

**Machine Learning Security**

**Course Number:** SEC-140
**Duration:** 4 days

**Overview**

Attacks on machine learning applications are gaining momentum and protecting against a machine learning (ML) breach is essential. This Machine Learning Security training course teaches developers the skills they need to protect their ML applications. Students learn specialized secure coding skills and how to avoid the security pitfalls of the Python programming language.

**Note:** To ensure ample one-on-one engagement with the instructor, this class is capped at 12 people, overriding Accelebrate’s default cap of 15.

**Prerequisites**

Students should be Python developers working on machine learning systems.

**Materials**

All attendees receive comprehensive courseware.

**Software Needed on Each Student PC**

Attendees will not need to install any software on their computer for this class. The class will be conducted in a remote environment that Accelebrate will provide; students will only need a local computer with a web browser and a stable Internet connection. Any recent version of Microsoft Edge, Mozilla Firefox, or Google Chrome will be fine.

**Objectives**

* Understand essential cyber security concepts
* Learn about various aspects of machine learning security
* Discover the possible attacks and defense techniques in adversarial machine learning
* Identify vulnerabilities and their consequences
* Learn the security best practices in Python
* Understand Input validation approaches and principles
* Manage vulnerabilities in third-party components
* Understand how cryptography can support application security
* Learn how to use cryptographic APIs correctly in Python
* Understand security testing methodology and approaches
* Be familiar with common security testing techniques and tools

**Outline**

* Cyber Security Basics
	+ What is security?
	+ Threat and risk
	+ Cyber security threat types
	+ Consequences of insecure software
		- Constraints and the market
		- The dark side
	+ Categorization of bugs
		- The Seven Pernicious Kingdoms
		- Common Weakness Enumeration (CWE)
		- CWE Top 25 Most Dangerous Software Errors
		- Vulnerabilities in the environment and dependencies
* Cyber Security in Machine Learning
	+ ML-specific cyber security considerations
	+ What makes machine learning a valuable target?
	+ Possible consequences
	+ Inadvertent AI failures
	+ Some real-world abuse examples
	+ ML threat model
		- Creating a threat model for machine learning
		- Machine learning assets
		- Security requirements
		- Attack surface
		- Attacker model – resources, capabilities, goals
		- Confidentiality threats
		- Integrity threats (model)
		- Integrity threats (data, software)
		- Availability threats
		- Dealing with AI/ML threats in software security
* Using ML in Cyber Security
	+ Static code analysis and ML
	+ ML in fuzz testing
	+ ML in anomaly detection and network security
	+ Limitations of ML in security
* Malicious Use of AI and ML
	+ Social engineering attacks and media manipulation
	+ Vulnerability exploitation
	+ Malware automation
	+ Endpoint security evasion
* Adversarial Machine Learning
	+ Threats against machine learning
	+ Attacks against machine learning integrity
		- Poisoning attacks
		- Poisoning attacks against supervised learning
		- Poisoning attacks against unsupervised and reinforcement learning
		- Evasion attacks
		- Common white-box evasion attack algorithms
		- Common black-box evasion attack algorithm
		- Transferability of poisoning and evasion attacks
	+ Some defense techniques against adversarial samples
		- Adversarial training
		- Defensive distillation
		- Gradient masking
		- Feature squeezing
		- Using reformers on adversarial data
		- Caveats about the efficacy of current adversarial defenses
		- Simple practical defenses
	+ Attacks against machine learning confidentiality
		- Model extraction attacks
		- Defending against model extraction attacks
		- Model inversion attacks
		- Defending against model inversion attacks
* Denial of Service
	+ Denial of Service
	+ Resource exhaustion
	+ Cash overflow
	+ Flooding
	+ Algorithm complexity issues
	+ Denial of service in ML
		- Accuracy reduction attacks
		- Denial-of-information attacks
		- Catastrophic forgetting in neural networks
		- Resource exhaustion attacks against ML
		- Best practices for protecting availability in ML systems
* Input Validation Principles
	+ Blacklists and whitelists
	+ Data validation techniques
	+ What to validate – the attack surface
	+ Where to validate – defense in depth
	+ How to validate – validation vs transformations
	+ Output sanitization
	+ Encoding challenges
	+ Validation with regex
	+ Regular expression denial of service (ReDoS)
	+ Dealing with ReDoS
* Injection
	+ Injection principles
	+ Injection attacks
	+ SQL injection
	+ SQL injection basics
	+ Attack techniques
	+ Content-based blind SQL injection
	+ Time-based blind SQL injection
	+ SQL injection best practices
	+ Input validation
	+ Parameterized queries
	+ Additional considerations
	+ SQL injection and ORM
	+ Code injection
		- Code injection via input()
		- OS command injection
	+ General protection best practices
* Integer Handling Problems
	+ Representing signed numbers
	+ Integer visualization
	+ Integers in Python
	+ Integer overflow
	+ Integer overflow with ctypes and NumPy
	+ Other numeric problems
* Files and Streams
	+ Path traversal
	+ Path traversal-related examples
	+ Additional challenges in Windows
	+ Virtual resources
	+ Path traversal best practices
	+ Format string issues
* Unsafe Native Code
	+ Native code dependence
	+ Best practices for dealing with native code
* Input Validation in Machine Learning
	+ Misleading the machine learning mechanism
	+ Sanitizing data against poisoning and RONI
	+ Code vulnerabilities causing evasion, misprediction, or misclustering
	+ Typical ML input formats and their security
* Security Features
	+ Authentication
		- Authentication basics
		- Multi-factor authentication
		- Authentication weaknesses - spoofing
		- Password management
	+ Information exposure
		- Exposure through extracted data and aggregation
		- Privacy violation
		- System information leakage
		- Information exposure best practices
* Time and State
	+ Race conditions
		- File race condition
		- Avoiding race conditions in Python
	+ Mutual exclusion and locking
		- Deadlocks
	+ Synchronization and thread safety
* Errors
	+ Error handling
		- Returning a misleading status code
		- Information exposure through error reporting
	+ Exception handling
		- In the except, catch block. And now what?
		- Empty catch block
		- The danger of assert statements
* Using Vulnerable Components
	+ Assessing the environment
	+ Hardening
	+ Malicious packages in Python
	+ Vulnerability management
		- Patch management
		- Vulnerability management
		- Bug bounty programs
		- Vulnerability databases
		- Vulnerability rating – CVSS
		- DevOps, the build process and CI / CD
		- Dependency checking in Python
	+ ML Supply Chain Risks
		- Common ML system architectures
		- ML system architecture and the attack surface
		- Protecting data in transit – transport layer security
		- Protecting data in use – homomorphic encryption
		- Protecting data in use – differential privacy
		- Protecting data in use – multi-party computation
	+ ML frameworks and security
		- General security concerns about ML platforms
		- TensorFlow security issues and vulnerabilities
* Cryptography for Developers
	+ Cryptography basics
	+ Cryptography in Python
	+ Elementary algorithms
		- Random number generation
		- Hashing
	+ Confidentiality protection
		- Symmetric encryption
	+ Homomorphic encryption
		- Basics of homomorphic encryption
		- Types of homomorphic encryption
		- FHE in machine learning
	+ Integrity protection
		- Message Authentication Code (MAC)
		- Digital signature
	+ Public Key Infrastructure (PKI)
		- Some further key management challenges
		- Certificates
* Security Testing
	+ Security testing methodology
		- Security testing – goals and methodologies
		- Overview of security testing processes
		- Threat modeling
	+ Security testing techniques and tools
		- Code analysis
		- Dynamic analysis
* Wrap Up
	+ Secure coding principles
		- Principles of robust programming by Matt Bishop
		- Secure design principles of Saltzer and Schröder
	+ And now what?
		- Software security sources and further reading
		- Python resources
		- Machine learning security resources