

— Public	Développeurs, chefs de projets techniques.
— Durée	4 jours - 28 heures
— Pré-requis	Développeurs en C/C++ Connaissance en assembleur est souhaitée
— Objectifs	Sensibiliser les développeurs sur la sécurité du code Connaître le rôle des acteurs et la classification des risques : CERT, CWE, OWASP Comprendre la structure mémoire (heap, stack, etc.), et savoir comment les vulnérabilités exploitent ces parties pour compromettre un système/ Savoir repérer les failles de sécurité dans le code Étudier des cas d'attaques contre un code tout en connaissant les mécanismes pour se protéger Connaître les bonnes pratiques à appliquer et les différentes approches pour produire un code sécurisé Renforcer la protection des logiciels contre les tentatives de rétroconception et de piratage Étudier les mécanismes de protection offerts au niveau d'un système d'exploitation et par les compilateurs
— Méthodes pédagogiques	Pour bien préparer la formation, le stagiaire remplit une évaluation de positionnement et fixe ses objectifs à travers un questionnaire. La formation est délivrée en présentiel ou distanciel (e-learning, classe virtuelle, présentiel et à distance). Le formateur alterne entre méthodes démonstratives, interrogatives et actives (via des travaux pratiques et/ou des mises en situation). La validation des acquis peut se faire via des études de cas, des quiz et/ou une certification. Cette formation est animée par un consultant-formateur dont les compétences techniques, professionnelles et pédagogiques ont été validées par des diplômes et/ou testées et approuvées par l'éditeur et/ou par Audit Conseil Formation.
— Moyens techniques	1 poste de travail complet par personne De nombreux exercices d'application Mise en place d'ateliers pratiques Remise d'un support de cours Passage de certification(s) dans le cadre du CPF Remise d'une attestation de stage
— Modalité d'évaluation des acquis	Evaluation des besoins et objectifs en pré et post formation Evaluation technique des connaissances en pré et post formation Evaluation générale du stage
— Délai d'accès	L'inscription à cette formation est possible jusqu'à 5 jours ouvrés avant le début de la session
— Accessibilité handicapés	Au centre d'affaires ELITE partenaire d'ACF à 20 m. Guide d'accessibilité à l'accueil.

INTRODUCTION

- C et C++ des langages peu sécurisés ?
- Connaître les risques liés à la programmation
- Les traces laissées par les développeurs

CLASSIFICATION DES RISQUES

- Différents acteurs : CERT, PCI, CWE, OWASP, MISRA, etc.
- Codage sécurisé d'une application
- Classification des risques selon CERT
- Guides pour le codage sécurisé
- Travaux pratiques : Consulter les guides de codage sécurisé, et déterminer les règles pertinentes du codage sécurisé à travers l'appréhension du calcul des risques

VULNÉRABILITÉS

- Modèle mémoire d'un programme exécutable
- Compilation
- Appels des fonctions
- Les tableaux et les chaînes de caractères
- Les pointeurs
- Gestion de la mémoire dynamique
- Sécurité des entiers
- Sorties formatées
- Gestion des fichiers
- Travail pratique 1: Exploration de certaines failles de sécurité : buffer overflow, stack smashing, pointer subterfuge, exécution d'un code arbitraire, return-to-libc et ROP attaques, etc.
- Travail pratique 2 : Mettre des solutions pour remédier à chaque type d'attaque.

LES BONNES PRATIQUES

- Macro et inline
- Gestion de la mémoire
- Gestion des erreurs et des exceptions
- Structures des classes
- Passer à C++14 et C++17 : conversions, pointeurs intelligents, espaces de noms, etc.
- Le multithreading : Gestion sécurisée des threads et des données partagées
- Travaux pratiques : Divers exercice pour mettre en œuvre des concepts modernes de C++14/17 afin d'écrire un code sécurisé

SÉCURISATION DE LA COMMUNICATION

- La cryptographie en C/C++ : les fonctions de hachage, le chiffrement symétrique et asymétrique, les bibliothèques cryptographiques.
- La sécurité des communications : les protocoles SSL/TLS, gestion des certificats numériques.
- Travaux pratiques : Mise en œuvre de protocoles sécurisés dans les applications C/C++.

RÉTROCONCEPTION ET INGÉNIERIE INVERSE EN C/C++

- Concepts de base de la rétroconception (reverse engineering)
- Outils de rétroconception : Introduction à IDA Pro et Ghidra
- Comprendre et naviguer dans les appels de fonction et les conventions d'appel (cdecl, stdcall, fastcall)
- Analyse des tables de dispatch virtuelles (vtables) pour le C++ orienté objet
- Décomposition des binaires : Analyse des segments de code et reconstruction du flux de contrôle
- Mise en place de protections contre la rétroconception
- Recommandations pour sécuriser le code et prévenir la rétroconception
- Travaux pratiques : Rétroconception d'un binaire simple avec IDA Pro/Ghidra

DURCISSEMENT DE LA SÉCURITÉ (HARDENING)

- Outils d'analyse syntaxique
- Utilisation de valgrind : identifier les fuites de mémoire
- Techniques de mitigation des vulnérabilités (ASLR, DEP, canaries, etc.).
- Mécanismes de protection offerts par les systèmes d'exploitation
- Mécanismes de protection offerts par GCC
- Les options de sécurité de GCC
- Validation du Hardening.
- Travail pratique 1 : Utiliser un analyseur syntaxique pour identifier les failles
- Travail pratique 2 : Identifier les fuites de mémoire avec Valgrind
- Travail pratique 3 : Tester plusieurs configurations de compilation avec gcc en appliquant le hardening. Valider et évaluer des configurations gcc.

NOUS CONTACTER

Siège social

16, ALLÉE FRANÇOIS VILLON
38130 ÉCHIROLLES

Téléphone

04 76 23 20 50 - 06 81 73 19 35

Suivez-nous sur les réseaux sociaux, rejoignez la communauté !



ACF Audit Conseil Formation



@ACF_Formation

Dernière mise à jour : 18/09/2024

PROFIL Formateur : Les formateurs sont recrutés selon plusieurs critères :
Expérience, pédagogie, dynamisme et prévoyance.