CSSLP® – Certified Secure Software Lifecycle Professional (E-Learning)

ISC2

- **Nível:**
- **Duração:** h

Sobre o curso

Desenvolve competências técnicas avançadas e adquire os conhecimentos essenciais em autenticação, autorização e auditoria ao longo de todo o ciclo de vida do desenvolvimento de software, com base nas melhores práticas, políticas e procedimentos definidos pelos especialistas em cibersegurança da ISC2.

A certificação CSSLP valida competências avançadas em segurança aplicacional. Obter esta certificação é uma forma reconhecida de impulsionar a tua carreira e de integrar as melhores práticas do mercado em segurança em todas as fases do ciclo de vida de desenvolvimento de software (SDLC).

Destinatários

O CSSLP é ideal para profissionais de software development e segurança responsáveis por aplicar boas práticas em cada fase do SDLC – desde o software design e implementação até aos testes e deployment – incluindo profissionais nas seguintes funções:

- Software Architect
- Software Engineer
- Software Developer
- Application Security Specialist
- Software Program Manager
- Quality Assurance Tester
- Penetration Tester
- Software Procurement Analyst
- Project Manager
- Security Manager
- IT Director/Manager

Objetivos

- Resumir os conceitos que, quando aplicados, resultam em software seguro.

- Utilizar o secure software life cycle para gerir a proteção de dados.
- Comparar as categorias de secure software requirements para definir salvaguardas adequadas.
- Desenhar e construir secure software architectures.
- Aplicar práticas de segurança para implementar software seguro.
- Realizar testes para garantir a segurança do software.
- Implementar práticas de segurança quando o software é colocado em produção e mantido num ambiente operacional.

Pré-requisitos

Para obteres esta certificação, tens de passar no exame e ter pelo menos quatro anos de experiência profissional remunerada e cumulativa como profissional de software development lifecycle em um ou mais dos oito domínios do ISC2 CSSLP Exam Outline.

Metodologia

O Official ISC2 Online Self-Paced CSSLP Training é uma forma inovadora de preparação para a certificação que utiliza artificial intelligence para personalizar o teu percurso de aprendizagem. Identifica as áreas que exigem atenção adicional e orienta a tua preparação para o exame de forma verdadeiramente personalizada.

Estuda de forma mais inteligente, com estas vantagens principais:

- **Instrução personalizada** – O conteúdo, o ritmo e a dificuldade adaptam-se ao teu conhecimento individual, velocidade de aprendizagem e nível de confiança.
- **Maior envolvimento** – Feedback imediato e conteúdo dinâmico permitem aprender a um nível adequado.
- **Poupança de tempo** – O tempo de formação é otimizado com foco nas áreas que exigem maior revisão.
- **Melhores resultados de aprendizagem** – A plataforma identifica áreas que exigem revisão adicional e fornece apoio direcionado para maximizar a tua compreensão do conteúdo.

A formação Certified Secure Software Lifecycle Professional (CSSLP) tira partido do poder da artificial intelligence, orientando os formandos através de uma experiência de aprendizagem self-paced adaptada às suas necessidades específicas. Abrange as competências avançadas e os conhecimentos necessários para authentication, authorization e auditing ao longo do software development life cycle (SDLC), utilizando best practices, políticas e procedimentos estabelecidos pelos especialistas em cibersegurança da ISC2.

Programa

- Conceitos de Secure Software
- Gestão do Secure Software Lifecycle
- Requisitos de Secure Software
- Arquitetura e Design de Secure Software

- Implementação de Secure Software
- Testes de Secure Software
- Secure Software Deployment, Operações e Manutenção
- Secure Software Supply Chain

Domínio 1: Conceitos de Secure Software

1.1 – Compreender conceitos fundamentais

- Confidencialidade (por exemplo, encryption)
- Integridade (por exemplo, hashing, digital signatures, code signing, fiabilidade, modificações, autenticidade)
- Disponibilidade (por exemplo, redundância, replicação, clustering, escalabilidade, resiliência)
- Authentication (por exemplo, multi-factor authentication (MFA), identity & access management (IAM), single sign-on (SSO), federated identity, biometria)
- Authorization (por exemplo, access controls, permissões, entitlements)
- Responsabilização (por exemplo, auditing, logging)
- Nonrepudiation (por exemplo, digital signatures, blockchain)
- Normas de governance, risk and compliance (GRC) (por exemplo, autoridade reguladora, requisitos legais, indústria)

1.2 – Compreender princípios de security design

- Least privilege (por exemplo, access control, need-to-know, run-time privileges, Zero Trust)
- Segregation of Duties (SoD) (por exemplo, controlo multipartes, secret sharing, split knowledge)
- Defense in depth (por exemplo, controlos em camadas, diversidade geográfica, diversidade técnica, sistemas distribuídos)
- Resiliência (por exemplo, fail safe, fail secure, ausência de single point of failure, failover)
- Economy of mechanism (por exemplo, single sign-on (SSO), password vaults, eficiência de recursos)
- Complete mediation (por exemplo, gestão de cookies, gestão de sessões, caching de credenciais)
- Open design (por exemplo, princípio de Kerckhoffs, peer review, open source, crowd source)
- Least common mechanism (por exemplo, compartmentalization/isolation, allow/accept list)
- Psychological acceptability (por exemplo, complexidade da palavra-passe, passwordless authentication, layouts de ecrã, CAPTCHA)
- Reutilização de componentes (por exemplo, common controls, bibliotecas)

Domínio 2: Gestão do Secure Software Lifecycle

2.1 – Gerir a segurança no âmbito de uma metodologia de software development (por exemplo, Agile, waterfall)

2.2 – Identificar e adotar normas de segurança (por exemplo, implementação de security frameworks, promoção da sensibilização para a segurança)

2.3 – Definir estratégia e roadmap

- Marcos e checkpoints de segurança (por exemplo, control gate, break/build criteria)

2.4 – Definir e desenvolver documentação de segurança

2.5 – Definir métricas de segurança (por exemplo, nível de criticidade, tempo médio de remediação, complexidade, KPI, objetivos e resultados-chave)

2.6 – Descontinuar aplicações

- Políticas de End of Life (EOL) (por exemplo, remoção de credenciais, remoção de configuração, cancelamento de licenças, arquivo, SLA)
- Destino dos dados (por exemplo, retenção, destruição, dependências)

2.7 – Criar mecanismos de reporting de segurança (por exemplo, relatórios, dashboards, feedback loops)

2.8 – Incorporar métodos integrados de gestão de risco

- Regulamentos, normas e guidelines (por exemplo, ISO, PCI, NIST, OWASP, SAFECode, SAMM, BSIMM)
- Legal (por exemplo, propriedade intelectual, notificação de violação)
- Gestão de risco (por exemplo, avaliação de risco, análise de risco)
- Risco técnico vs. risco de negócio

2.9 – Implementar práticas de operação segura

- Processo de change management
- Plano de incident response
- Verification and validation
- Processo de Assessment and Authorization (A&A)

Domínio 3: Requisitos de Secure Software

3.1 – Definir requisitos de software security

- Funcionais (por exemplo, business requirements, use cases, stories)
- Não funcionais (por exemplo, segurança, operação, continuidade, deployment)

3.2 – Identificar requisitos de conformidade

- Autoridade reguladora
- Legal
- Específicos da indústria (por exemplo, defesa, saúde, comercial, financeiro, PCI)
- Aplicáveis a toda a empresa (por exemplo, development tools, normas, frameworks, protocolos)

3.3 – Identificar requisitos de classificação de dados

- Propriedade dos dados (por exemplo, data dictionary, data owner, data custodian)
- Data labeling (por exemplo, sensibilidade, impacto)
- Tipos de dados (por exemplo, estruturados, não estruturados)
- Ciclo de vida dos dados (por exemplo, geração, armazenamento, retenção, eliminação)
- Tratamento de dados (por exemplo, PII, informação publicamente disponível)

3.4 – Identificar requisitos de privacidade

- Âmbito da recolha de dados

- Anonimização de dados (por exemplo, pseudo-anónimos, totalmente anónimos)
- Direitos do utilizador (legais) e preferências (por exemplo, eliminação de dados, direito ao esquecimento, preferências de marketing, partilha e utilização de terceiros, termos de serviço)
- Retenção de dados (por exemplo, durante quanto tempo, onde, o quê)
- Requisitos transfronteiriços (por exemplo, residência dos dados, jurisdição, limites de processamento de dados multinacionais)

3.5 – Definir aprovisionamento de acesso a dados

- User provisioning
- Service accounts
- Processo de reaprovação

3.6 – Desenvolver misuse e abuse

- Identificação de controlos de mitigação

3.7 – Desenvolver uma matriz de rastreabilidade de requisitos de segurança

3.8 – Definir requisitos de segurança para fornecedores terceiros

Domínio 4: Arquitetura e Design de Secure Software