Бауманский учебный центр «Специалист»

Программа курса Python: безопасность и разработка защищенных приложений, контейнеризация и мониторинг

Код курса: **ПИТБЕЗОП**

Тема	Ак. часов
Модуль 1. Введение в безопасность приложений • Триада CIA (Confidentiality, Integrity, Availability). • Разница между безопасностью инфраструктуры и безопасностью кода. • Обзор OWASP Top 10 (актуальная версия). • Анализ кейсов реальных взломов Python-приложений.	**4**
Модуль 2. Архитектура безопасного Python: стандарты, ревью и анализ кода • Корпоративные стандарты Python: PEP 8, PEP 257 (docstrings), PEP 484 (type hints). • Практики эффективного Code Review: чек-листы безопасности (OWASP Code Review Guide), уязвимые паттерны в Python. Статический и динамический анализ. • Документирование архитектурных решений: mkdocs, Changelog, Security. Внутренняя база знаний по уязвимостям. • **Практическая сессия:** Анализ и рефакторинг legacy-кода (поиск уязвимостей). • AI-ассистент для поиска уязвимостей в коде на Python.	**4**
Модуль 3. Валидация входных данных и Инъекции • Принцип «Не доверяй входным данным». • SQL-инъекции: защита через ORM (SQLAlchemy, Django ORM) и параметризированные запросы. • Command Injection: опасности os.system, subprocess, eval. • XSS (Cross-Site Scripting) в веб-приложениях на Python. • Использование библиотек для валидации (pydantic, marshmallow).	**4**
Модуль 4. Безопасное управление памятью и ресурсами в Python • Управление памятью в Python: сборщик мусора (gc), циклы ссылок, утечки памяти в долгоживущих процессах. tracemalloc для отладки. • Контекстные менеджеры (with): управление файлами, сетевыми соединениями, блокировками. Написание кастомных контекстных менеджеров. • Финализаторы и деструкторы: __del__ vs __exit__, опасности использования __del__.	**2**
Модуль 5. Защита данных, работа с ФС и обработка ошибок • Безопасная сериализация/десериализация: pickle (опасности, RCE), альтернативы (json, orjson, msgpack, pyarrow). Подпись данных (hmac). • Контроль доступа к файловой системе: защита от path traversal, безопасное открытие файлов (shutil, pathlib), временные файлы (tempfile). • Обработка исключений: безопасное проектирование fail-safe приложений, избегание голого except. • Логирование с учетом безопасности: structlog, python-json-logger, маскирование PII (passwords, tokens, emails). Защита от log injection. • **Практический пример:** Разработка защищенного модуля импорта/экспорта данных (JSON) с валидацией и безопасным логированием.	**4**

Модуль 6. Аутентификация, Авторизация и Управление сессиями • Хранение паролей: хеширование (bcrypt, argon2), соль, пейпинг. • Управление сессиями и JWT-токенами. • Защита от Brute-force и Credential Stuffing. **Практические примеры:** • Реализация безопасной системы регистрации/входа. • Настройка безопасных куки (HttpOnly, Secure, SameSite).	**4**
Модуль 7. Специфические уязвимости Python • Опасности сериализации: модуль pickle (RCE уязвимость). • Безопасная работа с yaml (опасность yaml.load). • Path Traversal (обход путей) при работе с файлами. • **Практический пример:** Замена pickle на безопасные альтернативы (json, protobuf).	**2**
Модуль 8. Конфигурация и Секреты • Почему нельзя хардкодить секреты в коде? • Переменные окружения (.env). • Использование секрет-менеджеров (HashiCorp Vault, AWS Secrets Manager). **Практика:** • Рефакторинг кода: вынос всех ключей в переменные окружения. • Использование библиотеки python-decouple или dynaconf.	**2**
Модуль 9. Инструменты автоматизации (DevSecOps) • Статический анализ кода (SAST). • Динамический анализ (DAST). • Настройка bandit (security linter for Python). • Введение в SonarQube.	**4**
Модуль 10. Контейнеризация защищенных Python-приложений • Docker для Python: написание защищенных Dockerfile (multistage, non-root user, distroless images). Минимизация attack surface. • Сканирование образов на уязвимости: Trivy, Grype, Docker Scout. • Безопасное хранение секретов: Docker Secrets, BuildKit (secrets mount). Никогда не класть .env в образ. • Оркестрация (Docker Compose): network policies, security contexts для Pythonмикросервисов. • **Практическая сессия:** Контейнеризация уязвимого приложения, сканирование образа, исправление Dockerfile.	**6**
Модуль 11. Практическая работа по разработке web-сервиса отвечающего всем современным стандартам безопасности	**4**
Аудиторная нагрузка в классе с преподавателем	**40 ак.ч. + 40 ак.ч. самостоятельно**

По окончании обучения на курсе проводится итоговая аттестация. Аттестация проводится в виде теста на последнем занятии или на основании оценок практических работ, выполняемых во время обучения на курсе.

Контакты центра

Для частных лиц
+7 (495) 232-32-16

Для организаций
+7 (495) 780-48-44

E-mail
info@specialist.ru