Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЁТ
к лабораторной работе №5
на тему

ЗАЩИТА ОТ АТАКИ НА ПЕРЕПОЛНЕНИЕ НА ПЕРЕПОЛНЕНИЕ БУФЕРА

Выполнил: студент гр.253504
Фроленко К.Ю.
Проверил: ассистент кафедры информатики
Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современных программных системах безопасность обмена данными и корректное управление памятью являются фундаментальными задачами. Особое внимание уделяется уязвимостям, связанным с переполнением буфера, которые могут привести к непредсказуемому поведению программ и даже к выполнению произвольного кода. Такие атаки являются классическим примером эксплуатации ошибки, возникающей в результате небрежного обращения с памятью, особенно в языках программирования низкого уровня, таких как C и C++.

Данная работа посвящена демонстрации принципов защиты от переполнения буфера. Для этого разработана программа, позволяющая в одном варианте работать с использованием небезопасного метода копирования строк, а в другом – с применением корректной техники, предотвращающей выход за границы выделенной памяти. Основная идея заключается в сравнении двух подходов: уязвимого режима, где копирование осуществляется с помощью функции `strcpy`, и защищённого режима, использующего `strncpy` с обязательной установкой завершающего нулевого символа.

Целью данной работы является не только наглядное представление проблемы, но и демонстрация того, как даже простейшие меры контроля позволяют значительно повысить устойчивость приложения к потенциальным атакам. Такой подход имеет большое значение в условиях современного развития информационных технологий, когда вопросы безопасности становятся приоритетными для разработчиков.

1 ФОРМУЛИРОВКА ЗАДАЧИ

Основной задачей исследования является разработка демонстрационной программы, позволяющей сравнить два метода копирования строк в контексте защиты от переполнения буфера. Программа должна обеспечить возможность выбора между уязвимым режимом, в котором используется функция `strcpy`, и защищённым режимом с применением `strncpy`.

В ходе работы пользователь вводит строку, предназначенную для имитации атаки. В случае уязвимого режима программа копирует всю строку в буфер фиксированного размера, что приводит к переполнению, если введённая строка превышает выделенный объем памяти. Напротив, защищённый режим демонстрирует корректное поведение: функция `strncpy` ограничивает копирование заданным количеством символов, что предотвращает возможность повреждения соседних областей памяти. Таким образом, итоговое решение позволяет наглядно сравнить последствия неправильного и правильного управления памятью, а также понять, почему даже простые методы защиты имеют большое значение для обеспечения безопасности приложения.

2 ПРИМЕР ВЫПОЛНЕНИЯ ПРОГРАММЫ

При запуске программы пользователь получает возможность выбрать режим работы. На экране отображается приглашение, предлагающее ввести символ «u» для работы в уязвимом режиме или «p» для защищённого режима. Далее программа запрашивает строку, которая имитирует атаку.

На Рисунке 1 представлена демонстрация работы программы в уязвимом режиме. Здесь функция, использующая `strcpy`, копирует всю введённую строку, независимо от её длины. Результатом такого подхода является полное копирование строки, что может привести к переполнению буфера и повреждению данных, находящихся в соседних областях памяти. На скриншоте видно, как вывод программы содержит всю строку, что свидетельствует о наличии уязвимости.

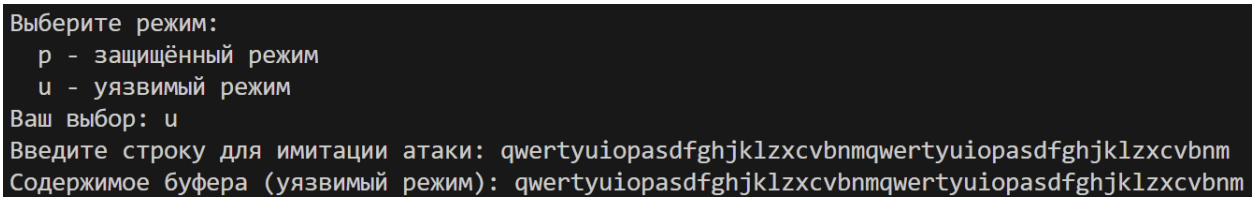

Рисунок 1 – Демонстрация работы уязвимого режима

На Рисунке 2 изображён результат выполнения программы в защищённом режиме. При выборе данного режима функция, использующая `strncpy`, копирует в буфер лишь ограниченное количество символов. Таким образом, даже при попытке передачи строки, значительно превышающей допустимый объём, копирование осуществляется корректно, а избыточные данные обрезаются. Это позволяет избежать переполнения и последующего повреждения памяти. На приведённом скриншоте можно наблюдать, что вывод программы содержит только часть исходной строки, что и является желаемым поведением.

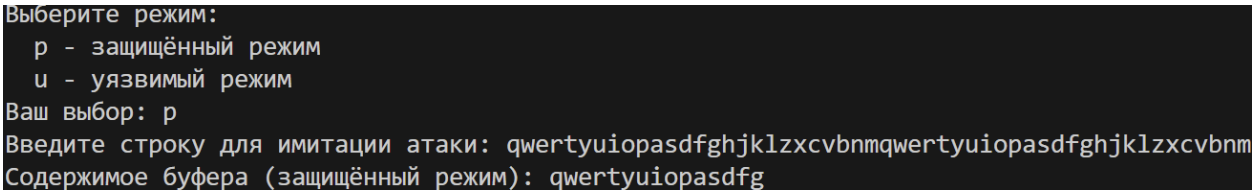

Рисунок 2 – Демонстрация работы защищённого режима

Эти примеры демонстрируют наглядное отличие двух подходов: небезопасного копирования, способного привести к критическим ошибкам, и корректного метода, обеспечивающего защиту от атак на переполнение буфера. Такой сравнительный анализ позволяет глубже понять важность соблюдения правил управления памятью в программировании.

ЗАКЛЮЧЕНИЕ

В ходе данной работы было реализовано демонстрационное приложение, позволяющее исследовать проблему переполнения буфера и оценить эффективность простейших методов защиты от данной уязвимости. Проведённое экспериментальное тестирование убедительно показало, что использование небезопасной функции `strcpy` приводит к полному копированию входных данных, даже если они превышают размер выделенного буфера, что создаёт угрозу для корректного функционирования программы. В свою очередь, применение функции `strncpy` с соответствующими ограничениями гарантирует, что в буфер будет скопировано только допустимое количество символов, что значительно снижает риск возникновения переполнения.

Таким образом, выполненная работа не только демонстрирует теоретические аспекты проблемы, но и подтверждает практическую значимость использования проверенных методов защиты в программировании. Наглядное сравнение двух режимов позволяет осознать, что даже малейшие недочёты в управлении памятью могут привести к серьёзным проблемам, а правильная реализация механизмов защиты является залогом создания надёжного и безопасного программного обеспечения.