SCHEDA DELL'INSEGNAMENTO (SI)

SOFTWARE SECURITY

SSD ING-INF/05

DENOMINAZIONE DEL CORSO DI STUDIO: LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

ANNO ACCADEMICO 2023-2024

INFORMAZIONI GENERALI - DOCENTE

DOCENTE: ROBERTO NATELLA
TELEFONO: 081 7683820
EMAIL: ROBERTO.NATELLA@UNINA.IT

INFORMAZIONI GENERALI - ATTIVITÀ

INSEGNAMENTO INTEGRATO (EVENTUALE):	NESSUNO
MODULO (EVENTUALE):	NESSUNO
CANALE (EVENTUALE):	NESSUNO
ANNO DI CORSO (I, II, III):	II ANNO
SEMESTRE (I, II):	II SEMESTRE
CFU:	6 CFU

INSEGNAMENTI PROPEDEUTICI (se previsti dall'Ordinamento del CdS)

Nessuno

EVENTUALI PREREQUISITI

Sono sufficienti come prerequisiti le nozioni di base acquisite nel corso della Laurea Triennale in Ingegneria Informatica sui sistemi operativi, l'ingegneria del software, le basi di dati, le architetture dei calcolatori elettronici, e le reti di calcolatori.

OBIETTIVI FORMATIVI

L'obiettivo generale dell'insegnamento è di fornire agli studenti nozioni e competenze per la progettazione, lo sviluppo, la validazione e la gestione di sistemi software sicuri. Questo obiettivo è complementare altri insegnamenti della area di "Cyber-Security" della Laurea Magistrale in Ingegneria Informatica.

Gli obiettivi formativi dell'insegnamento includono:

- Fornire concetti avanzati sulle minacce, sulle vulnerabilità, e sugli attacchi ai sistemi software.
- Fornire competenze specialistiche di progettazione e sviluppo sicuro del software.
- Fornire competenze specialistiche per la validazione del software mediante analisi statica e dinamica.
- Fornire concetti avanzati sulle tecniche adottate nel software malevolo.
- Fornire competenze specialistiche per la prevenzione e la rilevazione di attacchi dovuti a software malevolo.

RISULTATI DI APPRENDIMENTO ATTESI (DESCRITTORI DI DUBLINO)

Conoscenza e capacità di comprensione

Lo studente deve dimostrare di conoscere e saper comprendere le minacce, le vulnerabilità, e gli attacchi nei sistemi software, e le tecniche utilizzate nel software malevolo. Il percorso formativo intende fornire agli studenti le conoscenze e gli strumenti metodologici per comprendere le attuali e future problematiche di sicurezza dei sistemi software a fronte della loro continua evoluzione nel tempo, e per identificare efficacemente le problematiche di sicurezza e comunicarle agli stakeholder coinvolti nella realizzazione e gestione dei sistemi software sicuri.

Capacità di applicare conoscenza e comprensione

Lo studente deve dimostrare di essere in grado di realizzare nuovi sistemi software sicuri, applicando le capacità e gli strumenti metodologici e operativi acquisiti nel percorso formativo nel contesto di tutte le fasi del processo di sviluppo software (analisi, progettazione, sviluppo, testing, gestione). Il percorso formativo è orientato a favorire la capacità di effettuare scelte di progetto dei sistemi software in maniera consapevole delle problematiche di sicurezza, e di saper riconoscere e risolvere le minacce, le vulnerabilità, gli attacchi, e le tecniche di software malevolo all'interno di un sistema software.

PROGRAMMA-SYLLABUS

Introduzione alla software security: Il ciclo di sviluppo di software sicuro, tassonomie di vulnerabilità (CVE, CVSS, CWE, OWASP), top design flaws & bugs.

Vulnerabilità software: Buffer overflows, vulnerabilità di memory & type safety e altri attacchi (format string, double free, out-of-bounds reads), return-oriented programming, control-flow integrity, undefined behaviors, web vulnerabilities (session hijacking, CSRF, XSS, SQL/command injection).

Progettazione sicura del software e certificazione: Processi e standard di software security (Microsoft Security Development Lifecycle, Common Criteria, OWASP, CERT), principi di progettazione sicura, attack e threat modeling, security requirements e abuse cases, secure software supply chain, DevSecOps.

Programmazione software sicura: Tecniche di difesa OS/compiler/language/framework-based, strategie di input validation, espressioni regolari, tecniche di error handling e resource management.

Tecniche di identificazione delle vulnerabilità: Analisi statica del software (type checking, compiler-based analysis, quality scanners, security scanners), analisi dinamica del software (source/binary code instrumentation, sanitizers), fuzzing (generation/mutation fuzzing, coverage-driven fuzzing, library-based fuzzing, protocol fuzzing).

Software malevolo: Forme di software malevolo (virus, RAT, keyloggers, rootkits, etc.), tattiche e tecniche degli attaccanti (Cyber Kill Chain, Diamond Model, MITRE ATT&CK), tecniche statiche e dinamiche di reverse engineering e di analisi di codice binario, malware signatures.

MATERIALE DIDATTICO

Libri di testo:

- Wenliang Du, ***"Computer & Internet Security: A Hands-on Approach"***, 2a edizione, 2019
- A. Takanen, J. DeMott, C. Miller, A. Kettunen, ***"Fuzzing for Software Security Testing and Quality Assurance"***, 2a edizione, 2018
- B. Chess, J. West, ***"Secure Programming with Static Analysis"***, 1a edizione, 2007
- M. Sikorski, A. Honig, ***"Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software"***, 1a edizione, 2012

MODALITÀ DI SVOLGIMENTO DELL'INSEGNAMENTO

Le attività dell'insegnamento saranno strutturate in:

- Lezioni frontali, per circa il 60% delle ore totali
- Attività di laboratorio per applicare e approfondire le conoscenze acquisite, per circa il 40% delle ore totali

Le lezioni si avvarranno di dimostrazioni pratiche basate su ambienti di virtualizzazione, e sugli strumenti di sicurezza software maggiormente utilizzati nei sistemi operativi Linux e Windows.

VERIFICA DI APPRENDIMENTO E CRITERI DI VALUTAZIONE

a) Modalità di esame:

L'esame si articola in prova	
scritta e orale	
solo scritta	
solo orale	
discussione di elaborato progettuale	X
altro	

b) Modalità di valutazione:

La valutazione avviene tramite la discussione di un elaborato progettuale, che lo studente sviluppa in autonomia da solo o in collaborazione con un piccolo gruppo di altri studenti. La discussione mira ad accertare la acquisita conoscenza da parte dello studente delle vulnerabilità e attacchi nei sistemi software, e la sua capacità di applicare operativamente tali conoscenze nella progettazione e/o nella validazione della sicurezza di un sistema software. La valutazione finale tiene in considerazione la complessità del progetto affrontato, la qualità del software eventualmente sviluppato e/o il grado di approfondimento nella validazione della sicurezza di un sistema software, e il grado di maturità dimostrato nella esposizione delle problematiche di sicurezza.