.
[2] Plusieurs tutoriels et vidéo youtube traitent de l'installation de DVWA, notamment sur Kali

	5. Configurer et tester le lancement de l'application web et l'accès à la base de données utilisée par l'application.	https://www.kalilinux.in/2020/01/setup-dvwa-kali-linux.html	
Séance 2	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	Le chef de projet de l'équipe sécurité a rédigé des plans de tests de vulnérabilités pour les besoins de la démonstration aux développeurs. Cette séance est consacrée aux injections SQL les moins connues. Différents niveaux de sécurité seront proposés 1. Effectuer les scénarios de tests fournis **2.** Compléter le rapport de tests fourni **3.** Etudier le comportement de l'application et le code correspondant pour chaque niveau de sécurité **4.** Documenter, dans un tableau de synthèse, les failles de sécurité testées, les risques identifiés et les mesures de codage sécurisé à mettre en place pour s'en prémunir.	• Fiches de savoirs techniques des failles de sécurité des applications web (*injections SQL avancées*) • Machine virtuelle hébergeant l'application vulnérable • Plan de tests d'injections SQL (*si l'application DVWA est utilisée, les tests devront être effectués pour les différents niveaux de sécurité proposés, sinon s'en inspirer[3] pour amener progressivement l'étudiant à atteindre la meilleure pratique pour se protéger contre les injections SQL*). • Rapport de tests de sécurité à compléter • Tableau de synthèse des vulnérabilités à compléter (cible de la menace, niveau du risque, technique d'attaque, contre-mesure…)	• Les tests de sécurité sont effectués pour analyser les vulnérabilités de l'application conformément au plan de tests. • Le rapport de tests est complété. • Les vulnérabilités trouvées et les mesures de codage pour les contrer sont documentées dans le tableau de synthèse qui sera livré au client.
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	Le chef de projet de l'équipe sécurité souhaite également que vous intégriez dans votre démonstration les attaques	• Fiches de savoirs techniques des failles de sécurité des applications web (*XXE – XML eXternal Entities, XEE – XML Entity Expansion)*	• Les tests d'attaques XXE et XEE sont réalisés

[3] Il est également possible de consulter les CVE (Common Vulnerabilities and Exposures) pour d'autres idées d'injections SQL

	XXE et XEE assez méconnues par les développeurs. 1. Effectuer une recherche sur Internet pour étudier ce type d'attaques et leur impact sur le serveur web 2. Concevoir un test d'attaque XXE 3. Documenter cette faille de sécurité dans le document de synthèse 4. Concevoir un test d'attaque XEE 5. Documenter cette faille de sécurité dans le document de synthèse	• Machine virtuelle hébergeant l'application vulnérable • Tableau de synthèse des vulnérabilités à compléter (cible de la menace, niveau du risque, technique d'attaque, contre-mesure…)	• Les vulnérabilités XXE et XEE sont documentées en termes d'impacts et de contre-mesures possibles dans le tableau de synthèse.
Séance 4	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	Afin de vous aider dans la préparation de vos démonstrations, le chef de projet de l'équipe sécurité vous fournit des plans de tests qui ciblent les vulnérabilités dues aux attaques XSS (Cross Site Scripting ou scripts intersites) qu'elles soient de type stockées (Stored XSS), non permanentes (reflected XSS) ou basées sur le DOM (*Document Object Model – DOM based XSS)*. (*L'application DVWA permet de tester ces injections de code avec des niveaux de sécurité différents allant du niveau le plus faible au plus sécurisé*).	• Fiches de savoirs techniques des failles de sécurité des applications web (*attaques XSS – Cross Site Scripting – persistantes, non persistantes et basées sur le DOM – Document Object Model*), voir https://owasp.org/www-project-top-ten/ et les productions OWASP de Patrice Dignan sur le réseau Certa • Machine virtuelle hébergeant l'application vulnérable • Plans de tests exploitant les failles de sécurité XSS dans l'application • Tableau de synthèse des vulnérabilités à compléter (cible de la menace, niveau du risque, technique d'attaque, contre-mesure…)	• Les failles de sécurité XSS sont testées sur l'application vulnérable. • Les failles de sécurité XSS et les mesures de codage nécessaires pour s'en prémunir sont documentées dans le tableau de synthèse livrable au client.

	1. Réaliser les plans de tests d'attaques XSS stockées, non persistantes ou basées sur le DOM 2. À partir des résultats de tests et de vos recherches sur Internet, mettre à jour le document de synthèse avec chacune des failles de sécurité XSS, des exemples de scénario d'attaque et les bonnes pratiques de codage à appliquer pour les contrer.		
Séance 5	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	Le chef de projet de l'équipe sécurité vous suggère également de sensibiliser les développeurs de votre client sur les menaces liées au téléchargement de fichiers et aux dysfonctionnements qu'elles peuvent provoquer sur les serveurs web. 1. Dérouler les scénarios d'injection de type *upload* fournis (*pour les différents niveaux de sécurité proposés par l'application*) 2. Réaliser des captures d'écran des résultats de tests obtenus 3. Documenter cette faille de sécurité dans le document de synthèse et préciser les mesures à prendre en termes de codage et de configuration du serveur web pour éviter ce type de menaces.	• Fiches de savoirs techniques des failles de sécurité des applications web (*injections de scripts malicieux)* • Différents scénarios de tests (*productions OWASP du réseau Certa, tutoriels ou vidéos d'injection de code malicieux sur DVWA disponibles sur internet peuvent être proposés)* • Exemples de scripts PHP malicieux. • Tableau de synthèse des vulnérabilités à compléter (cible de la menace, niveau du risque, technique d'attaque, contre-mesure…)	• Les scénarios de pénétration présentant les failles de sécurité au téléchargement de fichiers malicieux sont exécutés sur l'application de test. • Les codes sources correspondants sont étudiés et analysés. • La faille de sécurité ainsi que les mécanismes de sécurité à appliquer pour s'en prémunir sont documentés dans le tableau de synthèse livrable au client.

<table>
<tr><td>Séquence 3.5 B2D</td><td colspan="4">Réaliser un audit de sécurité basé sur les tests d’intrusions d’une application web</td></tr>
<tr><td rowspan="2">Durée totale en heures du scénario pour la séquence

18 h</td><td colspan="4">Votre client, petite entreprise, a fait développer il y a quelques années une application web par un prestataire externe. Depuis peu, certains internautes ont signalé sur les commentaires que l’accès au site leur retournait une erreur mentionnant des soucis de sécurité. Conscient du préjudice que cela pourrait avoir sur son entreprise, le gérant de l’entreprise cliente vous demande d’effectuer un audit de sécurité de son application web.

Vous effectuez les tests d’intrusion de l’application de votre client sous la responsabilité du chef de projet de l’équipe sécurité de votre entreprise prestataire.</td></tr>
<tr><td>Compétences travaillées</td><td>Savoirs associés</td><td>Indicateurs de performance</td><td>Prérequis / Transversalités</td></tr>
<tr><td>• Participer à la vérification des éléments contribuant à la qualité d’un développement informatique
• Prévenir les attaques.
• Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures.</td><td><u>Savoirs technologiques</u>
• Développement informatique : méthodes, normes, standards et bonnes pratiques.
• Aspects réglementaires du développement applicatif : protection de la vie privée dès la conception, protection des données par défaut, sécurité par défaut, droit des individus.
• Sécurité du développement d’application : gestion de projet, architectures logicielles, rôle des protocoles, authentification, habilitations et privilèges des utilisateurs, confidentialité des échanges, tests de sécurité, audit de code.
• Vulnérabilités et contre-mesures sur les problèmes courants de développement.</td><td>• Les bonnes pratiques de sécurité sont mises en œuvre à toutes les étapes d’un développement informatique.
• Des tests de sécurité sont prévus et mis en œuvre.
• L’accès aux données respecte les règles de sécurité.
• Les échanges de données entre applications sont protégés.
• Les contre-mesures mises en place corrigent et préviennent les incidents de sécurité.
• Les contre-mesures sont documentées de manière à en assurer le suivi.
• La communication écrite et orale est adaptée à l’interlocuteur.</td><td><u>Prérequis</u> :

Compétence 1.3 – 1ère année

Bloc 3 – 1ère année

Séquence 3.5 - A2D

<u>Transversalités</u> :

Bloc 2.1 / 2.2 – 2ème année</td></tr>
</table>

		• Outils de contrôle de la sécurité : plans de secours, traçabilité et audit technique. Outils de contrôle de la sécurité : plans de secours, traçabilité et audit technique. <u>Savoirs économiques, juridiques et managériaux</u> • Les risques des cyberattaques pour l'organisation : économique, juridique, atteinte à l'identité de l'entreprise. • Obligations légales de notification en cas de faille de sécurité. • Responsabilité du concepteur de solutions applicatives.		

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h 30	La portée des prestations des tests d'intrusion est différente d'un client à un autre. Afin d'accompagner au mieux ses auditeurs de sécurité (*pentesters)* dans l'étude d'un contrat de prestation, le service juridique de votre entreprise a réalisé une liste des principaux points de vigilance auxquels il faut être attentif. Pour les besoins d'audit interne, ceux-ci sont recensés dans un questionnaire à remplir avant la signature du contrat. En tant que personne chargée des tests d'intrusion pour auditer les vulnérabilités	• Fiches de savoirs juridiques des obligations légales de l'auditeur de sécurité (*pentester*) • Contrat de prestation de tests d'intrusion (*pentest*) en étude entre votre entreprise et votre client • Infrastructure applicative • Liste des points de vigilance à compléter (*contexte global du déroulement de la mission, descriptif des biens sensibles du SI à auditer, responsables du projet, périmètre technique de l'intrusion, responsabilités, méthodologie…)* • Questionnaire à remplir	• Le contrat de prestation du test d'intrusions est analysé par l'auditeur de sécurité. • Le périmètre technique d'intervention est délimité et soumis à validation. • La liste des vérifications est complétée. Les éléments manquants sont signalés. • Le questionnaire lié à l'analyse et à la compréhension du contrat est complété.

	potentielles et hypothétiques du système de votre client, vous devez prendre connaissance du contrat. 1. Étudier les termes du contrat de la prestation 2. Relever les responsabilités qui vous incombent en tant que personne responsable des tests d'intrusion 3. Relever les risques juridiques éventuels liées à vos opérations de tests d'intrusion 4. Vérifier la liste des points de vigilance et, pour chacun d'eux, intégrer une capture d'écran de la partie du contrat qui le traite 5. Compléter le questionnaire fourni par le service juridique 6. Analyser l'infrastructure applicative et délimiter les éléments concernés par votre intervention. 7. Faire valider le périmètre du test d'intrusion auprès de votre client.		
Séance 2	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h 30	La responsable RSSI vous demande de préparer la machine qui servira pour effectuer les tests d'intrusions au sein de l'environnement de votre client.	• Fiche de savoirs technologiques des outils d'audit et de tests d'intrusion et de sécurité (tutoriels, vidéos…) • Infrastructure applicative (*comportant les VM et un schéma réseau)* • Dossier technique numérique comportant :	• La machine virtuelle de tests d'intrusion est installée et configurée conformément au dossier documentaire. • Les outils d'audit sont installés sur la machine virtuelle

	1. Importer la machine virtuelle vous permettant de réaliser les tests d'intrusion 2. Installer un scanner de vulnérabilités Web (*proxy ZAP développé par l'OWASP, Wfuzz, OpenVAS*…) 3. Installer un cadre applicatif (*framework*) de test d'intrusion 4. Installer un outil d'injection SQL et d'énumération de bases de données (*par exemple SQLMap*) Afin de simuler l'environnement du client, vous devez également importer l'infrastructure applicative de votre client.	○ la machine virtuelle de test d'intrusions ○ les éléments de configuration de la machine virtuelle ○ la description des outils d'audit à installer • Fiche méthodologique d'installation de l'environnement du client *<u>Note aux auteurs</u> : certaines machines virtuelles de tests d'intrusion disponibles sur internet peuvent comporter les scanners de vulnérabilités demandées ou des équivalents (par exemple Kali Linux). Dans ce cas-là, on fournira un descriptif des outils installés et leur rôle dans les tests d'intrusions.* *Outils de pentest :* <u>10 outils de pen test pour hackers éthiques - Le Monde Informatique</u>	• L'environnement technique du client est installé et prêt pour les tests d'intrusion
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	Parmi les besoins exprimés dans le contrat, le client souhaite une modélisation graphique des menaces détectées ou hypothétiques. La responsable RSSI a fait le choix d'un outil qui permet de modéliser les menaces à l'aide d'un diagramme de flux de données. 1. Explorer les fonctionnalités de cet outil	• Fiches de savoirs technologiques des typologies des menaces • Fiche de savoirs technologiques des outils de modélisation des menaces • Outil de modélisation des menaces et ressources pour son installation et sa prise en main. • Cas d'étude *Note aux auteurs : la séance a été testée avec l'outil de modélisation des menaces de Microsoft qui s'appuie sur le modèle STRIDE (Spoofing –*	• L'outil de modélisation des menaces est installé • Le diagramme des menaces du cas d'étude est créé • Le rapport de modélisation des menaces est généré dans le format souhaité.

	2. Installer l'outil en question 3. Créer un diagramme de flux de données en vous basant sur le cas d'étude fourni par votre RSSI 4. Générer le rapport complet de la modélisation des menaces	*Tampering – Repudiation – Information disclosure – Denial of service – Elevation of privilege). Les ressources utilisées sont :* *https://docs.microsoft.com/fr-fr/azure/security/develop/threat-modeling-tool* *https://docs.microsoft.com/fr-fr/azure/security/develop/threat-modeling-tool-getting-started* *https://docs.microsoft.com/fr-fr/learn/modules/tm-introduction-to-threat-modeling/* *https://docs.microsoft.com/fr-fr/learn/modules/tm-create-a-threat-model-using-foundational-data-flow-diagram-elements/*	
Séance 4	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**

3 h	Vous avez intégré l'entreprise cliente pour réaliser la prestation de test d'intrusions en appliquant la stratégie de la boîte grise (*pentest Greybox*).
	
	1ère étape de la méthodologie : collecte d'informations et énumération des actifs de l'application cible (phase de reconnaissance)
	
	1. Connecter votre machine sur le réseau de l'entreprise
		• Fiche de savoirs technologiques de la méthodologie et des stratégies de test d'intrusion (https://www.login-securite.com/2019/02/22/le-pentest-de-a-a-z-methodologie-et-bonnes-pratiques-pour-securiser-son-si/, https://blog.e-xpertsolutions.com/comprendre-la-modelisation-des-menaces/)
		• Fiche de savoirs technologiques des outils d'audit, de tests d'intrusion et de modélisation des menaces (tutoriels, vidéos…)
			• La machine de tests est connectée sur le réseau de l'entreprise cliente
			• Le rapport d'audit de sécurité est complété avec :
			  ○ les informations collectées
			  ○ le descriptif des actions menées pour les collecter
			  ○ et la liste des outils exploités.
			• Le rapport d'audit de sécurité est transmis de manière sécurisée au RSSI pour le suivi de la prestation.
			• Les résultats des différents scans sont stockés dans des fichiers conformément au contrat

	2. Identifier le serveur web et de base de données grâce à l'analyse de paquets 3. Lister les noms de domaine qualifiés du site web de votre client 4. Recenser tous les services en écoute sur le serveur web et le serveur de base de données grâce à l'analyse des ports réseau. Stocker le résultat de l'analyse dans un fichier 5. Recueillir des informations sur les serveurs (version d'OS, configuration TLS, liste des services hébergés…) 6. Compléter le rapport avec les informations collectées sur le serveur Web 7. Compléter le rapport avec les informations collectées sur le serveur de bases de données	• Fiches de savoirs sur les vulnérabilités des serveurs (voir fiches du bloc 3 – 1ère année) • Environnement technique du client • Serveur Web présentant des vulnérabilités (*services en écoute illégitimes, méthodes HTTP présentant des risques potentiels, mauvaises configuration, version ancienne de TLS…)* • Serveur de base de données présentant des vulnérabilités • Application web présentant des vulnérabilités • Exemple de rapport d'audit de sécurité • Rapport d'audit de sécurité à compléter	
Séance 5	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
4 h	**2ème phase de la méthodologie : recherche de vulnérabilités des actifs de l'application cible (phase de détection)** En vous appuyant sur les informations collectées : 1. Effectuer une analyse (*scan*) de vulnérabilités (OWASP ZAP par exemple) du serveur de base de	• Rapport d'audit de sécurité à compléter • Serveur Web présentant des vulnérabilités • Serveur de base de données présentant des vulnérabilités • Application web présentant des vulnérabilités *Note aux auteurs : Les services et l'application qui seront utilisés devront comporter des vulnérabilités qui permettront d'illustrer les*	• Les résultats des différents scans automatiques et manuels de vulnérabilités sont enregistrés dans des fichiers. • Le rapport est complété avec les différentes vulnérabilités détectées et potentielles des actifs ciblés. Pour chaque vulnérabilité, on indiquera les effets en termes de menaces, le type de la vulnérabilité en référence à des typologies comme la liste CWE (*Common Weakness Enumeration*), la probabilité de réalisation.

	données et enregistrer le résultat dans un fichier 2. Effectuer une analyse de vulnérabilités du serveur web et enregistrer le résultat dans un fichier 3. Réaliser une analyse de vulnérabilités de l'application 4. Recenser les vulnérabilités de l'application en les classant 5. Compléter cette analyse avec des recherches sur des sites qui recensent des vulnérabilités officielles (*Common Vulnerabilities and Exposures, Top 10 OWASP, CNIL, ANSSI…)* 6. Modéliser les menaces à l'aide d'un outil et générer le rapport correspondant	*différentes étapes du test d'intrusions exploitant la stratégie de la boîte grise (pentest Greybox*). *L'organisation MITRE publie sur son site deux listes de diffusion qui regroupent différentes vulnérabilités trouvées par les organisations et personnes de la sécurité informatique (CVE – Common Vulnerabilities and Exposures - https://cve.mitre.org/) et les failles et faiblesses dans la conception et l'architecture d'une application (CWE – Common Weakness Enumeration - https://cwe.mitre.org/). Ces deux listes pourraient être des sources d'inspiration des vulnérabilités à exploiter.*	• Les menaces sont modélisées • Le rapport de modélisation des menaces est généré.
Séance 6	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
4 h	**4ème phase de la méthodologie : exploitation des vulnérabilités des actifs de l'application cible** 1. Lancer les commandes de l'outil d'injection SQL (sqlmap) permettant d'exploiter les failles de sécurité liée au serveur de bases de données 2. Explorer la liste des sites qui référencent des *exploits* publics connus pour pirater un système. 3. Rechercher des exploits que vous pouvez exploiter pour réaliser des	• Liste des informations qu'on souhaite obtenir sur les bases de données via les tests de pénétration • Liste des sites de bases d'informations des exploits publics (*Exploit-db.com*) et de leur exploitation à travers des outils (*Metasploit, Searchsploit*) • Exemple de rapport de tests d'intrusion • Rapport de tests d'intrusion *Note aux auteurs : La catégorie « Free» du site https://pentesterlab.com/exercises/ peut être*	• Les tests d'intrusion sont effectués en respectant le périmètre contractuel d'intervention. • Les tests d'intrusion sont documentés (commandes, résultat, conclusion) dans un rapport. • L'environnement technique du client est nettoyé des manipulations des tests d'intrusion (*porte dérobée, utilisation de compte créé ou à privilège, modification des accès*…) qui pourraient compromettre davantage le système.

	tests d'intrusion dans l'application de votre client. 4. Effectuer les *exploits* correspondants aux vulnérabilités détectées dans l'application lors de la phase précédente **5.** Mettre à jour le rapport d'audit en précisant les vulnérabilités qui n'ont pu aboutir **6.** Renseigner le rapport de tests d'intrusions. Pour chaque type de vulnérabilité, préciser la commande ou *l'exploit* utilisé, le résultat obtenu et la conclusion du test. **7.** Remettre l'environnement technique du client dans son état d'origine.	*source d'inspirations pour exploiter certaines vulnérabilités.*	
Séance 7	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**5ème phase : présentation des correctifs** En vous appuyant sur les bonnes pratiques de codage sécurisé et sur votre veille active concernant les bibliothèques tierces et les protocoles utilisés : 1. Proposer des recommandations de sécurité permettant de corriger les vulnérabilités détectées 2. Mettre à jour le rapport de tests d'intrusion.	• Liste des rapports à livrer au client	• Le rapport de tests d'intrusion est mis à jour avec les recommandations de correctifs de sécurité adaptés, y compris en termes de choix d'architecture logicielle et de composants certifiés à privilégier ou à désactiver. • Le rapport de tests d'intrusion est transmis au RSSI et complété.

<table>
<tr><td>Séquence
3.5 C2D</td><td colspan="4">Analyser et corriger les vulnérabilités détectées à l’issue d’un audit de sécurité d’une application web</td></tr>
<tr><td>9 h</td><td colspan="4"><u>Rappel du contexte</u> : Votre client, petite entreprise, a fait développer il y a quelques années une application web par un prestataire externe. Depuis peu, certains internautes ont signalé sur les commentaires que l’accès au site leur retournait une erreur mentionnant des soucis de sécurité. Conscient du préjudice que cela pourrait avoir sur son entreprise, le gérant de l’entreprise vous demande d’effectuer un audit de son application web.

Suite au rapport de tests d’intrusions réalisé de la plateforme web, le client souhaite mettre en place les correctifs que vous lui avez recommandés. Il fait appel à nouveau aux services de votre entreprise.

Rattaché au chef de projet de l’équipe sécurité de votre entreprise, vous intervenez au sein de l’équipe technique de votre client pour développer les correctifs de sécurité et pallier les vulnérabilités détectées.</td></tr>
<tr><td></td><td>Compétences travaillées</td><td>Savoirs associés</td><td>Indicateurs de performance</td><td>Prérequis / Transversalités</td></tr>
<tr><td></td><td>• Participer à la vérification des éléments contribuant à la qualité d’un développement informatique.
• Mettre en œuvre et vérifier la conformité d’une solution applicative et de son développement à un référentiel, une norme ou un standard de sécurité
• Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures.</td><td><u>Savoirs technologiques</u>
• Développement informatique : méthodes, normes, standards et bonnes pratiques.
• Aspects réglementaires du développement applicatif : protection de la vie privée dès la conception, protection des données par défaut, sécurité par défaut, droit des individus.
• Sécurité du développement d’application : gestion de projet, architectures logicielles, rôle des protocoles, authentification, habilitations et privilèges des utilisateurs, confidentialité des échanges, tests de sécurité, audit de code.</td><td>• Le respect des bonnes pratiques de développement informatique est vérifié (les structures de données sont normalisées, les accès aux données sont optimisés, le code est modulaire et robuste, les tests sont effectués).
• Les préoccupations de sécurité sont prises en compte à toutes les étapes d’un développement informatique.
• Les bonnes pratiques de sécurité sont mises en œuvre à toutes les étapes d’un développement informatique.
• Des tests de sécurité sont prévus et mis en œuvre.</td><td><u>Prérequis</u> :
B3.5 – B2D
<u>Transversalités</u> :
B2.1 – B2.2 – 2ème année</td></tr>
</table>

		• Vulnérabilités et contre-mesures sur les problèmes courants de développement. <u>Savoirs économiques, juridiques et managériaux</u> • Responsabilité du concepteur de solutions applicatives. • Les risques des cyberattaques pour l'organisation : économique, juridique, atteinte à l'identité de l'entreprise. • Réglementation en matière de lutte contre la fraude informatique : infractions, sanctions	• Les échanges de données entre applications sont protégés Les contre-mesures mises en place corrigent et préviennent les incidents de sécurité. Les contre-mesures sont documentées de manière à en assurer le suivi.	

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
4 h	Le chef de projet de l'équipe sécurité a validé vos contre-mesures au niveau de l'application et complété le document par les mesures à prendre au niveau du serveur de base de données et des échanges entre les deux machines. 1. Analyser le rapport de tests d'intrusion et identifier les faux positifs 2. Mettre à jour le rapport de tests d'intrusion 3. Effectuer les corrections logicielles définitives nécessaires au code de l'application pour la sécuriser (*remplacement de composants par*	• Rapport de tests d'intrusion de la plateforme web • Diagramme de modélisation des menaces • Rapport de modélisation des menaces • Plateforme web.	• Les vulnérabilités présentant un haut risque sont corrigées. • Les accès au serveur de bases de données sont sécurisés et appliquent la stratégie du moindre privilège. • Les échanges entre le serveur web et le serveur de bases de données sont sécurisés.

	des composants certifiés, désactivation d'un module ou d'une fonctionnalité, codage sécurisé…) 4. Apporter les corrections nécessaires pour mieux sécuriser le serveur de bases de données et les échanges avec le serveur web.		
Séance 2	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	1. Réaliser un test post-audit des vulnérabilités les plus critiques après remédiation. 2. Réaliser un inventaire détaillé des outils et bibliothèques tierces utilisés pour le suivi des mises à jour.	• Machine de tests avec les outils d'audit • Plateforme web • Scénario du test post-audit fourni par la RSSI • Rapport de test post-audit à compléter	• Le test post-audit des vulnérabilités est réalisé selon le scénario établi. • Le rapport du post-audit des vulnérabilités présente uniquement des failles de sécurité résiduelles (acceptées au vu de faible niveau de leur impact). • L'inventaire des technologies utilisées (composants, outils, bibliothèques) est réalisé.
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
3 h	L'application a été déployée en pré-production. Des incidents de sécurité de niveau 2 ont été déclarés. Vous devez les traiter. L'analyse de ces incidents révèle des erreurs liées à la conception de la base de données. 1. Réaliser une rétro-conception de la base de données. 2. Corriger les vulnérabilités liées à la conception	• Tickets d'incidents de sécurité • Base de données de test • Application web	• La nouvelle base de données est modélisée. • Les vulnérabilités liées à la conception de la base de données sont corrigées. • Les corrections apportées sont testées dans l'environnement de test. • Le script de mise à jour des données dans l'environnement de pré-production est réalisé et testé avec des données fictives.

	3. Modéliser la nouvelle base de données 4. Tester les corrections dans l'environnement de test 5. Écrire un script qui permettra de mettre à jour les données dans l'environnement de pré-production 6. Pousser la nouvelle application sur le dépôt distant pour une nouvelle intégration/déploiement		

<table>
<tr><td>Séquence
3.5 D2D</td><td colspan="4">Participer au cycle de développement sécurisé d’une solution applicative</td></tr>
<tr><td>14 h</td><td colspan="4">Vous travaillez au sein de la direction du système d’information (DSI) d’une organisation cliente et vous intégrez l’équipe de développeurs chargée de la conception et du développement d’une application métier (architecture client lourd). Les équipes de développement de votre client travaillent en mode agile et intègrent les préoccupations de sécurité dès la phase de conception (security by design) sous la responsabilité d’une experte en sécurité informatique qui supervise les aspects sécuritaires du développement tout au long du cycle (Secure Software Development Life Cycle – S-SDLC,).

<u>Note aux auteurs</u> : cette séquence a été réalisée grâce aux ressources numériques trouvées sur les sites SAFECode (<u>https://safecode.org</u>, Fundamental Practices for Secure Software Development), Oracle (https://www.oracle.com/java/technologies/javase/seccodeguide.html, Secure Coding Guidelines for Java SE), IONOS (https://www.ionos.fr/digitalguide/sites-internet/developpement-web/quest-ce-que-le-fuzzing/), le livre blanc Cybersecurity by design de Thalès, <u>sécurité dans le pipeline CI/CD</u></td></tr>
<tr><td></td><td>Compétences travaillées</td><td>Savoirs associés</td><td>Indicateurs de performance</td><td>Prérequis / Transversalités</td></tr>
<tr><td></td><td>• Participer à la vérification des éléments contribuant à la qualité d’un développement informatique
• Prendre en compte la sécurité dans un projet de développement d’une solution applicative
• Mettre en œuvre et vérifier la conformité d’une solution applicative et de son développement à un référentiel, une norme ou un standard de sécurité.
• Prévenir les attaques</td><td><u>Savoirs technologiques</u>
• Développement informatique : méthodes, normes, standards et bonnes pratiques.
• Aspects réglementaires du développement applicatif : protection de la vie privée dès la conception, protection des données par défaut, sécurité par défaut, droit des individus.
• Sécurité du développement d’application : gestion de projet, architectures logicielles, rôle des protocoles, authentification, habilitations et privilèges des utilisateurs, confidentialité des échanges, tests de sécurité, audit de code.</td><td>• Le respect des bonnes pratiques de développement informatique est vérifié (les structures de données sont normalisées, les accès aux données sont optimisés, le code est modulaire et robuste, les tests sont effectués).
• Les préoccupations de sécurité sont prises en compte à toutes les étapes d’un développement informatique.
• Les bonnes pratiques de sécurité sont mises en œuvre à toutes les étapes d’un développement informatique.
• Des tests de sécurité sont prévus et mis en œuvre.</td><td><u>Prérequis</u> :
B2.5 A2D
B2.2 A2D (CI/CD)
<u>Transversalités</u> :
Bloc 2 – 2ème année</td></tr>
</table>

	• Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures	• Environnements de production et de développement : fonctionnalités de sécurité, techniques d'isolation des applicatifs. <u>Savoirs économiques, juridiques et managériaux</u> • Responsabilité du concepteur de solutions applicatives.	• L'accès aux données respecte les règles de sécurité. • Les composants utilisés sont certifiés, sécurisés et actualisés.	

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 1</u>** : sensibilisation des membres de l'équipe projet agile Pré-requis du S-SDLC, la RSSI souhaite évaluer la maturité de la sécurité applicative de l'équipe projet. 1. Répondre au QCM relatif aux développeurs-testeurs de l'équipe projet 2. Compléter vos connaissances en consultant les ressources fournies. 3. Mettre en place des outils de veille technologique efficaces	• QCM d'évaluation sur la sécurité des applications et de leur développement portant sur les outils de protection contre les vulnérabilités les plus connues, l'analyse et la sécurisation du code, la sécurité par la conception (*secure by design*), la modélisation des menaces, le RGPD, les bonnes pratiques de développement, les méthodes de détection d'erreurs dans le code… • Liste de sites web traitant de la sécurité applicative pour effectuer une veille technologique.	• QCM complété • Des outils sont mis en place pour effectuer une veille technologique efficace sur la sécurité
Séance 2	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 2</u>** : analyse du cahier des charges et des exigences de sécurité de l'application	• Fiche de savoirs technologiques des bonnes pratiques de sécurité en conception d'une solution applicative (*réduction de la surface d'attaque, défense en profondeur, séparation*	• La cartographie des surfaces d'attaque est réalisée et comporte les points d'entrée, les services et les logiciels présents dans ce périmètre.

	À partir de l'analyse documentaire et de sécurité : 1. Identifier et décrire dans un tableau les points d'entrée qui peuvent être des surfaces d'attaque de l'application 2. Recenser dans un tableau les actifs physiques et logiques : services présents derrière ces points d'entrée, logiciels utilisés et leur version, données personnelles 3. Identifier les différents acteurs de l'application, leur rôle, leurs droits d'accès autorisés et la couche SI concernée par ces droits 4. Analyser la résilience de la défense en profondeur mise en place en précisant l'impact d'une défaillance d'un contrôle de sécurité 5. Rédiger des recommandations de sécurité de l'application et de ses dépendances en vous appuyant sur les bonnes pratiques de développement sécurisé	*des privilèges, application de paramètres par défaut sûrs, minimisation de la complexité, principe du moindre privilège, séparation des tâches, tolérance aux fautes, utilisation de fonctions sûres…)* • Fiche de savoirs technologiques des normes internationales et référentiels relatifs au management de la sécurité de l'information et à la sécurité des applications et des données (ISO/IEC 27001, ISO/IEC 27002, ISO/IEC 31000, ISO/IEC 27034, PCI-DSS et PA-DSS, HIPAA, RGPD) • Dossier documentaire comportant : le schéma de l'infrastructure du SI, pyramide des contrôles de sécurité mis en place dans chaque couche du SI, paramétrage par défaut respectant la sécurité • Nouveau schéma de l'infrastructure. • Plan de tests des contrôles de sécurité de chaque couche du SI	• Le tableau récapitulatif des acteurs et des droits attribués en fonction de leur rôle est réalisé. • Le rapport de test de la résilience de la défense en profondeur mise en place est rédigé. • Des recommandations de sécurité applicative sont proposées. • L'ensemble des documents sont envoyés au RSSI
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 3 :</u>** configuration de l'environnement de développement Votre mission consiste à participer à la l'installation et à la configuration de	• Fiche de savoirs technologiques des outils d'intégration et de déploiement continus (voir la séquence B2.2A2D) • Dossier documentaire de l'environnement de développement comportant les outils	• L'outil de gestion de bugs est installé et configuré. • Les outils de tests de sécurité et d'analyse statique sont installés. • L'environnement de compilation est sécurisé. • Le scanner de vulnérabilité est installé

	l'environnement de développement et d'intégration avec les outils de sécurité 1. Installer un outil de suivi de bugs de sécurité. 2. Configurer des outils d'analyse de sécurité de code statique (*Static Analysis Security Testing - SAST*) et dynamique (DAST) 3. Activer les options de sécurité de compilation et les défauts de sécurité au sein de votre IDE 4. Installer et configurer un scanner de vulnérabilités 5. Créer un pipeline CI/CD pour tester l'ensemble des outils depuis le développement jusqu'au déploiement dans l'environnement de pré-production. 6. Intégrer dans le pipeline CI la détection automatique de failles de sécurité connues dans des référentiels (*Git via GitLeaks, WhiteSourceBolt, OWASP*...)	d'analyse de code statique (*Static Application Security Testing)*, liste des méthodes et API obsolètes et/ou interdites présentant des menaces de sécurité, le logiciel de gestion de versions. • Machine virtuelle comportant l'outil de CI/CD Jenkins et jouant le rôle de l'environnement de développement et d'intégration • Machine virtuelle comportant Docker et simulant l'environnement de pré-production de l'application. • Archive d'une application de test comportant des défauts de conception et des vulnérabilités.	• La journalisation des événements est mise en place de manière sécurisée • Le pipeline CI/CD mis en œuvre sollicite l'ensemble des outils installés et intègre la détection automatique de vulnérabilités grâce aux analyses dans les référentiels de vulnérabilités
Séance 4	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 4 :</u>** revue de code La revue de code est réalisée de manière incrémentale, à chaque sprint. Un développeur de l'équipe a codé un récit d'utilisateur lors du sprint précédent	• Fiche de savoirs technologiques des bonnes pratiques de sécurité en conception d'une solution applicative (*réduction de la surface d'attaque, défense en profondeur, séparation des privilèges, application de paramètres par défaut sûrs, minimisation de la complexité, principe du moindre privilège, séparation des*	• Le suivi des bugs de sécurité est mis en place • Les tests de conformité aux exigences et aux bonnes pratiques de sécurité sont effectués. • Le rapport des vulnérabilités lié au récit d'utilisateur est rédigé

	et il a poussé la nouvelle version de l'application. 1. Tester l'exécution de l'application 2. Lancer l'outil SAST pour analyser les défauts de sécurité du code de l'application 3. Analyser le résultat du test et identifier les exigences de sécurité qui ne sont pas respectées dans le code du récit d'utilisateur 4. Vérifier la conformité du code avec les bonnes pratiques de codage sécurisé (*validation des entrées, contrôles d'accès, utilisation de fonctions sûres, RGPD, surface d'attaque dans le code…)* 5. Rédiger un rapport des vulnérabilités suite à la revue de code du récit d'utilisateur	*tâches, tolérance aux fautes, utilisation de fonctions sûres…)* • Fiches de savoirs technologiques des typologies des menaces de développement applicatif • Liste des vérifications de la conformité du code aux exigences de sécurité de l'application (objectifs de la revue de code) et aux bonnes pratiques de codage sécurisé • Récit d'utilisateur • Machine virtuelle de développement et d'intégration comportant le code de l'application • Modèle du rapport de revue de code	• Les vulnérabilités sont classées selon le modèle STRIDE (*Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, Elevation of privilege*). • Les vulnérabilités sont documentées.
Séance 5	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 5 :</u>** recherche de vulnérabilités par injection de données aléatoires (*fuzzing*) Les failles de sécurité détectées dans les sprints précédents sont corrigées et/ou atténuées. On vous demande de tester la sécurité de l'application avant son déploiement à la direction métier.	• Fiche de savoirs technologiques des mécanismes de tests par injection de données aléatoires • Machine virtuelle de pré-production • Machine virtuelle de développement et d'intégration • Fiche descriptive de la configuration du scanner de vulnérabilités et du lancement du scénario de tests de *fuzzing*	• Le suivi des bugs de sécurité est mis en place. • Le pipeline CD est configuré. • L'application est déployée dans l'environnement de pré-production. • Les tests par injection de données aléatoires sont lancés dans l'environnement de pré-production. • Le rapport de tests est rédigé.

	1. Lancer la machine virtuelle de pré-production et démarrer l'image Docker 2. Déployer manuellement l'application dans le conteneur Docker 3. Lancer à l'aide du scanner de vulnérabilités installé dans l'environnement de développement les tests d'injection de données aléatoires dans l'application déployée dans l'environnement de pré-production. Vérifier le bon déroulement du test 4. Modifier le pipeline de déploiement pour intégrer le lancement automatique du test d'injection de données aléatoires 5. Lancer le pipeline de déploiement et vérifier le lancement du test de vulnérabilité 6. Rédiger le rapport de tests 7. Documenter les vulnérabilités rencontrées.		• Les vulnérabilités rencontrées sont documentées ainsi que leur impact grâce à votre veille technologique.
Séance 6	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**<u>Sprint 6 :</u>** déroulement de scénario de sécurité Le responsable de produit a rédigé un récit d'utilisateur qui a été implémenté par un développeur lors d'un sprint.	• Exemples de scénarios de sécurité : https://safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf • Scénario de sécurité selon le même modèle que les exemples fournis • Récit d'utilisateur	• Le récit d'utilisateur est testé dans l'environnement de pré-production. • Les tâches du *backlog* sont exécutées dans l'environnement de pré-production.

	L'experte en sécurité a rédigé un scénario de sécurité qu'elle vous demande de dérouler pour vérifier les pratiques de sécurité mises en œuvre. 1. Lancer l'application et vérifier le récit d'utilisateur dans l'environnement de pré-production 2. Traiter les tâches du *backlog* correspondant au scénario de sécurité du récit d'utilisateur 3. Reporter dans un document le résultat obtenu à l'issue de chacune des tâches et le résultat escompté 4. Identifier les bonnes pratiques de codage sécurisé manquantes qui engendrent les failles de sécurité détectées 5. Mettre en place les bonnes pratiques manquantes pour répondre à la politique de sécurité mise en place.		• Les résultats d'exécution des tâches du *backlog* obtenues et attendus sont reportés dans un document par des captures écran. • Les bonnes pratiques de codage sont mises en œuvre pour corriger les vulnérabilités détectées.
Séance 7	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	**Sprint 7 :** rédaction d'un scénario de risque en déclinant le besoin de sécurité sur les axes de la sécurité DICP L'experte en sécurité informatique a rédigé un scénario de risque accidentel et intentionnel relatifs à un récit d'utilisateur.	• Fiche de savoirs technologiques des scénarios de risque (https://www.ssi.gouv.fr/uploads/2018/11/guide-securite-numerique-agile-anssi-pa-v1.pdf) • Exemples de scénarios de sécurité : https://safecode.org/publication/SAFECode_Agile_Dev_Security0712.pdf	• Le scénario de sécurité est rédigé selon le modèle fourni. • Les mesures préconisées sont mises en œuvre dans l'environnement de développement. • L'application est déployée dans l'environnement de pré-production via le pipeline CI/CD. • L'application est opérationnelle dans l'environnement de pré-production.

	1. Rédiger le scénario de sécurité ayant pour vocation de supprimer et/ou de réduire le risque intentionnel évoqué 2. Après validation du scénario de sécurité, mettre en place les mesures préconisées **3.** Déployer l'application dans l'environnement de pré-production.	• Scénario intentionnel et accidentel liés au récit d'utilisateur	• Les bonnes pratiques de sécurité applicative sont mises en œuvre tout au long du cycle de développement de l'application.

Séquence 3.5 E2D	**Sécuriser des API avec des solutions de gestion des identités et des autorisations (IAM)**			
13 h	Vous participez à un projet de sécurisation des API REST de votre organisation cliente. Votre client souhaite déléguer le processus d'authentification et d'autorisations à des solutions existantes robustes qui offrent la fonctionnalité SSO (*Single Sign On*). Les équipes de développement peuvent ainsi exposer leurs API REST en toute sécurité et de manière rapide. Le client souhaite tester plusieurs solutions de gestion des identités et des accès (IAM) pour faire son choix. *Note aux auteurs : cette séquence peut être réalisée avec des solutions basées sur keycloak ou OpenID Connect et OAuth2.*			
	Compétences travaillées	**Savoirs associés**	**Indicateurs de performance**	**Prérequis / Transversalités**
	• Mettre en œuvre et vérifier la conformité d'une solution applicative et de son développement à un référentiel, une norme ou un standard de sécurité. • Gérer les accès et les privilèges appropriés.	<u>Savoirs technologiques</u> • Authentification, privilèges et habilitations des utilisateurs : principes et techniques • Sécurité des applications *Web* : risques, menaces et protocoles	• Les accès et privilèges respectent les règles organisationnelles : – les utilisateurs sont authentifiés ; – les habilitations sont configurées ;	<u>Prérequis</u> : Bloc 2 – C2D <u>Transversalités</u> : Bloc 2 – 2ème année

	• Analyser les connexions (logs).	• Sécurité du développement d'application : gestion de projet, architectures logicielles, rôle des protocoles, authentification, habilitations et privilèges des utilisateurs, confidentialité des échanges, tests de sécurité, audit de code.	– l'accès aux données est contrôlé ; – les privilèges sont restreints. • Le système d'authentification est conforme aux règles de sécurité. • Les composants utilisés sont certifiés, sécurisés et actualisés.	

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h	Le chef de projet vous confie le test d'une des solutions d'IAM sur une API REST du client. Avant d'attaquer l'implémentation du module, il vous demande de vous documenter et de mettre en place une veille technologique efficace sur les solutions qui mettent en œuvre la gestion IAM. 1. Réaliser un comparatif des solutions d'IAM proposées dans le tableau comparatif. 2. Étudier la solution choisie à l'issue de votre comparatif et vérifier si elle présente des vulnérabilités connues	• Fiches de savoirs technologiques relatifs aux concepts de base des solutions de gestion centralisée des identités et des accès (IAM) dans les API micro-services ainsi que la délégation de droits : *OpenID Connect, OAuth2.0 – RFC 6749/6750, solutions SSO, SAML – Security Assertion Markup Language, keycloack, JWT - Json Web Token - RFC7519, JSON Web signature – RFC7515, spring security* • Ressources numériques : (https://www.ionos.fr/digitalguide/serveur/securite/oauth/, https://www.youtube.com/watch?v=xg5wGSPmeGI, https://www.youtube.com/watch?v=I5tFlK5PPjc, http://keycloak.jboss.org) • Tableau comparatif à compléter	• Une veille technologique des solutions d'IAM est mise en place. • Le comparatif des solutions d'IAM est réalisé selon les critères précisés dans le tableau.

Séance 2	Tâches à réaliser	Ressources fournies	Résultats attendus
3 h	Le chef de projet vous demande d'installer et de configurer la solution d'IAM. Le client a fourni un clone d'une application de test sur le dépôt distant du projet 1. Installer l'environnement de test 2. Installer le code de l'application de test et gérer les dépendances 3. Installer la solution d'IAM 4. Se connecter à l'interface web d'administration de la solution 5. Modifier le mot de passe par défaut du compte d'administration par défaut et tester l'authentification 6. Configurer la solution d'IAM et définir l'API de test et son paramétrage	• Code de l'API REST de test disponible sur un dépôt distant • L'environnement de test de la solution (VM ou machine physique) • La solution d'IAM (archive ou image docker) • Guide pratique d'installation et de configuration de la solution d'IAM	• L'environnement de test est installé. • L'API de test est installée dans l'environnement de test. • La solution d'IAM est installée et configurée. • Les fichiers de configuration sont exportés et mis dans un dépôt de gestion des versions distant.
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
3 h	L'API est utilisée par trois profils d'utilisateurs différents. 1. Créer un utilisateur de chaque profil dans la solution d'IAM 2. Créer des rôles dans la solution d'IAM 3. Affecter les rôles aux utilisateurs en fonction de leur profil 4. Modifier le code de l'API pour intégrer la solution d'IAM (dépendances, paramètres et classes de configuration),	• Guide pratique d'installation et de configuration de la solution d'IAM • Cahier des charges du projet comportant la stratégie d'authentification et des accès des utilisateurs à l'API • Rapport d'activités à compléter • Cas d'utilisation de l'authentification via la solution d'IAM. Le scénario doit prévoir le cas d'une altération ou d'une révocation du jeton d'authentification.	• Des utilisateurs de l'API sont créés dans la solution d'IAM. • Les rôles sont affectés aux utilisateurs selon leur profil. • Le contenu des jetons d'accès et d'ID sont décodés et sauvegardés dans un rapport d'activités • Le rapport de tests est rédigé. Des captures écran permettent d'illustrer les résultats des tests réalisés.

	5. Demander le jeton d'accès de l'utilisateur admin en ligne de commande 6. Décoder le jeton d'accès JWT et l'analyser 7. Décoder le jeton d'identification et repérer les informations personnelles de l'utilisateur 8. Vérifier la signature du jeton d'identification 9. Tester le cas d'utilisation de l'authentification des utilisateurs		
Séance 4	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
2 h	1. Mettre en place la gestion des autorisations conformément aux exigences de sécurité définies dans le cahier des charges 2. Sécuriser les points d'entrées de l'API grâce aux rôles créés dans l'IAM 3. Tester les accès aux ressources protégées de l'API 4. Rédiger un rapport de tests des accès à l'API et réaliser des captures écrans des résultats obtenus	• Cahier des charges du projet • Rapport d'activités à compléter	• Les autorisations sont configurées conformément aux exigences de sécurités définies. • Les tests des points d'entrée et des accès sont réalisés et un rapport de tests est rédigé • Les fichiers de configuration des autorisations sont commentés et poussés de manière incrémentale dans le dépôt distant du projet. • Le rapport d'activités est complété
Séance 5	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h	Le client souhaite que les utilisateurs puissent se connecter à l'application en utilisant des fournisseurs d'authentification externes	• Cahier des charges du projet	• Les étapes de configuration des fournisseurs externes et des tests d'authentification sont documentées dans un rapport à destination du client.

	1. Configurer les fournisseurs d'authentification externes dans la solution d'IAM 2. Implémenter l'authentification SSO sécurisée. 3. Mettre à jour le code de l'API pour intégrer les fournisseurs d'authentification externes 4. Tester l'authentification d'un utilisateur à travers un réseau social		• Les utilisateurs se connectent sur l'API via un fournisseur externe. • Le code de l'application est poussé sur le dépôt distant du projet
Séance 6	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
3 h	Afin de réagir rapidement à des intrusions d'applications REST, le client vous demande de mettre en place un dispositif qui permet de tracer les accès aux applications et à la console d'administration de l'IAM. 1. Configurer la journalisation des accès des utilisateurs dans la solution d'IAM en respectant le cahier des charges. 2. Tester la configuration en vérifiant le type d'événements journalisés dans l'IAM 3. Configurer la journalisation d'événements exécutés dans la console d'administration 4. Configurer une notification par mail sur certains événements 5. Tester la configuration des notifications mises en place	• Cahier des charges du projet	• Les accès des utilisateurs sont journalisés dans la base de données de la solution d'IAM conformément aux types d'événements configurés. • La notification par mail des événements configurés est opérationnelle. • Une documentation sur les types d'événements d'accès et de la mise en place de leur journalisation est rédigée à l'attention des développeurs.

	6. Configurer et activer la journalisation des événements liés aux erreurs de connexion dans un fichier 7. Réaliser des accès infructueux (à l'aide d'un programme) et visualiser le fichier de logs 8. Rédiger une documentation relative à la mise en place de la journalisation des événements dans la solution d'IAM		

Séquence 3.5F2D	**Sécuriser une base de données NoSQL**			
10 h	Votre client utilise une application métier qui exploite des données non structurées et des données à caractère personnel stockées dans une base de données MongoDB. Votre client a entendu parler de nouvelles vulnérabilités dans les bases de données NoSQL. Il a sollicité votre entreprise pour auditer sa base de données afin de mettre en place les mesures nécessaires qui vont assurer l'intégrité et la confidentialité des données de l'application et empêcher une élévation de privilèges qui risque de rendre l'application vulnérable aux attaques de sécurité. Votre mission consiste à prendre en charge la sécurité applicative.			
	Compétences travaillées	**Savoirs associés**	**Indicateurs de performance**	**Prérequis / Transversalités**
	• Participer à la vérification des éléments contribuant à la qualité d'un développement informatique • Mettre en œuvre et vérifier la conformité d'une solution	<u>Savoirs technologiques</u> • Aspects réglementaires du développement applicatif : protection de la vie privée dès la conception, protection des	• Le respect des bonnes pratiques de développement informatique est vérifié (les structures de données sont normalisées, les accès aux données sont optimisés, le code est modulaire et robuste, les tests sont effectués).	<u>Prérequis</u> : B2.3. C2D <u>Transversalités</u> :

	applicative et de son développement à un référentiel, une norme ou un standard de sécurité • Prévenir les attaques • Analyser des incidents de sécurité, proposer et mettre en œuvre des contre-mesures.	données par défaut, sécurité par défaut, droit des individus. • Vulnérabilités et contre-mesures sur les problèmes courants de développement. • Gestion des droits d'accès aux données : principes et techniques <u>Savoirs économiques, juridiques et managériaux</u> Les risques des cyberattaques pour l'organisation : économique, juridique, atteinte à l'identité de l'entreprise.	• Des tests de sécurité sont prévus et mis en œuvre. • L'accès aux données respecte les règles de sécurité. • Les contre-mesures mises en place corrigent et préviennent les incidents de sécurité. • Les contre-mesures sont documentées de manière à en assurer le suivi.	B2.3 – 2ème année

Séance 1	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
1 h	1. Analyser le cahier des charges 2. Installer l'environnement de test 3. Tester l'authentification à la base de données à auditer 4. Installer l'image Docker nosqlmap d'audit 5. Installer l'outil d'audit mongoaudit	• Cahier des charges comportant le contexte, les spécifications de l'application, les informations d'authentification à la base de données (BDD) NoSQL de test et l'infrastructure logicielle • Environnement de test comportant le serveur et la base de données NoSQL ainsi que le serveur de l'application	• L'environnement de test est installé et configuré • L'accès aux bases de données avec les informations d'authentification est opérationnel. • L'image docker nosqlmap est installée
Séance 2	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
3 h	1. Lancer l'outil d'audit nosqlmap. 2. Configurer l'outil et lancer une analyse de sécurité sur la base de données 3. Identifier les injections NoSQL qui ont abouti	• Fiche de savoirs technologiques des concepts de base et des bonnes pratiques en matière de cybersécurité des bases de données NoSQL • Plan de tests comportant des exemples d'injections NoSQL	• Les audits de la base de données, du serveur et de l'application sont réalisés. • Le rapport d'audit de Mongoaudit est généré. • Les résultats des tests de sécurité sont reportés dans un rapport de tests. Pour chaque test, il

	4. Lancer une nouvelle analyse sur le serveur de l'application et de la base de données 5. Lancer la suite de tests fournie par MongoDB 6. Compléter vos tests avec le plan de tests fourni par votre collègue spécialiste des injections NoSQL		sera indiqué le résultat escompté et le résultat obtenu (captures d'écran des résultats).
Séance 3	**Tâches à réaliser**	**Ressources fournies**	**Résultats attendus**
3 h	Analyser les vulnérabilités trouvées. Pour chacune d'elles, préciser dans un tableau : 1. la source de la faille (code applicatif, base de données) 2. le type de la vulnérabilité : injection, non-respect des bonnes pratiques de codage et de RGPD, mauvaises configurations, infrastructure 3. l'impact en termes d'atteinte d'intégrité, de confidentialité des données, y compris des données à caractère personnel, et d'élévation de privilèges Les vulnérabilités doivent être classées par type.	• Fiche de savoirs technologiques des concepts de base et des bonnes pratiques en matière de cybersécurité des bases de données NoSQL • Rapport d'audit de Mongoaudit • Rapport de tests de sécurité	• Les vulnérabilités sont identifiées, analysées et classées par type dans un tableau

Séance 4	Tâches à réaliser	Ressources fournies	Résultats attendus
3 h	Vous devez mettre en place un plan d'action pour résoudre les failles détectées. 1. Proposer des contre-mesures pour corriger les vulnérabilités liées à l'application et à la base de données. 2. Mettre en œuvre les contre-mesures validées par le client 3. Mesurer l'efficacité des contre-mesures mises en place	• Fiche de savoirs technologiques des concepts de base et des bonnes pratiques en matière de cybersécurité des bases de données NoSQL • Tableau des vulnérabilités • Rapport d'audit de Mongoaudit	• Le tableau des vulnérabilités est complété par les propositions de contre-mesures aux vulnérabilités. • Les contre-mesures des vulnérabilités sont mises en place dans l'environnement de test • Les vulnérabilités sont corrigées et testées dans l'environnement de test. • Les contre-mesures efficaces contre les vulnérabilités sont précisées dans le tableau pour mesurer l'efficacité des corrections apportées.