МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Волинський національний університет імені Лесі Українки
Факультет інформаційних технологій і математики
Кафедра комп'ютерних наук та кібербезпеки

СИЛАБУС

вибіркового освітнього компонента

БЕЗПЕЧНЕ ПРОГРАМУВАННЯ

підготовки бакалавра

Луцьк – 2026

Силабус освітнього компонента «Безпечне програмування**»** підготовки бакалавра, всіх галузей знань, всіх спеціальностей, за всіма освітніми програмами.

Розробник: Гаращенко В.В. старший викладач кафедри комп'ютерних наук та кібербезпеки.

Погоджено

Гарант освітньо-професійної програми:  Чернящук Н.Л.

Силабус освітнього компонента затверджено та погоджено на засіданні кафедри комп'ютерних наук та кібербезпеки

протокол № 6 від 15.01.2026 р.

Завідувач кафедри:  Гришанович Т. О.

І. Опис освітнього компонента

Найменування показників	**Характеристика освітнього компонента**
	Вибірковий
Денна форма навчання	Рік підготовки 2
150/5 кредитів	Семестр 4
	Лекції 10 год.
	Лабораторні 20 год.
	Самостійна робота 110 год.
ІНДЗ: немає	Консультації 10 год.
	Форма контролю: залік

ІІ. Інформація про викладача

ПІП Гаращенко Володимир Вікторович
Науковий ступінь -
Вчене звання -
Посада ст. викладач кафедри комп'ютерних наук та кібербезпеки
Контактна інформація ел. скринька: Harashchenko.Volodymyr@vnu.edu.ua
Дні занять https://ps.vnu.edu.ua/cgi-bin/timetable.cgi?n=700

ІІІ. Опис освітнього компонента

1. Анотація освітнього компонента

Силабус вибіркового освітнього компонента «Безпечне програмування» складено з урахуванням можливості формування індивідуальної освітньої траєкторії здобувачів освіти підготовки бакалавра. Дисципліна знайомить з основними принципами розробки захищеного програмного забезпечення, методами виявлення вразливостей та способами запобігання кібератакам на рівні вихідного коду.

2. Пререквізити та постреквізити

Пререквізити. Знання та вміння, отримані в результаті вивчення базових дисциплін з програмування (C/C++, Java, Python або C#), розуміння алгоритмів, структур даних та принципів роботи операційних систем. Знання основ комп'ютерних мереж буде перевагою.

Постреквізити. Знання та вміння, отримані в результаті вивчення дисципліни, можуть бути корисні при проектуванні архітектури складних систем, проведенні аудиту коду, тестуванні на проникнення (пентестінг), а також для професійної розробки продуктів, що відповідають сучасним стандартам кібербезпеки.

3. Мета і завдання освітнього компонента

Мета дисципліни: сформувати систему знань та практичних навичок, необхідних для створення програмних продуктів, стійких до зовнішніх і внутрішніх загроз. Курс

спрямований на те, щоб навчити розробника мислити як зловмисник для ефективного захисту системи, мінімізуючи ризики витоку даних або несанкціонованого доступу.

Завдання:

- навчитися ідентифікувати типові вразливості (Buffer Overflow, SQL-injection, XSS тощо);
- опанувати методи статичного (SAST) та динамічного (DAST) аналізу коду;
- вивчити стандарти безпечної розробки (наприклад, OWASP);
- навчитися застосовувати механізми шифрування, автентифікації та авторизації;
- розвинути навички написання «чистого» коду, який є стійким до експлуатації помилок проектування.

4. Soft skills

- **Критичне та аналітичне мислення** – здобувачі вчаться ставити під сумнів надійність кожного рядка коду та передбачати нестандартні сценарії поведінки користувачів.
- **Увага до деталей** – безпека часто залежить від однієї пропущеної перевірки чи неправильного типу даних; дисципліна вимагає максимальної концентрації.
- **Етична відповідальність** – розуміння правових та моральних аспектів роботи з даними користувачів і потенційними вразливостями систем.
- **Вирішення проблем (Problem Solving)** – пошук нестандартних способів захисту системи в умовах обмежених ресурсів або застарілих технологій.
- **Безперервне навчання (Lifelong Learning)** – сфера кібербезпеки змінюється щодня, тому важливо вміти швидко знаходити та засвоювати інформацію про нові загрози.
- **Комунікація та командна робота** – вміння аргументовано пояснити колегам ризики певного технічного рішення та спільно розробити план усунення вразливостей.
- **Стресостійкість** – здатність зберігати спокій та приймати зважені рішення під час виявлення критичних багів або імітації кібератак.

1. Структура освітнього компонента.

Назви змістових модулів і тем	Усього	Лек.	Лабор.	Сам. роб.	Конс.	Форма контролю / Бали
Змістовий модуль 1. Методи розробки захищеного програмного коду та аналіз вразливостей						
Тема 1. Керування пам'яттю та запобігання низькорівневим вразливостям. Програмна реалізація захисту від переповнення буфера (Stack/Heap Overflow). Використання безпечних аналогів стандартних функцій, механізми Stack Canaries та контроль меж масивів. Особливості безпечної роботи з покажчиками в мовах	23	2	4	15	2	Звіт/6/6

C/C++ та концепція безпеки пам'яті в Rust.						
Тема 2. Програмування механізмів валідації та фільтрації вхідних даних. Реалізація стратегій «білих списків» та регулярних виразів для перевірки користувацького вводу. Захист від ін'єкцій (SQL, Command Injection) через параметризацію запитів та використання ORM-систем. Контекстне кодування даних для запобігання XSS-атакам.	23	2	4	15	2	Звіт /6/6
Тема 3. Криптографічні примітиви та безпечне зберігання секретів у коді. Практичне застосування алгоритмів хешування (Argon2, bcrypt) з використанням «солі» для захисту паролів. Програмна реалізація симетричного та асиметричного шифрування. Робота з криптографічно стійкими генераторами випадкових чисел (CSPRNG) та менеджері секретів (Secret Managers).	23	2	4	15	2	Звіт /6/6
Тема 4. Безпека логіки виконання та обробка виключних ситуацій. Проектування надійних обробників помилок (Exception Handling), що не розкривають внутрішню архітектуру системи. Запобігання логічним вразливостям та станам гонитви (Race Conditions) у багатопотокових застосунках. Безпечна десеріалізація об'єктів та захист від зовнішніх сутностей XML (XXE).	23	2	4	15	2	Звіт /6/6
Тема 5. Автоматизований аналіз безпеки (SAST/DAST) та аудит коду. Інтеграція інструментів статичного аналізу (Linter, Security Scanners) у процес розробки. Проведення ручного аудиту коду (Code Review) на відповідність стандартам OWASP. Автоматизація	23	2	4	15	2	Звіт/6/6

пошуку вразливостей у сторонніх залежностях та бібліотеках.						
Разом за модулем 1	115	10	20	75	10	60
Контрольна робота				20		20
Тестування				15		20
Всього годин/Балів	150	10	20	110	10	100

2. Завдання для самостійного опрацювання.

№ з/п	**Тема (опрацювати)**	Кількість год.
1	Опрацювання та аналіз лекційного матеріалу	20
2	Опрацювання додаткових джерел та відео-роликів мережі Інтернет	15
3	Підготовка до лабораторних робіт	20
4	Підготовка до контрольної роботи	20
5	Підготовка до тестування	15
6	Підготовка відповідей на питання до заліку	20

IV. Політика оцінювання

Політика викладача щодо здобувача освіти

Усі учасники освітнього процесу повинні дотримуватись вимог чинного законодавства України, Статуту і Правил внутрішнього розпорядку ВНУ імені Лесі Українки, загально-прийнятих моральних принципів, правил поведінки та корпоративної культури; підтримувати атмосферу доброзичливості, відповідальності, порядності й толерантності. Атмосфера на заняттях повинна бути творчою, відкритою до конструктивної критики. Недопустимі запізнення на заняття; користування мобільним телефоном, планшетом чи іншими мобільними пристроями під час заняття; списування. Очікується, що всі студенти відвідають усі лекції і практичні заняття курсу.

Політика щодо академічної доброчесності

Дотримання академічної доброчесності здобувачами передбачає: самостійне виконання завдань поточного та підсумкового контролю (для осіб з особливим освітніми потребами ця вимога застосовується з урахуванням їх індивідуальних потреб і можливостей); посилання на джерела інформації у разі використання ідей, тверджень, відомостей; дотримання норм законодавства про авторське право; надання достовірної інформації про результати власної навчальної (наукової, творчої) діяльності.

Порушенням академічної доброчесності вважається: академічний плагіат, самоплагіат, фабрикація, фальсифікація, списування. За порушення академічної доброчесності здобувачі освіти можуть бути притягнені до такої академічної відповідальності: повторне проходження оцінювання; повторне проходження відповідного освітнього компонента освітньої програми.

Під час модульного та підсумкового контролю (заліку) студентам заборонено користуються такими засобами як мобільний телефон, планшет, конспект, навчальна література, інші джерела інформації, в тому числі Інтернет-ресурси.

Політика щодо дедлайнів та перескладання

Усі передбачені завдання мають бути виконані у встановлений термін. Несвоєчасно виконані завдання оцінюються на нижчу оцінку. Виключенням можуть бути завдання, які не вдалося зробити з поважних причин, в такому випадку студент може доробити вказані завдання у вказаний термін.

Якщо здобувач вищої освіти був відсутній на заняттях з будь-якої причини, то він (вона) вивчає матеріал самостійно, використовуючи навчальні посібники, конспекти лекцій, матеріали дистанційного курсу, у випадку розміщення його на платформі дистанційного навчання Moodle, виконує всі домашні завдання (завдання подані на самостійну роботу). Прозвітуватися про виконання завдань можна, використовуючи дистанційний курс, прикріпивши виконанні завдання у відповідні комірки та попередити викладача про здане завдання, або під час консультацій або надіслати виконане завдання на корпоративну пошту викладача. Зворотній зв'язок з викладачем для з'ясування всіх питань: використання форуму, чату дистанційного курсу, корпоративної пошти університету або відповідної бесіди у певному месенджері.

Можливість визнання результатів навчання, отриманих у формальній, неформальній та інформальній освіті

Під час вивчення освітнього компонента можливе визнання інших результатів навчання, отриманих у формальній, неформальній та/або інформальній освіті. Порядок визнання результатів навчання для здобувачів вищої освіти, набутих у: формальній освіті (академічна мобільність студентів на території України чи поза її межами, для студентів, які переводяться, поновлюються з інших ЗВО (вітчизняних чи іноземних); неформальній та/або інформальній освіті здійснюється згідно «ПОЛОЖЕННЯ про визнання результатів навчання, отриманих у формальній, неформальній та/або інформальній освіті у Волинському національному університеті імені Лесі Українки».

Можливість отримати додаткові (бонусні) бали

Відповідно до пункту 4.5 Положення про поточне та підсумкове оцінювання знань здобувачів вищої освіти Волинського національного університету імені Лесі Українки здобувачам освіти, які брали участь у роботі конференцій, підготовці наукових публікацій, в олімпіадах, конкурсах студентських наукових робіт, спортивних змаганнях, мистецьких конкурсах тощо й досягли значних результатів, може бути присуджено додаткові (бонусні) бали, які зараховуються як результати поточного контролю з відповідного ОК. Систему бонусних балів погоджує науково-методична комісія факультету (інституту).

І так, здобувачі освіти мають можливість отримати додаткові бали за вказаний вид робіт з ОК «Програмування» відповідно до таблиці витягу з протоколу № 1 засідання НМТ ФІТІМ ВНУ ім. Лесі Українки від 3.09.2025 р.

Системи бонусних балів для здобувачів освіти

Вид діяльності	Рівень / результат	Кількість бонусних балів
Студентські олімпіади	І місце	7
	ІІ місце	5
	ІІІ місце	3
	Участь в олімпіаді	2
Конкурси студентських наукових робіт	Диплом І ступеня	7
	Диплом ІІ ступеня	5
	Диплом ІІІ ступеня	3
Підготовка наукових публікацій	Публікація в WoS / Scopus	10
	Фахова стаття	7
	Нефахова стаття	5
	Публікація тез	2
Участь у конференціях	Виступ на конференції	2
Першість України з командного програмування	І місце	10
	ІІ місце	8
	ІІІ місце	6
	Участь	4

V. Підсумковий контроль

Підсумковий контроль з даної дисципліни передбачено у вигляді заліку.

Оцінювання здійснюється за 100-бальною шкалою. Оцінка включає в себе оцінювання всіх видів запланованої навчальної роботи протягом семестру: нараховується за якісне виконання лабораторних, індивідуальних робіт та виконання самостійної роботи. Максимальна кількість балів, яку може отримати студент під час поточного оцінювання, у випадку заліку, за семестр – 100 балів.

Залік викладач виставляє за результатами поточної роботи за умови, що здобувач освіти виконав ті види навчальної роботи, які визначено силабусом ОК. У випадку, якщо здобувач освіти не відвідував окремі аудиторні заняття (з поважних причин), на консультаціях він має право відпрацювати пропущені заняття та добрати ту кількість балів, яку було визначено на пропущені теми. У дату складання заліку викладач записує у відомість суму поточних балів, які здобувач освіти набрав під час поточної роботи (шкала від 0 до 100 балів).

У випадку, якщо здобувач освіти протягом поточної роботи набрав менше як 60 балів, він складає залік під час ліквідації академічної заборгованості. У цьому випадку бали, набрані під час поточного оцінювання анульовуються. Максимальна кількість балів на залік під час ліквідації академічної заборгованості, як правило, 100. У день складання заліку за основною сесією заборонено проводити додаткові опитування здобувача освіти, а також здобувач освіти не має права доздавати будь-який вид робіт, передбачений силабусом освітнього компоненту. На залік, під час ліквідації академічної заборгованості, здобувач отримує комплексне завдання, яке охоплює всі теми і всі форми контролю, які пропонувалися при вивченні освітнього компонента. Порядок проведення заліку-ліквідації – залік відбувається у вигляді виконання комплексного завдання.

Питання до заліку-ліквідації та приклади практичних завдань

1. Поясніть механізм виникнення вразливості переповнення буфера (Buffer Overflow) та її наслідки для виконання програми.
2. Які існують програмні засоби захисту від переповнення стека на рівні компілятора та операційної системи (Stack Canaries, ASLR, DEP)?

3. Чому функції gets(), strcpy() та sprintf() вважаються небезпечними та які захищені аналоги слід використовувати замість них?
4. Опишіть концепцію безпечного керування пам'яттю (Memory Safety) та як вона реалізована в сучасних мовах програмування (наприклад, Rust або Java).
5. Сформулюйте основні принципи валідації вхідних даних: різниця між «білими» та «чорними» списками фільтрації.
6. Яким чином використання параметризованих запитів (Prepared Statements) повністю нівелює загрозу SQL-ін’єкцій?
7. Що таке контекстне кодування (Context-aware Encoding) і як воно запобігає виконанню шкідливих скриптів (XSS) у браузері користувача?
8. Поясніть ризики використання динамічної генерації команд ОС на основі вводу користувача та методи їхнього безпечного виконання.
9. У чому полягає небезпека «жорсткого кодування» (Hardcoding) конфіденційних даних (паролів, ключів) у вихідному коді програми?
10. Які криптографічні вимоги висуваються до генераторів випадкових чисел (CSPRNG) у задачах безпеки?
11. Опишіть алгоритм безпечного збереження паролів: роль «солі» (Salt), ітерацій та спеціалізованих функцій хешування (Argon2, bcrypt).
12. Які типові помилки допускають програмісти при реалізації протоколів шифрування даних (TLS/SSL) у своїх застосунках?
13. Як неправильна обробка виключних ситуацій (Exception Handling) може допомогти зловмиснику отримати інформацію про внутрішню структуру системи?
14. Поясніть сутність вразливості небезпечної десеріалізації (Insecure Deserialization) та методи захисту від неї.
15. Що таке атака типу Race Condition (стан гонитви) і як використання м'ютексів та атомарних операцій забезпечує безпеку багатопотокових програм?
16. Опишіть механізм атаки XXE (XML External Entity) та способи налаштування XML-парсерів для захисту від неї.
17. У чому полягає різниця між статичним (SAST) та динамічним (DAST) аналізом безпеки програмного коду?
18. Чому перевірка сторонніх залежностей та бібліотек (SCA — Software Composition Analysis) є критично важливою для безпеки кінцевого продукту?
19. Які основні вектори атак на API (наприклад, Broken Object Level Authorization) повинен враховувати розробник під час проектування інтерфейсів?
20. Опишіть роль та основні етапи аудиту коду (Security Code Review) у життєвому циклі розробки захищеного ПЗ.

Приклади практичних завдань

1. **Безпечне керування пам'яттю та запобігання Buffer Overflow**
   Контекст: На мові C++ надано функцію, яка копіює дані з мережевого буфера в локальний масив фіксованого розміру за допомогою `strcpy()`.

- Розрахуйте обсяг даних, який призведе до переповнення стека та перезапису адреси повернення.

- Проведіть рефакторинг коду: замінюйте небезпечну функцію на `strncpy()` або використовуйте контейнер `std::string`, який автоматично керує пам'яттю.
- Додайте перевірку довжини вхідних даних перед початком будь-яких маніпуляцій з буфером.

2. **Аудит та виправлення вразливостей системи авторизації**

Опис ситуації: Вам надано фрагмент коду (псевдокод), який відповідає за перевірку пароля під час входу в систему. У цьому коді допущено три критичні помилки безпеки.

```
# ВРАЗЛИВИЙ КОД
def login(user_input_name, user_input_password):
    # 1. Отримуємо дані з БД (паролі зберігаються у відкритому вигляді!)
    user = db.execute("SELECT * FROM users WHERE username = '" + user_input_name + "'")

    if user:
        if user['password'] == user_input_password:
            print("Доступ дозволено!")
        else:
            print("Помилка: невірний пароль для користувача " + user_input_name)
    else:
        print("Користувача не існує")
```

Завдання:

1. **Знайдіть помилки:**
   - **Помилка 1 (SQL Injection):** Поясніть, що станеться, якщо ввести в поле імені: `' OR '1'='1`.
   - **Помилка 2 (Конфіденційність):** Чому не можна зберігати паролі у відкритому вигляді (Plain text)?
   - **Помилка 3 (Витік інформації):** Чому небезпечно виводити різні повідомлення («невірний пароль» vs «користувача не існує»)?

   Підказка: це допомагає зловмиснику підібрати існуючі логіни.
2. **Виправте код (Рефакторинг):**
   - Замініть прямий запит до БД на **параметризований**.
   - Замініть перевірку відкритого пароля на перевірку **хешу** (наприклад, використовуючи функцію `check_password_hash`).
   - Зробіть повідомлення про помилку **універсальним** (наприклад: «Невірний логін або пароль»).
3. **Результат:** Надайте виправлений варіант коду, де ці три вразливості відсутні.

Шкала оцінювання знань здобувачів освіти з освітніх компонентів, де формою контролю є залік

Оцінка в балах	Лінгвістична оцінка

90–100	Зараховано
82–89	
75–81	
67–74	
60–66	
1–59	Незараховано (необхідне перескладання)

VII. Рекомендована література та інтернет-ресурси

1. OWASP Top 10 : [Електронний ресурс]. — URL: https://owasp.org/www-project-top-ten/
2. SEI CERT C Coding Standard : [Електронний ресурс] / Software Engineering Institute, Carnegie Mellon University. — URL: https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards
3. CWE – Common Weakness Enumeration : [Електронний ресурс] / MITRE. — URL: https://cwe.mitre.org/
4. Google Secure Coding Guidelines : [Електронний ресурс] / Google Engineering Practices. — URL: https://google.github.io/eng-practices/
5. Microsoft Security Development Lifecycle (SDL) : [Електронний ресурс]. — URL: https://www.microsoft.com/en-us/securityengineering/sdl/
6. Гайдамакін О. В. Технології безпечного програмування : навч. посіб. / О. В. Гайдамакін. — Львів : Вид-во Львівської політехніки, 2022. — 180 с.
7. Гололобов В. В. Методи та засоби безпечного програмування : навч. посіб. / В. В. Гололобов, М. С. Сєвєрінов. — Харків : ХНУРЕ, 2021. — 124 с.
8. Гончаренко Ю. Ю., Кушнір О. П. Аналіз вразливостей програмного забезпечення та методи безпечного програмування // Системи обробки інформації. — 2018. — № 4 (155). — С. 45–52.