Курс “Комп’ютерна вірусологія“

Безпека програмного коду

Ст. викладач
каф. Інформатики ФІ
Кирієнко Оксана Валентинівна
ok.kyrienko@gmail.com

Безпечне програмування

- Як зробити корисну програму?
- Що в нашій програмі може призвести до вразливості?
- Як це попередити?

Безпечне програмування

Ресурси присвячені безпеці застосунків:

- The Open Web Application Security Project - https://www.owasp.org/index.php/Category:Technology
- Common Weakness Enumeration - http://cwe.mitre.org/

Безпечне програмування

Не винаходьте колесо!

Ваші експерименти можуть дорого коштувати. Користуйтесь готовими перевіреними **бібліотеками**!

Якщо, за умовами використання сторонніх бібліотек неможливо, **напишіть свою бібліотеку**, ретельно протестуйте її самі та передайте на тестування спеціалістам. Використовуйте можливості цієї бібліотеки в своєму коді.

Безпечне програмування

Поспішайте повільно!

- Самотестування - The Personal Software Process – методологія для розробників програмного забезпечення - http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=5283
- Пошук дурних помилок – утиліти автоматизованого аналізу – https://www.owasp.org/index.php/Static_Code_Analysis

Безпечне програмування

- Тестування на безпеку!
  - Authentication
  - Authorization
  - Session Management
  - Input Validation
  - Error Handling
  - Secure Deployment
  - Cryptographic Controls
- Ревизия кода - Code Review:
  - Buffer Overruns and Overflows
  - OS Injection
  - SQL Injection
  - Data Validation
  - Cross-Site Scripting
  - Cross-Site Request Forgery
  - Logging Issues
  - Session Integrity
  - Race Conditions

Проектування безпеки

Основні принципи безпеки:

- простота механізмів (economy of mechanism);
- безпека за замовчуванням (fail-safe defaults);
- всепроникний захист (complete mediation);
- відкритий дизайн (open design);
- розподіл повноважень (separation of privilege);
- мінімум привілей (least privilege);
- мінімізація розподілу ресурсів (least common mechanism);
- психологічна прийнятність (psychological acceptability).

The Protection of Information in Computer Systems – сформульовані в 1975 році.

Проектування безпеки

Чи залежить безпека програми від структури програми?

- вірогідність зробити в ній помилку;
- чи стане ця помилка вразливістю;
- наскільки серйозна буде ця вразливість.

Наприклад:

яка помилка призведе до більш серйозних наслідків:

в коді, який має доступ до важливих даних, чи в коді, який такого доступу не має?

Типові методи реалізації безпеки програм - шаблони

- шаблони безпеки трьох різних рівнів: шаблони архітектури, шаблони дизайну и шаблони реалізації. *Security Design Patterns* розроблені в 2009 році *Software Engineering Institute* http://resources.sei.cmu.edu/asset_files/TechnicalReport/2009_005_001_15110.pdf

- структурні паттерни безпечних программ *Security Design Patterns* розроблена в 2004 році *Open Group* - http://pubs.opengroup.org/onlinepubs/9299698999/toc.pdf

Пошук вразливостей програм

- Переповнювання буфера.
- Неконтрольоване введення.
- Помилки синхронізації
- …

Відшуковувати "дірки", недекларовані властивості програмного забезпечення, "злоякісні фрагменти коду" вірусних програм, досліджувати захист систем можна за допомогою спеціальних відлагоджувальних засобів.

Пошук вразливостей програм - **Переповнювання буферу.**

Ціль такої атаки:

- читання секретних змінних
- модифікація секретних змінних
- передача управління на секретну функцію програми
- передача управління на код, який передається жертві зловмисником

Наслідки переповнення буферу

- змінюється логіка виконання програмних іструкцій
- програма аварійно завершується, «зависає», вилітає
- нічого не відбувається

Як дослідити програму?

- високорівневе дослідження (загальноприйнятий підхід), - передбачає дослідження вихідних текстів програм, описаних мовами високого рівня,
- низькорівневе дослідження (альтернативний підхід), полягає у дослідженні вихідних текстів програм (обернена обробка), отриманих з виконуваних кодів шляхом дизасемблювання.

Відновлення початкового коду

Основні методи пошуку вразливих місць:

- **„Біла скринька”** — аналіз передусім вихідного коду
- **„Чорна скринька”** — дослідження за допомогою тестових даних
- **„Сіра скринька”** — поєднання двох підходів (приклад — запуск програми в середовищі відлагоджувача і подання на вхід тестових даних)

Відновлення початкового коду

Інструменти відновлення початкового коду:

- відлагоджувачі (наприклад, SoftICE чи його „спадкоємець” Syser)
- дизасемблер (IDA, IDA Pro, HexRays)
- декомпілятор

Див. *Крис Касперски* **Техника хакерских атак. Фундаментальные основы хакерства**

Искусство дизассемблирования

Крис Касперски, Ева Рокко
Искусство дизассемблирования
БХВ-Петербург, 2008

Приклад дизасемблювання

https://habrahabr.ru/post/235487/

DirCrypt - один з найбільш злісних

перетворюється на муку.

Методи протидії дизасемблюванню

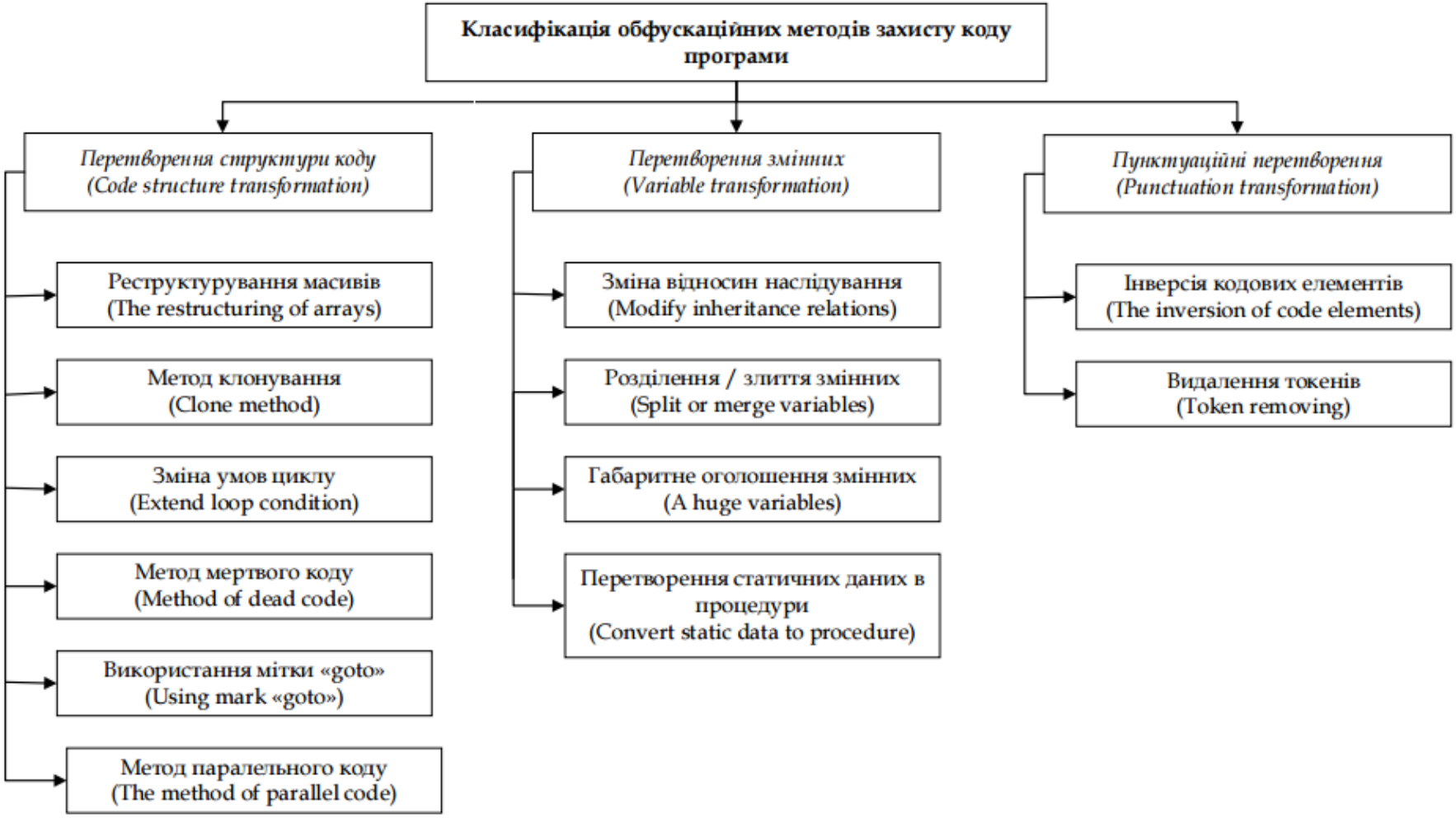
- Архівування.
- Засмічення коду програми.
- Використання мультипоточності.
- Придушення спроб зміни операційного середовища.
- Протидія встановленню контрольних точок.
- Шифрування та шифрування і дешифрування (динамічна зміна коду).
- Використання віртуальних машин.
- …

Методи протидії дизасемблюванню
обфускація

Заплутування коду програми - зміна вихідного тексту таким чином, що зберігається функціональність програми, але ускладнюється її аналіз, розуміння алгоритмів роботи та можлива модифікація програми.

Даний процес можливий за допомогою деобфускаційних програм: ***IDA Pro, Ariadne, JSNice, iMPROVE .NET, De4dot, ...*** які дозволяють читати та модифікувати виконувані файли, переводити їх у машинний код або перетворювати частину коду в зручне для аналізу проміжне представлення.

Класифікація обфускаційних методів захисту коду програми



АЛЕ …

Практично всі методи захисту вже відомі.

Сподіватися на 100% безпеку - не можна.

Проте, можна і потрібно розробляти нові ефективні системи захисту.