Министерство науки и высшего образования
Российской Федерации
Национальный исследовательский ядерный университет
«МИФИ»

Сборник научных работ студентов и аспирантов Института интеллектуальных кибернетических систем НИЯУ МИФИ

к 70-летнему Юбилею
ВМУ-ЭВУСА-В-К-КиБ-ИИКС

Москва 2024

УДК 004:007.5
ББК 32.81
С23

Сборник научных работ студентов и аспирантов Института интеллектуальных кибернетических систем НИЯУ МИФИ. – М.: НИЯУ МИФИ, 2024.

В сборник вошли статьи студентов и аспирантов ИИКС, посвященные таким перспективным направлениям развития компьютерных технологий, как информационная безопасность, стохастические методы защиты информации, искусственный интеллект и большие данные, киберфизические системы.

Большинство статей сборника написаны по материалам отчетов о самостоятельной работе по различным учебным дисциплинам, по материалам отчетов об учебно-исследовательской работе и по результатам выпускных квалификационных работ.

Издание адресовано научным работникам, преподавателям, аспирантам, студентам высших учебных заведений.

Составитель сборника
д.т.н., профессор Иванов Михаил Александрович

Рецензенты:
к.т.н., доцент Чугунков Илья Владимирович,
к.т.н., доцент Вавренюк Александр Борисович

ISBN 978-5-7262-3090-0

СОДЕРЖАНИЕ

УДК 004.421.5

ПРОГРАММНАЯ МОДЕЛЬ СТОХАСТИЧЕСКОГО КОДЕКА $(n, k, 2^{64})$-КОДА

К.В. Агиевец[1], М.А. Кондахчан[2]
[1]Студент группы Б21-563 НИЯУ МИФИ, agievets.k.v@gmail.com
[2]Студент группы Б21-503 НИЯУ МИФИ, mikarkon@gmail.com

Аннотация. При передаче данных необходимо решать задачи обнаружения и исправления ошибок, возникающих из-за действия помех в канале связи, обеспечения секретности информации и имитозащиты. Традиционные системы передачи данных эти задачи решают с использованием соответственно помехоустойчивого кодирования, шифрования, формирования (на стороне отправителя) и проверки (на стороне получателя) криптографического контрольного кода целостности (имитовставки). По этой причине они громоздкие и недостаточно эффективные.

Целью данной работы является программная реализация схемы передачи данных, обеспечивающей универсальную защиту передаваемых данных, которая решает перечисленные выше задачи, обеспечивая при этом заданную вероятность правильного приема информации в случае случайных ошибок в канале связи.

Полученные результаты: программно реализован алгоритм работы $(8, 4, 2^{64})$-кода, проведено тестирование процессов кодирования, прямого и обратного стохастического преобразования и декодирования в случае ошибок различной кратности.

Ключевые слова: стохастическое кодирование, генератор псевдослучайных чисел, стохастическое преобразование, преобразованный канал связи.

Введение

Свойства реальных каналов связи многообразны и редко соответствуют модели двоичного симметричного канала. В итоге невозможно рассчитать вероятности наступления событий, приводящих к ошибкам декодирования. Для того, чтобы обеспечить наперед заданную вероятность правильного приема информации, необходимо создать преобразованный (виртуальный) дискретный канал. Ни один классический код не сможет работать при использовании виртуального канала, так как при стохастическом преобразовании теряются свойства кода. Следовательно, необходим новый код, учитывающий факт наличия преобразованного канала связи. Главным результатом применения стохастического метода кодирования данных является решение всех трех задач защиты информации: обнаружения и исправления ошибок, возникающих из-за действия помех в канале связи, обеспечения секретности информации и защиту от навязывания ложных данных (имитозащиту) в рамках одного алгоритма обработки информации [1, 2].

Стохастический (8, 4, 2^{64})-код

Теория кодирования и криптография преследуют противоположные цели. При кодировании сообщение представляется в виде, который допускает в процессе передачи данных появления некоторого количества наиболее вероятных ошибок. Можно считать, что при кодировании ясность сообщения повышается. При шифровании же уменьшается ясность сообщения, скрывается его смысл от злоумышленника. Однако, совмещая теорию кодирования и криптографию, можно решить вышеперечисленные задачи защиты информации, используя один алгоритм.

Использование криптографии позволяет создать Q-ичный (в данной работе $Q = 2^{64}$) симметричный канал передачи данных, при этом используется стохастический код, работающий с Q-ичными символами.

Существует целое семейство стохастических (n, k, Q)-кодов, разработанных С.А. Осмоловским, для каналов с различными вероятностями возникновения помех [1]. В данной работе рассматривается (8, 4, 2^{64})-код, при этом блоки стохастического преобразования реализованы с использованием криптоалгоритма Магма [3, 4].

Принцип работы стохастического кода при обнаружении и исправлении ошибок схож с тем, который используется в современной теории линейных двоичных кодов. Он описывается следующим образом [1, 5, 6]:

- выявление факта искажения информации,
- локализация неискаженных Q-ичных символов,
- стирание искаженных символов,
- их восстановление на основании выполнившихся проверочных соотношений.

За основу взят двоичный (8, 4)-код Хэмминга с дополнительной проверкой на четность и минимальным кодовым расстоянием $d = 4$, позволяющий при двоичной реализации одновременно исправлять одну ошибку и обнаруживать две. Формирование проверочных соотношений задается матрицей $\boldsymbol{H}_{8,\,4}$:

$$\boldsymbol{H}_{8,\,4} = \begin{vmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}.$$

Сообщение

$$a = a_1 a_2 a_3 a_4$$

кодируется в кодовое слово

$$x = x_1x_2x_3x_4x_5x_6x_7x_8 = a_1a_2a_3a_4b_1b_2b_3b_4,$$

где a_i и b_i – соответственно информационные и проверочные Q-ичные символы. Принцип формирования избыточных Q-ичных символов:

$$b_1 = a_2 \oplus a_3 \oplus a_4,$$
$$b_2 = a_1 \oplus a_3 \oplus a_4,$$
$$b_3 = a_1 \oplus a_2 \oplus a_4,$$
$$b_4 = a_1 \oplus a_2 \oplus a_3.$$

Ниже представлены четыре проверочных соотношения (1)-(4), задаваемые матрицей $\boldsymbol{H}_{8,\,4}$, и все их линейные комбинации (5)-(15)

$$a_2 \oplus a_3 \oplus a_4 \oplus b_1 = 0, \quad (1)$$
$$a_1 \oplus a_3 \oplus a_4 \oplus b_2 = 0, \quad (2)$$
$$a_1 \oplus a_2 \oplus a_4 \oplus b_3 = 0, \quad (3)$$
$$a_1 \oplus a_2 \oplus a_3 \oplus b_4 = 0, \quad (4)$$
$$a_1 \oplus a_2 \oplus b_1 \oplus b_2 = 0, \quad (5)$$
$$a_2 \oplus a_3 \oplus b_2 \oplus b_3 = 0, \quad (6)$$
$$a_3 \oplus a_4 \oplus a_3 \oplus b_4 = 0, \quad (7)$$
$$a_1 \oplus a_3 \oplus b_1 \oplus b_3 = 0, \quad (8)$$
$$a_1 \oplus a_4 \oplus b_1 \oplus b_4 = 0, \quad (9)$$
$$a_2 \oplus a_4 \oplus b_2 \oplus b_4 = 0, \quad (10)$$
$$a_4 \oplus b_1 \oplus b_2 \oplus b_3 = 0, \quad (11)$$
$$a_3 \oplus b_1 \oplus b_2 \oplus b_4 = 0, \quad (12)$$
$$a_2 \oplus b_1 \oplus b_3 \oplus b_4 = 0, \quad (13)$$
$$a_1 \oplus b_2 \oplus b_3 \oplus b_4 = 0, \quad (14)$$
$$a_1 \oplus a_2 \oplus a_3 \oplus a_4 \oplus b_1 \oplus b_2 \oplus b_3 \oplus b_4 = 0. \quad (15)$$

При декодировании сообщения проверяется выполнение этих 15-ти соотношений. (8, 4, 2^{64})-код как и любой другой (8, 4, Q)-код позволяет исправлять два стирания, т.е. если известно шесть правильных 64-разрядных символов, то остальные символы можно исправить, выразив через правильно принятые символы. Схема алгоритма декодирования представлена на рис. 1.

Пусть в процессе передаче произошла ошибка в символе a_1. Тогда выполняются соотношения под номерами (1), (6), (7), (10), (11), (12), (13), так как в них не содержится символ a_1. Следовательно, при одиночной ошибке $N_c = 7$. Искаженный 64-разрядный символ можно восстановить, воспользовавшись одним из соотношений, куда входит a_1. Например, с помощью соотношения под номером (2):

$$a_1 = a_3 \oplus a_4 \oplus b_2.$$

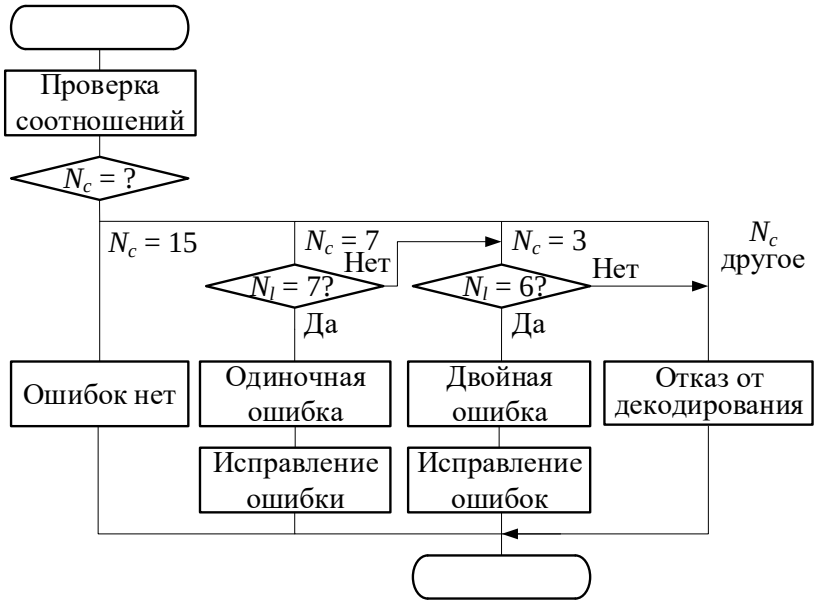

Рисунок 1 – Схема алгоритма декодирования стохастического $(8, 4, 2^{64})$-кода, где N_c – количество выполнившихся проверочных соотношений, N_l – число локализованных символов.

Пусть в процессе передачи произошла двойная ошибка и исказились символы a_1 и a_2. Точно выполняются соотношения под номерами (7), (11), (12). Однако ошибки могут взаимно уничтожиться в соотношениях, в которые a_1 и a_2 входят одновременно: (3), (4), (5), (15), при этом вероятность маскирования ошибок можно рассчитать. Соответственно, при двойной ошибке $N_c \in \{3, 7\}$. Сочетание соотношений (3), (4), (5), (7), (11), (12), (15) позволяет локализовать лишь 6 символов. Но опираться необходимо на выполнение соотношений под номерами (7), (11), (12), потому что в них не содержатся ни a_1, ни a_2. Символ a_1 после стирания можно восстановить с помощью соотношения под номером (2), а символ a_2 – с помощью соотношения под номером (1):

$$a_1 = a_3 \oplus a_4 \oplus b_2,$$
$$a_2 = a_3 \oplus a_4 \oplus b_1.$$

Рассмотрим блоки прямого и обратного стохастического преобразования (соответственно R и R^{-1}), обеспечивающие появление преобразованного канала связи с требуемыми свойствами: когда все преобразованные вектора ошибок Q-ичных символов на выходе блока R^{-1} равновероятны. На входе реального канала связи используется блок R, при этом параметр стохастического преобразования снимается с выхода генератора псевдослучайных чисел (ГПСЧ), схема которого показана на рис. 2. Совокупность блока R и ГПСЧ на стороне отправителя и блока R^{-1} и ГПСЧ на стороне получателя (рис. 2, 3) по сути образуют схему шифрования методом гаммиро-

вания, которая в отличии от традиционной на основе операции XOR не позволяет злоумышленнику вносить предсказуемые изменения в преобразованные данные (по сути шифртекст).

ГПСЧ реализован по схеме Counter Mode с использованием счетчика Q, реализованного на регистре сдвига с линейной обратной связью, и функции E зашифрования криптоалгоритма Магма [3, 4] в качестве функции выхода F_{out} генератора (рис. 2). Блоки R и R^{-1} реализуют функции E и D (соответственно за- и расшифрования) криптоалгоритма Магма (рис. 2, 3). Выбор криптоалгоритма определила требуемая разрядность Q-ичных символов. Имитозащита обеспечивается за счет избыточности, вносимой кодером.

На рис. 3 показана общая схема стохастического кодека, где e – вектор ошибок, действующий в реальном канале связи, e' – преобразованный вектор ошибок.

Результаты моделирования работы стохастического кодека

Результаты тестирования программной реализации стохастического $(8, 4, 2^{64})$-кода при внесении ошибок в информационные и проверочные Q-ичные символы различной кратности приведены на рис. 4-10.

Заключение

Реализация рассмотренной схемы передачи данных позволяет обеспечить универсальную защиту передаваемой информации в рамках единого алгоритма. Помехозащищенность достигается за счет использования стохастического $(8, 4, 2^{64})$-кода, построенного на основе (8, 4)-кода Хэмминга с дополнительной проверкой на четность. При этом за счет реализации преобразованного канала связи на основе алгоритмов, специфицированных в ГОСТ 34.12-2018, обеспечивается секретность передаваемой информации и фиксация факта умышленных ее искажений.

Перспективными направлениями для развития изложенного в данной работе решения являются:

- реализация и тестирование других стохастических (n, k, Q)-кодов;
- реализация систем передачи данных на основе кодов Рида-Соломона, БЧХ и LDPC с целью сравнения эффективности работы различных кодеков в режиме обнаружения и исправления ошибок, вызванных помехами в каналах связи;
- Light-Weight реализация преобразованного канала связи, ориентированная на использование в RFID- и IoT-системах.

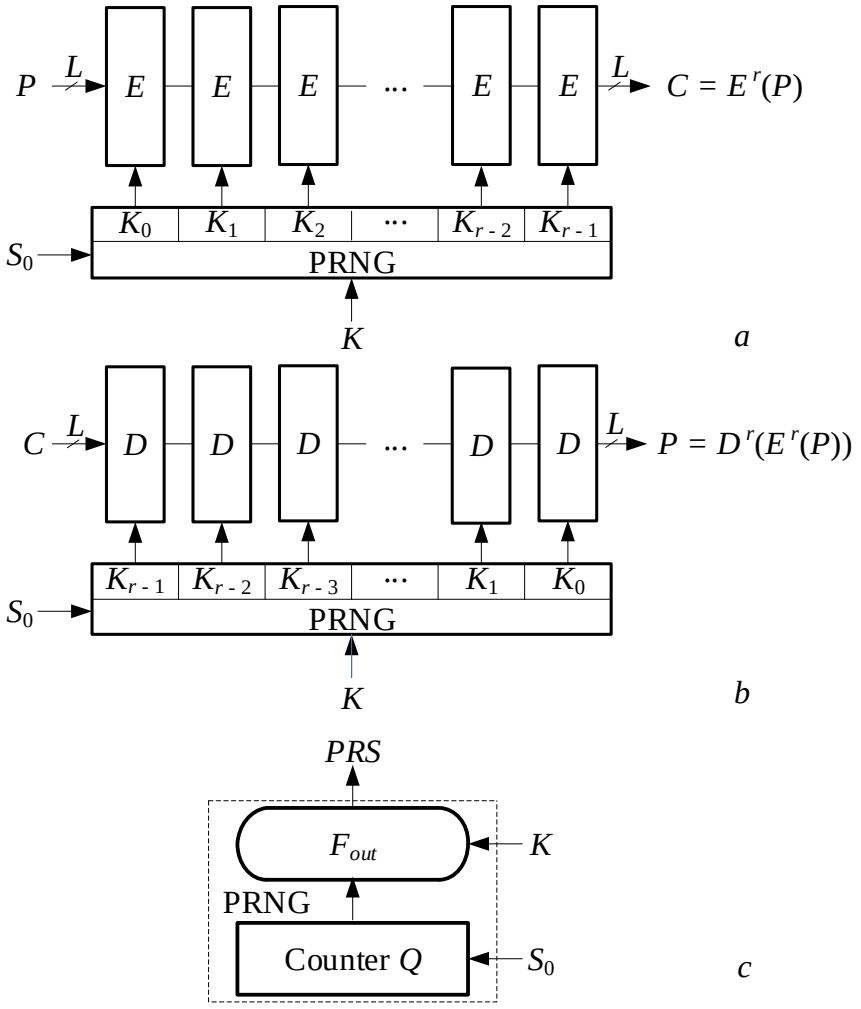

Рисунок 2 – Стохастическое преобразование: *a* – схема прямого преобразования *R*, *b* – схема обратного преобразования R^{-1}; *c* – схема ГПСЧ. PRS – Pseudo-Random Sequence, PRNG – Pseudo-Random Number Generator, *P* – Plaintext, *C* – Ciphertext, *K* – ключ, K_i – раундовые ключи, S_0 – синхропосылка.

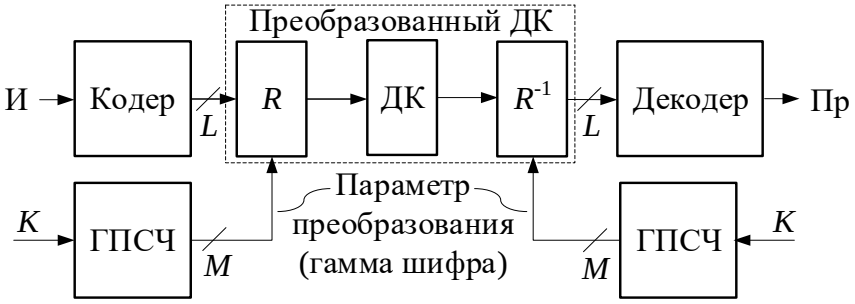

Рисунок 3 – Схема передачи данных по каналу связи с использованием стохастического кодирования. ДК – дискретный канал.
K – ключ, *L* – разрядность *Q*-ичных символов,
$M \geq L$ – разрядность параметра стохастического преобразования.

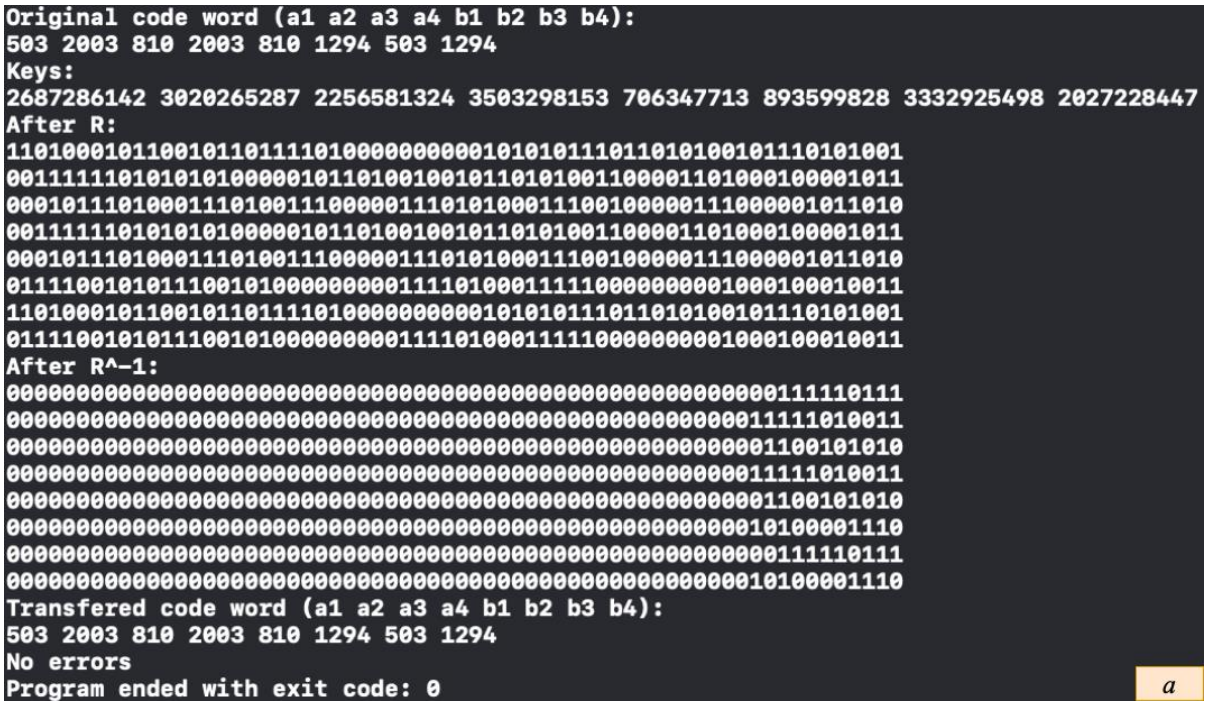

Рисунок 4 – Передача данных без ошибок.

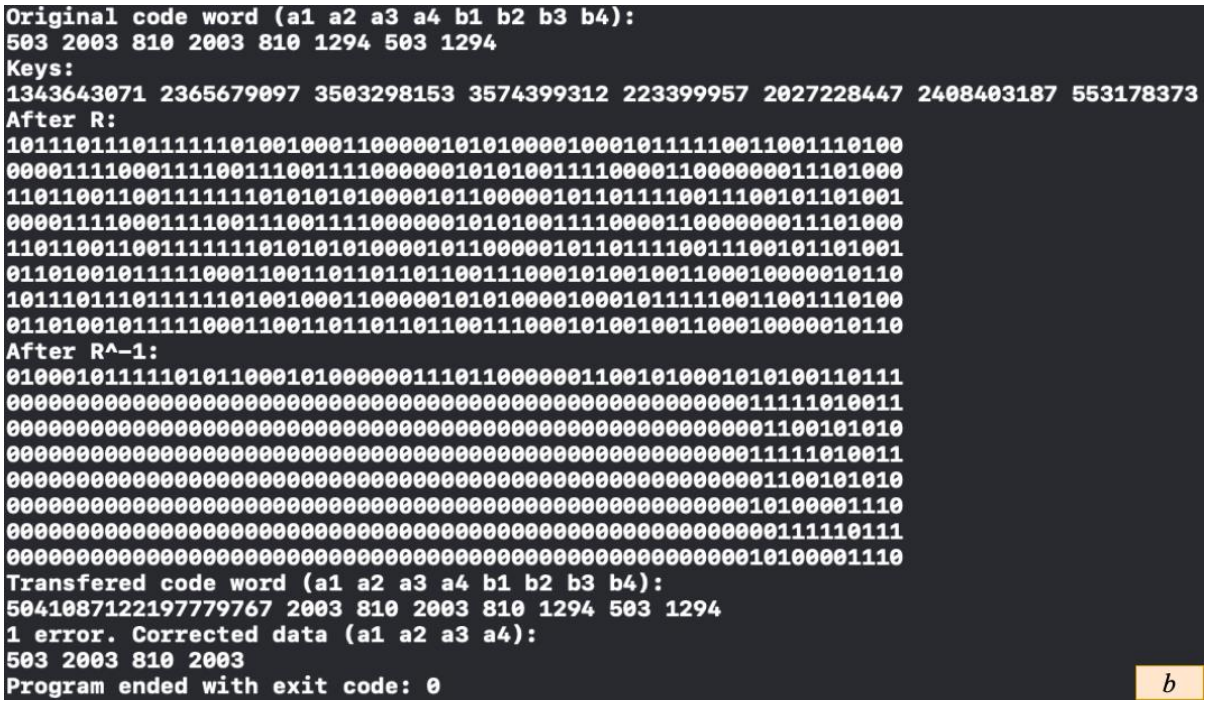

Рисунок 5 – Передача данных с ошибкой.
Исказился один информационный символ. Ошибка исправлена.

Рисунок 6 – Передача данных с ошибками.
Исказились два информационных символа. Ошибки исправлены.

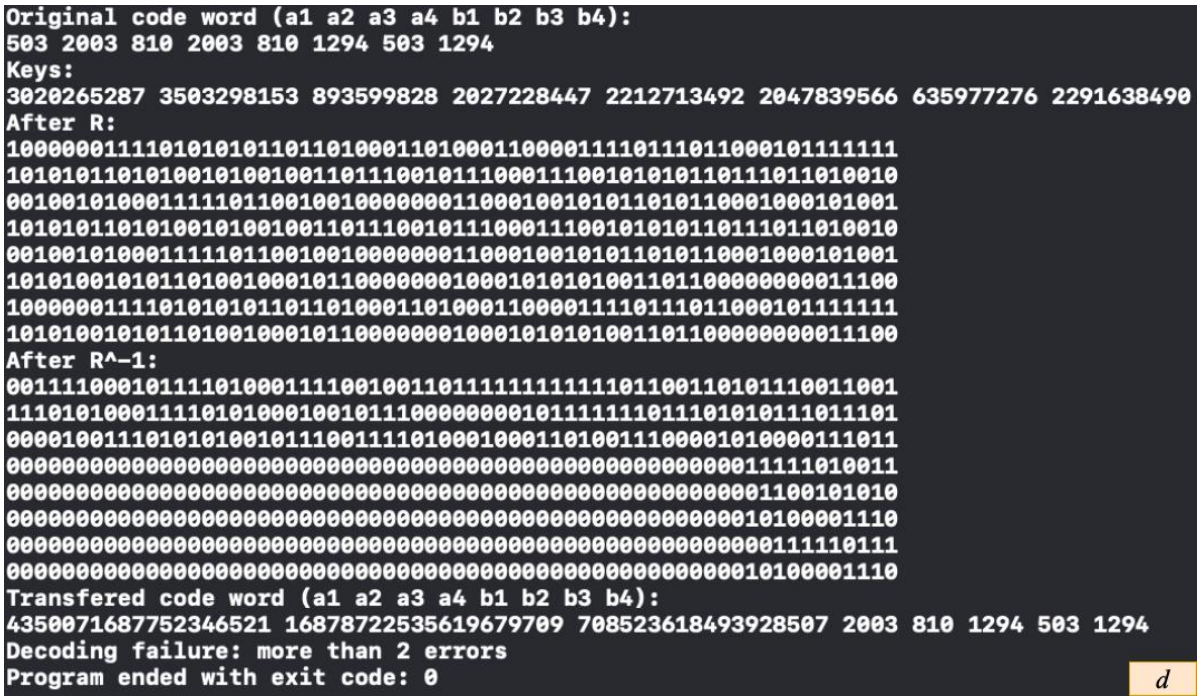

Рисунок 7 – Передача данных с ошибками,
Исказилось более двух информационных символов.
Отказ от декодирования.

```
Original code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 810 2003 810 1294 503 1294
Keys:
2583874451 1412695426 3332925498 2408403187 4095679132 635977276 1145819245 394827685
After R:
0000110010101110010111001001000001101010110000100011111111000000
0010100011011110010000111010000001001001010110010001111000010111
1011110001001001110010100011001101000100000101110000100011110110
0010100011011110010000111010000001001001010110010001111000010111
1011110001001001110010100011001101000100000101110000100011110110
0000011101110101011010000010101101101100011001100011011010100111
0000110010101110010111001001000001101010110000100011111111000000
0000011101110101011010000010101101101100011001100011011010100111
After R^-1:
0000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000111111010011
0000000000000000000000000000000000000000000000000000001100101010
0000000000000000000000000000000000000000000000000000111111010011
0110000011011001011111010100001111101001011011100101111111110111
0000000000000000000000000000000000000000000000000000010100001110
0000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000010100001110
Transfered code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 810 2003 6978746828219899895 1294 503 1294
b error, a correct
Program ended with exit code: 0
```

e

Рисунок 8 – Передача данных без ошибок в информационных символах. Исказился один проверочный символ.

```
Original code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 810 2003 810 1294 503 1294
Keys:
2365679097 3574399312 2027228447 553178373 2543909104 2291638490 394827685 1737984142
After R:
1010001100001000101001001001010011000001100010110100110111111011
1101111000000010101101010111000111001010110101110010000010010010
1010111011001101101101010100011111100111000011001011010011010111
1101111000000010101101010111000111001010110101110010000010010010
1010111011001101101101010100011111100111000011001011010011010111
1011010110101100000110100010110000111110110110111011110000110000
1010001100001000101001001001010011000001100010110100110111111011
1011010110101100000110100010110000111110110110111011110000110000
After R^-1:
0000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000111111010011
0000000000000000000000000000000000000000000000000000001100101010
0000000000000000000000000000000000000000000000000000111111010011
1101101100101100101000010100110001010110101100101000110010100
0001011111110010011001110010010100011110010110001010111011001010
0000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000010100001110
Transfered code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 810 2003 15793174628833841556 1725555016356900554 503 1294
b error, a correct
Program ended with exit code: 0
```

f

Рисунок 9 – Передача данных без ошибок в информационных символах. Исказились два проверочных символа.

```
Original code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 810 2003 810 1294 503 1294
Keys:
2256581324 893599828 2669322631 2047839566 3300722679 1579310740 3475968284 2842138530
After R:
0100000110001101001111011010101011010110001011111101010001011000
0000001011001011101101001100011001001000011001111110101001100001
0100001100011010001110010001111111011001001011110101110000111111
0000001011001011101101001100011001001000011001111110101001100001
0100001100011010001110010001111111011001001011110101110000111111
1101011110010000000110100000000010001100000001100001001001111101110
0100000110001101001111011010101011010110001011111101010001011000
1101011110010000000110100000000010001100000001100001001001111101110
After R^-1:
0000000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000011111010011
1110000010011101010100100011001011111000011010010100010011110000
0000000000000000000000000000000000000000000000000000011111010011
0000000000000000000000000000000000000000000000000000001100101010
1000100101001101101011111101001110110010111110011000010000000100
0000000000000000000000000000000000000000000000000000000111110111
0000000000000000000000000000000000000000000000000000010100001110
Transfered code word (a1 a2 a3 a4 b1 b2 b3 b4):
503 2003 16185183014708921584 2003 810 9893757280178504708 503 1294
1 error in a and 1 error in b. Corrected data (a1 a2 a3 a4):
503 2003 810 2003
Program ended with exit code: 0
```

Рисунок 10 – Передача данных с ошибками. Исказились один информационный и один проверочный символ. Ошибка исправлена.

Список литературы

1. Осмоловский С.А. Стохастические методы защиты информации. – М.: Радио и связь, 2003.
2. Иванов М.А. Способ обеспечения универсальной защиты информации, пересылаемой по каналу связи // Вопросы кибербезопасности. 2019, № 3 (31), с. 45-50.
3. ГОСТ 28147-89. Система обработки информации. Защита криптографическая. Алгоритм криптографического преобразования.
4. ГОСТ 34.12-2018. Информационная технология. Криптографическая защита информации. Блочные шифры.
5. Сагалович Ю.Л. Введение в алгебраические коды. (2011). [Электронный ресурс]. http://iitp.ru/upload/content/790/algebcodes.pdf (Дата обращения: 30.07.2024).
6. Блейхут Р. Теория и практика кодов, контролирующих ошибки. – М.: Мир, 1986.

УДК 004.421.5

ПОГРАММНАЯ РЕАЛИЗАЦИЯ КОДИРОВАНИЯ И ДЕКОДИРОВАНИЯ СООБЩЕНИЙ С ИСПОЛЬЗОВАНИЕМ КОДА РИДА-СОЛОМОНА

М.В. Ковтун
Студент группы Б21-503 НИЯУ МИФИ, frostohelp@ya.ru

Аннотация. Рассматривается помехоустойчивое кодирование как средство повышения надёжности передачи данных по каналам связи. Проводится сравнение характеристик блочных и непрерывных корректирующих кодов и описана целесообразность их комбинации в разных задачах. Реализовано кодирование и декодирование информации кодом Рида-Соломона с исправлением ошибок на языке Python. Применение схожих свойств БЧХ-кодов позволяет достичь лучшей производительности и простоты реализации (за счет синдромного декодирования). Экспериментально доказано, что при увеличении числа проверочных символов возрастают требования к вычислительной мощности, а время декодирования значительно превышает время кодирования.

Ключевые слова: помехоустойчивое кодирование, код Рида-Соломона, сверточный код, Python, исправление ошибок.

Введение

В системах передачи информации качественной характеристикой выступает надежность – способность системы гарантировать передачу данных с допустимыми ошибками. При этом кабельные и беспроводные линии передачи информации подвержены внешним помехам различных физических свойств (в виде шорохов, тресков, неразборчивости речи абонентов и появления звуков с других каналов), что приводит к неоднозначному распознаванию информации на стороне приемника. Магнитные и оптические носители информации часто в процессе использования приобретают физические повреждения, что делает невозможным считывание информации с отдельных участков поверхности носителя. В такой ситуации актуальным становится применение специальных технологий, позволяющих обнаруживать и исправлять ошибки, появляющиеся при воздействии помех.

Классифицируя методы защиты от помех, можно выделить технические методы (экранирование, заземление, фильтрация, изоляция и т.д.) и методы защиты на основе использования помехоустойчивого кодирования.

Кодирование позволяет согласовать формат сообщения с конкретным каналом связи или другим устройством, предназначенным для преобразования или хранения информации. Рассмотрим типовую структурную схему системы передачи дискретной информации (СПДИ) (рис. 1): сигнал формируется в источнике информации, поступает на вход кодера канала, где происходит кодирование с целью улучшения качества передачи сигнала путем повышения помехоустойчивости системы. Далее кодированный сигнал модулируется, проходит через зашумленный канал, где происходят некоторые

искажения сигнала, после чего демодулируется и поступает в декодер, где сообщение восстанавливается. Кодер с декодером образуют кодек источника – устройство или программное обеспечение, которое кодирует и декодирует информацию. Для его построения необходимы регистры сдвига, логические элементы и ключи [1].

Повышение скорости передачи информации в реальных СПДИ приводит к снижению помехоустойчивости и достоверности передачи.

Помехоустойчивое кодирование осуществляется путем введения в передаваемое кодовое слово достаточного количества избыточной информации (например, в виде проверочных символов). В реальных ситуациях длина кода ограничена допустимой сложностью устройств кодирования и декодирования, поэтому эффективность использования корректирующих кодов зависит от параметров кода и ограничений реализации канального кодека.

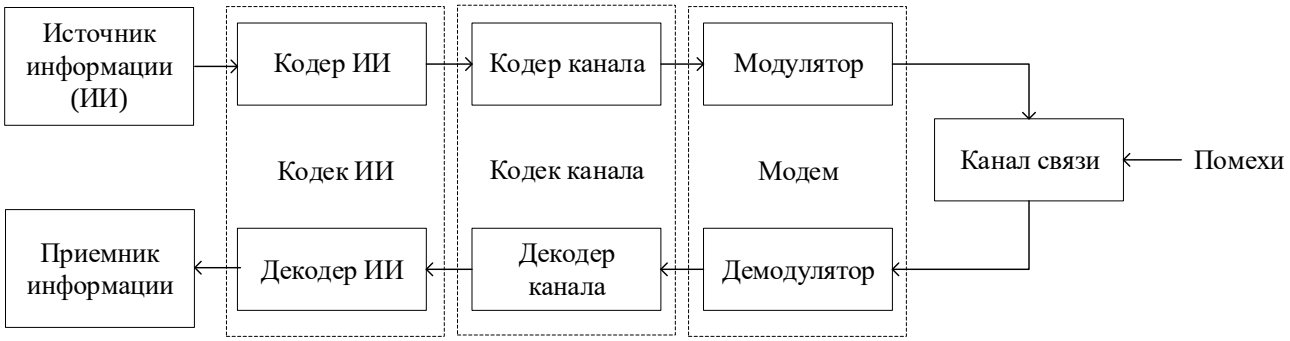

Рисунок 1 – Структурная схема СПДИ.

Кодирование включает в себя процесс преобразования сообщения (информационного слова) в кодовое слово. Каждому сообщению присваивается определенный набор кодовых символов, образующих кодовое слово. Набор кодовых слов, представляющих отдельные закодированные сообщения, образует код. К основным параметрам помехоустойчивых кодов относятся:

n – разрядность (количество символов) кодового слова (1),

k – разрядность (количество символов) информационного слова,

t – количество проверочных символов,

d – кодовое расстояние (минимальное число элементов, которыми любое кодовое слово отличается от другого по всем парам кодовых слов),

w – вес кода (количество ненулевых символов в кодовой комбинации).

$$n = k + t \qquad (1)$$

Существует два класса помехоустойчивых кодов: блочные коды и непрерывные коды.

Блочный код делит передаваемую информационную последовательность на отдельные блоки и добавляет к каждому блоку фиксированное количество

проверочных символов. Блоки кодируются и декодируются независимо друг от друга. Оптимальным линейным блочным кодом является недвоичный код Рида-Соломона, который имеет больше степеней свободы и может достигать максимального кодового расстояния (2):

$$d_{max} = t + 1 \qquad (2)$$

В непрерывных кодах, также известных как цепные, рекуррентные или сверточные, передаваемая информационная последовательность не разбивается на блоки, а проверочные символы размещаются между информационными в определенном порядке. Процессы кодирования и декодирования также выполняются в непрерывном режиме.

Поскольку коды Рида-Соломона и сверточные коды часто применяются при коррекции ошибок, сравним их основные характеристики (табл. 1) [2].

Каждый из этих кодов имеет свои достоинства и недостатки. Например, использование сверточных кодов приводит к меньшему расширению полосы пропускания, а коды Рида-Соломона могут указывать на наличие неисправимых ошибок. В блочных кодах в отличие от сверточных кодов нет памяти, кодирующее устройство по определенным правилам вносит избыточность. Поэтому на практике часто используется их комбинация: для повышения производительности в качестве внутреннего кода используется сверточный код, а в качестве внешнего – код Рида-Соломона.

Для исследования и демонстрации процессов кодирования и декодирования кодом Рида-Соломона была выполнена программная реализация на языке Python.

Коды Рида – Соломона – недвоичные блочные циклические коды (элементами кодового слова являются не биты (двоичные символы), а группы битов (недвоичные символы), например, байты), позволяющие исправлять ошибки в блоках данных.

В общем случае код Рида-Соломона определяется как RS (n, k) s-битных символов (кодер воспринимает k информационных символов по s битов каждый и добавляет проверочные символы для формирования n символов кодового слова).

Коды Рида-Соломона являются частным случаем кодов Боуза-Чоудхури-Хоквингема (БЧХ-кодов), корни порождающего полинома которого лежат в том же поле, над которым строится код, поэтому при реализации кода Рида-Соломона будем использовать логику БЧХ-кода, что позволит достичь лучшей производительности и простоты реализации [3].

Рассмотрим преимущества использования данной логики.

Оригинальная реализация кода Рида-Соломона включает полиномиальную интерполяцию и может требовать выбора наиболее популярного декодирования из

всех возможных. Это делает алгоритм сложным и неэффективным для реализации, особенно для больших значений n и k.

Таблица 1 – Сравнение кодов Рида-Соломона и сверточных кодов.

Основные характеристики	Коды Рида-Соломона (РС)	Сверточные коды (СК)
Тип	Блоковые коды	Последовательные коды
Быстродействие	Зависит от алгоритма декодирования, требует больше вычислительных ресурсов по сравнению с простыми алгоритмами СК	Обычно быстрее при декодировании по сравнению с кодом РС благодаря простым и быстрым алгоритмам (например, алгоритма Витерби)
Коррекция ошибок	Способны корректировать множественные ошибки и потерю данных, эффективны в сценариях с пакетами ошибок	Эффективны для исправления случайных ошибок, особенно в системах реального времени
Структура	Работают с фиксированным размером блоков данных. Представляют данные в виде полиномов над конечным полем	Работают с непрерывным потоком данных. Используют регистры сдвига и логические операции
Применение	Используются в запоминающих устройствах, беспроводной или мобильной связи, спутниковой связи, цифровом телевидении (DVB), высокоскоростных модемах (ADSL, xDSL, и т.д.)	Используются в мобильной связи, спутниковых коммуникациях и в других системах, где важна низкая задержка
Преимущества	Высокая способность к коррекции множества ошибок. Эффективны при работе с пакетами ошибок	Более высокая скорость декодирования. Меньшая задержка при обработке данных. Подходят для систем реального времени
Недостатки	Высокая вычислительная сложность при декодировании. Меньшая скорость декодирования по сравнению с некоторыми другими типами кодов	Менее эффективны при коррекции пакетов ошибок по сравнению с кодами РС. Обычно требуют больше памяти для реализации декодера

Логика БЧХ-кода предоставляет более простой и эффективный метод декодирования, использует синдромное декодирование, которое позволяет обнаруживать и исправлять ошибки, основываясь на вычисленных синдромах, что значительно упрощает алгоритм. Таким образом, можно быстро определять ошибки и их позиции, что делает кодирование и декодирование более практичными для реальных приложений.

Коды Рида-Соломона и BCH-коды являются подклассами циклических кодов и имеют схожие математические основы. Это позволяет использовать методы и алгоритмы, разработанные для одного подкласса, для другого. При этом можно сохранить преимущества обеих систем.

В реальных системах, таких как цифровое телевидение, DVD и компакт-диски, часто используется комбинация этих методов для обеспечения надежной коррекции ошибок при разумных вычислительных затратах.

Рассмотрим этапы кодирования (работаем в поле Галуа $\mathrm{GF}(2^8)$):

Шаг 1. Задаём примитивный полином (чтобы определить операции в $\mathrm{GF}(2^8)$), а также количество проверочных символов $t = 6$.

Примитивный полином (3) в шестнадцатеричной системе:

$$z^8 + z^6 + z^5 + z^4 + 1 = 0\text{x}171 \qquad (3)$$

Шаг 2. Генерируем порождающий полином $g(x)$ (4) (порождающий элемент $\alpha = 02$)

$$g(x) = \prod_{j=1}^{t}\left(x - \alpha^j\right) =$$

$$= 01x^6 + 7\text{E}x^5 + 36x^4 + \text{A}3x^3 + \text{BE}x^2 + 49x + 1\text{A} \qquad (4)$$

Функции для поиска корней и создания порождающего полинома представлены в классе class ReedSolomon (рис. 2), логика функционирования поля описана в классе class G2_8 (рис. 3).

```
def generator_poly(self):
    g = [1]
    for root in self._generator_roots():
        g = GF2_8.poly_mul(g, [1, root])
    return g
```

```
def _generator_roots(self):
    for i in range(1, self._t + 1):
        yield GF2_8.EXP[i]
```

Рисунок 2 – class ReedSolomon.

```
@staticmethod
def _init_class():
    GF2_8.EXP = [0] * (GF2_8.ORDER * 2)
    GF2_8.LOG = [0] * GF2_8.SIZE

    x = 1
    for i in range(GF2_8.ORDER):
        GF2_8.EXP[i] = x
        GF2_8.LOG[x] = i
        x <<= 1
        if x & GF2_8.SIZE:
            x ^= GF2_8.PRIM

    for i in range(GF2_8.ORDER, GF2_8.ORDER * 2):
        GF2_8.EXP[i] = GF2_8.EXP[i - GF2_8.ORDER]
```

Рисунок 3 – class G2_8.

Шаг 3. Задаём сообщение или в текстовом виде (рис. 4, *a*), или как массив целых положительных чисел (рис. 4, *b*). Размер сообщений не должен превышать $n = k + t$.

```
message = 'Reed-Solomon'
bmessage = message.encode('utf-8')
encoded_message = rs.encode(bmessage)
```

a

```
length = random_int(0, 256 - rs._t)
input_data = np.random.randint(0, 255, length).astype(np.uint8)
```

b

Рисунок 4 – Формирование сообщений:
a – в текстовом виде, *b* – генерацией массива.

Шаг 4. Производим кодирование (рис. 5), для чего:

– определим полином (5), коэффициенты которого являются элементами сообщения:

$$p(x) = \sum_{j=1}^{k} a_j x^{j-1} = a_k x^{k-1} + \cdots + a_2 x + a_1 \qquad (5)$$

– сдвинем $p(x)$ на t символов влево, чтобы освободить место под проверочные символы (6):

$$p(x) \cdot x^t \qquad (6)$$

– найдём остаток от деления на $g(x)$ (7):

$$s_r(x) = p(x) \cdot x^t \bmod g(x) \qquad (7)$$

– вычислим закодированное сообщение (8):

$$s(x) = p(x) \cdot x^t - s_r(x) \qquad (8)$$

```
class ReedSolomon:
    def __init__(self, t):
        self._t = t
        self._generator_polynomial = self.generator_poly()

    def encode(self, msg):
        p_shifted = list(msg) + [0] * self._t
        _, sR = GF2_8.poly_div(p_shifted, self._generator_polynomial)
        s = GF2_8.poly_sub(p_shifted, sR)
        return s
```

Рисунок 5 – Кодирование.

Шаг 5. Вносим ошибки явно (рис. 6, *a*) или случайным образом (рис. 6, *b*).

```
# Introduce errors
erroneous_message = encoded_message
erroneous_message[0] = ord('A')
erroneous_message[1] = ord('b')
```

a

```
for _ in range(num_errors):
    encoded[random_int(0, len(encoded) - 1)] = \
    random_int(0, 255)
```

b

Рисунок 6 – Внесение ошибок: *a* – явно, *b* – случайным образом.

Шаг 6. Восстановление полученного сообщения.

Получатель имеет сообщение вида (9), состоящее из отправленного сообщения и ошибок:

$$r(x) = sm(x) + e(x) \qquad (9)$$

Вычисляем синдром *s* и определяем, есть ли ненулевые разряды (рис. 7):

```
def syndromes(self, r):
    return [GF2_8.poly_eval(r, root) for root in self._generator_roots()]
```

```
is_valid_codeword = all(s == 0 for s in syndromes)
if is_valid_codeword:
    return r
```

Рисунок 7 – Вычисление синдрома.

Вычисляем локатор ошибок и количество ошибок (10) (переменная *l* в коде) (рис. 8):

$$\Lambda(x) = \prod_{k=1}^{\nu}(1 - xX_k)\text{, где } X_k = \alpha^{i_k} \qquad (10)$$

Вычисляем (11) позиции ошибок (рис. 9):

$$i_k = \log_\alpha(\alpha^{i_k}) = \log_\alpha(X_k) \qquad (11)$$

```
def error_locator(self, syndromes):
    err_loc = [1]
    old_loc = [1]
    l = 0

    for i in range(self._t):
        delta = GF2_8.poly_mul_at(err_loc, syndromes, i)
        old_loc.append(0)

        if delta != 0:
            if 2 * l <= i:
                new_loc = GF2_8.poly_sub(err_loc, GF2_8.poly_scale(old_loc, delta))
                old_loc = GF2_8.poly_scale(err_loc, GF2_8.div(1, delta))
                err_loc = new_loc
                l = i + 1 - l
            else:
                err_loc = GF2_8.poly_sub(err_loc, GF2_8.poly_scale(old_loc, delta))
    return err_loc[-(l + 1):]
```

Рисунок 8 – Вычисление локатора ошибок.

```
def error_positions(self, err_loc):
    err_pos = []
    for i in range(GF2_8.SIZE):
        if GF2_8.poly_eval(err_loc, i) == 0:
            err_pos.append(GF2_8.LOG[GF2_8.div(1, i)])
    return err_pos
```

Рисунок 9 – Вычисление позиций ошибок.

Определяем корректность позиций ошибок (рис. 10): если мы не нашли ν разных корней или если эти позиции находятся вне границ нашего сообщения, то сообщение передано больше, чем с $t/2$ ошибками, из-за чего восстановить его невозможно.

```
if not self.error_positions_valid(err_pos, err_loc, r):
    raise ReedSolomonException('Could not decode message.')
```

```
def error_positions_valid(self, err_pos, err_loc, r):
    if len(err_loc) - 1 != len(err_pos):
        return False
    return max(err_pos) < len(r)
```

Рисунок 10 – Вычисление позиций ошибок.

Используя алгоритм Форни, получим значения ошибок:

– найдем полиномиальный вычислитель (12) ошибок (рис. 11)

$$\Omega(x) = S(x) * \Lambda(\mathrm{x}) \qquad (12)$$

```
def error_evaluator(self, syndromes, err_loc):
    syndromes = syndromes[::-1]
    mul = GF2_8.poly_mul(syndromes, err_loc)
    syndromes = syndromes[::-1]
    return mul[-self._t:]
```

Рисунок 11 – Нахождение вычислителя ошибок.

– вычислим значения ошибок e_j (13), где $\Lambda(x)$ – локатор ошибок, рассчитанный ранее (рис. 12):

$$e_j = -\frac{\Omega(X_j^{-1})}{\Lambda'(X_j^{-1})} \qquad (13)$$

```
def error_polynomial(self, syndromes, err_loc, err_pos):
    err_eval = self.error_evaluator(syndromes, err_loc)
    err_loc_deriv = GF2_8.poly_deriv(err_loc)
    error_polynomial = [0] * (max(err_pos) + 1)

    for pos in err_pos:
        x_inv = GF2_8.div(1, GF2_8.EXP[pos])
        n = GF2_8.poly_eval(err_eval, x_inv)
        d = GF2_8.poly_eval(err_loc_deriv, x_inv)
        magnitude = GF2_8.div(n, d)
        error_polynomial[len(error_polynomial) - pos - 1] = magnitude
    return error_polynomial
```

Рисунок 12 – Нахождение значений ошибок.

Восстанавливаем исходное сообщение и проверяем, что восстановили корректно, т.е. количество ошибок было не более t/2 и все разряды синдрома равны 0 (рис. 13).

$$s'(x) = r(x) - e(x) \qquad (14)$$

```
e = self.error_polynomial(syndromes, err_loc, err_pos)
repaired = GF2_8.poly_sub(r, e)

if not self.is_valid_codeword(repaired):
    raise ReedSolomonException('Could not decode message.')
```

Рисунок 13 – Восстановление сообщения.

Шаг 7. Декодируем сообщение (рис. 14).

```
def decode(self, r):
    repaired = self.repair(r)
    decoded = self.remove_check_symbols(repaired)
    return decoded
```

```
def remove_check_symbols(self, s):
    return s[:len(s) - self._t]
```

Рисунок 14 – Декодирование сообщения.

Результат тестирования работы программы показывает, что сообщения восстанавливаются верно при наличии не более двух ошибок при количестве проверочных символов $t = 6$ (рис. 15).

```
Encoded:   [113, 42, 35, 55, 215, 180, 42, 213, 7, 245, 67, 71, 110, 91, 206]
Erroneous: [113, 42, 35, 55, 21, 180, 42, 213, 7, 245, 67, 71, 110, 91, 206]
Decoded:   [113, 42, 35, 55, 215, 180, 42, 213, 7]
******************************
Encoded:   [159, 95, 150, 144, 28, 123, 25, 14, 165, 9]
Erroneous: [159, 95, 150, 144, 28, 123, 25, 14, 165, 9]
Decoded:   [159, 95, 150, 144]
******************************
Encoded:   [134, 183, 213, 86, 224, 232, 166, 180, 234]
Erroneous: [134, 183, 213, 85, 224, 232, 166, 180, 3]
Decoded:   [134, 183, 213]
```

Рисунок 15 – Пример работы программы.

На следующем этапе было проведено тестирование с целью анализа зависимости времени работы программы от числа проверочных символов при фиксированном количестве информационных символов (табл. 2). На рис. 16 представлена зависимость среднего времени декодирования одного сообщения (AVG DEC TIME 1) от t.

Таблица 2 – Результаты экспериментов при заданных параметрах для кодирования и декодирования сообщений.

$k = 100$						
tests | errors	t	n	ENC TIME, c	DEC TIME, c	ENC TIME 1, c	DEC TIME 1, c
5000 | 3	6	106	3,818088	16,71241	0,0007636	0,0033425
5000 | 3	10	110	5,204664	25,04663	0,0010409	0,0050093
5000 | 3	20	120	8,39976	45,91281	0,00168	0,0091826
5000 | 3	50	150	18,24146	119,9818	0,0036483	0,0239964
5000 | 3	100	200	34,59369	278,108	0,0069187	0,0556216

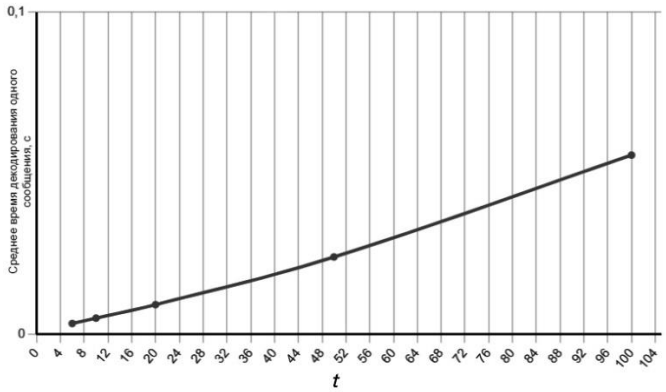

Рисунок 16 – График зависимости времени декодирования от t.

Заключение

Объем вычислительной мощности, необходимой для кодирования и декодирования для кодов Рида-Соломона при фиксированной длине сообщений, зависит от числа t. Большое значение t означает, что большее число ошибок может

быть исправлено, но это потребует большей вычислительной мощности по сравнению с вариантом при меньшем *t*.

Полученное время декодирования сообщений превышает время кодирования от четырех (при малых *t*) до восьми раз (при больших *t*).

Таким образом применение кодов Рида-Соломона для исправления ошибок делает задачу построения эффективных алгоритмов декодирования этих кодов весьма актуальной. В настоящее время реализуется множество различных алгоритмов декодирования кода Рида-Соломона, однако время декодирования все еще остается большим.

Список литературы

1. Пуговкин А. В. Основы построения инфокоммуникационных систем и сетей: учебное пособие. Томский Государственный университет систем управления и радиоэлектроники (ТУСУР). – Томск : Эль Контент, 2014. – 156 с.
2. TM Synchronization and Channel Coding, Recommendation for Space Data System Standards CCSDS 131.0-B-1, Issue 1, Blue Book, Consultative Committee for Space Data Systems, September, 2023. [Электронный ресурс]. https://public.ccsds.org/Pubs/131x0b5.pdf (Дата обращения: 08.08.2024).
3. Reed-Solomon error correction. [Электронный ресурс]. https://sigh.github.io/reed-solomon/ (Дата обращения: 08.08.2024).

УДК 004.89

АНАЛИЗ ОСОБЕННОСТЕЙ ПРИМЕНЕНИЯ МЕТОДОВ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА В АТОМНОЙ ЭНЕРГЕТИКЕ

Е. А. Кузина
Аспирант группы А23-501 НИЯУ МИФИ, ekaterina.kuzina2@yandex.ru

Аннотация. Современное развитие теории и успехи практического освоения методов искусственного интеллекта (ИИ) вызывают интерес к применению данной технологии в наукоемких сферах, одной из которых является атомная энергетика. Исторически исследование возможностей ИИ в данной области рассматривалось с точки зрения систем поддержки операторов. Развитие таких систем, в основном, имело целью обеспечение безопасности атомной электростанции в режимах эксплуатации, требовательных к когнитивным способностям оператора. На сегодняшний день в атомной энергетике отмечается перспективность применения ИИ для более широкого класса задач. В данной работе представлен анализ исследованных и перспективных методов искусственного интеллекта в задачах атомной энергетики. Наибольшее внимание уделяется применимости активно развивающегося направления ИИ – нейронных сетей – к оптимизации энерговыделения в активной зоне реактора. Обсуждается вопрос повышения доверия к указанной технологии с технической и регуляторной точек зрения.

Ключевые слова: искусственный интеллект, экспертные системы, машинное обучение, нейронные сети, интерпретируемость моделей, атомная энергетика.

Введение

Современное развитие теории и успехи практического освоения методов искусственного интеллекта (ИИ) вызывают интерес к применению данной технологии в наукоемких сферах, одной из которых является атомная энергетика. Исторически, в связи со сложностью контроля и управления процессами на атомной электростанции (АЭС), исследование возможностей ИИ в данной области рассматривалось с точки зрения систем поддержки операторов. Развитие таких систем, в основном, имело целью обеспечение безопасности АЭС в режимах эксплуатации, требовательных к когнитивным способностям оператора, например, при переходных режимах, в предаварийных и аварийных ситуациях [1, 2]. На сегодняшний день Международное агентство по атомной энергии (МАГАТЭ) отмечает перспективность применения ИИ в атомной энергетике для более широкого класса задач, к которым также относится оценка остаточного ресурса оборудования, оптимизация сложных процессов, анализ документации, моделирование новых материалов и другие [3].

В данной работе представлен анализ исследованных и перспективных методов искусственного интеллекта в атомной энергетике. В отличие от работ других авторов [4–7], в данном обзоре особое внимание уделяется применимости активно

развивающегося направления ИИ – нейронных сетей – к оптимизации энерговыделения в активной зоне реактора. Обсуждается вопрос повышения доверия к указанной технологии с технической и регуляторной точек зрения.

Методы ИИ в задачах атомной энергетики

Под искусственным интеллектом в широком смысле понимают программно-аппаратные средства для решения интеллектуальных задач. Исследования в этой области начались во второй половине XX века. Большинство разработок того периода сосредоточено на моделях, основанных на знаниях. На 80–90-е годы XX века пришлось активное развитие и применение данной технологии в форме различных экспертных систем (ЭС). Среди таких систем, ориентированных на использование на АЭС, стоит выделить интеллектуальную SCADA-систему “СПРИНТ-РВ”, доведенную до эксплуатации [8]. Данная система включает в себя интеллектуальную компоненту оперативной диагностики состояния энергоблока, представляющую собой экспертную систему на продукционных правилах [9]. Как отмечается в [8, 10], нетривиальными особенностями применения моделей, основанных на знаниях, являются задачи извлечения экспертных знаний и генерации знаний в новых ситуациях. Тем не менее, исследования в данной области для задач атомной энергетики не теряют своей актуальности (например, [11, 12]), что по мнению автора настоящей работы, связано с высокой степенью интерпретируемости таких решений.

В начале 90-ых годов возрастает интерес к другим методам ИИ – нейронным сетям, представляющим на сегодняшний день наиболее быстро развивающуюся область ИИ. В то время в атомной энергетике исследования проводились с нейронными сетями небольшой глубины (в основном, трехслойными), которые в том числе рассматривались в комбинации с существующими наработками в области экспертных систем [10].

Развитие вычислительной техники позволило применять более ресурсоемкие технологии в области искусственного интеллекта. К ним можно отнести эволюционное моделирование [13] и машинное обучение (в том числе, глубокие нейронные сети) [14].

Современный научный интерес во многих практических областях сконцентрирован на машинном обучении. К успешным реализациям соответствующих систем в атомной энергетике можно отнести опыт АО ОКБ “ГИДРОПРЕСС” решения задач обоснования безопасности водо-водяных энергетических реакторов (ВВЭР) [4], а также опытное внедрение АО “ВНИИАЭС” прототипа системы предиктивной аналитики [15], решающей задачу диагностики оборудования на АЭС.

Применение нейронных сетей в задачах оптимизации энерговыделения

Процесс энерговыделения в активной зоне (АЗ) реактора является ключевым для функционирования атомной электростанции. Управление этим процессом осуществляется в условиях нескольких ограничений:

- поддержка заданной номинальной мощности реактора;
- учет требований безопасности, относящихся к локальным характеристикам поля энерговыделения;
- экономный расход ядерного топлива.

Динамика характеристик поля энерговыделения в значительной степени зависит от решений, принимаемых человеком-оператором: например, перемещение в ручном режиме стрежней-поглотителей или перегрузка топлива. В связи с указанными особенностями возникает интерес к исследованию возможностей нейронных сетей для моделирования соответствующих физических и технологических процессов.

На предыдущих этапах применения теории нейронных сетей в задачах управления процессами в АЗ данная технология в основном использовалась для моделирования динамических параметров АЗ. При этом структуры применяемых в то время нейронных сетей, как правило, воспроизводили известные подходы теории идентификации и теории временных рядов [16, 17] или представляли простые (относительно современных архитектур) рекуррентные модели [18, 19]. Также стоит отметить распространенность гибридных моделей, основанных на нейронных сетях и методах нечеткой логики [16, 20, 21]. В современных работах нашли применение более продвинутые архитектуры рекуррентных сетей, например, долгой краткосрочной памяти (англ. Long Short-Term Memory, LSTM) как в задачах прогноза технологический параметров [22], так и планирования загрузки топлива [23].

Переходя к задаче загрузки и перезагрузки топлива, стоит отметить роль эволюционных алгоритмов в ее решении [24, 25]. В то же время исследования как классических [26], так и оригинальных архитектур нейронных сетей [27] демонстрируют применимость данной технологии к обсуждаемой задаче.

В машинном обучении для решения задач управления выделяют класс подходов, который называется обучением с подкреплением (англ. Reinforcement Learning, RL). Применение таких подходов представляет интерес и для задач атомной энергетики [28], однако распространенность соответствующих исследований на сегодняшний день незначительна. Примерами перспективных направлений исследований, к которым применима технология RL,

в рамках задачи оптимизации энерговыделения, по мнению автора настоящего обзора, являются управление стержнями-поглотителями, а также планирование перегрузки топлива.

Особенностью большинства рассмотренных выше исследований является обучение моделей на синтетических данных. Применение таких моделей на реальном объекте может быть затруднено в силу зашумленности истинных данных, полученных в результате съема информации с датчиков технологических параметров. Для снижения влияния шумов могут быть использованы, например, нейронные сети архитектуры автокодировщик [29].

Помимо архитектур, рассмотренных ранее, к перспективным для задач обсуждаемой области, но малоизученным архитектурам нейронных сетей можно отнести архитектуру трансформер [30], зарекомендовавшую себя в задачах обработки естественного языка. Данная архитектура в случае применения позиционного кодирования может быть рассмотрена в качестве альтернативы рекуррентным сетям.

Рассмотренные в данном разделе работы, посвященные нейросетевым подходам, свидетельствуют о целесообразности дальнейших исследований их применимости в том числе к решению задачи оптимизации энерговыделения в АЗ реактора. Кроме того, отмечены перспективные, но малоизученные на сегодняшний день направления.

Особенности применения нейронных сетей

При обсуждении применимости нейронных сетей в наукоемкой сфере, чем является атомная энергетика, нельзя опустить замечание о низкой интерпретируемости большинства таких моделей, получившее в литературе название “дилемма черного ящика”. Действительно, способность нейронной сети извлекать из данных признаки в отсутствие непосредственного влияния со стороны исследователя влечет за собой проблему обоснования полученных результатов. Эта особенность требует внимания в случае применения данной технологии в отношении объектов критической инфраструктуры, тем более представляющих потенциальную радиационную опасность.

Решение указанной проблемы можно рассматривать с двух точек зрения: технической (поиск способов интерпретации моделей нейронных сетей) и регуляторной (введение различных стандартов процессов разработки и эксплуатации нейронных сетей).

На сегодняшний день технические решения развиваются в следующих направлениях:

– Гибридные модели. В наукоемких сферах активно развивается концепция физически информированных (англ. physics-informed, theory-guided) моделей машинного обучения [31, 32]. Данный подход заключается в различных техниках внедрения в модель априорных знаний об исследуемом процессе;
– Объяснимые модели машинного обучения и техники объяснения моделей (англ. eXplainable Artificial Intelligence, XAI) [33];
– Новые архитектуры и подходы к обучению нейронных сетей. Например, замена обучаемых линейных весов на обучаемые функции активации в архитектуре нейронной сети Колмогорова-Арнольда (англ. Kolmogorov-Arnold Network, KAN) заметно повышает интерпретируемость модели [34].

Кроме того, одним из этапов внедрения методов ИИ может стать их предварительная отработка на цифровых двойниках исследуемых объектов.

Вопросы стандартизации применения технологий искусственного интеллекта активно обсуждаются на национальном и международном уровнях. Одним из главных результатов этой деятельности в России можно считать образование технического комитета по стандартизации в области искусственного интеллекта и разработка соответствующих стандартов [35], например, ГОСТ Р 59276-2020 “Системы искусственного интеллекта. Способы обеспечения доверия. Общие положения”. В то же время в МАГАТЭ исследуется вопрос безопасности применения ИИ на АЭС, а также организованы рабочие группы, разрабатывающие нормативные положения в отношении ИИ [36].

Таким образом, проблема интерпретируемости моделей ИИ и обеспечения доверия к ним является одной из ключевых в области практического применения нейронных сетей, в частности в атомной энергетике.

Заключение

В данной работе выполнен анализ особенностей применения искусственного интеллекта в атомной энергетике. Рассмотрены примеры отечественных реализаций методов ИИ, существующие и перспективные направления исследований, в частности применение нейросетевых подходов в задачах оптимизации энерговыделения в активной зоне реактора. Обсужден вопрос интерпретируемости моделей искусственного интеллекта и повышения доверия к данной технологии.

Список литературы

1. Анохин А.Н., Калинушкин А.Е., Горбаев В.А., Сивоконь В.П. Состояние и перспективы систем поддержки операторов АЭС // Известия вузов. Ядерная энергетика, 2016, №2, с. 5-16.
2. International atomic energy agency Use of expert systems in nuclear safety / Non-serial Publications. Vienna: IAEA, 1990.

3. International atomic energy agency Artificial Intelligence for Accelerating Nuclear Applications, Science and Technology / Non-serial Publications. Vienna: IAEA, 2022.
4. Николаева А.В., Увакин М.А., Пантюшин С.И., Сотсков Е.В., Антипов М.В., Николаев А.Л., Литышев А.В., Безруков Ю.А., Кавун О.Ю., Быков М.А. (АО ОКБ "ГИДРОПРЕСС") Искусственный интеллект в области использования атомной энергии - существующие возможности и перспективы // Вопросы атомной науки и техники. Серия: Физика ядерных реакторов, 2023, № 3, с. 4-16.
5. Lu Ch., Lyu J., Zhang L., Gong A., Fan Y., Yan J., Li, Xiu. Nuclear power plants with artificial intelligence in Industry 4.0 era: top-level design and current applications – A systemic review // IEEE Access, 2020, № 8, pp. 194315-194332.
6. Kim J., Lee S., Seong P.H. Autonomous Nuclear Power Plants with Artificial Intelligence / Lecture Notes in Energy. Springer Cham, 2023.
7. Huang Q., Peng Sh., Deng J., Zeng H., Zhang Z., Liu Y., Yuan P. A review of the application of artificial intelligence to nuclear reactors: where we are and what's next. // Heliyon, 2023, V. 9, p. e13883.
8. Башлыков А.А. СПРИНТ-РВ – интеллектуальная SCADA-система для построения средств человеко-машинного управления сложными и экологически опасными объектами и технологиями // Автоматизация, телемеханизация и связь в нефтяной промышленности, 2012, № 12, с. 8-20.
9. Башлыков А.А., Еремеев А.П. Методы и программные средства конструирования интеллектуальных систем поддержки принятия решений для объектов энергетики // Вестник Московского энергетического института, 2018, № 1, с. 72-85.
10. International atomic energy agency The potential of knowledge based systems in nuclear installations / Non-serial Publications. Vienna: IAEA, 1993.
11. Поваров В.П. Принципы разработки систем принятия решений в задачах управления ядерными блоками // Вестник Воронежского государственного технического университета, 2018, Т. 14, № 2, с. 87-91.
12. Hanna B., Son T., Dinh N. AI-Guided Reasoning-Based Operator Support System for the Nuclear Power Plant Management // Annals of Nuclear Energy, 2021, V. 154, p. 108079.
13. Schirru R., Pereira C.M.N.A., Martinez A.S. Genetic Algorithms Applied to the Nuclear Power Plant Operation // Fuzzy Systems and Soft Computing in Nuclear Engineering / Studies in Fuzziness and Soft Computing, 2000, V. 38, pp. 335-350.
14. Neudecker D., Dwivedi N., Alhassan E., Schnabel G. Machine Learning for Nuclear Data // Summary Report of the IAEA Consultants' Meeting. 2021.
15. Сборник "2020-2021 годы: краткие результаты научно-технической деятельности АО "ВНИИАЭС". [Электронный ресурс]. https://vniiaes.ru/upload/Сборник_ОР_ВНИИАЭС_2020_2021.pdf (Дата обращения 23.07.2024).
16. Boroushaki M., Ghofrani M., Bagher, Lucas C., Yazdanpanah M.J., Sadati, N. Axial offset control of PWR nuclear reactor core using intelligent techniques // Nuclear Engineering and Design, 2004, V. 227, pp. 285-300.
17. Zio E., Broggi M., Pedroni N. Nuclear reactor dynamics on-line estimation by Locally Recurrent Neural Networks // Progress in Nuclear Energy, 2009, V. 51, pp. 573-581.
18. Adali T., Bakal B., Sonmez M.K., Fakory R., Tsaoi C.O. Modeling nuclear reactor core dynamics with recurrent neural network // Neurocomputing, 1997, V. 15, № 3-4, pp. 363-381.
19. Mirvakili S. M., Faghihi F., Khalafi, H. Developing a computational tool for predicting physical parameters of a typical VVER-1000 core based on artificial neural network // Annals of Nuclear Energy, 2012, V. 50,pp. 82–93.
20. Na M. G., Upadhyaya B. R. A neuro-fuzzy controller for axial power distribution an nuclear reactors // IEEE Transactions on Nuclear Science, 1998, V. 45, № 1, pp. 59-67.
21. Khajavi M.N., Menhaj M.B., Suratgar A.A. A neural network controller for load following operation of nuclear reactors // Annals of Nuclear Energy, 2002, V. 29, № 6, pp. 751-760.
22. Bae J., Kim G., Lee S.J. Real-time prediction of nuclear power plant parameter trends following operator actions // Expert Systems with Applications, 2021, V. 186, p. 115848.

23. Ren C., He L., Lei J., Liu J., Huang G., Gao K., Qu H., Zhang Y., Li W., Yang X., Yu T. Neutron transport calculation for the BEAVRS core based on the LSTM neural network // Scientific Reports, 2023, V. 13, p. 14670.
24. Dechaine M.D., Feltus M.A. Nuclear-fuel management optimization using genetic algorithms // Nuclear Technology, 1995, V. 111, № 1, pp. 109-114.
25. Соболев А.В., Газетдинов А.С., Самохин Д.С. Генетический алгоритм в задачах оптимизации загрузки и перегрузок топлива ядерного реактора // Известия вузов. Ядерная энергетика, 2016, № 4, с. 67-77.
26. Sedighi M., Setayeshi S., Salehi A. PWR fuel management optimization using neural networks // Annals of Nuclear Energy, 2002, V. 29, pp. 41-51.
27. Thakur A., Sarkar D., Bharti V., Kannan U. Development of in-core fuel management tool for AHWR using artificial neural networks // Annals of Nuclear Energy, 2021, V. 150, p. 107869.
28. Gong A., Chen Y., Zhang J., Li X. Possibilities of reinforcement learning for nuclear power plants: Evidence on current applications and beyond // Nuclear Engineering and Technology, 2024, V. 56, № 6, pp. 1959-1974.
29. Hinton G.E., Salakhutdinov R.R. Reducing the Dimensionality of Data with Neural Networks // Science, 2006, V. 313, № 5786, pp. 504-507.
30. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser L., Polosukhin I. Attention Is All You Need // 31st Conference on Neural Information Processing Systems, 2017. [Электронный ресурс]. https://arxiv.org/pdf/1706.03762 (Дата обращения: 13.08.2024).
31. Karpatne A., Atluri G., Faghmous, J., Steinbach M., Banerjee A., Ganguly A., Shekhar S., Samatova N., Kumar V. Theory-guided Data Science: A New Paradigm for Scientific Discovery // IEEE Transactions on Knowledge and Data Engineering, 2017. [Электронный ресурс]. https://arxiv.org/pdf/1612.08544 (Дата обращения: 13.08.2024).
32. Radaideh M., Wolverton I., Joseph J., Tusar J., Otgonbaatar U., Roy N., Forget B., Shirvan K. Physics-informed reinforcement learning optimization of nuclear assembly design // Nuclear Engineering and Design, 2020, V. 372, p. 110966.
33. Ali S., Abuhmed T., El-Sappagh S., Muhammad Kh., Alonso-Moral J.M., Confalonieri R., Guidotti R., Ser J.D., Díaz-Rodríguez N., Herrera F. Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence // Information Fusion, 2023, V. 99, p. 101805.
34. Liu Z., Wang Y., Vaidya S., Ruehle F., Halverson J., Soljacc M., Hou T. Y., Tegmark M. KAN: Kolmogorov-Arnold Networks // ArXiv preprint, 2024. arXiv:2404.19756.
35. Действующие стандарты по направлению “Искусственный интеллект” / Росстандарт. Федеральное агенство по техническому регулированию и метрологии. [Электронный ресурс]. https://www.rst.gov.ru/portal/gost/home/standarts/aistandarts (Дата обращения: 24.07.2024).
36. Picot W. Повышение эффективности производства ядерной энергии с помощью искусственного интеллекта // Бюллетень МАГАТЭ. Ядерные инновации для мира без выбросов, 2023, Т. 64, № 3. [Электронный ресурс]. https://www.iaea.org/ru/bulletin/povyshenie-effektivnosti-proizvodstva-yadernoy-energii-s-pomoshchyu-iskusstvennogo-intellekta (Дата обращения: 20.07.2024).

УДК 004.04

ПОСТРОЕНИЕ ПЛАТФОРМЫ ДЛЯ АНАЛИЗА БОЛЬШИХ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ ПРОГРАММНЫХ ПРОДУКТОВ С ОТКРЫТЫМ КОДОМ ИЗ ЭКОСИСТЕМЫ APACHE

Д.О. Соловьев[1], Г.Н. Косилов[2]
[1]Студент группы С19-501 НИЯУ МИФИ, dmitriy.porter@mail.ru
[2]Аспирант группы А23-501 НИЯУ МИФИ, g.kosilov@yandex.ru

Аннотация. В статье рассматривается построение платформы для анализа больших данных с использованием программных продуктов с открытым кодом из экосистемы Apache, таких как HDFS, YARN, Hive, Spark, Kafka, Airflow и Superset. Описаны основные компоненты системы, их функции и преимущества, также приводятся примеры применения. Статья будет полезна специалистам, связанным с анализом данных, инженерам и разработчикам, занимающимся проектированием и внедрением систем анализа данных.

Ключевые слова: большие данные, анализ данных, open-source, программные продукты Apache.

Введение

В последние годы концепция «больших данных» стала важной частью цифровой трансформации. Большие данные представляют собой объемные и сложные наборы данных, характеризующиеся высокой скоростью поступления и разнообразием форматов. Большие данные генерируются из множества источников, таких как социальные сети, датчики IoT и финансовые транзакции. Анализ больших массивов информации становится критически важным для компаний в различных отраслях, так как позволяет оптимизировать бизнес-процессы, повышать эффективность и конкурентоспособность организаций. Исследования показывают, что компании, активно применяющие анализ больших данных, достигают лучших результатов по сравнению с конкурентами. Отсюда вытекает необходимость грамотного использования компаниями программных продуктов, предназначенных для построения систем и платформ для анализа данных. С учетом текущей ситуации, появилась новая особенность – необходимость использования продуктов с открытым программным кодом. Open-source решения, например, такие, как предоставляет компания Apache, дают компаниям гибкость и независимость от лицензионных ограничений, что является важным преимуществом в условиях ограниченного доступа к коммерческим продуктам.

Целью данной статьи является демонстрация практического подхода к построению платформы для анализа больших геоданных оператором сотовой связи с использованием open-source продуктов из экосистемы Apache. В частности, рассматриваются Apache HDFS для хранения данных, Apache YARN для управления вычислительными ресурсами, Apache Hive и Apache Spark для обработки и анализа данных, Apache Kafka для потокового обмена данными, Apache

Airflow для оркестрации (координирования) процессов и Apache Superset для визуализации.

Анализ больших данных

Построение хранилища данных

Хранение больших данных является сложным и многоуровневым процессом. Apache HDFS (Hadoop Distributed File System) и Apache YARN (Yet Another Resource Negotiator) наиболее подходящие для этого программные продукты из экосистемы Apache.

Apache HDFS – распределенная файловая система для хранения файлов больших размеров с возможностью потокового доступа к информации, поблочно распределенной по узлам вычислительного кластера, который может состоять из произвольного аппаратного обеспечения. HDFS, как и любая файловая система, это иерархия каталогов с вложенными в них подкаталогами и файлами [1]. HDFS обеспечивает надежное хранение данных путем разделения файлов на блоки и их распределенного хранения на нескольких узлах, что повышает отказоустойчивость системы. В случае сбоя одного узла, данные остаются доступными благодаря резервированию блоков на других узлах.

Apache YARN играет важную роль в управлении ресурсами кластера, координируя выполнение задач и обеспечивая эффективное использование вычислительных ресурсов. В условиях обработки больших объемов данных это программное обеспечение позволяет распределять задачи между узлами, что оптимизирует загрузку процессоров и памяти, что является крайне важным для поддержания высокой производительности и быстрого времени отклика. Также YARN позволяет использовать более обобщенный подход к хранению данных в Apache HDFS, т.е. делает использование механизма MapReduce лишь опцией, от которой можно перейти к обработке данных с помощью Apache Spark [2].

Для реализации подобных систем требуются значительные ресурсные мощности, включающие большое количество серверов с высоким объемом оперативной памяти и мощными процессорами. Обслуживание таких систем требует специалистов с глубокими знаниями, а также навыками работы с Linux-системами, так как большинство кластеров разворачиваются именно на этой операционной системе. Кроме того, необходимы знания в области сетевой архитектуры и оптимизации хранения данных.

В качестве примера рассмотрим компанию, предоставляющую услуги связи для населения. Данный оператор связи может использовать инфраструктуру для сбора абонентских геоданных, т.е. данных с привязкой к географическим координатам. С базовых станций передаются данные о местоположении

абонентов, которые затем агрегируются и хранятся в HDFS. Это позволяет оператору анализировать передвижения пользователей, что может быть использовано для оптимизации сети или разработки новых услуг.

Обработка данных

Обработка и анализ больших массивов данных с помощью Apache Hive и Apache Spark являются неотъемлемой частью современной аналитики. Apache Hive предоставляет SQL-подобный интерфейс для работы с большими данными, что упрощает доступ к ним ввиду стандартизированного языка. Hive идеально подходит для обработки структурированных данных и может использоваться для предварительной обработки данных, их агрегации и фильтрации.

Apache Spark представляет собой платформу с открытым исходным кодом для параллельной обработки и анализа слабоструктурированных данных. Она обеспечивает высокую скорость обработки данных благодаря использованию in-memory вычислений, т.е. благодаря хранению данных в оперативной памяти во время их обработки. Это делает Spark идеальным для работы с неструктурированными или слабоструктурированными данными, такими как текстовые файлы или логи. Spark обеспечивает широкие возможности для выполнения сложных задач для анализа данных, включая построение алгоритмов машинного обучения и исследование потоковых данных [3].

В рамках концепции MapReduce в Spark существуют специальные методы доступа к хранилищу данных. Архитектура Spark состоит из центрального координатора (master node), который управляет распределенным кластером из исполнительных узлов (worker nodes). Spark использует абстракцию resilient distributed dataset (устойчивый распределенный набор данных), позволяющую пользователям кэшировать данные в памяти, управлять разделением данных и выполнять вычисления в масштабе кластера [4].

Специалисты, работающие с этими инструментами, должны иметь опыт в области разработки распределенных систем и навыки работы с языками программирования, такими как Python, Scala или Java, которые часто используются для написания приложений на Spark.

Для примера использования этих технологий рассмотрим процессы внутри компании оператора связи. Обработку и анализ данных с базовых станций можно использовать для следующих целей.

- Анализ посетителей для развития городского парка. Данные о посещаемости парка можно использовать для планирования маркетинговых кампаний и улучшения инфраструктуры.
- Создание транспортной модели города. Анализ передвижений абонентов позволяет оптимизировать маршруты общественного транспорта, уменьшая затраты и повышая удобство для горожан.

- Анализ плотности населения. Данные используются для обоснования выделения бюджета на строительство новой инфраструктуры.

Организация потоков данных

Apache Kafka и Apache Airflow играют ключевую роль в организации потоков данных и управлении рабочими процессами. Apache Kafka предоставляет функцию брокера сообщений и была разработана для сценариев обработки потоков данных. Apache Kafka поддерживает метрики, отслеживание активности, агрегирование журналов, журналы коммитов и источник событий. Kafka обеспечивает высокую производительность и надежность, что делает это программное обеспечение идеальным выбором для обработки и передачи больших объемов данных в реальном времени [5].

Apache Airflow – это инструмент, который позволяет разрабатывать, планировать и осуществлять мониторинг сложных рабочих процессов. Airflow используется как планировщик ETL/ELT-процессов [6]. Главной особенностью является то, что для описания процессов используется язык программирования Python.

Для работы с этими инструментами необходимы специалисты с опытом в области DevOps и навыками автоматизации процессов. Кроме того, важно понимание архитектуры распределенных систем и знания в области работы с базами данных и сетевыми протоколами.

Примером использования может служить задача внутри компании оператора связи по проведению регулярного расчета и передачи данных о перемещениях абонентов для ежедневного анализа. Kafka используется для передачи данных из источников в систему обработки, а Airflow управляет процессом их обработки и подготовки отчетов для заказчика.

Визуализация данных

Apache Superset используется для визуализация данных с помощью создания дашбордов, т.е. специальных панелей, на которых отображаются ключевые метрики и показатели. Этот инструмент поддерживает интеграцию с остальными инструментами экосистемы Apache и поддерживает различные виды графиков и диаграмм, а также предоставляет возможности для создания кастомизированных визуализаций, что облегчает качественную интерпретацию данных и упрощает принятие управленческих решений, основанных на анализе данных, т.е. реализующих data-driven decision making подход.

В качестве примера можно привести создание ситуационного аналитического центра городских происшествий. Данные о происшествиях собираются по сотовой сети и обрабатываются в реальном времени, после чего визуализируются с помощью Superset, предоставляя в центр информацию для оперативного реагирования и планирования действий.

Заключение

В статье рассмотрен комплексный подход к построению платформы для анализа больших данных с использованием программных продуктов с открытым кодом из экосистемы Apache. Хранилище данных на основе HDFS и YARN обеспечивает надежное и масштабируемое хранение, в то время как Apache Hive и Spark позволяют эффективно обрабатывать и анализировать данные. Использование Kafka и Airflow оптимизирует потоки данных и рабочие процессы, а Superset предоставляет мощные возможности для конечной визуализации. При внедрении компетентными специалистами и при достаточных вычислительных мощностях эти решения способствуют инновационному развитию процесса анализа данных внутри компаний в условиях современной цифровой экономики без зависимостей от лицензионных продуктов.

Список литературы

1. Пискун Г.А. Обзор программной платформы Apache Hadoop для обработки и хранения больших данных / Г.А. Пискун, В.Ф. Алексеев, Т.М. Воронко // BIG DATA и анализ высокого уровня : Сборник научных статей IX международной научно-практической конференции: в 2-х частях , Минск, 17–18 мая 2023 года. – Минск: Белорусский государственный университет информатики и радиоэлектроники, 2023, с. 465-471.
2. A survey of open source tools for machine learning with big data in the Hadoop ecosystem / S. Landset, T.M. Khoshgoftaar, A.N. Richter, T. Hasanin // Journal of Big Data, 2015, Vol. 2, No. 1, pp. 1-36.
3. Пальмов С.В., Посиваткина А.А. Обзор основных возможностей Apache Spark // Евразийское Научное Объединение. 2020, № 5-2(63), с. 155-159.
4. Иванова В.Ю., Соловьев Д.О. Обзор методов обработки больших данных с использованием Apache Spark, библиотеки Pandas и SQL // Наукосфера, 2024, № 5-1, с. 43-47.
5. Рахматуллин Т.Г. Сравнительный анализ Apache Kafka и rabbitmq // Актуальные исследования, 2022, № 49-1(128), с. 35-40.
6. Трофимцов Е.В. Анализ платформы Apache Airflow // Актуальные проблемы прикладной математики, информатики и механики : сборник трудов Международной научной конференции, Воронеж, 12–14 декабря 2022 года /Воронежский государственный университет. – Воронеж: Научно-исследовательские публикации, 2023, с. 936-940.
7. Бушуев А. Как анализировать большие объемы данных: 10 проверенных инструментов для разработчика // Системный администратор, 2022, № 7-8(236-237), с. 54-57.

УДК 004.93

РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА ДЛЯ ГЕНЕРИРОВАНИЯ ИЗОБРАЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ГЕНЕРАТИВНЫХ НЕЙРОННЫХ СЕТЕЙ

А.Ф. Ахметов
студент группы С20-501 НИЯУ МИФИ, azat.akhmetov.1998@bk.ru

Аннотация. Данная работа опирается на принципы работы генеративных нейронных сетей, связана с различными их архитектурами и реализациями и представляет результаты разработки программного средства с их использованием. В процессе разработки и исследования использовались методы оценки показателей качества, аугментации наборов данных и переборов гиперпараметров моделей. В ходе исследования было разработано программное средство для генерирования изображения и подтверждена гипотеза о непрерывности значений векторного пространства признаков скрытого слоя. Программное обеспечение позволяет в зависимости от типа генерируемых данных выбирать модель и генерировать изображения соответствующего типа. Проведена оценка показателя эффективности, подтверждающего корректность генерации изображений для каждого из типов данных, использовавшихся в исследовании.

Ключевые слова: нейронные сети, генерирование изображений, аугментация набора данных, бинарный классификатор, компьютерное зрение, Python, библиотека PyTorch.

Введение

Генеративные нейронные сети позволяют решать ряд задач, имеющих большое значение для научно-технического прогресса, например, задачи аугментации (т.е. увеличения) наборов данных, моделирования процессов, имеющих характер data-driven и других. Отличительной способностью разработанного программного средства является возможность выбора между двумя принципиально разными моделями для генерации данных – модель с автоэнкодером, позволяющим восстанавливать зашумлённые данные, и модель, использующую операцию свёртки. Предлагаемое средство, реализующее генератор изображений, может быть использовано в учебных целях, а также для дальнейшего развития проекта, направленного на решение задач, актуальность которых обозначена выше. Разработка программного средства (ПС) велась на языке Python [1] с использованием библиотеки PyTorch.

Цели. Комплексное исследование предметной области, обзор источников, предложение о структуре программного средства (ПС), разработка алгоритма работы программного средства, реализация программного средства на заданном языке программирования.

В процессе разработки и исследования использовались методы оценки показателей качества, аугментации наборов данных [2] и переборов гиперпараметров моделей.

Использование генеративных моделей

Работа выполнялась в три этапа.

На первом этапе были выделены типы объектов, изображения которых могли быть сгенерированы или очищены от аддитивного гауссова шума с помощью модели (далее – SimpleGAN), использующей вариационный автоэнкодер [3] – результатом такого выделения стали рукописные цифры из открытого набора данных MNIST. Кроме того, был выбран набор данных, который будет основой для обучения концептуально более сложной модели – DCGAN [3], которая использует операцию свёртки, не имеет полносвязных слоёв кроме выходного и входного, а также использует функции активации ReLU и LeakyReLU.

На втором этапе было разработано необходимое программное средство [3] на языке Python [4] и проведена оценка [3] показателя качества каждой из моделей и выполнена проверка работы ПС. Пользователь ПС может выбрать алгоритм для решения задачи генерации изображений и затем выполнить обработку данных и вывести результат на экран. В зависимости от входных данных имеет смысл предлагать пользователю выбор алгоритма для решения задачи генерации изображений. После выбора алгоритма ПС производит обработку данных и выводит результат на экран. Схема алгоритма работы ПС представлена на рис. 1. За оценку показателя качества при генерации изображений был принят результат решения задачи бинарной классификации [5]. Классификатор [5, 6] предварительно обучали для распознавания истинных и сгенерированных изображений, после этого формировали тренировочную выборку, состоящую как из сгенерированных изображений, так и из истинных.

Для бинарной классификации используют часто показатель качества AUC ROC [5]. Результаты оценки показателя качества приведены в табл. 1.

Результаты работ ПС для моделей SimpleGAN и DCGAN показаны соответственно на рис. 2 и 3.

На третьем этапе разработки была экспериментально подтверждена гипотеза о непрерывности векторов скрытого состояния DCGAN. В ходе экспериментов с гиперпараметрами [7, 8] DCGAN было замечено, что вектора случайных величин, распределенных нормально, после обучения обладают свойством гладкости. Это значит, что если в пространстве признаков мы выберем две случайные точки и соединим их кривой, то точки на этой кривой будут также приемлемыми элементами изображениями. Пример такого явления показан на рис. 4 для пяти разных изображений (одна строка – результат движения по одной из кривых в пространстве векторов скрытого состояния).

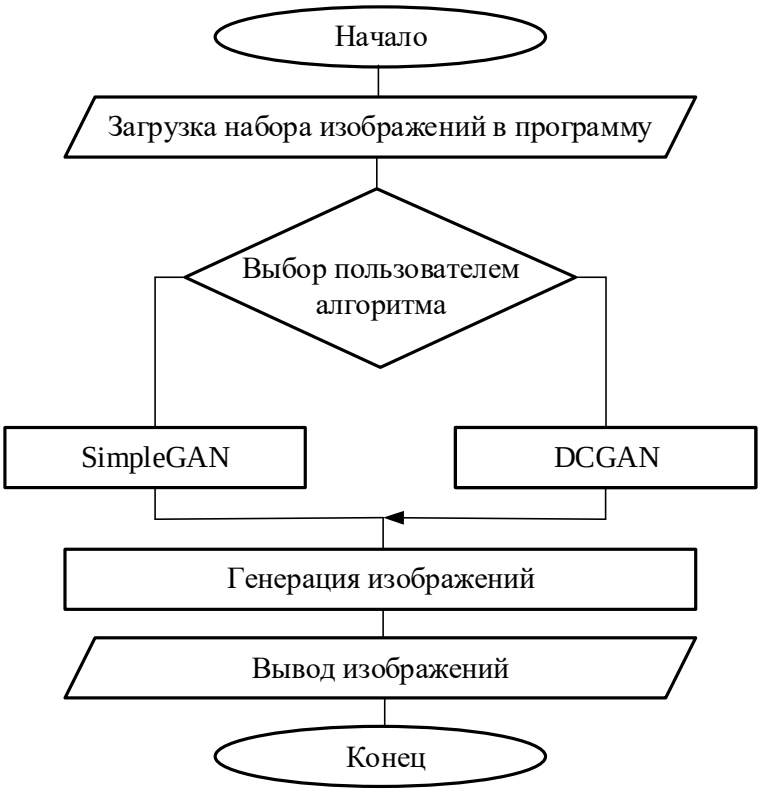

Рисунок 1 – Схема алгоритма работы ПС.

Таблица 1 – Оценки показателя качества.

Используемая модель	**Значение показателя качества AUC ROC**
SimpleGAN	0.81
DCGAN	0.76

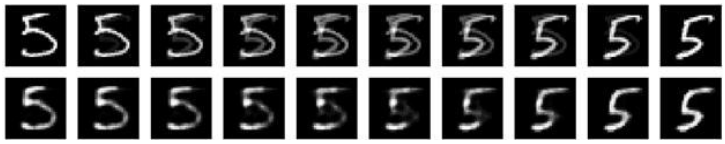

Рисунок 2 – Пример работы ПС для модели SimpleGAN.

Рисунок 3 – Пример работы ПС для модели DCGAN.

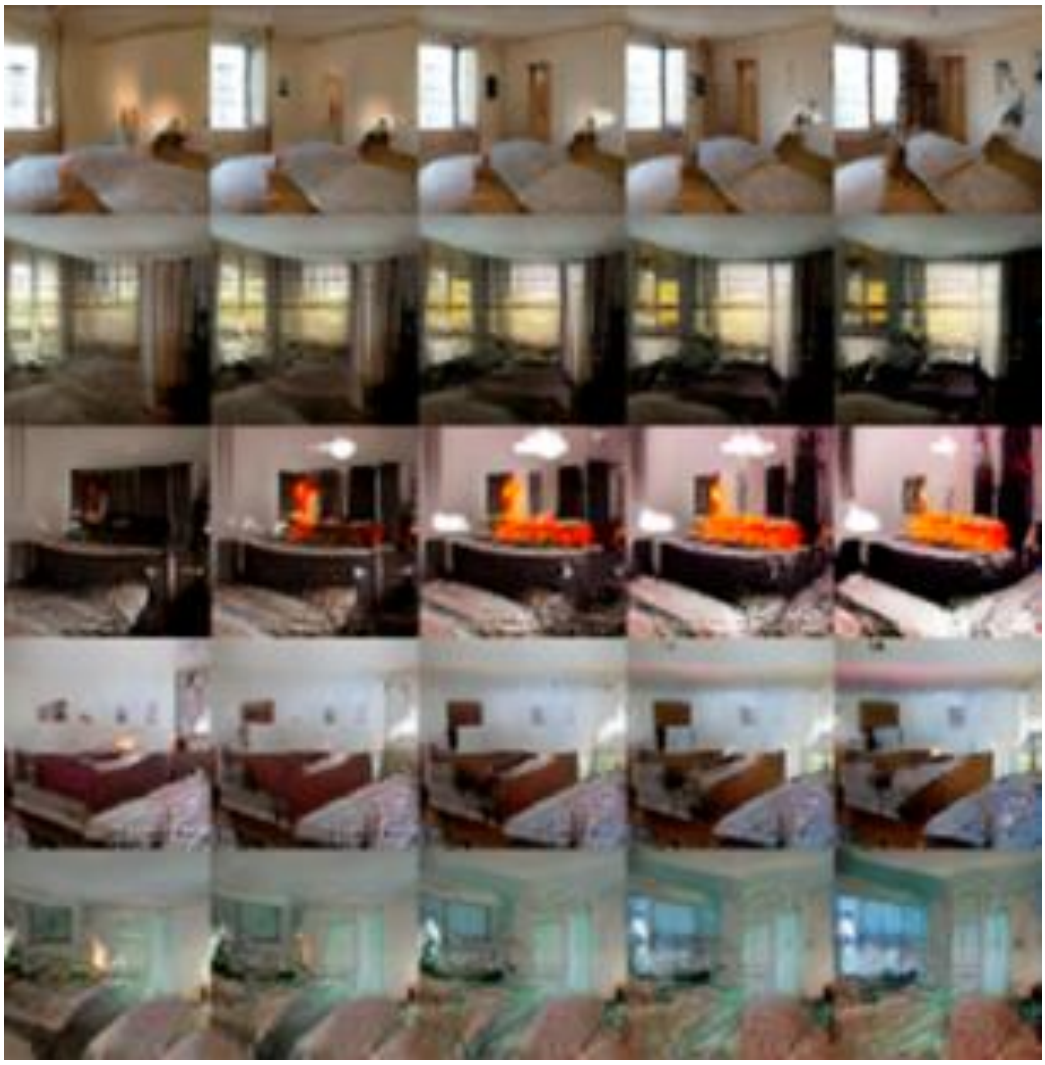

Рисунок 4 – Пример работы ПС для модели DCGAN.

Заключение

В процессе выполнения данной работы получены следующие основные результаты.

Изучена предметная область, выявлены и кратко описаны используемые при разработке ПС необходимые вспомогательные средства, а именно SimpleGAN и DCGAN. Выполнена проверка работы ПС на тестовом наборе данных, которая показала его работоспособность, т.е. возможность выполнять поставленные перед ним задачи. Выполнена проверка гипотезы о непрерывности векторов скрытого состояния модели DCGAN. Таким образом, успешно разработано ПС, решающее такую задачу компьютерного зрения [8], как генерация необходимых пользователю изображений. В дальнейшем возможно усовершенствование программного средства, например, добавление новых функциональных возможностей на языке Python [9], работа над дизайном ПС.

Список литературы

1. Саммерфилд М. Программирование на Python 3. Подробное руководство. – М.: Литрес, 2009. – 607 с.
2. Kaggle. Система организации конкурсов по исследованию данных, а также социальная сеть специалистов по обработке данных и машинному обучению. [Электронный ресурс]. https://www.kaggle.com (Дата обращения 12.05.2024).
3. Ахметов А.Ф. Разработка программного средства для генерирования изображений с использованием генеративных нейронных сетей (рукопись). / Отчет о НИР (8-й семестр), руков. Кулик С.Д. – М.: НИЯУ МИФИ, 2024. – 33 с.
4. Солем Я.Э. Программирование компьютерного зрения на языке Python. – М.: ДМК Пресс, 2016. – 312 с.
5. Клетте Р. Компьютерное зрение. Теория и алгоритмы. – М.: ДМК Пресс, 2014. – 325 с.
6. Любанович Б. Простой Python. Современный стиль программирования. 2-е изд. – СПб.: Издательский дом «Питер» 2021. – 592 с.
7. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: ДМК Пресс, 2005. – 231 с.
8. Чорбаа Н.А., Ле Ань Ту, Толстой И.М. Сравнительный анализ методов детектирования объектов на радиолокационных изображениях при помощи нейронных сетей // Научный результат. Информационные технологии. 2020. № 4. [Электронный ресурс]. https://cyberleninka.ru/article/n/sravnitelnyy-analiz-metodov-detektirovaniya-obektov-na-radiolokatsionnyh-izobrazheniyah-pri-pomoschi-neyronnyh-setey (Дата обращения: 21.12.2023).
9. Бэрри П. Изучаем программирование на Python. – М.: Литрес, 2019. – 581 с.

УДК 004.5

РАЗРАБОТКА НЕЙРОКОМПЬЮТЕРНОГО ИНТЕРФЕЙСА НА ОСНОВЕ TGAM1, ARDUINO UNO И СУХИХ ЭЛЕКТРОДОВ

А.П. Зинченко

студент группы С20-501 НИЯУ МИФИ, zap-02@mail.ru

Аннотация. Данная работа опирается на принципы работы нейроинтерфейсов, связана с техническими решениями и их реализациями и представляет результаты разработки программного обеспечения для них. В процессе разработки и исследования использовались методы цифровой схемотехники вычислительных устройств и инструментальные средства создания программного обеспечения. В ходе исследования были разработаны устройство нейрокомпьютерного интерфейса и прототип программного средства для сбора и хранения данных нейрокомпьютерного интерфейса на основе TGAM1, Arduino UNO и сухих электродов. Программное обеспечение позволяет сохранять и визуализировать сигналы в режиме реального времени, что важно для оперативного анализа полученных экспериментальных данных. Выполнены успешно тесты, подтверждающие функциональность нейроинтерфейса в различных сценариях.

Ключевые слова: нейрокомпьютерный интерфейс, электроэнцефалография, альфа ритм, бета ритм, сухой электрод, Python, PostgreSQL.

Введение

Современные нейроинтерфейсы [1] позволяют устанавливать прямое соединение между нервной системой человека и устройствами, открывая новые возможности для взаимодействия с техникой и медициной. Эта революционная технология открывает перед нами новые горизонты: от улучшения качества жизни людей с ограниченными возможностями до расширения человеческих способностей. Отличительной особенностью разработанного программного обеспечения (ПО) является дублирование информации в текстовый файл и базу данных PostgreSQL, что обеспечивает удобное хранение и последующую обработку данных. Предлагаемое средство, реализующее нейроинтерфейс, может быть использовано в учебных целях, а также для дальнейшего развития проекта, направленного на распознавание различных состояний человека (например, для медицинской диагностики). Разработка ПО велась на языке Python с использованием библиотек matplotlib и sqlalchemy.

Цели. Комплексное исследование предметной области, обзор источников, предложение о структуре программного средства, разработка алгоритма работы программного средства, реализация программного средства на заданном языке программирования для съёма данных (в базу данных и/или файл) и их отображения в графическом представлении.

В процессе разработки и исследования использовались методы цифровой схемотехники вычислительных устройств и инструментальные средства разработки ПО.

Нейрокомпьютерный интерфейс

Работа выполнялась в два этапа.

На первом этапе был успешно разработан приемник (рис.1) сигналов мозговой активности для нейроинтерфейса. На втором этапе [2] бы разработан прототип программного средства для сбора и хранения данных нейрокомпьютерного интерфейса на основе TGAM1 [3], Arduino UNO [4] и сухих электродов. Основные характеристики TGAM1 представлены в табл.1. Программное обеспечение позволяет сохранять и визуализировать сигналы в режиме реального времени, что важно для оперативного анализа полученных экспериментальных данных. Выполнены успешно тесты, подтверждающие функциональные возможности нейроинтерфейса в различных сценариях.

Рисунок 1 – Экспериментальное применение нейроинтерфейса (глаза закрыты).

Таблица 1 – Основные характеристики TGAM1 [3].

№	Параметры	Значение
1	Опорное напряжение	2.97V ~ 3.63V
2	Защита от электростатического разряда	4kV Контактный разряд 8kV Воздушный разряд
3	Стандарт выходного интерфейса	UART(Serial)
4	Выходная скорость передачи данных	1200, 9600, 57600 б/с

Кратко остановимся на ритмах ЭЭГ [5]. ЭЭГ – это запись электрической активности мозга, которая отражает его состояние. Ритмы ЭЭГ – регулярные колебания этой активности, связанные с определенными мозговыми процессами.

Основные ритмы ЭЭГ.

Альфа-ритм (8-13 Гц): наблюдается в состоянии покоя, при закрытых глазах. Преобладает в затылочной области.

Бета-ритм (14-35 Гц): характерен для умственной активности, открытых глаз. Более выражен в лобных долях.

Тета-ритм (4-7 Гц): появляется при неглубоком сне, кислородном голодании, наркозе.

Дельта-ритм (0,5-3 Гц): свидетельствует о глубоком сне, наркозе.

При переходе от покоя к активности альфа-ритм сменяется на бета-ритм. Это явление называется реакцией активации. В работе нас интересуют альфа и бета ритмы, поскольку они отражают состояние человека в покое и при умственной активности. Мы исследуем, как меняются эти ритмы при переходе от покоя с закрытыми глазами к активным действиям с открытыми глазами.

Пример полученных данных с помощью нейроинтерфейса кратко представлен в табл. 2.

Таблица 2 – Пример полученных данных (человек активен).

Показатели				
Секунды отсчета	**low alpha**	**high alpha**	**low beta**	**high beta**
1	30069	21214	8101	8951
2	27353	4792	11756	7496
3	59366	15807	4790	4956
4	36206	7312	9096	7979
5	2376	16180	4740	3430
6	52888	31557	362	11206
7	47424	52593	12235	8564
8	34219	21078	16181	4131
9	3288	6629	16866	6500
10	30069	21214	8101	8951
Средние значения	28891	27560	8825	6833
Медианные значения	24654	15994	7884	6783

В качестве программной части проекта, отвечающей за сохранение входных сигналов, было решено разработать программу, сохраняющую данные в базу данных PostgreSQL, а также в отдельный файл на внешнем устройстве. Использован язык SQL [6, 7]. Данное решение обуславливается дублированием сохраняемых сигналов, чтобы в случае потери одного из компонентов, существовал способ восстановления потерянных данных из второго источника. Для восстановления данных в PostgreSQL отдельно была разработана команда, запускаемая из терминала СУБД PgAdmin4.

Данные, сохраняемые в базе данных с помощью СУБД PgAdmin4, пред-

ставляются в виде специальной таблицы (см. табл. 3), для которой была разработана необходимая структура.

Таблица 3. Пример таблицы хранения данных.

Вид сигнала	Тип данных
signal_strength	big integer
attention	big integer
meditation	big integer
delta	big integer
theta	big integer
low_alpha	big integer
high_alpha	big integer
low_beta	big integer
high_beta	big integer
low_gamma	big integer
high_gamma	big integer

Пример графика альфа сигналов представлен на рис.2.

Для визуализации экспериментальных данных в режиме реального времени была успешно разработана программа, отображающая графики, обновляемые с частотой один раз в секунду. Эта программа позволяет предоставить пользователю четыре основных параметра, которые позволяют оценить активность человека:

1) low alpha: относительная мощность нижней половины альфа-ритма, выраженная в мкВ, в диапазоне от 0 до 100;
2) high alpha: относительная мощность верхней половины альфа-ритма, выраженная в мкВ, в диапазоне от 0 до 100;
3) low beta: относительная мощность нижней половины бета-ритма, выраженная в мкВ, в диапазоне от 0 до 40;
4) high beta: относительная мощность верхней половины бета-ритма, выраженная в мкВ, в диапазоне от 0 до 40.

Необходимое программное обеспечение было разработано на языке Python [8-10] и представляет собой классический исполняемый файл, который позволяет выводить графики в режиме реального времени (см. рис. 3).

Заключение

В ходе исследования были разработаны устройство нейрокомпьютерного интерфейса и прототип программного средства для сбора и хранения данных нейрокомпьютерного интерфейса на основе TGAM1, Arduino UNO и сухих электродов. Программное обеспечение позволяет сохранять и визуализировать сигналы в режиме реального времени, что важно для оперативного анализа полученных экспериментальных данных. Выполнены успешно тесты, подтверждающие функциональные возможности нейроинтерфейса в различных сценариях.

Исследование демонстрирует успешность ПО для нейроинтерфейса на базе чипа TGAM1, открывая новые перспективы для его применения в различных областях, включая медицину и технологии помощи людям с ограниченными возможностями. Систему необходимо в дальнейшем развивать, особенно в области опознавания состояния человека с помощью алгоритмов машинного обучения.

Можно полагать что проведенное исследование и выполненная разработка программного обеспечения сможет предоставить дополнительную информацию о работе нейроинтерфейсов и их возможном потенциале в будущем.

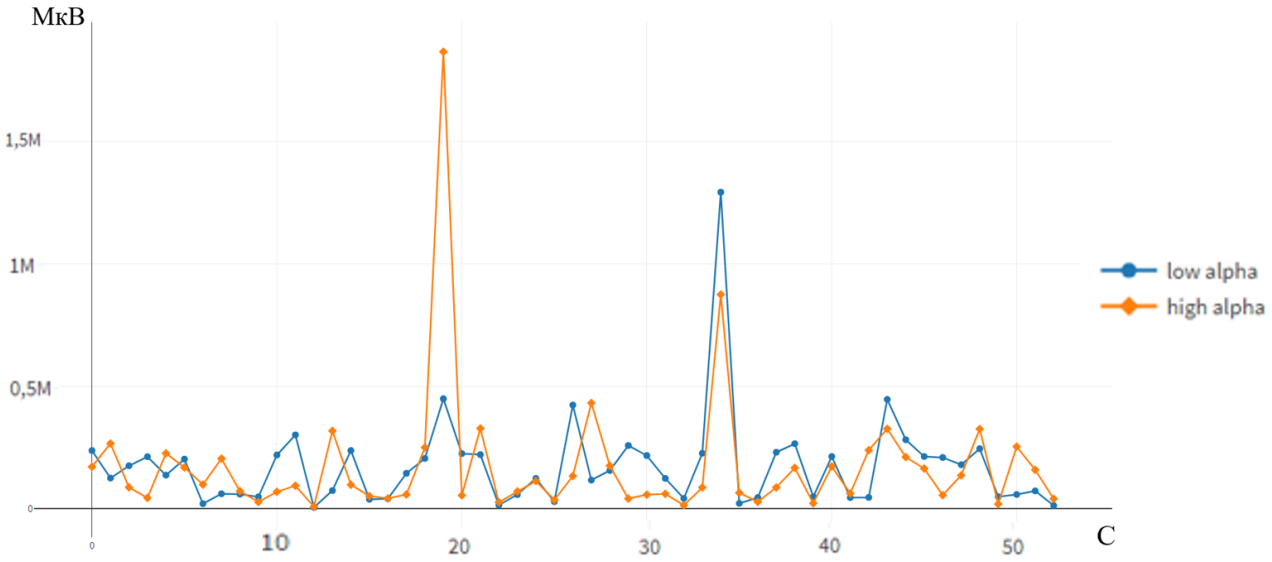

Рисунок 2 – Пример альфа сигналов (не экранированных) при закрытых глазах.

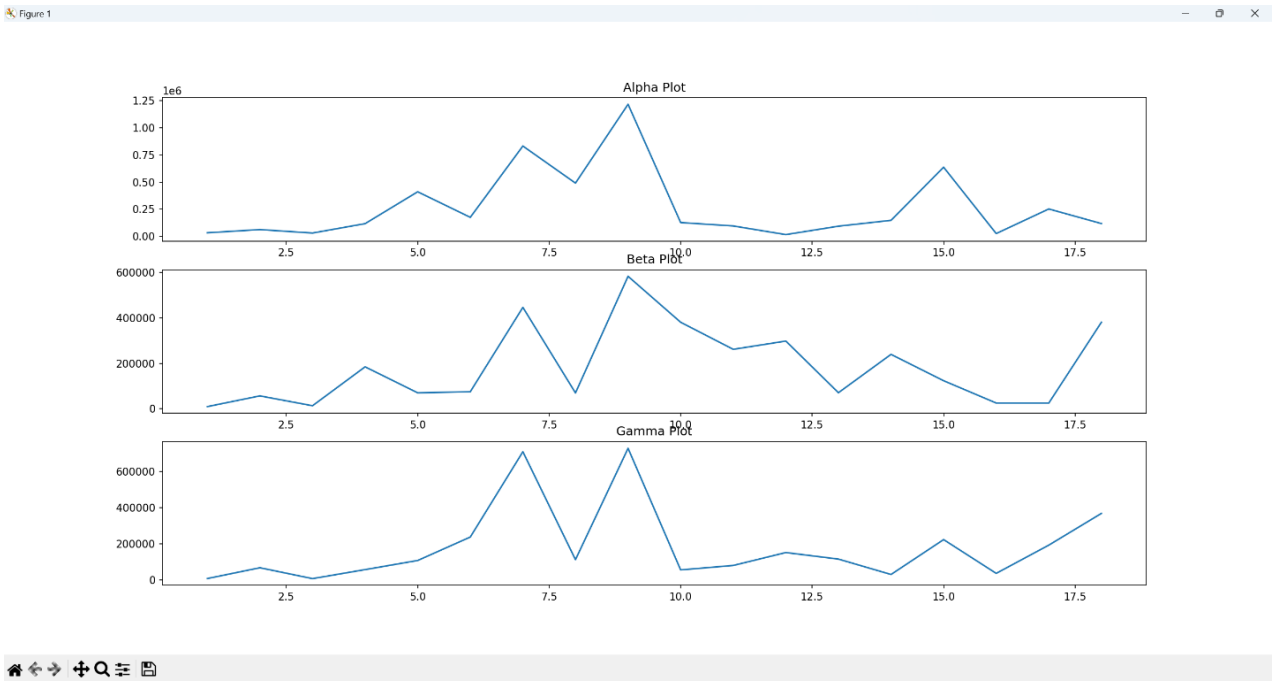

Рисунок 3 – Пример работы программы для визуализации сигналов.

Список литературы

1. Цулейскири Е.Г. Применение нейрокомпьютерных интерфейсов для реабилитации и улучшения условий жизни пациентов с нарушениями моторных функций нервной системы [Электронный ресурс]. https://interagro.donstu.com/wp-content/uploads/2022/12/203-206.pdf (Дата обращения 20.04.2024).
2. Зинченко А.П. Разработка прототипа программного средства для сбора и хранения данных специального нейрокомпьютерного интерфейса на основе TGAM1, Arduino UNO и сухих электродов (рукопись). / Отчет о НИР (8-й семестр), руков. Кулик С.Д. – М.: НИЯУ МИФИ, 2024. – 30 с.
3. TGAM1 SPEC SHEET [Электронный ресурс]. https://opendevices.ru/wp-content/uploads/2013/05/neurosky_eeg_brainwave_chip_and_board_tgam1.pdf (Дата обращения 11.12.2023).
4. Паоло А. Изучаем Arduino. Руководство для начинающих. – М.: ДМК Пресс, 2021. – 237 с.
5. Анохин К.В. Ритмы ЭЭГ. CMI Brain Research. [Электронный ресурс]. https://cmi.to/ритмы-ээг/ (Дата обращения 23.12.2023).
6. Шилдс У. SQL: быстрое погружение. – СПб.: Издательский дом Питер, 2022. – 224 с.
7. Кригель А., Трухнов Б. SQL. Библия пользователя". 2-е изд. – М.: Диалектика, 2010. – 764 с.
8. Любанович Б., Простой Python. Современный стиль программирования. 2-е изд. – СПб.: Издательский дом «Питер» 2021. – 592 с.
9. Бэрри П. Изучаем программирование на Python. – М.: Литрес, 2019. – 581 с.
10. Саммерфилд М. Программирование на Python 3. Подробное руководство. – М.: Литрес, 2009. – 607 с.

УДК 004.5

РАСПОЗНАВАНИЕ ПАТТЕРНОВ СУБВОКАЛИЗАЦИИ И ИССЛЕДОВАНИЕ ИХ ПРИМЕНЕНИЯ В ЗАДАЧЕ УПРАВЛЕНИЯ МОБИЛЬНЫМ РОБОТОМ

Б.О. Лавров
студент группы М22-512 НИЯУ МИФИ, lavrov.bogdan@list.ru

Аннотация. В статье рассмотрено распознавание паттернов субвокализации (тихой речи) и исследование их применения в задаче управления мобильным роботом. Процесс распознавания паттернов ЭМГ сигнала обычно состоит из трех этапов: предварительная обработка сигнала, т.е. снижение влияния внешних шумов и улучшение соотношение сигнал/шум; извлечение полезной информации; классификация. Система состоит из набора электродов, крепящихся к горлу и подключенных к компьютеру, который анализирует сигнал и преобразует его в текст или голос. Представлены результаты эксперимента, который показал, что модель робота не полностью адаптирована к условиям реального использования и требует дополнительной настройки или обучения.

Ключевые слова: паттерны, классификация ЭМГ сигналов, метрики качества модели, метод выбора признаков, модель машинного обучения.

Введение

При любом процессе возбуждения живой ткани возникает биоэлектрический потенциал. В определенных пределах существует прямая зависимость между напряжением, создаваемым мышцами, и уровнем биоэлектрического потенциала. В связи с этим альтернативой произносимой (вокализованной) речи может быть непроизносимая речь, т.е. внутренняя речь – процесс мысленного представления слова без его проговаривания. В процессе формирования речи помимо акустических сигналов организм генерирует и другие биологические сигналы, которые возникают в организме во время произнесения речи и могут варьироваться от движений артикуляционных органов (гортань, голосовые связки, язык, небо, зубы, губы, носоглотка) до активности нейронов в мозге. В зависимости от конкретных расстройств, поражающих человека, некоторые этапы процесса формирования речи могут быть нарушены, в то время как другие остаются неизменными. Поэтому биологические сигналы, исходящие от неповрежденных частей речевого аппарата, могут быть зарегистрированы, обработаны и преобразованы либо в звуковой сигнал, либо в текст. Иными словами, путем регистрации биосигналов, поступающих от элементов речевого аппарата, можно распознать речь без ее произнесения, что является чем-то большим, чем мысли, но меньшим, чем фактическая речь, и представляет собой желание что-то сказать. Наиболее известными субвокальными системами являются:

– Система субвокального распознавания NASA для астронавтов, которая позволяет им общаться в условиях высокого шума или отсутствия звука; система состоит из набора электродов, крепящихся к горлу и подключенных к компьютеру, который анализирует сигнал и преобразует его в текст или голос [1];
– AlterEgo – система, разработанная в Массачусетском технологическом институте, которая позволяет человеку общаться с компьютером или другими людьми без открытия рта; устройство надевается на голову и имеет четыре электрода, сигнал передается на беспроводные наушники, которые воспроизводят ответ компьютера или собеседника [2].
– Subvocal – система, разработанная в Вустерском политехническом институте, которая позволяет человеку управлять музыкальными инструментами с помощью субвокализации [3].

Основной целью изучения ЭМГ сигналов, является извлечение полезной информации о мышечной активности. Эта задача носит название распознавания паттернов ЭМГ сигнала, которая обычно состоит из трех этапов [4]:

1) предварительная обработка сигнала, т.е. снижение влияния внешних шумов и улучшение соотношение сигнал/шум;
2) извлечение полезной информации;
3) классификация.

Ключевые технические параметры для исследования субвокализации с помощью ЭМГ могут быть следующими:

– расположение электродов,
– количество каналов данных – обычно используются от двух до восьми каналов данных.
– частота целевого сигнала – обычно целевой сигнал имеет частоту от 0 до 500 Гц.
– частота считывания данных – обычно частота считывания данных составляет от 1000 до 5000 Гц.

Способы обработки сигнала

Несмотря на удобство поверхностной ЭМГ, существуют также и недостатки, которые негативным образом влияют на качество исследований. Когда электромиографический датчик установлен непосредственно на коже, то он неизбежно подвергается действию как внешних, так и физиологических факторов, влияющих на качество обнаружения мышечной активности. Помимо шумов мышц, которые располагаются в зоне чувствительности электрода и вносят паразитные помехи посредством собственной электрической активности, на качество регистрации влияет целый ряд факторов [5-7]:

– Шум, вызванный электронным оборудованием;
– Шум окружающей среды;
– Артефакт, связанный с движением;
– Вход в режим насыщения;
– Естественная нестабильность сигнала.

Способы обработки сигнала для распознавания паттернов субвокализации могут включать следующие этапы:

– Фильтрация: применение различных фильтров для удаления шума и артефактов из сигнала, таких как фильтры высоких и низких частот, полосовые фильтры, вейвлет-фильтры и т.д.;
– Извлечение признаков: это процесс преобразования входного сигнала в набор числовых характеристик, которые отражают его свойства и позволяют отличать различные паттерны субвокализации; эти характеристики называются признаками и используются для обучения моделей распознавания речи; среди основных особенностей ЭМГ сигнала можно выделить временные и частотные характеристики.

Временные характеристики сигнала

Суммирование ЭМГ сигнала представляет собой суммирование сигналов ЭМГ в окне определенного размера, определяемого количеством отсчетов N. Впоследствии данная сумма сравнивается с ранее определенным порогом и принимается решение об активности мышцы.

$$G = \sum_{i=1}^{N} |x_i| ,$$

где x_i — это измеренный сигнал ЭМГ, N — размер исследуемого окна (число отсчетов).

Среднее абсолютное значение (MAV) является одной из самых популярных характеристик, используемых при анализе сигналов ЭМГ. MAV имеет несколько модификаций. Условно их можно разделить на два типа. Первый тип — это модификация с добавлением новой переменной w_i, которая является своего рода весом промежутка внутри окна. Второй тип – это модификация с использованием нестационарной переменной w_i.

В [8] для анализа предлагается энергия ЭМГ сигнала, рассчитанная как сумма квадратов значений амплитуды ЭМГ сигнала. Другими характеристикой для анализа ЭМГ сигнала являются дисперсия (VAR) и среднее абсолютное значение степенной функции считываемого сигнала. Среднеквадратичное значение (RMS) также является популярной характеристикой для анализа сигнала ЭМГ. Длина волны (WL) – это мера сложности ЭМГ сигнала. Этот параметр определен как совокупная величина изменения амплитуды сигнала ЭМГ за временной сегмент. Изменение средней амплитуды

(AAC) почти эквивалентно WL, за исключением того, что длина волны усреднена. Пересечение нуля (ZC) – это мера частотной информации сигнала ЭМГ, определенная во временной области. Значение амплитуды сигнала ЭМГ может несколько раз пересечь нулевой уровень амплитуды. Изменение знака наклона (SSC) – это ещё один способ представления частотной информации сигнала ЭМГ. SSC определяется количеством раз, когда наклон сигнала ЭМГ меняет знак.

Частотные характеристики

Частотные характеристики сигнала вычисляются на основе преобразования Фурье. Спектральная плотность мощности (P) – это мера распределения энергии сигнала по частотам. Она показывает, какая часть общей энергии сигнала приходится на каждую частоту. P вычисляется с помощью преобразования Фурье от временного представления сигнала.

Средняя частота (MNF) рассчитывается как сумма произведений спектра мощности ЭМГ на частоту, разделенная на суммарную мощность спектра. MNF также называют центральной частотой f_c или спектральным центром тяжести.

Исследование ЭМГ сигналов

Для измерения активности мышц в работе были использованы датчики ЭМГ Glove от Seeed Studio. Датчик получает сигнал от мышц, затем обрабатывается с двукратным усилением и подается на Arduino.

Сигналы ЭМГ регистрировались с кожи горла с частотой дискретизации 2.8 кГц в комнате с постоянной температурой. Использовались только одноразовые ЭМГ электроды. Вокруг участника эксперимента была установлена разная оргтехника: принтер, ноутбук, блоки питания, сетевые переключатели и т.д.; таким образом, естественные магнитные поля, окружающие человека в повседневной жизни, не были искусственно занижены. Затем испытуемый последовательно произносил шёпотом и обычной речью слова «вперед», «направо», «назад», которые записывались в соответствующие файлы. Запись набора данных осуществлялась несколько раз в разное время.

Модели для работы с данными

Для каждого файла данных (240 файлов) были рассчитаны все временно-частотные признаки (31 параметр), описанные выше. Для выявления наиболее важных признаков, была построена ковариационная матрица, которая предоставляет информацию о взаимосвязи (корреляции) между признаками. Чем ближе к нулю ковариационное значение, тем лучше (это означает, что признаки независимы).

Анализ ковариационной матрицы показал, что к сильно коррелирующим признакам относятся MAV, MAV_1, MAV_2, TM3, TM4, TM5, LOG, SSC. Их решено исключить из анализа.

Используя метод главных компонент (PCA), было решено снизить размерность пространства признаков. Путем определения кумулятивной выборочной дисперсии признаков выясняется, какие признаки определяют большую часть дисперсии данных. Методом главных компонент было выбрано 13 признаков из 31, которые определяют 98% дисперсии данных.

Для классификации паттернов были исследованы следующие модели: SCV, DecisionTreeClassifier, GradientBoostingClassifier, LogisticRegression, RandomForestClassifier. Наиболее высокую точность распознавания 83.5% показал RandomForestClassifier, для которого в дальнейшем были подобраны наилучшие гиперпараметры. В результате анализа установлены следующие закономерности влияния гиперпараметров на точность модели:

- n_estimators: значения 300, 500, 700, показывают наилучшие средние результаты;
- min_samples_split: небольшие значения, вроде 2 и 7, показывают наилучшие результаты; хорошо выглядит и значение 23; можно исследовать несколько значений этого гиперпараметра, превышающих 2, а также – несколько значений около 23;
- min_samples_leaf: возникает такое ощущение, что маленькие значения этого гиперпараметра дают более высокие результаты; а это значит, что мы можем испытать значения между 2 и 7;
- max_features: вариант sqrt даёт самый высокий средний результат;
- max_depth: тут чёткой зависимости между значением гиперпараметра и результатом работы модели не видно, но есть ощущение, что значения 2, 3, 7, 11, 15 выглядят неплохо;
- bootstrap: значение False показывает наилучший средний результат.

Используя эти результаты, с помощью алгоритма GridSearchCV был выполнен более тонкий поиск наилучшей комбинации гиперпараметров: 'bootstrap': False, 'max_depth': 13, 'max_features': 'sqrt', 'min_samples_leaf': 7, 'min_samples_split': 2, 'n_estimators': 700. Средняя точность классификации паттернов составила 86%.

Тестирование предварительно обученной модели в режиме реального времени проводилось на мобильной роботизированной тележке. Произносились последовательно слова-команды: «направо», «налево» «назад», «вперед» и «стоп». Всего было выполнено 20 циклов. Для каждого слова-команды была составлена матрица ошибок и рассчитаны: полнота (*recall*) –

способность алгоритма обнаруживать данное слово, точность результата измерений (*precision*), точность измерений (*accuracy*) и *f*-мера – конечный показатель эффективности модели.

Полученные результаты показали, что модель не полностью адаптирована к условиям реального использования и требует дополнительной настройки и обучения.

Заключение

В статье рассмотрены методы распознавания паттернов субвокализации ЭМГ сигнала. Проведен обзор способов обработки сигнала для распознавания паттернов субвокализации. Разработана модель классификации ЭМГ сигнала на основе RandomForestClassifier, которая на тестовых данных имеет точность предсказания 86%. Полученный классификатор в процессе проверки продемонстрировал низкое качество распознавания мышечной активности в режиме реального времени. Тестирование модели в режиме реального времени показало, что она не может точно распознавать команды пользователя. Для улучшения результатов необходимо провести дополнительную тренировку модели на более широком наборе данных. Полученные выводы могут служить основой для дальнейших исследований и развития технологии распознавания субвокализации.

Список литературы

1. Bluck J. NASA Develops System To Computerize Silent, 'Subvocal Speech'. NASA Ames Research Center, Moffett Field, Calif., March 2004. [Электронный ресурс]. https://www.nasa.gov/centers/ames/news/releases/2004/04_18AR.html. (Дата обращения 04.05.2023).
2. Kapur A., Kapur S., Maes P. AlterEgo: A Personalized Wearable Silent Speech Interface. 2018, pp. 43-53.
3. A Review on Electromyography Decoding and Pattern Recognition for Human-Machine Interaction / M. Simao, N. Mendes, O. Gibaru, P. Neto // IEEE 460 Access, 2019, V. 7, pp. 39564-39582.
4. Resolving the 425 limb position effect in myoelectric pattern recognition / A. Fougner, E. Scheme, A. Chan, K. Englehart, O. Stavdahl // IEEE Trans. Neural Rehabil. Syst. Eng., 2011, V. 19, N. 6, pp. 644-651.
5. Reaz M., Hussain M., Mohd-Yasin F. Techniques of EMG signal analysis: detection, processing, classification and applications. // Biol Proced Online, 2006, V. 8, pp. 11-35.
6. Examining the adverse effects of limb position on pattern recognition based myoelectric control / E. Scheme, A. Fougner, A. Chan, K. Englehart // IEEE inginering in Medicine and Biology Soc., 2010, pp. 6337-6340.
7. Du S., Vuskovic M. Temporal vs. spectral approach to feature extraction from prehensile EMG signals. // In Proceedings of IEEE International Conference on Information Reuse and Integration. 2004, pp. 344–350.
8. Phinyomark A., Phukpattaranont P., Limsakul C. Feature reduction and selection for EMG signal classification. // Expert Systems with Applications, 2012, V. 39, № 8, pp. 7420-7431.

УДК 004.942

РАЗРАБОТКА И ОЦЕНКА ФУНКЦИОНАЛЬНЫХ ВОЗМОЖНОСТЕЙ ТРЕХМЕРНОЙ МОДЕЛИ ДЛЯ ТЕСТИРОВАНИЯ АЛГОРИТМОВ НАВИГАЦИИ РОБОТА PATROLBOT

Д.А. Маркова
Аспирант группы А23-502 НИЯУ МИФИ, destromo@yandex.ru

Аннотация. Статья посвящена разработке трехмерной модели второго этажа НИЯУ МИФИ для проверки алгоритмов навигации автономного робота PatrolBot. Модель, созданная в Blender3D и экспортированная в Gazebo, позволяет имитировать реальные условия эксплуатации робота, что важно для точности испытаний и отладки систем управления. Основные этапы включают детализацию модели с использованием текстур и визуальных элементов, а также последующие испытания в симуляционной среде для оценки функциональных возможностей и корректности алгоритмов. Результаты подтверждают эффективность использования виртуальной модели для предварительных тестов, снижая риски и затраты на эксперименты с реальными роботами.

Ключевые слова: трехмерное моделирование, алгоритмы навигации, робототехника, симуляционная среда, Blender3D, Gazebo, ROS (Robot Operating System).

Введение

В современном мире робототехника проникает во все сферы жизни, от промышленности до быта, что делает разработку интеллектуальных систем управления (ИСУ) актуальной научно-инженерной задачей. Основной проблемой при разработке ИСУ является необходимость их тестирования, которое, если проводить его на реальных роботах, становится крайне затратным и трудоемким. Это обуславливает поиск альтернативных методов отладки.

Применение симуляционных платформ типа Gazebo и ROS в разработке алгоритмов для управления автономными роботами стало ключевым моментом в современной робототехнике. Эти инструменты предоставляют возможность проводить тесты в полностью контролируемой среде, что позволяет ученым и инженерам проводить обширное тестирование без риска для оборудования и существенного увеличения затрат. Симуляции также ускоряют процесс разработки, поскольку они позволяют моделировать разнообразные операционные сценарии и немедленно оценивать результаты изменений в алгоритмах.

В данной работе предлагается методика отладки ИСУ на трехмерной модели, что позволяет значительно снизить затраты и ускорить процесс разработки. Разработка была поделена на следующие этапы:

1) Моделирование работы робота – виртуальное тестирование алгоритма на разработанной модели;
2) Создание трехмерной модели среды – в данном случае второго этажа НИЯУ МИФИ;

3) Разработка и интеграция алгоритма управления – создание и внедрение алгоритмов навигации и взаимодействия с препятствиями.

Проверка адекватности и эффективности методики была проведена на основе ряда тестов, симулирующих реальные условия работы мобильного робота. Результаты подтвердили, что предложенный подход позволяет не только адекватно оценить работоспособность ИСУ, но и выявить потенциальные недостатки алгоритма до его реализации в натуральных условиях.

Теоретическое обоснование проблемы

В последние годы значительные исследования были сосредоточены на развитии и интеграции робототехнических систем в различные сферы, включая область образования и научных исследований. Особое внимание уделяется разработке и моделированию роботизированных систем с использованием таких инструментов, как ROS (Robot Operating System) и Gazebo. Разработанные по такой технологии роботы используются для различных целей. Ниже приведены примеры таких роботизированных систем.

1. Управления роботами поисково-спасательной службы (SAR) в условиях, где инфраструктура связи может быть полностью уничтожена [1]. В работе представлена коммуникационная архитектура, основанная на технологии LoRa LPWAN.

2. Моделирование, симуляция и управление роботизированным манипулятором [2]. Автор описывает применение математического моделирования и программного обеспечения, такого как MATLAB и Simulink, для анализа и контроля роботизированной системы. Что подчеркивает значимость теоретических основ мехатроники и автоматического управления в робототехнике, а также их роль в образовательных программах по ИТ.

3. Автономная навигация лесного робота с использованием сканирующего лазера [3]. Рассматривается проект Innovative Forest Plantation, направленный на автоматизацию задач ухода за лесными насаждениями. Исследование фокусируется на автономной навигации с использованием SLAM для эффективного перемещения робота в полуструктурированной среде, где традиционные методы локализации, такие как GPS, неэффективны. Использование ROS и Gazebo в этом контексте подчеркивает их важность для разработки и тестирования сложных систем автономной навигации.

Данные примеры явно демонстрируют, что совместное применение ROS и Gazebo достаточно эффективно и протестировано множеством авторов. Таким образом можно сделать вывод, что эту совокупность технологий следует использовать для моделирования разрабатываемой системы автономной навигации.

Описание проекта PatrolBot

ROS и Gazebo представляют собой технологии в современной робототехнике, которые существенно помогают и облегчают процесс разработки, тестирования и внедрение роботизированных систем.

ROS представляет собой гибкую фреймворк-платформу для написания программного обеспечения для роботов. Он включает в себя набор инструментов, библиотек и соглашений, которые призваны упростить задачу создания сложного и надежного поведения роботов в различных средах [4]. ROS предоставляет инструменты визуализации (например, RViz) и отладки.

Gazebo предлагает среду для симуляции, которая точно имитирует физические и визуальные аспекты роботизированных систем в трехмерной среде. Среда идеально интегрируется с ROS, что было рассмотрено выше на примере работ других авторов [5].

Использование ROS и Gazebo существенно облегчит интеграцию алгоритмов искусственного интеллекта и машинного обучения при разработке ИСУ для многих вариантов интеллектуальных робототехнических комплексов (ИРТК).

На первом этапе необходимо было создать модель робота. В процессе создания в Blender3D первоначально был выбран примитив "конус" для формирования основы корпуса робота. Эта базовая форма была модифицирована, были изменены её размеры и пропорции, чтобы точно соответствовать дизайну и функциональным особенностям реального корпуса PatrolBot. Далее, для создания колес робота был использован примитив "цилиндр", который подвергся аналогичной детализации. Размеры и форма цилиндра были изменены, а также были добавлены дополнительные элементы, включая обод и шину, для придания колесам более реалистичного вида. Кроме того, были тщательно подобраны и настроены материалы и текстуры, чтобы добиться максимальной визуальной точности (рис. 1).

В процессе моделирования также были созданы лидары и камера робота, используя подходы, аналогичные разработке корпуса и колес. Для лидаров применялся примитив "цилиндр", который затем детализировался с добавлением элементов, имитирующих сканер, а для камеры использовался примитив "куб", который преобразовывался для достижения реалистичной формы корпуса камеры и объектива. Создание бамперов включало использование комбинации примитивов "куб" и "цилиндр", которые были адаптированы для воссоздания физической структуры бамперов робота. Все элементы были объединены в единую модель с учетом их взаимного расположения и взаимодействия, используя инструменты Blender3D, такие как "родительские связи" и "ограничения".

Рисунок 1 – Модель PatrolBot.

Рисунок 2 – Модель коридора второго этажа НИЯУ МИФИ.

После создания и детализации всех компонентов модели был создан набор текстур для корпуса, колес и дополнительных деталей, используя различные методы, такие как текстурный рисунок, шум и карты окружения, чтобы улучшить визуальное качество модели. Затем модель была экспортирована в формат URDF с использованием специального плагина для последующего использования в симуляторе Gazebo.

Создание модели второго этажа НИЯУ МИФИ

Для отладки интеллектуальной системы управления (ИСУ) предложено использовать трехмерную модель второго этажа НИЯУ МИФИ, созданную в Blender3D и экспортированную в формате SDF для симуляций в Gazebo (рис. 2). Процесс разработки модели включал итерации детализации, в ходе которых добавлялись текстуры, цвета и другие визуальные атрибуты для достижения высокой степени реалистичности виртуальной среды.

В начале процесса моделирования второго этажа НИЯУ МИФИ в Blender3D были выполнены точные измерения с использованием лазерного дальномера. С помощью лазерного дальномера Bosch и соответствующего программного обеспечения были измерены размеры помещения, включая длину стен, высоту потолков и ширину дверных проемов. Эти данные стали основой для последующего моделирования в Blender3D.

В рамках данного исследования была предложена методика создания трехмерной модели для имитационного тестирования алгоритмов навигации автономного робота. Инициация проекта началась с создания первоначальной грубой модели второго этажа в программе Blender3D. Используя базовые геометрические фигуры, такие как кубы и плоскости, было выполнено приблизительное моделирование форм и размеров помещений.

Для повышения степени детализации модели, к первоначальной конструкции были добавлены дополнительные элементы: текстуры, цвета, материалы и другие визуальные детали, что позволило воссоздать реалистичное представление интерьера второго этажа (рис. 3). Каждый этап моделирования был направлен на максимальное приближение виртуальной среды к реальным условиям, что является критически важным для точности последующих испытаний алгоритмов навигации.

После завершения этапа детализации, модель была экспортирована в формат SDF (Simulation Description Format) при помощи специализированного плагина, что обеспечило совместимость с симулятором Gazebo. Экспериментально было проверено, что данная модель может быть успешно использована для симуляции работы мобильного робота.

После транспортировки модели в симулятор Gazebo было проведено экспериментальное тестирование (рис. 4). Этот этап включал проверку поведения мобильного робота в моделированной среде, анализ возможных коллизий и другие тесты для оценки корректности и реалистичности симуляции. Обнаруженные в ходе тестирования несоответствия или ошибки требовали возвращения модели в Blender3D для доработки, после чего обновленная версия модели повторно экспортировалась в Gazebo для дальнейшего тестирования.

Каждый этап разработки и тестирования подкреплялся документированием результатов, что позволяло систематически настраивать параметры и улучшать качество виртуальной среды, а также повышать точность алгоритмов навигации ИСУ. Результаты экспериментов подтвердили эффективность предложенной методики.

Рисунок 3 – Модель лаборатории «Робототехнтика» (В-208).

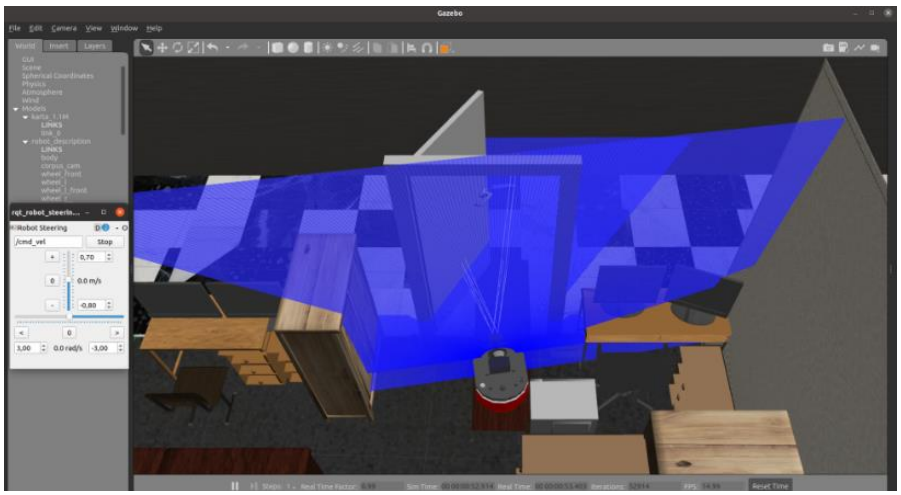

Рисунок 4 – Тестирование готовых моделей в ROS и Gazebo.

Заключение

Разработанная трехмерная модель среды позволила выполнить проверку алгоритмов навигации автономного робота PatrolBot. Модель, созданная в Blender3D и экспортированная в Gazebo, позволила имитировать реальные условия эксплуатации робота и повысить качество отладки систем управления. Полученные результаты подтвердили эффективность использования виртуальной модели для предварительного тестирования ИСУ, снизить затраты на эксперименты с реальными роботами.

Список литературы

1. Ben Abdallah, F., Bouali, A., Meausoone, P.-J. Autonomous Navigation of a Forestry Robot Equipped with a Scanning Laser // AgriEngineering, 2023, V. 5, № 1, pp. 1-11.
2. Удивительная техника. – М.: Эксмо, Наше слово, 2016. – 176 с.
3. Тывес Л. И. Механизмы робототехники. Концепция развязок в кинематике, динамике и планировании движений. – М.: Ленанд, 2014. – 208 с.
4. Винницкий Ю.А., Поляков К.Ю. Конструируем роботов на ScratchDuino. Первые шаги. – М.: Мир, 2016. – 183 с.
5. Мобильные роботы: робот-колесо и робот-шар. Под ред. Борисова А.В., Мамаева И.С., Караваева Ю.Л. – М.: Гостехиздат, 2013. – 532 с.

УДК 004.89

СИСТЕМА ЖЕСТОВОГО УПРАВЛЕНИЯ ДЛЯ ИНКЛЮЗИВНОГО ИСПОЛЬЗОВАНИЯ ДОМАШНИХ И ОБЩЕСТВЕННЫХ ТЕХНОЛОГИЙ

Д.А. Маркова[1], Р.В. Боронин[2]
аспирант группы А23-502 НИЯУ МИФИ, destromo@yandex.ru
аспирант группы А23-502 НИЯУ МИФИ, boronin.rostislav@yandex.ru

Аннотация. В работе представлена система управления умным домом с использованием языка жестов, разработанная для повышения доступности домашних и общественных технологий для людей с нарушениями слуха или зрения. Система интегрирует технологии Python, OpenCV, Mediapipe, TensorFlow и Keras для распознавания жестов в реальном времени и управления устройствами, такими как лампы, розетки и шторы. Тестирование системы показало точность распознавания жестов на уровне 81%. Опрос среди пользователей выявил, что 85% респондентов считают систему удобной и полезной. Результаты подтверждают эффективность предложенной системы для улучшения качества жизни людей с нарушениями слуха.

Ключевые слова: жестовое управление, инклюзивные технологии, умный дом, Python, OpenCV, Mediapipe, TensorFlow, Keras, распознавание жестов.

Введение

Технологии для управления умным домом посредством жестового языка набирает актуальность в контексте стремления к увеличению доступности и инклюзивности домашних и общественных технологий для лиц с ограниченными возможностями. Особую значимость данная проблематика приобретает в свете демографических и социотехнических тенденций последних лет, подтверждающих необходимость интеграции доступных технологических решений в повседневную жизнь этих категорий населения.

По данным Всемирной организации здравоохранения, более 430 миллионов человек в мире нуждаются в реабилитационных услугах из-за утраты слуха, что составляет более 5% населения планеты [1]. В условиях, когда большая часть населения с нарушениями слуха проживает в странах с низким и средним уровнем дохода, эффективность и доступность домашних и общественных технологий становятся особенно значимыми.

Проект по разработке системы управления умным домом через язык жестов представляет собой попытку расширить возможности взаимодействия для людей с различными ограничениями функций, включая нарушения слуха, зрения и речи.

На текущем этапе "умный дом" служит тестовой платформой, однако предвидится, что технология найдет применение в различных сферах, где необходимо безбарьерное взаимодействие человека с цифровыми системами и устройствами.

Основной задачей является создание системы, которая бы интерпретировала жесты и преобразовывала их в команды для выполнения различных действий в доме, делая повседневные задачи более доступными для глухих и слабослышащих людей.

Практическое применение системы управления умным домом через язык жестов охватывает широкий спектр областей, обеспечивая значительные улучшения в доступности и коммуникации для людей с нарушениями слуха. В медицинских учреждениях такая система может значительно упростить общение между пациентами и медицинским персоналом. Юридические учреждения, применяя эту технологию, могут обеспечить более доступное и понятное обслуживание для глухих клиентов, что особенно важно при работе с юридическими документами и проведении консультаций [2].

Применение системы в муниципальных службах или центрах обслуживания населения может упростить процесс получения различных услуг, делая их более доступными для граждан с нарушениями слуха. Это способствует повышению их самостоятельности и уверенности во взаимодействии с государственными структурами [3].

В образовательных учреждениях использование данной технологии может радикально изменить учебный процесс для студентов с нарушениями слуха, предоставляя им более полные и равные возможности для обучения и социальной адаптации.

Описание работы системы

В системе управления умным домом с использованием языка жестов интеграция технологий Python, OpenCV, Mediapipe, TensorFlow и Keras обеспечивает комплексный подход к обработке и распознаванию жестов в реальном времени. Вот как эти компоненты работают вместе.

Библиотека OpenCV используется для обработки видео. С помощью OpenCV система может получать изображения рук пользователя, что является первым шагом в процессе распознавания жестов [4].

Фреймворк Mediapipe обеспечивает детектирование ключевых точек рук на изображениях, полученных с помощью OpenCV. Mediapipe использует предварительно обученные модели машинного обучения для определения положения пальцев и ладони, что позволяет точно интерпретировать жесты пользователя.

После того как ключевые точки рук определены, данные о жестах передаются в модель глубокого обучения, созданную с помощью TensorFlow и Keras. Эти фреймворки используются для разработки, тренировки и внедрения нейронных сетей, которые могут классифицировать различные жесты на основе данных о движении рук.

Классифицированные жесты затем интерпретируются как команды, которые могут быть использованы для управления различными функциями умного дома – например, включение и выключение света, управление температурой или медиа системами [5].

В качестве основного языка программирования Python используется для связывания всех этих процессов в единую систему. Скрипты на Python организуют поток данных между различными библиотеками и модулями, обрабатывают ошибки и управляют пользовательским интерфейсом.

На первом этапе выбирается жест, который будет использоваться для обучения. Это может быть любой жест, который четко различим и хорошо виден на изображениях.

Затем создается большое количество фотографий, демонстрирующих выбранный жест. Эти фотографии должны включать различные углы, освещение и нюансы положения руки, чтобы система могла точно распознавать жест в различных условиях. Для каждого жеста сделано множество последовательных снимков.

Собранные фотографии передаются в систему машинного обучения. На рис. 1 показан пример интерфейса, который может использоваться для этого – Teachable Machine от Google. Это инструмент, позволяющий пользователю загрузить изображения и быстро обучить модель распознавать различные объекты или жесты.

В процессе обучения модель анализирует визуальные данные с фотографий, изучая особенности жеста. Это может включать определение ключевых точек на руках, распознавание уникальных паттернов и текстур кожи, а также другие визуальные маркеры, которые помогают системе распознавать жест в будущем.

После первоначального обучения модель тестируется на новых изображениях, чтобы проверить, насколько эффективно она распознает жесты. Если распознавание происходит с ошибками, можно провести дополнительные корректировки, добавив новые данные или настроив параметры модели.

Рисунок 1 – Тестирование жеста.

Рисунок 2 – Реализованные жесты.

Для реализации системы управления умным домом был выбран умный дом от "Алисы" – голосового помощника от компании Яндекс. Выбор этой платформы обусловлен следующими моментами:

– "Алиса" является одним из наиболее популярных голосовых помощников в России, что делает её интеграцию в умные дома востребованной и актуальной для большого количества пользователей;
– Платформа поддерживает широкий спектр умных устройств;
– Яндекс предоставляет обширные возможности для разработчиков по интеграции их решений с "Алисой", включая доступ к API и документации; это значительно облегчает процесс разработки и внедрения новых функций в систему умного дома.

В данный момент реализованы следующие жесты:

– Алиса – активирует голосового помощника.
– Открыть шторы — жест для команды открытия штор.
– Закрыть шторы – жест для команды закрытия штор.
– Включить розетку – жест для включения розетки.
– Выключить розетку – жест для выключения розетки.
– Включить свет – жест для включения света.
– Выключить свет – жест для выключения света.

Дизайн сайта для управления умным домом был сгенерирован с помощью Midjourney. На основной странице сайта представлены устройства, которые могут быть управляемыми, такие как лампа и розетка. Статус этих устройств отображается с помощью индикаторов.

На рис. 3 видно, что лампа включена, что обозначено зеленым индикатором, а розетка выключена, что обозначено красным индикатором. Также на странице есть камера, которая используется для распознавания жестов. При активной камере отображается статус "Камера включена" с зеленым индикатором, а в блоке распознавания показывается, какой жест в данный момент распознается системой.

Рисунок 3 – Статус устройства.

Для оценки удобства и эффективности системы управления умным домом с использованием жестов был проведен опрос среди глухих людей. В опросе при-

няли участие 20 человек. 17 человек (85%) отметили систему как удобную и полезную в повседневном использовании, тогда как 3 человека (15%) выразили нейтральное или отрицательное мнение о системе.

В табл. 1 представлены результаты анализа эффективности системы распознавания жестов, выполненного с использованием матрицы ошибок. Элементы, расположенные на главной диагонали матрицы, такие как 0,68 для жеста "Алиса", 0,92 для "Открыть шторы", 0,96 для "Закрыть шторы" и т.д., отражают долю правильно идентифицированных экземпляров для каждого жеста. Данные значения демонстрируют эффективность системы в точном распознавании соответствующих команд. Ненулевые значения вне диагонали отражают ошибки классификации, когда один жест ошибочно распознавался как другой, эта информация важна для дальнейшего анализа и улучшения точности системы.

Таблица 1 – Результаты тестирования системы.

	Алиса	Открыть шторы	Закрыть шторы	Выключить свет	Включить свет	Включить розетку	Выключить розетку
Алиса	0,68	0,04	0,08			0,08	0,12
Открыть шторы		0,92					0,08
Закрыть шторы			0,96				0,04
Выключить свет			0,08	0,88			0,04
Включить свет		0,04	0,12		0,8		0,04
Включить розетку			0,08	0,04	0,04	0,84	
Выключить розетку		0,12			0,28		0,6

Для подсчёта точности работы системы была использована следующая формула:

$$ACC = \frac{TP+TN}{TP+TN+FP+FN},$$

где TP – количество корректных распознаваний, TN – количество случаев, когда нужный жест не распознался, FP – ошибка первого рода (ложноположительное срабатывание), FN – ошибка второго рода (ложноотрицательное срабатывание).

Итоговая точность: 81%.

Эта формула выбрана, потому что она обеспечивает комплексное понимание эффективности классификатора, учитывая, как его способность правильно идентифицировать целевые классы (жесты), так и избегать ошибок.

Относительно достигнутой точности в 81%: этот показатель можно считать достаточно хорошим для начальных этапов разработки системы, особенно учитывая сложность задачи распознавания жестов в реальном времени.

Заключение

В рамках данного исследования была создана и протестирована система управления умным домом с использованием языка жестов. Основной целью проекта было создание инклюзивного интерфейса, который позволил бы людям с нарушениями слуха эффективно управлять бытовыми и технологическими устройствами в доме без необходимости вербального общения.

В процессе работы было выбрано несколько жестов для управления различными устройствами, такими как лампы, розетки и шторы. Эти жесты содержат команды для включения и выключения света, открытия и закрытия штор, а также активации голосового помощника "Алиса". Проведенные тесты показали, что система имеет высокую точность распознавания жестов, составляющую 81%. Опрос среди глухих пользователей (20 человек) показал, что 85% участников сочли систему удобной и полезной, что подтверждает её эффективность и актуальность. Разработанная система демонстрирует значительный потенциал для улучшения качества жизни людей с нарушениями слуха, предлагая удобный и интуитивно понятный способ взаимодействия с умными устройствами в доме.

Список литературы

1. Всемирная организация здравоохранения. Глухота и потеря слуха. [Электронный ресурс]. https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss (Дата обращения: 29.07.2024).
2. Jiyeon Yu, Angelica de Antonio, Elena Villalba-Mora. Deep Learning (CNN, RNN) Applications for Smart Homes: A Systematic Review. // Computers, 2022, 11(2). [Электронный ресурс]. https://www.mdpi.com/2073-431X/11/2/26 (Дата обращения: 08.08.2024).
3. V. Chang, R.O. Eniola, L. Golightly, Q.A. Xu. A. An Exploration into Human–Computer Interaction: Hand Gesture Recognition Management in a Challenging Environment // SN Computer Science, Volume 4, article number 441, 2023.
4. A Dynamic Gesture Recognition Interface for Smart Home Control based on Croatian Sign Language. Luka Kraljević, Mladen Russo, Matija Pauković, Matko Šarić // Applied Sciences, 2020, V. 10, № 7, pp. 1-16.
5. G.R.S. Murthy, R.S. Jadon. A Review of Vision Based Hand Gestures Recognition // International Journal of Information Technology and Knowledge Management, Jul-Dec 2009, Vol. 2, No. 2, pp. 405-410.

УДК 004.056

РАЗРАБОТКА МЕТОДИКИ ТЕСТИРОВАНИЯ НА ПРОНИКНОВЕНИЕ СЕТЕВОЙ ИНФРАСТРУКТУРЫ ОРГАНИЗАЦИИ

М.В. Ванин
Студент группы М22-508 НИЯУ МИФИ, vaninmv.5582@gmail.com

Аннотация: Проанализированы пять известных методик проведения тестирования на проникновение для выявления уязвимостей сетевой инфраструктуры типовой организации. На основании проведённого анализа разработана собственная методика, включающая в себя четыре этапа ее выполнения. Для двух этапов методики подробнее конкретизированы подэтапы. Сделан вывод, что использование разработанной методики проведения тестирования на проникновение позволяет успешно выявлять уязвимости в сетевой инфраструктуре для их последующего устранения, пока ими не успели воспользоваться злоумышленники.

Ключевые слова: сетевая инфраструктура, информационная безопасность, методика, тестирование на проникновение.

Введение

В связи с постоянно возрастающими и усложняющимися угрозами со стороны киберпреступников [1] появилась необходимость проведения тестирования на проникновение (ТНП) – имитации кибератак, в ходе которых выявляются уязвимости защищаемой системы и проводится анализ ее защищенности. Благодаря этому можно заранее подготовиться к реальной атаке, устранить ведущие к успешности ее реализации уязвимости и тем самым не допустить ущерба.

Однако далеко не всегда существует понимание как должно проводиться ТНП. Известно множество различных методик проведения ТНП, которые отличаются своими характеристиками. Также на текущий момент существует большое количество программных инструментов, которые могут использоваться для проведения ТНП, но далеко не все из них хорошо подходят для автоматизированного применения.

В связи с этим была поставлена цель исследования – упорядочить деятельность пентестеров сетевых инфраструктур организаций на основе использования разработанной методики проведения ТНП.

В ходе работы проанализированы наиболее известные методики проведения ТНП: Open Web Application Security Project (OWASP) Web Application Testing Guide [2], Open Source Security Testing Methodology Manual (OSSTMM) [3], Information Systems Security Assessment Framework (ISSAF) [4], Information Security Testing and Assessment из стандарта NIST 800-115 [5] и Penetration Testing Execution Standard (PTES) [6]. Далее, основываясь на их позитивном опыте, разработана собственная методика проведения ТНП, устраняющая их недостатки и включающая в себя четыре этапа (рис. 1):

1) сбор информации;
2) определение угроз информационной безопасности (ИБ);
3) поиск уязвимостей;
4) попытка использования (эксплуатация) найденных уязвимостей.

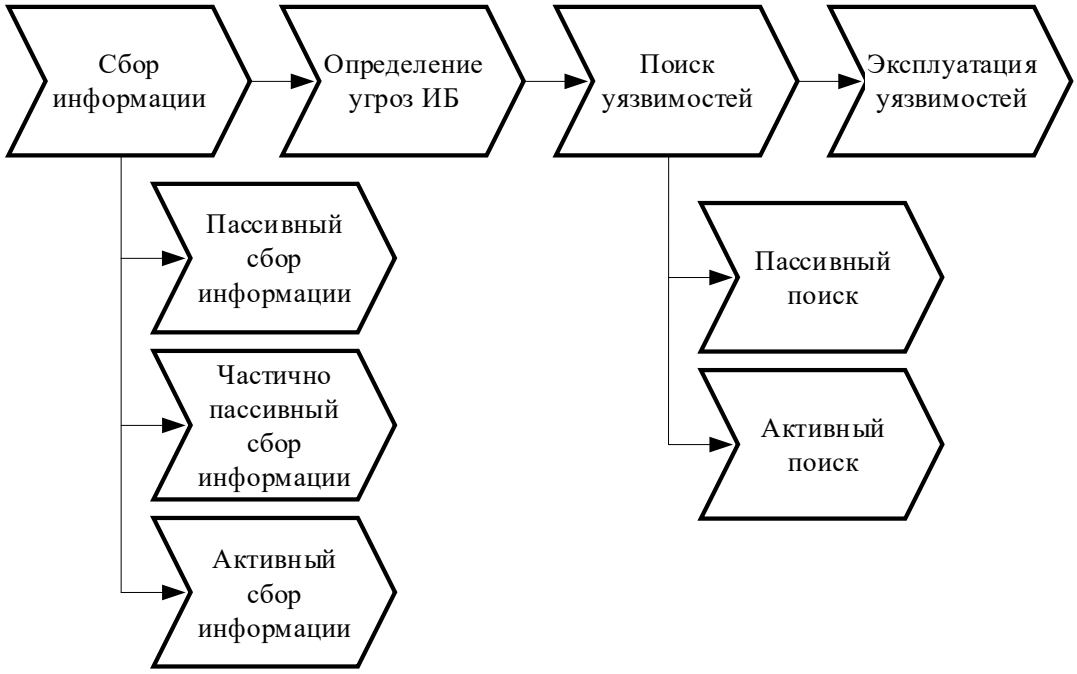

Рисунок 1 – Графическое представление этапов методики.

Сбор информации

Сбор информации представляет собой поиск информации в открытых источниках («разведку») о целевой организации с целью нахождения представленных в этих источниках сведений различного характера, которые могут быть так или иначе полезны при проведении ТНП на этапах выявления уязвимостей и их эксплуатации. Для данных целей рекомендуется использовать методы OSINT (Open Source INTelligence) [7]. В частности, на данном этапе могут быть обнаружены «уязвимые» к методам социальной инженерии сотрудники.

Этап сбора информации состоит из трёх уровней, различающихся по глубине поиска.

Первый уровень состоит из легко получаемой информации, которая может быть собрана с применением автоматизированных инструментов.

Второй уровень включает в себя информацию, которая может быть получена с помощью поверхностного анализа информации, полученной с применением автоматизированных инструментов. Получаемая таким образом информация может дать понимание бизнес-процессов исследуемой организации (Клиента), местоположения офиса(-ов), данные о сотрудниках компании и другие полезные в проведении ТНП сведения.

Третий уровень заключается в наиболее глубоком анализе информации, полученной на предыдущих двух уровнях. В частности, анализ может включать

в себя исследование страниц сотрудников в социальных сетях, глубокое исследование бизнес-процессов организации или обработка промежуточных результатов, получаемых в ходе проведения следующих этапов ТНП.

При проведении исследования необходимо в первую очередь определить принадлежащие организации домены и поддомены. Анализ содержимого общедоступных интернет-страниц, находящихся на данных доменах и поддоменах, может дать информацию об используемых в компании технологиях, контактных телефонах, адресах электронной почты и людях, личные страницы которых в социальных сетях могут содержать полезную информацию.

Сбор информации по открытым источникам (OSINT) может быть трёх типов:

1) *пассивный* применяется в случаях, когда требуется избегать любого воздействия на сетевую инфраструктуру организации с целью предотвращения обнаружения действий атакующих; данный тип является трудозатратным и «требовательным» к техническим средствам; для данного типа сбора информации обычно используются сервисы третьих лиц, которые предоставляют доступ к ранее собранной информации об организации;
2) *частично пассивный* заключается в сборе информации о цели с использованием методов, которые маскируются под обычную интернет-активность; это включает в себя получение публично доступной информации о серверах, без глубокого исследования серверов, такого как перебор директорий и файлов на сервере; данный тип сбора информации не включает в себя сканирование открытых портов, он ограничивается сбором метаданных из опубликованных в открытом доступе файлов и документов организации;
3) *активный* подразумевает использование инструментов, которые могут быть обнаружены специалистами Клиента; в частности, производится агрессивное исследование его сетевой инфраструктуры, перебор часто встречающихся имён файлов и директорий на серверах и использование сканеров уязвимостей.

Сбор цифровых следов является фазой сбора информации, которая требует взаимодействия с целевым устройством для получения информации о нём. Одна из основных целей сбора цифровых следов – определение списка устройств, которые попадают в область интереса атакующих. Существует большое количество техник, с помощью которых может быть собрана данная информация. Например, такими техниками являются различные DNS-запросы, использование «whois»-сервисов, исследование веб-приложений, сканирование портов и сервисов.

Определение угроз ИБ

Следующим этапом проведения ТНП является формирование модели угроз ИБ для тестируемой сетевой инфраструктуры. Минимально необходимой

информацией для формирования модели угроз ИБ является знание активов, представленных в сети, технических средств и навыков, которыми обладает атакующий, и бизнес-процессов, в которых задействуются данные активы.

С учётом того, что ТНП проводится по методу двойного слепого тестирования [8], моделирование угроз ИБ на высоком уровне не может опираться на внутреннюю документацию Клиента. Тем не менее, на основании полученных на этапе сбора информации данных, должен быть определён список доступных для тестирования активов, угроз, которые несёт возможная компрометация данных активов, а также минимальный уровень компетенций атакующего, который может воздействовать на данный актив.

В перечень активов должны быть включены не только технические активы, но и различные политики, планы, другая внутренняя документация компании, в том числе описание конфигурации различных технических активов, финансовые и другие документы, учётные данные пользователей и другая важная информация.

Пример табличного описания угрозы представлен в табл. 1.

Поиск уязвимостей

Процесс поиска уязвимостей подразумевает обнаружение свойств информационной системы, обусловливающих возможность реализации угроз безопасности обрабатываемой в ней информации. Во время поиска уязвимостей любого типа должны соблюдаться ограничения, наложенные договорённостями с Клиентом. Список исследуемых объектов может включать в себя сети, сегменты сетей, конечные точки, приложения и т.п.

Таблица 1 – Пример описания угрозы ИБ.

Угроза ИБ	Источник угрозы ИБ	Актив				Метод реализации угрозы ИБ на СОИА	Последствия реализации угрозы ИБ			
		Информационный актив (ИА)	Значимые свойства ИБ в порядке приоритета	Среда обработки ИА (СОИА)	Уязвимость СОИА		Для ИА	Степень возможности реализации угроз ИБ	Степень тяжести последствий нарушения ИБ	Значимость и уровень риска ИБ
Угроза несанкционированного доступа к идентификационной информации	Внешний нарушитель	Идентификационные данные (на АРМ администратора)	Конфиденциальность (К)	АРМ администратора	BDU:2023-00531	Манипулирование ресурсами	Нарушение К	Минимальная (2)	Среднее (2)	Допустимый (5)

Активный поиск заключается в прямом взаимодействии с компонентами тестируемой сетевой инфраструктуры на предмет выявления уязвимостей. Это может быть реализовано с использованием инструментов, работающих на четвертом уровне модели взаимодействия открытых систем OSI/ISO, либо же с

использованием инструментов, работающих с протоколами более высокого уровня. Инструменты для проведения активного поиска подразделяются на два типа по степени вовлечённости атакующего в их работу: автоматизированные и ручные.

Автоматизированные инструменты тестирования применяются для взаимодействия с целевой системой, обработки полученных ответов. На основании полученных данных они определяют наличие уязвимостей.

Пассивный поиск включает в себя анализ метаданных и мониторинг сетевого трафика. Анализ метаданных подразумевает исследование параметров файла, таких как автор, дата создания, дата модификации, метки геолокации, наименование компании и т.д. Метаданные могут также содержать информацию о внутренних IP-адресах, именах внутренних серверов, а также другие данные, которые могут быть полезными при проведении ТНП. Мониторинг трафика подразумевает наличие подключения к внутренней сети компании, благодаря чему может осуществляться зеркалирование пакетов и дальнейший их анализ.

При использовании инструментов поиска уязвимостей необходимо проводить сравнение результатов, полученных в ходе работы различных инструментов. Ввиду специфики их работы при одинаковых входных данных результаты могут кардинально отличаться.

В процессе поиска уязвимостей нужно выстраивать дерево шагов, предпринимаемых в ходе ТНП. По мере обнаружения новых систем, сервисов и потенциальных уязвимостей в дерево добавляются новые листья и ветви.

Точность анализа уязвимостей и их эксплуатации повышается при воспроизведении условий тестирования в лабораторных условиях. Нередко системы Клиента содержат специфичные настройки, поиск уязвимостей в которых может потребовать большого времени. Воссоздав специфику системы Клиента на полностью подконтрольных тестирующим системах, можно протестировать работу программных средств, использующих одну или несколько уязвимостей целевого объекта (так называемых эксплойтов), для достижения наилучшего результата в ходе проведения ТНП.

Для поиска уязвимостей могут использоваться и сторонние сервисы, такие как базы данных уязвимостей или базы знаний вендоров используемых в сетевой инфраструктуре Клиента аппаратных и программных продуктов.

Эксплуатация уязвимостей

Фаза эксплуатации уязвимостей фокусируется исключительно на получении доступа к системам и ресурсам Клиента. Для достижения данной цели используются эксплойты. Таким образом, на данном этапе применяется широкий

спектр различных программ и подпрограмм, конкретный набор которых зависит от специфики атакуемого узла или сети.

Заключение

Исследование было начато с анализа типовой сетевой инфраструктуры организации, предполагаемой к тестированию. Она является неоднородной и включает в себя как физические конечные устройства, например, компьютеры пользователей, так и виртуальные машины, используемые для запуска необходимого в работе программного обеспечения (ПО). Исходя из рассмотренной инфраструктуры были выбраны методики проведения ТНП, для которых проводился их сравнительный анализ. Далее с учётом достоинств и недостатков рассмотренных методик сформированы критерии выбора ПО, с помощью которого будет реализовываться методика проведения ТНП. После этого была разработана собственная методика проведения ТПН сетевой инфраструктуры организации. Для разработанной методики проведения ТНП был создан комплекс ПО, позволяющий её автоматизировать. Он составлялся на основании ранее сформулированных критериев отбора ПО. Для этого комплекса описан сценарий его применения. После этого проведена апробация разработанной методики на тестовом стенде, показавшая ее актуальность и применимость в реальных условиях.

Список литературы

1. Отчёт РТК-Солар «Атаки на российские компании в III квартале 2023 года». [Электронный ресурс]. https://rt-solar.ru/analytics/reports/3889. (Дата обращения: 30.05.2024).
2. OWASP Web Application Testing Guide. [Электронный ресурс]. https://owasp.org/www-project-web-security-testing-guide/stable/. (Дата обращения: 08.06.2024).
3. OSSTMM. [Электронный ресурс]. https://www.isecom.org/OSSTMM.3.pdf. (Дата обращения 30.06.2024).
4. ISSAF. [Электронный ресурс] https://www.untrustednetwork.net/files/issaf0.2.1.pdf. (Дата обращения: 30.06.2024).
5. NIST 800-115. [Электронный ресурс]. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/ nistspecialpublication800-115.pdf. (Дата обращения: 30.06.2024).
6. PTES. [Электронный ресурс]. https://buildmedia.readthedocs.org/media/pdf/pentest-standard/latest/pentest-standard.pdf. (Дата обращения: 30.06.2024).
7. Stodelov D., Miloslavskaya N. Open Source INTelligence Tools. In: Félix Francisco Ramos Corchado and Alexei V. Samsonovich (Eds.). 2022 Annual International Conference on Brain-Inspired Cognitive Architectures for Artificial Intelligence: The 13th Annual Meeting of the BICA Society. Procedia Computer Science, Vol. 213, pp. 83-88.
8. ГОСТ Р 56045-2021/ISO/IEC TS 27008:2019 Информационные технологии (ИТ). Методы и средства обеспечения безопасности. Рекомендации по оценке мер обеспечения информационной безопасности. – М.: Стандартинформ, 2021. 90 с.

УДК 004.056

МЕТОДИКА ПРЕДОТВРАЩЕНИЯ УТЕЧКИ КОНФИДЕНЦИАЛЬНЫХ ДАННЫХ КЛИЕНТОВ И СОТРУДНИКОВ ТИПОВОГО БАНКА

Д.А. Прокопчук[1], Д.Н. Стоделов[2]
[1]Студент группы М22-508 НИЯУ МИФИ, dmitriyprokopchuk@yandex.ru
[2]Аспирант группы А22-544 НИЯУ МИФИ, dstodelov@yandex.ru

Аннотация: Исследуются методы предотвращения утечки конфиденциальных данных клиентов и сотрудников типового банка. Рассматриваются процессы обработки и защиты персональных данных в банковских организациях, проводится всесторонний анализ существующих методов поиска и маркировки данных из открытых источников (OSINT). Выявляются недостатки традиционных и современных методов маркировки данных, а также необходимость разработки новых подходов. Разработана методика маркировки данных, включающая алгоритмы и критерии оценки ее результативности, а также спроектирован и апробирован программный модуль для маркировки и отслеживания перемещения данных. Проведено тестирование разработанных решений на тестовом стенде и сформулированы рекомендации по их внедрению в бизнес-процессы банка. Работа направлена на повышение уровня информационной безопасности банковских учреждений, минимизацию рисков утечек и оптимизацию внутренних процессов защиты данных.

Ключевые слова: информационная безопасность, персональные данные, банки, маркировка данных, OSINT, утечка данных, программное обеспечение, меры противодействия.

Введение

В эпоху цифровой трансформации, когда банки активно используют информационные технологии для обработки и хранения персональных данных (ПДн) клиентов, обеспечение информационной безопасности (ИБ) становится критически важной задачей. Современные банки сталкиваются с увеличивающимися рисками, связанными с несанкционированным доступом (НСД) и утечками данных. В связи с этим защита ПДн клиентов и сотрудников требует комплексного подхода, включающего в себя использование современных методов маркировки данных для их последующего отслеживания и анализа.

Обзор нормативной и правовой базы

Основным законодательным актом Российской Федерации, регулирующим отношения, связанные с обработкой ПДн, является Федеральный закон № 152-ФЗ от 27.07.2006 «О персональных данных». Согласно этому закону, ПДн определяются как любая информация, относящаяся прямо или косвенно к идентифицированному или определяемому физическому лицу. Это могут быть имя, фамилия, дата рождения, место жительства, паспортные данные, отпечатки пальцев и другие сведения.

Помимо Федерального закона № 152-ФЗ, защита ПДн регулируется рядом

других нормативных правовых актов. Конституция Российской Федерации гарантирует защиту частной жизни, включающую в себя ПДн. Трудовой кодекс содержит главу 14 «Защита персональных данных», которая описывает общие положения о защите ПДн работников.

Федеральный закон № 149 «Об информации, информационных технологиях и о защите информации» регулирует отношения, связанные с обработкой информации и защитой информации. Постановления Правительства РФ № 1119 и № 687, а также Постановление ФСТЭК № 21 устанавливают требования к защите ПДн при их обработке в информационных системах.

Процессы обработки и защиты персональных данных

Процессы обработки и защиты ПДн в банках включают сбор, запись, систематизацию, хранение, уточнение и другие операции. Каждая банковская организация, заключающая трудовые договоры или предоставляющая услуги, автоматически становится контролером данных и обязана обеспечивать их защиту.

Для организации защиты ПДн банки применяют различные технические меры, включая использование антивирусных решений, межсетевых экранов, систем обнаружения вторжений и других средств обеспечения ИБ [1-3]. Важную роль играют также организационные меры, такие как разработка и внедрение внутренних политик и процедур, направленных на защиту ПДн.

Анализ существующих методов поиска и маркировки данных

Существующие методы поиска и маркировки данных в основном базируются на использовании открытых источников информации (Open Source INTelligence, OSINT). Эти методы включают активное сканирование, сбор информации о сетевой инфраструктуре [4], использовании социальных сетей и других общедоступных ресурсов. Однако, несмотря на эффективность традиционных методов маркировки, таких как водяные знаки и электронные подписи, они имеют свои ограничения и недостатки.

Одним из важных аспектов маркировки данных является их защита от НСД и последующего распространения. Современные методы маркировки данных включают использование цифровых водяных знаков, стеганографии и токенизации (замены конфиденциальных данных на неконфиденциальные заместители, называемые токенами). Каждый из этих методов имеет свои преимущества и ограничения, и их выбор зависит от специфики банка, типа защищаемых данных и требований законодательства.

Анализ существующих методов маркировки данных показал, что они не всегда обеспечивают необходимый уровень защиты информации. Основными

проблемами являются ограниченная способность методов маркировки к адаптации под новые угрозы и уязвимости, а также недостаточная результативность в условиях сложных и многоуровневых атак [5].

В связи с этим возникает необходимость разработки новых методов маркировки данных. Проведенный анализ показал, что основные направления совершенствования методов маркировки данных включают разработку алгоритмов скрытой маркировки, интеграцию методов машинного обучения для выявления аномалий и угроз, а также создание комплексных решений, объединяющих несколько методов защиты.

Разработка методики маркировки данных

Разработка методики маркировки данных клиентов и сотрудников банка является ключевым этапом в обеспечении их защиты. Для этого необходимо определить требования к методике, разработать алгоритмы маркировки и критерии оценки ее результативности. Они сформулированы следующим образом.

1. Конфиденциальность: методика должна обеспечивать защиту данных от НСД и утечки.
2. Целостность: методика должна гарантировать неизменность данных в процессе их обработки и хранения.
3. Аутентичность: методика должна позволять идентифицировать источник данных и подтверждать их подлинность.
4. Отслеживаемость: методика должна обеспечивать возможность отслеживания перемещения данных внутри и вне организации.
5. Гибкость: методика должна быть адаптирована к различным типам данных и специфике банка.
6. Соответствие законодательству: методика должна соответствовать требованиям законодательства в области защиты ПДн [6].

Разработка алгоритмов маркировки данных включает создание скрытых маркеров (например, QR-кодов), которые позволяют отслеживать перемещение данных. Алгоритмы маркировки должны обеспечивать следующее:

- внедрение маркеров: скрытые маркеры должны быть встроены в данные таким образом, чтобы они были незаметны для пользователя, но легко идентифицируемы с помощью специальных инструментов;
- шифрование данных: маркеры должны содержать зашифрованную информацию о документе и его пользователе, что обеспечивает дополнительный уровень защиты;
- отслеживание данных: алгоритмы должны позволять отслеживать перемещение данных внутри и вне организации, выявлять случаи НСД и утечки информации.

Для оценки результативности разработанной методики маркировки данных предлагается использовать следующие критерии:

- точность маркировки: степень точности, с которой маркеры внедряются в данные и идентифицируются при их извлечении;
- надежность маркировки: способность маркеров сохранять свои свойства и идентифицируемость в различных условиях эксплуатации данных;
- эффективность отслеживания: способность алгоритмов отслеживания данных выявлять случаи НСД и утечки информации;
- удобство использования: удобство работы с методикой для сотрудников банка, включая простоту внедрения маркеров и анализа данных;
- соответствие законодательству: соответствие методики требованиям законодательства в области защиты ПДн [7].

Разработан алгоритм маркировки данных (рис. 1), основанный на использовании водяных знаков в документах Word, с применением QR-кода для хранения информации о пользователе и времени работы с документом.

1. Выбор алгоритма хеширования для создания хеш-суммы из метаданных пользователя и времени работы с документом.
2. Формирование метаданных о пользователе, работающем с документом (например, идентификатор пользователя, IP-адрес, имя компьютера и т.д.).
3. Вычисление хеш-суммы на основе выбранного алгоритма хеширования и собранных метаданных и текущего времени.
4. Генерация QR-кода, содержащего полученную хеш-сумму. Это будет использоваться в качестве водного знака в документе.
5. Внедрение водного знака в колонтитул документа Word. Водной знак должен быть неприметным для пользователя, чтобы он не мог его обнаружить и удалить.
6. Периодическое обновление водного знака (например, каждый час или каждые несколько часов), вычисляя новую хеш-сумму на основе текущего времени и метаданных пользователя. Это обеспечит актуальность информации в QR-коде.
7. Извлечение информации из водного знака при возникновении подозрений на и его декодирование для получения хеш-суммы. Затем можно восстановить метаданные пользователя и время работы с документом, сравнив хеш-суммы с доступными данными.
8. Идентификация виновника утечки на основе извлеченных метаданных и времени работы с документом.

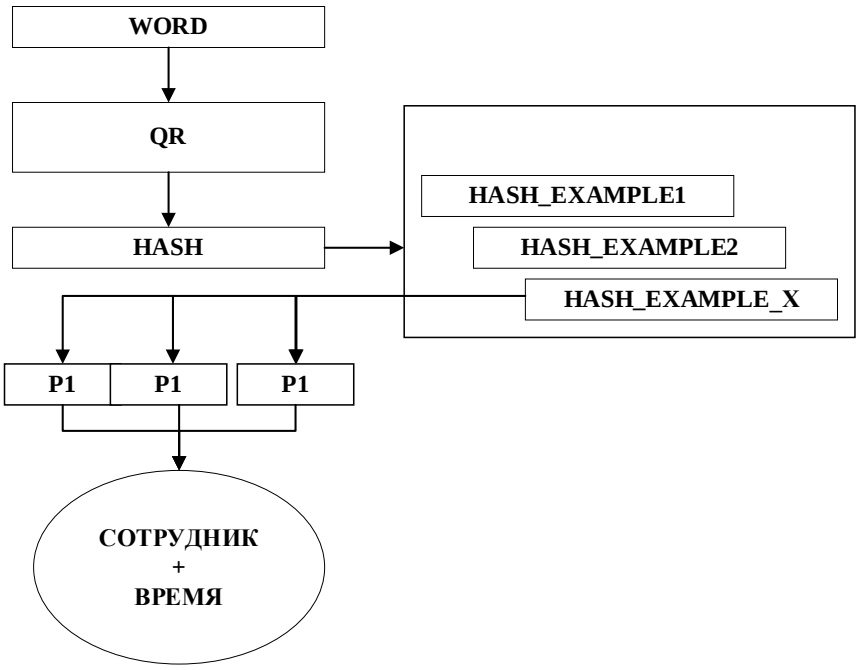

Рисунок 1 – Разработанный метод маркировки и определения источника утечки.

Проектирование и разработка программных модулей, реализующих методику маркировки данных

Проектирование и разработка программных модулей для автоматизации процесса маркировки данных являются важной частью исследования. Программный модуль должен обеспечивать автоматизацию внедрения маркеров, их отслеживание и анализ данных. При проектировании и реализации определялись системные требования целевой системы, разрабатывался алгоритм создания и внедрения маркеров, производилась реализация модулей программного продукта.

Алгоритмы создания и внедрения маркеров включают в себя следующие этапы:

1) генерация маркеров: создание уникальных скрытых маркеров для каждого документа или группы данных;
2) внедрение маркеров: встраивание маркеров в данные с учетом требований конфиденциальности и целостности;
3) шифрование информации: шифрование информации, содержащейся в маркерах, для обеспечения дополнительной защиты данных [8].

Модуль отслеживания перемещения данных должен обеспечивать следующее:

1) мониторинг данных: постоянный мониторинг перемещения данных внутри и вне организации;
2) выявление угроз: выявление случаев НСД и утечки информации на основе анализа данных из маркеров;

3) интеграция с системой безопасности: интеграция модуля с общей системой безопасности банка для обеспечения комплексного подхода к защите данных.

Также разработан интуитивно понятный и функциональный пользовательский интерфейс для управления процессами маркировки и отслеживания данных. Интерфейс обеспечивает удобный доступ к основным функциям модуля и интегрируется с другими системами безопасности банка.

Апробация программных решений

Апробация разработанных программных решений проводилась на тестовом стенде. Этот этап включал в себя планирование и проведение тестирования, анализ результатов и оптимизацию системы.

Тестирование программных решений проводилось следующим образом:

1) разработка тестовых сценариев: создание сценариев, имитирующих различные условия эксплуатации данных и возможные угрозы;
2) проведение самих тестов: проведение тестов на тестовом стенде с использованием разработанных сценариев;
3) анализ результатов: анализ результатов тестирования для выявления слабых мест и недостатков программных решений.

На основе анализа результатов тестирования проводилась оптимизация системы, что включало в себя следующие действия:

1) улучшение алгоритмов: оптимизация алгоритмов маркировки и отслеживания данных для повышения их результативности;
2) исправление ошибок: устранение выявленных в ходе тестирования ошибок и недостатков;
3) оптимизация производительности: улучшение производительности программного модуля для обеспечения его результативной работы в реальных условиях.

Тестирование программных решений различными пользователями подтвердило удобство и функциональность интерфейса.

Заключение

Проведенный анализ существующих методов маркировки данных, таких как водяные знаки, электронные подписи, стеганография и токенизация, выявил их основные проблемы и ограничения. Традиционные методы часто оказываются недостаточно адаптируемыми к новым угрозам, обладают ограниченной функциональностью и сложны во внедрении в сложные информационные системы банка. В ответ на эти вызовы в работе предложены новые алгоритмы скры-

той маркировки данных, включающие использование QR-кодов и других идентификаторов. Эти маркеры могут быть незаметно внедрены в документы и файлы, что позволяет отслеживать их перемещение и защищать информацию от несанкционированного доступа.

Особое внимание уделено внедрению шифрования в маркеры, что добавляет дополнительный уровень защиты данных. Это позволяет сохранить конфиденциальность информации даже при ее утечке [9]. Для автоматизации процессов маркировки и отслеживания данных спроектированы и разработаны программные решения, которые интегрируются в общую систему безопасности банка и обеспечивают результативное отслеживание перемещения данных внутри и вне организации. Важными функциями программного модуля являются генерация и внедрение маркеров, шифрование данных, а также мониторинг и анализ перемещений данных.

Апробация и тестирование программных решений проводились на тестовом стенде, где моделировались различные условия эксплуатации данных и возможные угрозы. Тестирование подтвердило функциональность и эффективность разработанных решений, а выявленные проблемы и недостатки были устранены, что позволило оптимизировать алгоритмы и повысить общую производительность системы.

Рекомендуется интеграция разработанной методики маркировки данных в бизнес-процессы банка для повышения уровня ИБ и минимизации рисков утечек данных. Важно также организовать обучение сотрудников банка для освоения новых инструментов и методик, что позволит эффективно использовать разработанные решения на практике [10]. Регулярный мониторинг и обновление системы маркировки данных необходимы для адаптации к новым угрозам и обеспечения долгосрочной безопасности информации.

На основе проведенного исследования сформулированы рекомендации по внедрению методики маркировки данных и программных решений в бизнес-процессы банка.

На текущем этапе осуществляется регулярный мониторинг работы системы маркировки данных и ее обновление для адаптации к новым угрозам и требованиям.

Список литературы

1. ГОСТ Р ИСО/МЭК 27000-2012 Информационная технология (ИТ). Методы и средства обеспечения безопасности. Системы менеджмента информационной безопасности. Общий обзор и терминология. – Введ. 2013-12-01. – М.: Стандартинформ, 2014. – 22 с.
2. ISO/IEC 27001:2013 Information technology – Security techniques – Information security management systems – Requirements. – 23 p.

3. ГОСТ 15971-90 Системы обработки информации. Термины и определения. – Введ.1992-01-01. – М.: Гос. комитет СССР по управлению качеством, 1991. – 14 с.
4. ГОСТ Р ИСО/МЭК 27033-1-2011 Информационная технология (ИТ). Методы и средства обеспечения безопасности. Безопасность сетей. Часть 1. Обзор и концепции. – Введ.2012-01-01. – М.: Стандартинформ, 2012. – 73 с.
5. ГОСТ 15971-90 Системы обработки информации. Термины и определения. – Введ.1992-01-01. – М.: Гос. комитет СССР по управлению качеством, 1991. – 14 с.
6. ГОСТ Р 50922-2006 Защита информации. Основные термины и определения. – Введ. 2008-02-01. – М.: Стандартинформ, 2008. – 12 с.
7. СТО БР ИББС-1.0-2014 Обеспечение информационной безопасности организаций банковской системы Российской Федерации. Общие положения. – Введ. 2014-06-01. – Москва, 2014. – 101 с.
8. ISO/IEC 27032:2023 Information technology – Security techniques – Guidelines for cybersecurity. – 50 p.
9. ГОСТ Р 53114-2008 Защита информации. Обеспечение информационной безопасности в организации. Основные термины и определения. – Введ. 2009-10-01. – М.: Стандартинформ, 2009. – 20 с.
10. International Data Corporation. [Электронный ресурс]. https://cronos.asia/it-tehnologii/skolko-vesit-vsya-informaciya-v-internete (Дата обращения: 11.04.2024).

УДК 004.056

ОБЕСПЕЧЕНИЕ НЕПРЕРЫВНОСТИ ФУНКЦИОНИРОВАНИЯ DLP-СИСТЕМЫ В СЛУЧАЕ АВАРИЙНОЙ СИТУАЦИИ В ТЕРРИТОРИАЛЬНО РАСПРЕДЕЛЁННЫХ ЦОД

В.Ю. Семилеткин
Студент группы М22-508 НИЯУ МИФИ, svu_2017@mail.ru

Аннотация. Представлены результаты исследования методов обеспечения непрерывности функционирования DLP-системы в территориально распределённых (ТР) центрах обработки данных (ЦОД). Показано, что существующая зависимость организаций всех форм собственности от информационных технологий (ИТ) повышает их уязвимость к различным угрозам, включая угрозы экстремизма и терроризма. Обоснован выбор отечественных средств виртуализации. Рассчитаны основные показатели надежности для системы, включающей ПК СВ «Брест» и DLP-систему «СёрчИнформ КИБ». На примере «СёрчИнформ КИБ» рассмотрены особенности функционирования DLP-системы при ее развертывании на облачной платформе. Приведены результаты оценки эффективности разработанных политик и поисковых критериев для DLP-системы, ориентированных на выявление случаев распространения материалов, содержащих призывы к экстремизму и терроризму. Сделан вывод, что надежное и непрерывное функционирование DLP-системы существенно снижает вероятность утечек информации и других инцидентов информационной безопасности.

Ключевые слова: информационные системы, информационная безопасность, киберпространство, терроризм, территориально-распределенные ЦОД, экстремизм, DLP-системы.

Введение

Вопросы обеспечения безопасности информационных систем (ИС) играют все более возрастающую роль в современной экономике и управлении. Стабильность работы этих систем имеет решающее значение для устойчивого роста организаций всех форм собственности. Усиливающаяся зависимость и бизнеса, и государственных учреждений от ИТ повышает их уязвимость к различным угрозам, включая угрозы экстремизма и терроризма [1-3]. Данные обстоятельства диктуют необходимость разработки более совершенных методов защиты данных и обеспечения непрерывного функционирования ИС отечественных компаний.

Важнейшей задачей, решение которой необходимо для обеспечения ИБ, является предотвращение утечек конфиденциальной информации. Одним из наиболее эффективных технических средств, позволяющих бороться с утечками, служат DLP-системы [4, 5]. Они помогают контролировать и фильтровать данные, проходящие через сети организации, а также обнаруживают и блокируют несанкционированные попытки передачи важной

и критической информации. Эффективное применение DLP-систем способствует не только предотвращению атак, но и снижению потенциального ущерба от различных инцидентов, уменьшая риски для руководства и сотрудников. В этой связи особое значение приобретает обеспечение защиты и непрерывности работы подобных решений в особенности на фоне угрозы терактов, которые могут привести к серьезным нарушениям в работе ключевых ИС.

Для усиления защиты и повышения устойчивости к внешним угрозам наиболее адекватными мерами на сегодняшний день являются использование возможностей ТР ЦОД и внедрение средств виртуализации. ЦОД – отказоустойчивые решения, которые призваны обеспечить непрерывную работу корпоративных приложений, сервисов и сайтов практически в любых условиях. В свою очередь виртуализация играет важную роль в повышении гибкости и эффективности ИТ-инфраструктуры современных предприятий. Она позволяет значительно увеличить загрузку аппаратных мощностей, улучшает управление ИТ-затратами и упрощает контроль над информацией и приложениями.

В статье обосновывается использование средств виртуализации для обеспечения бесперебойного функционирования DLP-системы (на примере программного комплекса «СёрчИнформ КИБ» [6]) в случае аварийной ситуации в ТР ЦОД. Также приводятся результаты разработки политик и поисковых критериев для DLP-системы, ориентированных на выявление случаев распространения материалов, содержащих призывы к экстремизму и терроризму.

Экстремизм и терроризм как угроза национальной безопасности

Преступления экстремистской и террористической направленности, к сожалению, становятся привычным явлением в жизни общества, состоящего из множества социальных групп, разделяемых между собой как национальной либо расовой принадлежностью, так и религиозными или идеологическими предпочтениями. Противодействие экстремистской и террористической деятельности, связанной с причинением существенного вреда общественным отношениям, обеспечивающим основы конституционного строя, является важнейшим направлением в современной государственной политике противодействия преступности.

В связи с проникновением экстремизма и терроризма в киберпространство их угроза с каждым годом только возрастает. Киберпространство стало интегральной частью современного общества, обеспечивая глобальное взаимодействие, доступ к информации и возможность обмена идеями. Однако, по-

мимо своих позитивных аспектов, оно также предоставляет новые возможности для экстремистских и террористических группировок, которые используют цифровые инструменты в своих интересах [3, 7]. Перед специалистами в области информационной безопасности (ИБ) встает очевидная задача контролировать распространение угроз экстремизма и терроризма в сети и, в случае необходимости, блокировать передачу соответствующей информации при помощи специально разработанных средств (в том числе таких, как DLP-системы).

Использование отечественных средств виртуализации для обеспечения непрерывности функционирования DLP-системы

Ранее отмечалось, что DLP-системы играют ключевую роль в защите конфиденциальной информации. В условиях современной цифровой экономики, когда данные могут располагаться в нескольких ЦОД, важно обеспечить их бесперебойное функционирование, в том числе в случаях аварийных ситуаций и террористических атак. Виртуализация представляет собой эффективное решение для достижения этой цели, предлагая гибкость управления ресурсами и повышенную устойчивость систем. Использование технологий виртуализации для DLP-систем в ЦОД позволяет гарантировать непрерывность сервиса за счет быстрого переноса и восстановления данных на другом сервере внутри ЦОД либо на другом ЦОД, упростить управление и масштабирование за счет реализации централизованного контроля и распределения вычислительных мощностей среди множества виртуальных машин; обеспечить изоляцию и безопасность процессов, что в свою очередь позволяет предотвратить распространение угроз между различными рабочими нагрузками, способствуя устойчивости DLP-системы к внешним и внутренним угрозам; оптимизировать расходы на необходимое оборудование.

Проведенный анализ позволил установить, что оптимальным набором характеристик для виртуализации DLP-систем в ЦОД обладает Программный комплекс «Средства виртуализации «Брест» (ПК СВ «Брест») компании «РусБИТех-Астра» [8]. ПК СВ «Брест» соответствует существующим законодательным требованиям в области ИБ и обеспечивает более высокий уровень защиты от внутренних и внешних рисков по сравнению с иностранными аналогами. Кроме того, применение российских технологий облегчает процесс технической поддержки и обновления программного обеспечения.

В ходе выполненных работ проведен расчет показателей надежности для основных компонент ЦОД, обеспечивающих функционирование DLP-системы: аппаратной части (на примере Huawei FusionServer 2288H V5),

ПК СВ «Брест» и DLP-системы «СёрчИнформ КИБ». Итоговые данные расчета по каждой оцениваемой функциональной подсистеме представлены в табл. 1.

При выходе из строя по причине аварии, терактов или иных экстремистских проявлений одного или нескольких серверов в одном ЦОД с развернутой при помощи ПК СВ «Брест» DLP-системой «СёрчИнформ КИБ» предполагается автоматическая миграция виртуальных машин на резервные или неповрежденные вычислительные мощности внутри того же или другого ЦОД (при наличии связи между ЦОД).

Таблица 1 – Показатели надежности для основных компонент ЦОД, обеспечивающих функционирование DLP-системы.

Параметр	Значение		
	Аппаратная часть	ПК СВ «Брест»	СёрчИнформ КИБ
Интенсивность отказов, 1/ч	0.0001	0.0005	0.0005
Средняя наработка до отказа, ч	10000	2500	2500
Максимальное время восстановления после сбоев, ч	8	8	8
Коэффициент готовности	0.999	0.99	0.99

Разработана методологии восстановления работоспособности DLP-системы, которая включает в себя как немедленные реакции на аварийные ситуации, так и долгосрочные стратегии по восстановлению и оптимизации системы. Методология предусматривает использование модульных подходов и резервирования критически важных компонентов системы. Также предложена процедура регулярных аудитов и тренировок персонала для повышения оперативной готовности к аварийным ситуациям. Проведённые практические испытания методологии на модельных и реальных сценариях аварий в ЦОД позволили оценить её эффективность и выявить потенциальные уязвимости. Результаты испытаний подтвердили значительное сокращение времени восстановления системы и уменьшение потерь данных.

Использование DLP-системы «КИБ СёрчИнформ» в ТР ЦОД

Развертывание DLP-системы «СёрчИнформ КИБ» на облачной платформе дает дополнительные преимущества: бесшовную интеграцию в готовую облачную инфраструктуру с усиленной защитой [9]. Общая схема работы облачной DLP-системы представлена на рис. 1. При этом она обладает полным набором функциональных возможностей: контролирует все каналы передачи информации, качественно анализирует трафик и предоставляет продвинутые

инструменты для расследования инцидентов.

«СёрчИнформ КИБ» позволяет отслеживать: активность в почте (на корпоративном или публичном домене, с доступом через браузер или почтовый клиент); активность в социальных сетях, мессенджерах, телефонии (портативные, десктопные и веб-версии); действия в облачных хранилищах (загрузку, скачивание, удаление и изменение данных); действия с файлами (создание, изменение, удаление файлов на рабочей станции пользователя); действия со съемными устройствами (в том числе подключения к удаленным рабочим столам и виртуальным машинам, использование виртуальных дисков); активность в браузерах (поисковые запросы, посещение сайтов, скачивание и загрузку файлов); активность за ПК (продуктивность и эффективность работы, использование разрешенных/нелегитимных программ и приложений); происходящее на мониторах пользователей (скриншоты и онлайн-просмотр мониторов пользователей, даже если они подключаются к виртуальным средам); другие виды активности пользователей (распечатку документов, аудиопереговоры, действия с клавиатурой, установку/удаление программного обеспечения и т.д.) [10].

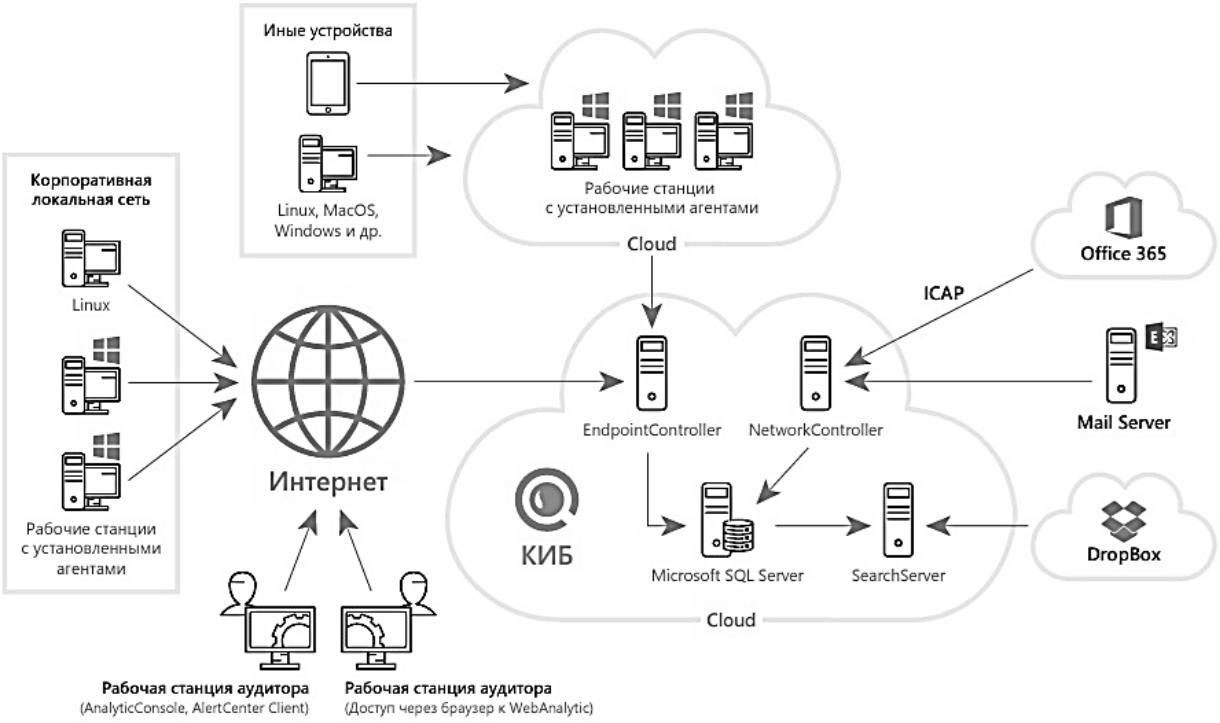

Рисунок 1 – Общая схема работы облачной DLP-системы
на примере «СёрчИнформ КИБ».

В рамках работ по обеспечению непрерывности функционирования DLP-системы «СёрчИнформ КИБ» разработан комплекс критериев и условий поиска, входящих в состав политик безопасности, ориентированных

на предотвращение угрозы распространения экстремизма и терроризма. Для проверки эффективности данных политик создан виртуальный стенд, на котором осуществлялось тестирование и корректировка всех разработанных элементов, и показано, что наилучшего результата позволяет добиться комплексирование поисковых запросов, реализующих фразовый поиск, поиск по словарям и поиск по атрибутам перехваченных сообщений.

Также разработан комплекс технических правил блокировки нежелательных действий пользователей корпоративной сети, связанных с распространением экстремизма и терроризма: контентная блокировка записи на сменные носители информации файлов, имеющих в своем содержании совпадения с данными из списка экстремистских материалов (в том числе с возможностью теневого копирования); блокировка передачи соответствующих материалов при помощи популярных мессенджеров; блокировка посещения сайтов экстремистского характера и отправки на них и др.

Тщательное изучение и анализ механизмов блокировок позволили убедиться в корректности политик и настроек, а также продемонстрировать их эффективность при использовании в корпоративной среде для обеспечения безопасности и грамотного управления ИБ.

Заключение

В работе рассмотрены вопросы обеспечения непрерывности функционирования DLP-системы в случае аварийной ситуации в ТР ЦОД. Установлено, что основные угрозы связаны с рядом технических проблем (например, поломкой оборудования и перебоями в электроснабжении), а также с внешними воздействиями, такими как стихийные бедствия, теракты и целенаправленные кибератаки. Показано, что использование средств виртуализации для поддержания работоспособности DLP-систем в случае аварийных ситуаций в ТР ЦОД не только способствует повышению устойчивости и безопасности ИС, но и обеспечивает их высокую доступность и гибкость управления, что крайне важно для современных организаций в условиях высоких требований к непрерывности и защите данных. На основе результатов анализа разработана методология восстановления работоспособности системы, подтвердившая свою эффективность в ходе практических испытаний. На конкретных примерах обоснована важность политик, используемых для мониторинга распространения материалов, содержащих призывы к экстремизму и терроризму, а также и настроенных правил блокировки, которые играют ключевую роль в обеспечении ИБ и предотвращении возможных инцидентов. Эти политики и правила не только минимизируют риски, но и позволяют специалистам эффективно анализировать полученные

данные, особенно в контексте идентификации и реагирования на террористические и экстремистские угрозы.

Список литературы

1. Осипов А. Автоматизация и ИИ – обязательные элементы современной системы ИБ // Информационная безопасность, 2024. № 2. [Электронный ресурс]. https://www.itsec.ru/ articles/avtomatizaciya-i-ii-obyazatelnye-elementy-sovremennoj-sistemy-ib (Дата обращения: 25.07.24).
2. Курило А.П., Милославская Н.Г., Сенаторов М.Ю., Толстой А.И. Основы управления информационной безопасностью. Серия «Вопросы управление информационной безопасностью. Выпуск 1»: учебное пособие. – М.: Горячая линия-Телеком, 2012. – 244 с.
3. Красинский В.В., Машко В.В. Кибертерроризм: криминологическая характеристика и квалификация // Государство и право, 2023, № 1, с. 79-91.
4. Ли Д. Возможности современных DLP-систем: как защитить внутренние данные компании от утечек [Электронный ресурс]. https://www.anti-malware.ru/analytics/Technology_Analysis/Modern-DLP-systems-capabilities (Дата обращения: 25.07.24).
5. Morozov V., Miloslavskaya N. DLP Systems as a Modern Information Security Control. In: Samsonovich A., Klimov V. (eds) Biologically Inspired Cognitive Architectures (BICA) for Young Scientists. BICA 2017. Advances in Intelligent Systems and Computing, 2018, Vol. 636, Q4 pp. 296-301.
6. «СёрчИнформ КИБ» [Электронный ресурс]. https://searchinform.ru/products/kib/ (Дата обращения: 25.07.24).
7. Информационная безопасность в системе национальной безопасности [Электронный ресурс]. https://searchinform.ru/informatsionnaya-bezopasnost/osnovy-ib/informatsionnaya-bezopasnost-v-otraslyakh/bezopasnost-informatsionnykh-sistem/informatsionnaya-bezopasnost-v-sisteme-natsionalnoj-bezopasnosti/ (Дата обращения: 25.07.24).
8. ПК СВ «Брест» [Электронный ресурс]. https://astragroup.ru/software-services/application-software-astra-group/brest/ (Дата обращения: 25.07.24).
9. DLP в облаке [Электронный ресурс]. https://searchinform.ru/services/cloud-dlp/ (Дата обращения: 25.07.24).
10. Дрозд А.В., Морозов В.Е, Милославская Н.Г. Основы аналитики в DLP-системах. Программный комплекс «КИБ СёрчИнформ»: учеб.-метод. пособие. – М.: Горячая линия – Телеком, 2023. – 368 с.

УДК 004.056

МЕТОДИКА ТЕСТИРОВАНИЯ НА ПРОНИКНОВЕНИЕ КОНТЕЙНЕРНОЙ ТЕХНОЛОГИИ KUBERNETES

Д.И. Денисенко
Студент группы М22-508 НИЯУ МИФИ, ddlreserch@gmail.com

Аннотация. При разработке клиент-серверных приложений повсеместно используют контейнерные технологии, призванные увеличить скорость разработки и минимизировать количество отказов программного обеспечения. Однако внедрение этих технологий создает уникальные проблемы в области безопасности. Учитывая их сильное влияние на приложения, крайне важно применять структурированную методику тестирования на проникновение, которая учитывает их отличительные особенности. Описанная методика тестирования на проникновение в значительной степени ориентирована на Docker и Kubernetes и предполагает глубокое понимание принципов их работы. В центре внимания методики оказался этап развертывания жизненного цикла разработки программного обеспечения, на котором происходит контроль и управление процессами развертывания контейнерных приложений. Несмотря на то, что методика разработана с учетом особенностей контейнерных сред, она не исключает интеграции с другими методиками, такими как методики тестирования сети или приложений. Методика может быть дополнена другими методиками для обеспечения комплексной оценки безопасности, так как она сосредоточена на среде контейнеризации и не учитывает специфику приложений, работающих в контейнерах. Тем не менее, она является самостоятельной и предоставляет методы тестирования в отсутствие дополнительных методик тестирования.

Ключевые слова: тестирование на проникновение, тестовый стенд, Kubernetes, Docker, кластер, атаки, безопасность, клиент-серверные приложения.

Введение

В области разработки клиент-серверных приложений обеспечение безопасности на протяжении всего жизненного цикла имеет большое значение. Поскольку клиент-серверные приложения становятся все более сложными возникает необходимость внедрения мер безопасности от начала разработки до этапа предоставления их конечному пользователю [1]. Согласно проекту документа от 19.12.2023 ГОСТ Р 56939 «Защита информации. Разработка безопасного программного обеспечения. Общие требования» [2] интеграция процессов обеспечения безопасности в процесс разработки имеет решающее значение для выполнения регулярных проверок кода и устранения возникающих в ПО уязвимостей [3]. В этом контексте конвейер непрерывной интеграции и непрерывной доставки ПО становится важным элементом, который обеспечивает быстрые циклы этапов разработки ПО и устранение ошибок на разных этапах, тем самым повышая уровень безопасности приложений. Технология Kubernetes [4-6] облегчает процессы масштабирования и развертывания приложений в различных средах, занимает основную роль в этом конвейере, обеспечивая надежную среду для контейнеризации приложений и управления ими соответственно. Обеспечение

безопасности должно производится на всех этапах жизненного цикла, для чего в документе содержатся общие требования к каждому этапу цикла. Однако методики оценивания соответствия реализации процессов разработки безопасного ПО требованиям не являются предметом рассмотрения данного документа и выбираются с учетом специфики конкретного приложения. В качестве такой методики возможно использовать методику тестирования на проникновение [7, 8].

Идентификация ресурсов

При обсуждении методики тестирования на проникновение контейнерных технологий важно учитывать две точки зрения: внутренний взгляд изнутри контейнера и внешний взгляд с хоста. Внутри контейнер воспринимается как изолированная среда, сродни автономному процессу. Такая изоляция ограничивает доступ, предоставляя доступ только к тем файлам и ресурсам, которые связаны с контейнером, позволяя выполнять команды исключительно в его пределах.

Внешнее тестирование на проникновение охватывает хост-систему, где как хост-процессы, так и контейнеры отображаются как процессы, находящиеся на хосте. Это позволяет видеть все доступные процессы на хосте, включая взаимодействие с демоном Docker [9, 10] и его дочерними процессами.

Проведение базовой идентификации доступных ресурсов в кластере Kubernetes необходимо для выявления потенциальных уязвимостей. Ниже представлен перечень базовых первостепенных проверок для получения первичной информации о кластере:

- для разрешения DNS-имен в пространстве имен Kubernetes по умолчанию с целью выявления внутренних IP-адресов используется команда

  nslookup kubernetes.default;

- сканирование портов сервера Kubernetes API с помощью команды

  nmap -p- <IP-адрес Kubernetes API>,

  чтобы обнаружить открытые порты и службы, которые могут быть уязвимы;

- пространства имен помогают организовать ресурсы внутри кластера; для их перечисления используется команда

  kubectl get namespaces;

- получение списка модулей в пространстве имен осуществляется выполнением команды

  kubectl get pods -n <namespace>;

- проверка версии сервера Kubernetes определяется посредством команды

kubectl version;

- чтобы составить список и проанализировать все развернутые образы контейнеров и их версии:

kubectl get services -A -o wide;

- дамп всех развертываний для анализа конфигураций выполняется командой

kubectl get deployments -A -o yaml > deployments.yaml;

- действия, разрешенные текущему пользователю, можно определить, выполнив

kubectl auth can-i –list;

- аудит настроек средств контроля доступа на наличие ролей с высокими привилегиями осуществляется путем получения информации с команд;
- вход в модуль для определения особенностей его функционирования осуществляется командой

kubectl exec -ti <pod_name> -n <namespace> -- /bin/sh;

- получение токена сервисной учетной записи модуля, который можно использовать для взаимодействия с API Kubernetes, через команду

kubectl exec -ti <pod> -n <namespace> -- cat /run/secrets/ kubernetes.io/ serviceaccount/token;

- используя ранее найденные токены, получить доступ к секретам в пространствах имен возможно выполнив команду

curl -v -H Authorization: Bearer <jwt_token>https://<master_ip> :<port>/api/v1/namespaces/ <namespace>/secrets;

В сценариях, где выполнение команды *kubectl* затруднено из-за ограничений на загрузку или установку утилиты kubectl, несмотря на наличие доступа к сети и необходимых разрешений или сертификатов для взаимодействия с API-сервером Kubernetes, для управления ресурсами кластера требуются альтернативные способы. Один из таких способов заключается в репликации функциональных возможностей команд *kubectl* путем прямого взаимодействия с API-сервером Kubernetes. Это позволяет выполнять задачи управления кластером без использования *kubectl*. Чтобы эмулировать команды *kubectl* важно понимать структуру запросов, которые генерирует *kubectl*. Для перечисления всех сетевых политик, применяемых в пространствах имен, определения правил маршрутизации трафика и сетевой сегментации необходимо выполнить команду

kubectl get networkpolicies --all-namespaces.

Используя результаты выполнения команды, необходимо проанализировать каждую сетевую политику, чтобы понять структуру маршрутизации трафика.

Повышение привилегий с учетной записью службы

Kubernetes использует для управления контейнерами сертификат учетной записи службы, который привязывается к контейнеру при инициализации модуля. В соответствии со стандартной конфигурацией сертификат и токен учетной записи службы обеспечивают связь с сервером API служб Kubernetes. Изначально учетные данные службы не содержат высоких привилегий. Однако администраторы кластера обладают возможностью повысить привилегии учетной записи службы через команду

kubectl create clusterrolebinding service-account-admin-binding --clusterrole=cluster-admin --serviceaccount=default: default.

Выполнение команды добавляет учетной записи службы роль администратора кластера, предоставляя административные привилегии в кластере.

Тестирование сервера API

Сервер API Kubernetes выступает в качестве центрального интерфейса для управления ресурсами кластера. Получение контроля над главным узлом, на котором хранятся файлы конфигурации и учетные данные администратора для сервера API, означают полную компрометацию кластера. Сервер API требует аутентификации для обеспечения безопасного доступа. Однако при неправильной настройке, когда сервер API запущен с флагами

--insecure-bind-address=0.0.0.0 и *--insecure-port=8080*,

это требование не соблюдается, что приводит к несанкционированному доступу.

Тестирование компонента kubelet

В Kubernetes каждый узел управляется службой kubelet. Порт 10250 служит важным каналом связи между kubelet и API-сервером Kubernetes, позволяя kubelet получать информацию о задачах, которые ему необходимо выполнить. Администраторы Kubernetes могут включить анонимный доступ в определенных конфигурациях, что можно проверить командой

curl -k https://<kubernetes-node-ip>:10250/pods.

Используя конечную точку */pods*, становится возможным получение подробной информации о кластере, такой как пространства имен, идентификаторы модулей и конфигурация контейнера. Кроме того, возможно исполнение команд в модулях, используя команду

curl -k https://<kubernetes-node-ip>:10250/run/<namespace>/<pod>/<container> -d cmd=<command>.

Тестирование панели мониторинга Kubernetes

В Kubernetes панель мониторинга предлагает веб-интерфейс пользователя, облегчающий управление кластерами Kubernetes. Несмотря на наличие аутентификации, поддерживающей вход в систему с помощью файлов конфигурации или токенов, конфигурацией панели мониторинга можно манипулировать, чтобы обойти аутентификацию с помощью опции *enable-skip-login*.

Это позволяет пользователям получать доступ к панели мониторинга без проверки подлинности. Однако доступ к панели мониторинга не означает предоставление привилегий для управления кластером. Kubernetes использует управление доступом на основе ролей для управления разрешениями, где параметры доступа определяется ролями, назначенными различным учетным записям служб. По умолчанию учетная запись службы, связанная с информационной панелью Kubernetes, обладает ограниченными привилегиями, что ограничивает ее способность выполнять задачи по управлению кластером, если явно не настроено иное. Распространенная практика заключается в повышении уровня доступа к панели мониторинга путем привязки роли администратора кластера к учетной записи службы. После применения этой конфигурации доступ к панели мониторинга обеспечивает полный административный контроль над кластером.

Тестирование базы данных etcd

База данных etcd – это распределенное хранилище ключей и значений, используемое Kubernetes и другими системами для хранения важных данных и управления ими. Инструмент *etcdctl* предоставляет интерфейс командной строки для взаимодействия с базой данных, позволяя выполнять различные операции, такие как чтение и запись ключей. Обычно распределенное хранилище ключей использует порт 2379 для взаимодействия с клиентами. Доступ к этому порту через общедоступную сеть без надлежащего контроля доступа может привести к утечке конфиденциальной информации. Kubernetes использует etcd в качестве хранилища данных по умолчанию, где хранятся все данные конфигурации кластера, его состояние и метаданные. Получив доступ к etcd, возможно использовать несколько тактик для повышения привилегий и получения контроля над кластером Kubernetes. Основной метод заключается в извлечении токенов, хранящихся

в etcd, которые используются для аутентификации на сервере API

Заключение

Описанная методика ориентирована на Kubernetes и предполагает глубокое понимание принципов его работы. В центре внимания методики находится этап развертывания жизненного цикла разработки ПО, на котором происходит контроль и управление процессами развертывания контейнерных приложений. Несмотря на то, что методика разработана с учетом особенностей контейнерных сред, она не исключает интеграции с другими методиками и может быть дополнена другими методиками для обеспечения комплексной оценки безопасности. Методика сосредоточена на среде контейнеризации и не учитывает специфику приложений, работающих в контейнерах.

Список литературы

1. ГОСТ Р ИСО/МЭК 27000-2021 «Информационные технологии. Методы и средства обеспечения безопасности. Система менеджмента информационной безопасности. Общий обзор и терминология». – М.: Стандартинформ, 2021.
2. Проект национального стандарта ГОСТ Р 56939 «Защита информации. Разработка безопасного программного обеспечения. Общие требования». [Электронный ресурс]. https://fstec.ru/tk-362/standarty/proekty/proekt-natsionalnogo-standarta-gost-r-56939 (Дата обращения: 7.03.2024).
3. ГОСТ Р 50922-2006 «Защита информации. Основные термины и определения». Введ. 2008-02-01. – М.: Стандартинформ, 2008.
4. Исследование VK Cloud – Kubernetes в России. 2023-2024 гг. [Электронный ресурс]. https://cloud.vk.com/promopage/state-of-kubernetes/. (Дата обращения: 7.03.2024).
5. State of Kubernetes Security Report. 2023 г. [Электронный ресурс]. https://www.redhat.com/en/resources/state-kubernetes-security-report-2023. (Дата обращения: 7.03.2024).
6. Center for Internet Security. CIS Kubernetes Benchmark. 2022, pp. 67-70.
7. Journal of Cybersecurity. Recent advances in penetration testing. 2022, pp. 90-94.
8. OWASP Foundation. Web Security Testing Guide. [Электронный ресурс]. https://owasp.org/www-project-web-security-testing-guide/stable/. (Дата обращения: 27.02.2024).
9. Docker Documentation. Docker Security. [Электронный ресурс]. https://docs.docker.com/engine/security/. (Дата обращения: 25.03.2024).
10. Center for Internet Security. CIS Docker Benchmark. 2022, pp. 78-83.

УДК 004.056

РАЗРАБОТКА СКАНЕРА СКРЫТЫХ КАНАЛОВ В ПРОТОКОЛЕ ПЕРЕДАЧИ ГИПЕРТЕКСТА СИСТЕМ БАНК-КЛИЕНТ ПРИ ПОМОЩИ КЛАССИФИКАТОРОВ МАШИННОГО ОБУЧЕНИЯ

А.Ю. Симачев
Студент группы М22-508 НИЯУ МИФИ, simachev.anton@mail.ru

Аннотация: Представлены результаты разработки сканера для выявления скрытых каналов в протоколе передачи гипертекста систем банк-клиент с использованием классификаторов машинного обучения. В работе рассматриваются архитектура систем банк-клиент, анализируются актуальные сетевые протоколы и выделяются основные признаки наличия скрытых каналов. Также представлены методы машинного обучения, применяемые для классификации трафика. Проведен анализ результативности разработанного сканера, подтверждающий его эффективность в условиях реального сетевого трафика.

Ключевые слова: скрытые каналы, протокол HTTP/3, системы банк-клиент, машинное обучение, классификаторы, информационная безопасность.

Введение

С увеличением числа систем банк-клиент возросла необходимость в обеспечении их безопасности. Одной из основных угроз является наличие скрытых каналов, которые позволяют злоумышленникам передавать данные незаметно для стандартных систем защиты.

Системы банк-клиент стали неотъемлемой частью современного банковского дела. Они предоставляют удобный и оперативный доступ к банковским услугам через интернет, что существенно упрощает работу клиентов с банком. Однако с ростом их популярности возрастает и количество угроз безопасности. Безопасность систем банк-клиент является критически важной задачей, поскольку они обрабатывают чувствительные данные, такие как личные данные клиентов и финансовую информацию. Уязвимости в этих системах могут привести к утечке данных, финансовым потерям и подрыву доверия клиентов. Одной из главных угроз безопасности являются сквозные атаки, при которых злоумышленники проникают в систему и передают данные через скрытые каналы, оставаясь незамеченными для стандартных средств защиты. В контексте систем банк-клиент, это может означать передачу данных, таких как пароли и личная информация, через законные на первый взгляд HTTP/3-запросы. HTTP/3 является последней версией протокола передачи гипертекста, которая использует протокол QUIC для улучшения скорости и безопасности передачи данных. Он предоставляет ряд преимуществ, таких как улучшенная производи-

тельность и уменьшенная задержка, что делает его привлекательным для использования в системах банк-клиент. Несмотря на свои преимущества, HTTP/3 также имеет уязвимости, которые могут быть использованы злоумышленниками для создания скрытых каналов передачи данных. Это связано с его сложной структурой и новизной, что делает его менее изученным по сравнению с предыдущими версиями протокола.

Для эффективного обнаружения скрытых каналов в HTTP/3 протоколе необходимо использовать современные методы анализа данных, такие как машинное обучение. Классификаторы машинного обучения могут быть обучены на выявление аномалий и необычных паттернов в сетевом трафике. На первом этапе необходимо собрать большой объем данных сетевого трафика, включающего как легитимные запросы, так и запросы с использованием скрытых каналов. Затем данные анализируются для выявления характеристик, которые могут указывать на наличие скрытых каналов. На основе полученных характеристик создается и обучается модель машинного обучения, способная классифицировать запросы. Разработанная модель проходит тестирование на различных наборах данных для проверки ее точности и надежности. Использование машинного обучения для выявления скрытых каналов в протоколе HTTP/3 позволяет значительно повысить точность и скорость обнаружения потенциальных угроз, что способствует улучшению общей безопасности систем банк-клиент.

Цель данной работы – создать сканер для выявления скрытых каналов в протоколе передачи гипертекста (HTTP/3) систем банк-клиент с использованием классификаторов машинного обучения.

Архитектура систем банк-клиент

Пользователь взаимодействует с системой банк-клиент с помощью веб-браузера, который является универсальным инструментом доступа к интернет-ресурсам. Веб-браузер позволяет пользователю вводить URL-адреса, переходить по ссылкам и загружать веб-страницы, предоставляющие интерфейс для работы с банковскими услугами. При использовании системы банк-клиент пользователь отправляет различные типы запросов на сервер банка, в том числе запросы на просмотр баланса счета и выполнение переводов.

Актуальные сетевые протоколы систем банк-клиент

Системы банк-клиент полагаются на различные сетевые протоколы для передачи данных и обеспечения надежного и безопасного взаимодействия между пользователями и банковскими серверами. Эти протоколы выполняют

различные функции, начиная от разрешения доменных имен до маршрутизации пакетов данных через интернет. Основные из них – это HTTP/3, DNS, BGP, OSPF и RIP. HTTP/3 является новейшей версией протокола передачи гипертекста, предназначенной для ускорения и повышения безопасности интернет-коммуникаций [1]. Он использует протокол QUIC, который работает поверх UDP (User Datagram Protocol). Благодаря использованию QUIC, HTTP/3 значительно уменьшает задержки при установке соединений и передаче данных. Протокол включает в себя встроенное шифрование, что делает коммуникации более защищенными по сравнению с предыдущими версиями HTTP. Также он обеспечивает более стабильное соединение, особенно в условиях нестабильных сетей, за счет встроенных механизмов восстановления утраченных пакетов. DNS (Domain Name System) – это протокол, который используется для преобразования доменных имен в IP-адреса. Он играет ключевую роль в интернете, обеспечивая пользователям возможность обращаться к веб-сайтам и другим ресурсам по удобным для восприятия именам, таким как example.com. Когда пользователь вводит URL адрес банка в браузере, DNS преобразует этот адрес в соответствующий IP-адрес сервера банка. Системы банк-клиент могут использовать расширения DNS, такие как DNSSEC, для обеспечения целостности и аутентичности DNS-запросов. BGP (Border Gateway Protocol) – это протокол динамической маршрутизации, который используется для обмена маршрутизируемой информацией между автономными системами в интернете. BGP помогает находить наиболее эффективные маршруты для передачи данных, что улучшает скорость и надежность соединений. Протокол также позволяет быстро перенастраивать маршруты в случае сбоев, что минимизирует перерывы в обслуживании. OSPF (Open Shortest Path First) и RIP (Routing Information Protocol) – это протоколы динамической маршрутизации, используемые для обмена маршрутной информацией внутри одной автономной системы. OSPF использует алгоритм на основе состояния канала для определения наикратчайшего пути и имеет высокую скорость конвергенции, быстро адаптируясь к изменениям в сети.

Скрытые каналы HTTP/3 и признаки их наличия

Для выявления скрытых каналов необходимо анализировать признаки, которые могут свидетельствовать о передаче данных через скрытые каналы. Основные признаки включают необычную частоту или последовательность определённых типов кадров, непропорционально большое количество потоков или фреймов, а также нестандартные значения параметров в кадрах SETTINGS. Необычная частота или последовательность определённых типов кадров может

указывать на использование скрытых каналов, так как злоумышленники могут изменять типы кадров или их последовательность для маскировки данных. Например, последовательности кадров, которые не соответствуют обычному паттерну трафика, могут быть признаком передачи данных через скрытые каналы. Непропорционально большое количество потоков или фреймов также может свидетельствовать о наличии скрытых каналов. Злоумышленники могут создавать большое количество дополнительных потоков или фреймов для разделения и передачи данных небольшими частями, что затрудняет их обнаружение стандартными методами анализа трафика. Анализ количества и распределения потоков и фреймов может помочь выявить такие аномалии. Нестандартные значения параметров в кадрах SETTINGS являются ещё одним признаком, который может указывать на скрытые каналы. Кадры SETTINGS используются для настройки параметров соединения между клиентом и сервером [2]. Злоумышленники могут изменять стандартные значения параметров в этих кадрах для передачи дополнительных данных. Анализ этих значений и выявление отклонений от нормальных параметров может помочь в обнаружении скрытых каналов. Для анализа данных и выявления этих признаков можно использовать методы машинного обучения. Классификаторы машинного обучения, обученные на наборах данных, содержащих примеры как легитимного трафика, так и трафика с использованием скрытых каналов, могут эффективно различать нормальные и аномальные паттерны. Обучение моделей включает сбор и предварительную обработку данных, выбор признаков, обучение и валидацию моделей. Использование машинного обучения позволяет автоматизировать процесс анализа и повысить точность обнаружения скрытых каналов. Выявление скрытых каналов в протоколах передачи данных, таких как HTTP/3, требует анализа различных признаков аномального поведения трафика. Основные признаки включают необычную частоту или последовательность кадров, непропорционально большое количество потоков или фреймов, и нестандартные значения параметров в кадрах SETTINGS. Применение методов машинного обучения позволяет автоматизировать и улучшить процесс обнаружения скрытых каналов, что способствует повышению безопасности систем банк-клиент.

Классификаторы машинного обучения

Для классификации трафика и выявления скрытых каналов используются различные методы машинного обучения. В данной работе рассматриваются следующие классификаторы: наивный байесовский классификатор, классификатор LSTM, Random Forest, Feedforward Neural Network и Support Vector Machine (SVM) [3].

Наивный байесовский классификатор – это простой и эффективный метод для начального анализа данных. Он основывается на применении теоремы Байеса с предположением о независимости признаков. Этот классификатор быстро обучается и работает, что делает его хорошим выбором для предварительного анализа больших объемов данных.

Классификатор LSTM (Long Short-Term Memory) представляет собой рекуррентную нейронную сеть, способную учитывать временные зависимости в данных. LSTM используется для анализа последовательностей данных и может эффективно выявлять скрытые паттерны во временных рядах, что делает его подходящим для задач анализа сетевого трафика, где временные зависимости играют важную роль.

Random Forest – это ансамблевый метод, основанный на использовании множества деревьев решений. Каждый отдельный классификатор (дерево) обучается на случайном подмножестве данных, а финальное решение принимается на основе голосования всех деревьев. Этот метод отличается высокой точностью и устойчивостью к переобучению, что делает его эффективным инструментом для анализа сложных данных.

Feedforward Neural Network представляет собой многослойную нейронную сеть, способную выявлять сложные зависимости в данных. Такие сети состоят из нескольких слоев нейронов, где каждый слой обучается выделять определенные признаки из данных. Feedforward Neural Network может обучаться на больших объемах данных и выявлять сложные взаимосвязи, что делает её мощным инструментом для классификации трафика.

Support Vector Machine (SVM) – это метод, использующий гиперплоскости для разделения классов данных. SVM находит оптимальную гиперплоскость, которая максимально разделяет классы в многомерном пространстве признаков. Этот метод эффективен для задач классификации, где необходимо найти четкое разделение между классами, и он хорошо работает с линейно и нелинейно разделимыми данными.

Применение этих методов в совокупности позволяет эффективно классифицировать трафик и выявлять скрытые каналы. Наивный байесовский классификатор обеспечивает быструю и простую начальную оценку данных. Классификатор LSTM учитывает временные зависимости и выявляет паттерны в последовательностях данных. Random Forest объединяет результаты множества деревьев решений, обеспечивая высокую точность и устойчивость. Feedforward Neural Network выявляет сложные зависимости и адаптируется к особенностям данных. SVM использует гиперплоскости для точного разделения классов, что позволяет четко классифицировать трафик [4]. Все эти методы

вместе позволяют создавать комплексные системы анализа, способные обнаруживать скрытые каналы в сетевом трафике с высокой точностью.

Построение модулей сканера скрытых каналов

Сканер скрытых каналов разработан с учётом функциональных и программных требований, а также ресурсов, необходимых для его работы. Архитектура сканера включает следующие модули: модуль захвата, модуль анализа, модуль классификации и модуль представления [5].

Модуль захвата отвечает за сбор сетевого трафика. Этот модуль постоянно мониторит сеть, перехватывая и записывая весь входящий и исходящий трафик. Для этого могут использоваться различные инструменты и технологии, такие как пакеты захвата на уровне сетевого интерфейса или использование специализированных устройств для мониторинга трафика. Основная цель модуля захвата – обеспечить полное и точное представление о сетевом трафике для последующего анализа [6].

Модуль анализа выполняет предварительную обработку данных и выявляет подозрительные признаки. Этот модуль очищает и нормализует захваченные данные, устраняя шум и некорректные записи. После этого он анализирует данные на наличие аномалий и признаков скрытых каналов, таких как необычная частота или последовательность кадров, непропорционально большое количество потоков или фреймов, а также нестандартные значения параметров в кадрах SETTINGS. Выявленные подозрительные признаки передаются в следующий модуль для более детального анализа.

Модуль классификации использует методы машинного обучения для определения наличия скрытых каналов. Этот модуль применяет различные классификаторы, такие как наивный байесовский классификатор, классификатор LSTM, Random Forest, Feedforward Neural Network и Support Vector Machine (SVM), для анализа предварительно обработанных данных и выявления скрытых каналов. Каждый классификатор обучен на большом наборе данных, включающем как легитимный трафик, так и примеры скрытых каналов, что позволяет им эффективно различать нормальные и аномальные паттерны.

Модуль представления визуализирует результаты анализа и предоставляет отчёты. Этот модуль отображает результаты работы сканера в удобной для восприятия форме, предоставляя пользователю графики, таблицы и отчёты о выявленных скрытых каналах. Визуализация включает информацию о типах и частоте подозрительных кадров, идентифицированных потоках и фреймах, а также о значениях параметров, которые были определены как

аномальные. Пользователь может использовать эти отчёты для дальнейшего расследования и принятия мер по устранению угроз.

В совокупности, эти модули обеспечивают комплексный подход к выявлению скрытых каналов в сетевом трафике. Модуль захвата гарантирует полный сбор данных, модуль анализа выявляет подозрительные признаки, модуль классификации точно определяет наличие скрытых каналов, а модуль представления предоставляет результаты в удобной и понятной форме. Такая архитектура позволяет эффективно обнаруживать и предотвращать использование скрытых каналов, улучшая общую безопасность сетевых систем.

Анализ результативности сканера

Анализ результативности сканера показал его высокую эффективность в условиях реального сетевого трафика. Использование различных классификаторов машинного обучения позволило достичь высокой точности в выявлении скрытых каналов. Экспериментальные данные подтверждают адекватность разработанной модели и её применимость в реальных системах банк-клиент. В ходе испытаний сканер был протестирован на различных наборах данных, имитирующих реальный сетевой трафик, включая трафик с внедрёнными скрытыми каналами. Полученные результаты показали, что сканер успешно идентифицирует скрытые каналы с высокой точностью, минимизируя количество ложных срабатываний. Экспериментальные данные подтвердили, что разработанная модель способна адаптироваться к изменениям в сетевом трафике и эффективно выявлять скрытые каналы в различных условиях. Это делает сканер применимым в реальных системах банк-клиент, где безопасность и точность обнаружения угроз являются критически важными. Высокая результативность сканера в реальных условиях свидетельствует о его потенциале для широкого использования в банковских и других критически важных системах, требующих защиты от скрытых угроз.

Заключение

В рамках данного исследования рассматривалась возможность разработки сканера скрытых каналов в протоколе передачи гипертекста систем банк-клиент. Использование методов машинного обучения позволило значительно повысить эффективность выявления скрытых каналов. В ходе разработки была создана архитектура, включающая модуль захвата для сбора сетевого трафика, модуль анализа для предварительной обработки данных и выявления подозрительных признаков, модуль классификации для опреде-

ления наличия скрытых каналов с помощью различных классификаторов машинного обучения, и модуль представления для визуализации результатов анализа и предоставления отчётов. Экспериментальные данные показали высокую точность работы сканера в реальных условиях, подтверждая его адекватность и применимость в системах банк-клиент. Использование различных классификаторов, таких как наивный байесовский классификатор, LSTM, Random Forest, Feedforward Neural Network и SVM, обеспечило комплексный подход к анализу трафика и выявлению скрытых каналов. Каждый из этих методов внёс свой вклад в общую точность и надёжность системы, что позволило минимизировать количество ложных срабатываний и улучшить выявление скрытых угроз. Будущие исследования будут направлены на дальнейшую оптимизацию алгоритмов, чтобы повысить их производительность и эффективность. Планируется изучение дополнительных факторов, влияющих на безопасность систем банк-клиент, таких как новые виды скрытых каналов, изменение паттернов трафика и развитие технологий шифрования. Также будет уделено внимание адаптации сканера к изменениям в сетевой инфраструктуре и улучшению методов обработки и анализа данных для повышения общей безопасности и устойчивости банковских систем.

Список литературы

1. Hypertext Transfer Protocol Version 2 (HTTP/2): RFC 7540. 2015.
2. BrainJS. JavaScript naive Bayesian classifier. [Электронный ресурс]. https://github.com/BrainJS/classifier/ (Дата обращения: 06.08.2024).
3. Draft-ietf-quic-invariants. [Электронный ресурс]. https://www.rfc-editor.org/auth48/C430 (Дата обращения: 06.08.2024).
4. Vladimir Mladenovic. Evaluation of HTTP/3 Protocol for Internet of Things and Fog Computing Scenarios. [Электронный ресурс]. https://www.researchgate.net/publication/ 354965913_Evaluation_of_HTTP3_Protocol_for_Internet_of_Things_and_Fog_Computing_ Scenarios. (Дата обращения: 06.08.2024).
5. Епишкина А.В., Когос К.Г. Об оценке пропускной способности скрытых информационных каналов, основанных на изменении длин передаваемых пакетов // Информатика, вычислительная техника и управление, 2015, с. 78-82.
6. Симачев А.Ю. Оценка скрытых каналов по памяти в протоколе QUIC и методы их обнаружения // Научно-исследовательский журнал для студентов и преподавателей «StudNet», 2022, Т. 5, № 5, с. 3644-3652.