ФИЗИКО-МАТЕМАТИЧЕСКИЕ НАУКИ / PHYSICS AND MATHEMATICS

Научная статья
УДК 004.934.056.5
https://doi.org/10.35266/1999-7604-2025-4-7

Комплексное решение многоуровневой стратегии защиты программных средств обработки акустической информации

Николай Евгеньевич Балакирев, Игорь Сергеевич Сергеев ✉, ***Кирилл Андреевич Коновалов***
Московский авиационный институт (национальный исследовательский университет), Москва, Россия

Аннотация. В статье представлена многоуровневая стратегия защиты программных средств и обрабатываемых ими акустических данных, разработанная на примере комплексов «Голосовая почта» и «Акустический оптимизатор». В основе подхода лежит структурное преобразование звукового потока с потерями, не влияющими на его информационное содержание, что позволяет использовать компактную форму представления данных в качестве механизма сокрытия. Предлагаемая стратегия сочетает программные и аппаратные методы защиты, включая индивидуализацию экземпляров программ, обфускацию кода, внедрение внешних аппаратных модулей, а также формализацию процедур в виде операторной схемы. Реализация методов обеспечивает устойчивость программных решений к анализу, декомпиляции и подмене, а также повышает защищенность звуковой информации от несанкционированного доступа.

Ключевые слова: защита программных средств, структуризация, оптимизация, обфускация, сжатие с потерями, акустическая информация, цифровая обработка, кодирование данных

Для цитирования: Балакирев Н. Е., Сергеев И. С., Коновалов К. А. Комплексное решение многоуровневой стратегии защиты программных средств обработки акустической информации // Вестник кибернетики. 2025. Т. 24, № 4. С. 58–72. https://doi.org/10.35266/1999-7604-2025-4-7.

Original article

Comprehensive approach to multi-layered protection strategy of acoustic information processing software

Nikolay E. Balakirev, Igor S. Sergeev ✉, ***Kirill A. Konovalov***
Moscow Aviation Institute (National Research University), Moscow, Russia

Abstract. The article presents a multi-layered protection strategy for software systems and the acoustic data they process, developed using the “Voicemail” and “Acoustic Optimizer” applications. The fundamental aspect of this methodology is a structural transformation applied to the audio stream, incorporating lossy compression to retain the original information, which in turn facilitates the utilization of a compact data representation for the purpose of concealment. The proposed strategy combines software and hardware protection methods, including program instance individualization, code obfuscation, integration of external hardware modules, and procedure formalization using an operator-based model. The implementation of these methods ensures the resilience of software solutions against analysis, reverse engineering, and tampering while also enhances the protection of acoustic data against unauthorized access.

Keywords: software protection, structuring, optimization, obfuscation, lossy compression, acoustic information, digital signal processing, data encoding

For citation: Balakirev N. E., Sergeev I. S., Konovalov K. A. Comprehensive approach to multi-layered protection strategy of acoustic information processing software. *Proceedings in Cybernetics*. 2025;24(4):58–72. https://doi.org/10.35266/1999-7604-2025-4-7.

ВВЕДЕНИЕ

Настоящая статья посвящена разработке многоуровневой стратегии обеспечения стойкости программных средств и обрабатываемых ими данных к попыткам несанкционированного вскрытия.

Любые методы шифрования обладают конечным сроком устойчивости: при наличии достаточного времени и вычислительных ресурсов зашифрованные данные могут быть вскрыты. В классической работе Клода Шеннона “Communication Theory of Secrecy Systems” сформулировано, что единственно теоретически неразрушимым является шифр с равной длиной ключа и сообщения – например, одноразовый блокнот (one-time pad). Любой другой шифр при определенной вычислительной мощности и наличии ресурса времени может быть вскрыт [1].

В большинстве случаев шифрование предполагает применение операций по преобразованию или перемешиванию кодируемых символов с использованием ключей различной длины, при этом длина итогового потока данных остается, как правило, неизменной. Стойкость алгоритма в значительной степени зависит от длины ключа: чем он больше, тем больше времени требуется для его перебора и соответственно для вскрытия зашифрованной информации.

В рамках настоящего исследования под передаваемыми данными будут подразумеваться звуковые данные. Одним из потенциальных способов обеспечения их защиты является применение алгоритмов сжатия с потерями, как, например, те, что используются в форматах MP3 или AAC [2]. Однако широкое распространение и высокая известность математических преобразований, лежащих в основе таких методов (в частности, преобразования Фурье), ограничивают их применение в задачах защиты данных.

Предлагаемый в статье подход основан на структурном представлении звукового потока во временной области с потерями промежуточных значений амплитуды, не влияющими на восприятие сигнала и не нарушающими его информационного содержания [3]. Такое представление обеспечивает компактную форму хранения и передачи акустической информации (в дальнейшем – КоФ).

Не имея информации о структуре КоФ и способах ее декодирования, злоумышленник оказывается неспособным преобразовать данные в исходный звуковой поток. Таким образом, защита обеспечивается не только за счет сжатия, но и за счет сложности самой процедуры восстановления, зависящей от реализованных программных алгоритмов и структуры представления данных.

Фактически ключом к расшифровке информации, преобразованной в КоФ, является либо наличие исходного кода программных средств, реализующих все этапы преобразования и восстановления, либо знание алгоритмических принципов, лежащих в основе процессов структуризации, уплотнения и кодирования звуковой информации. В первом случае злоумышленник получает доступ к исходному коду программного продукта, во втором – осуществляет «реверс-инжиниринг», создавая функциональный аналог с использованием иного кода, но с сохранением логики обработки данных.

Процедура вскрытия требует значительных усилий, специальных инструментов и квалификации, что существенно повышает порог для потенциального нарушителя и увеличивает временные затраты на анализ. В этом случае можно утверждать, что длиной ключа будет длина реализованного алгоритма, помноженная на сложность алгоритма КоФ. Логично предполагать, что время вскрытия такой программы злоумышленником будет на порядок больше, чем вскрытие ключа шифрования. Из этого следует, что первоочередной задачей является обеспечение защищенности самой программы от кражи, модификации и анализа – всеми возможными средствами. Отсюда следует, что приоритетной задачей становится защита самой программной реализации от несанкционированного копирования, анализа и модификации с использованием как существующего набора средств защиты, так и разработкой собственных инструментов в рамках многоуровневой стратегии защиты.

В рамках проводимых исследований, направленных на анализ и синтез информационного содержания акустических сигналов [4], был также получен прикладной результат в виде значительного сокращения объема передаваемых данных. Это стало возможным за счет структуризации звукового потока на основе амплитудных характеристик с минимальными потерями, не влияющими на восприятие сигнала. Как показано в работе [3], восстановление звукового потока из структурированной формы обеспечивает полное сохранение информационного содержания. Таким образом, наряду с задачами оптимизации объема данных предлагаемые преобразования могут рассматриваться как эффективный механизм скрытой передачи информации, аналогичный криптографическим подходам.

Дальнейшие разделы статьи посвящены вопросам реализации многоуровневой защиты данных и программных решений, а также анализу применяемых инструментов.

МАТЕРИАЛЫ И МЕТОДЫ

С учетом сложности реализуемой многоуровневой стратегии защиты программных средств и генерируемых данных представляется целесообразным описывать весь процесс в рамках формализованной операторной модели. В данной модели каждый этап обработки данных рассматривается как функция, допускающая множество возможных реализаций, которые далее будут обозначаться как оператор-процедуры. Предполагается, что к одному и тому же набору данных может быть применен единый алгоритм преобразования, реализованный в различных вариантах, приводящих к идентичному результату.

Для дальнейшего построения операторной схемы передачи и хранения данных введем используемые обозначения.

Служебные символы и их назначение:

(…) – в круглых скобках будет указываться множество элементов, на которое распространяется алгоритм защиты данных;

[…] – в квадратных скобках будет указываться множество элементов, на которое распространяется стратегия (процедура) защиты алгоритма;

– логическое объединение (связывание) процедур в рамках выбранной стратегии;

⇒ – символ преобразования одной последовательности данных в другую последовательность.

Обозначения данных:

a_i – исходный элемент потока данных волнового явления,

где $a_i \in A = (a_1, a_2, \ldots a_i, \ldots, a_n, \ldots)$;

a_{i^c} – закодированный элемент последовательности данных, где $a_{i^c} \in A^c = \left(a_{1^c}, a_{2^c}, \ldots, a_{i^c}, \ldots, a_{n^c} \ldots\right)$ без изменений их объема относительно исходного потока;

S_j – последовательность, где $S_j \in S$, т. е. $(S_1, S_2, \ldots S_i, \ldots, S_k, \ldots)$, элемент структурированного потока данных, полученного из потока $(a_1, a_2, \ldots a_i, \ldots, a_n, \ldots)$, и при этом $k < n$;

g_t – последовательность, где где $g_j \in G$, т. е. $(g_1, g_2, \ldots g_t, \ldots, g_w, \ldots)$, элемент упакованных (оптимизированных) данных относительно исходного потока звуковых данных;

u_{i^c} – последовательность, где $u_{i^c} \in U^c$, т. е. $\left(u_{1^c}, u_{2^c}, \ldots, u_{i^c}, \ldots, u_{k^c}, \ldots\right)$, элемент оптимизированных данных относительно структурированного потока данных;

$a_{i^c} \in A^c = \left(a_{1^c}, a_{2^c}, \ldots, a_{i^c}, \ldots, a_{n^c} \ldots\right)$ без изменений их объема относительно исходного потока.

Обозначения общих функций-алгоритмов:

F – функция-алгоритм прямого однозначного отображения или преобразования одной последовательности данных в другую последовательность;

V – функция-алгоритм обратного однозначного отображения или преобразования одной последовательности данных в другую последовательность относительно прямой функции-алгоритма;

R_f^j – j-я реализация функции-алгоритма прямого однозначного отображения или преобразования одной последовательности данных в другую последовательность;

R_v^j – j-я реализация функции-алгоритма обратного однозначного отображения или преобразования одной последовательности данных в другую последовательность.

Тогда общепринятое преобразование при кодировании данных может быть представлено как:

$$F(a_1, a_2, \dots a_i, \dots, a_n, \dots) \Rightarrow \left(a_{1^c}, a_{2^c}, \dots, a_{i^c}, \dots, a_{n^c} \dots\right), \quad (1)$$

А обратная функция будет выглядеть следующим образом:

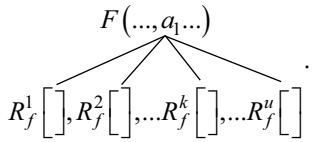
$$V\left(a_{1^c}, a_{2^c}, \dots, a_{i^c}, \dots, a_{n^c} \dots\right) \Rightarrow (a_1, a_2, \dots a_i, \dots, a_n, \dots). \quad (2)$$

Тогда общая схема преобразований при обмене передающей и принимающей сторон будет следующей:

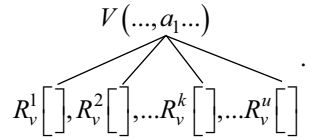
$$F(a_1, a_2, \dots a_i, \dots, a_n, \dots) \Rightarrow \left(a_{1^c}, a_{2^c}, \dots, a_{i^c}, \dots, a_{n^c} \dots\right),$$
$$V\left(a_{1^c}, a_{2^c}, \dots, a_{i^c}, \dots, a_{n^c} \dots\right) \Rightarrow (a_1, a_2, \dots a_i, \dots, a_n, \dots).$$

Обозначения общих операторов-процедур, реализующих алгоритмы кодирования:

Представление алгоритмов кодирования данных будет выглядеть следующим образом:



Аналогичная ситуация для обратной функции:



Как правило, для обычного кодирования процедура соответствует единственному экземпляру реализации:

$$R_f \equiv F \text{ и } R_v \equiv V.$$

В качестве таковой используется любой алгоритм кодирования без потерь. Далее рассматриваются только алгоритмы уплотнения (в том числе и кодирования) информации с потерями.

Обозначения конкретных операторов-процедур, реализующих алгоритм, обеспечивающий кодирование:

C_f^k – *k-я* процедура реализации прямой функции-алгоритма структуризации

$$C\left(a_1, a_2, \dots, a_i, \dots, a_n, \dots\right) => \left(S_{1^c}, S_{2^c}, \dots, S_{i^c}, \dots, S_{k^c}, \dots\right),$$

где $k < n$, (3)

D_v^k – *k-я* процедура реализации обратной функции-алгоритма деструктуризации (восстановления из структурного представления):

$$D\left(S_{1^c}, S_{2^c}, \dots, S_{i^c}, \dots, S_{k^c}, \dots\right), => \left(a_1, a_2, \dots, a_i, \dots, a_n, \dots\right),$$

где $k < n$, (4)

Q_f^k – *k-я* процедура реализации прямой функции-алгоритма фрагментного уплотнения структурного представления.

G_v^k – *k-я* процедура реализации обратной функции-алгоритма фрагментного восстановления структурного представления.

Тогда алгоритмы для прямого преобразования будут следующего вида:

$$Q\left(S_{1^c}, S_{2^c}, \dots, S_{i^c}, \dots, S_{k^c}, \dots\right) => \left(u_{1^c}, u, \dots, u_{i^c}, \dots, u_{k^c}, \dots\right),$$

где $k < n$, (5)

Тогда алгоритмы для обратного преобразования будут следующего вида:

$$G\left(u_{1^c}, u, \dots, u_{i^c}, \dots, u_{k^c}, \dots\right) => \left(S_{1^c}, S_{2^c}, \dots, S_{i^c}, \dots, S_{k^c}, \dots\right),$$

где $k < n$, (6)

H_f^k – *k-я* процедура реализации прямой функции-алгоритма уплотнения фрагментов на основе информационного содержания.

M_v^k – *k-я* процедура реализации обратной функции-алгоритма восстановления фрагментов на основе информационного содержания.

Z – общая многоуровневая стратегия защиты процесса передачи и хранения звуковых данных с указанием цепочки вариантов реализации процедур, включая в себя и мероприятия организационного характера.

Тогда операторная схема-модель организации защиты относительно данных будет выглядеть следующим образом.

Операторная схема 1:

$$Z \Rightarrow H\left(Q\left(C(\dots 0_i, \dots)\right)\right). \quad (7)$$

Операторная схема 1 применяется в случае применения фиксированной стратегии защиты данных. Такая стратегия соответствует постоянной стратегии защиты и может быть использована для некритичных обменов информацией.

Операторная схема 2:

$$Z^m \Rightarrow H^m\left(Q^m\left(C^m\left(\ldots 0_i,\ldots\right)\right)\right). \qquad (8)$$

Операторная схема 2 применяется в случае порядковой стратегии защиты данных.

В данном случае возможно для каждой организации предоставить собственную стратегию защиты, что обеспечивает защищенность при раскрытии одной из стратегий.

Операторная схема 3:

$$Z^{k,m,n} \Rightarrow H^k(Q^m(C^n(\ldots a_i, \ldots) \qquad (9)$$

Операторная схема 3 применяется в случае применения переборной стратегии защиты данных. Такая схема предполагает динамический характер изменения используемых операторов. Таким образом, один и тот же результат возможно получить разными стратегиями, которые применимы как для оптимизации данных, так и для их защиты в соответствии с таблицей.

Симметричные процессы защиты можно использовать при восстановлении исходной информации с таким же набором стратегий. Стратегии приема и передачи должны быть синхронизированы в соответствии с совместимостью прямой и обратной процедурой. Количество стратегий определяется суммой вариаций для соответствующих процедур:

Таблица

Стратегии защиты данных

	C_1	…	$C_к$	Q_1	…	Q_m	H_1	…	H_n
Z_1	+					+	+		
Z_2		+		+					+
…									
Z_u	+			+		+			

Примечание: составлено авторами на основании данных, полученных в исследовании.

$$C_u = k = m = n. \qquad (10)$$

Например, при $k = m = n = 3 \Rightarrow 27$ вариантов стратегий. Предполагаемая согласованность процедур обеспечивает дополнительную защиту данных и процедур. Эффект можно усилить при распределении процедур по нескольким исполнителям (процессорам), дополняя стратегию различными маршрутами преобразования.

Более сложная процедура реализуется для защиты программной реализации алгоритмов, в которой присутствуют как программно-аппаратные методы, так и организационные мероприятия.

Точно так же допустимо защитить программный продукт в виде использования в динамике той или иной процедуры защиты самой программы. Указанная возможность представлена формулой (11):

$$Z => [H]\#[Q]\#[C], \qquad (11)$$

что представляет собой логическое объединение защищенных процедур. Для этого возможно использование обфускации, индивидуализации экземпляров программы. Возможно применение любого из вышеперечисленных способов в единственном варианте и для ограниченного перечня процедур.

В конечном счете можно обеспечить комплексную защиту как данных, так и процедур в зависимости от уровня угроз и важности информации в соответствии с формулой (12):

$$Z_j \Rightarrow \left\{\left[H^k\right]\left(\left[Q^m\right]\left(\left[C^n\right](\ldots a_i,\ldots)\right)\right)\right\}. \qquad (12)$$

Далее будет рассмотрен каждый из вариантов защиты относительно данных и программного кода.

Защита звуковых данных, полученных при структуризации

Реализуя программные продукты по передаче речевых сообщений и компактному хранению звуковых данных с потерями (например, MP3 и AAC), в общепринятом варианте очень часто используют преобразование Фурье и вейвлет-преобразования, которые широко известны и не представляют какой-либо дополнительной сложности для воспроизве-

дения злоумышленниками в процессе обратного преобразования [5].

Предлагаемый подход представления звуковых данных основан на процессе структуризации звукового потока [4], где также происходит частичная потеря данных, но она предполагает утрату лишь той части информации, которая может быть восстановлена в пределах коридора из возможной вариации, таким образом, исключая потерю качества воспроизведения. Информация о процессе структуризации и кодировании звуковых данных не является общедоступной, в связи с чем появляется вопрос о защите генерируемых данных на основе структуризации.

Следует отметить, что программные средства «Голосовая почта» и «Акустический оптимизатор», используя структуризацию, реализованы на основе качественного подхода с использованием системы отношений для структурного представления звукового потока с возможностью сохранения его информационного содержания [3]. При данном подходе основными этапами обработки звукового потока стали выделение характерных элементов (характерных точек), структуризация (реализуемая операторами-процедурами C и D, формула (3), основанная на установлении отношений между характерными элементами [3, 4].

Как видно из предлагаемой выше модели защиты, в рамках обозначенной проблемы передачу звуковых сообщений, т. е. данных, можно осуществлять вне рамок стандартных общеупотребимых механизмов, например, таких как сквозное шифрование, которое можно осуществить через стандартизированные подходы [6–8], с использованием в особой оптимизированной и уплотненной форме [9].

На примере программных средств для работы со звуковыми данными предлагается сценарий преобразования согласно формуле (7) для исходного звукового потока, призванный затруднить действия злоумышленника. Мероприятиями в реализации вышесказанного используются несколько собственных наработок [9–11] и ряд общепринятых и специализированных решений [12–16].

Защита данных при оптимизации представления структуры

В структурном представлении звукового потока, при наличии общей формы представления структуры, все поля имеют максимальные размеры с учетом возможного разброса значений этих полей. Однако при внимательном рассмотрении оказывается, что достаточно большие участки звуковых данных не выходят за границы очень узкого диапазона значений. В рамках отсутствия речевых данных разброс значений может находиться в пределах 1–2 разрядов, тогда как под значение амплитуды выделяется 8 (или даже 16, 24) разрядов. Фиксация в двух разрядных полях сразу дает возможность уменьшить объем данных в 4 раза. Безусловно, это предполагает достаточно большой объем таких данных, что как раз и подтверждается на практике [11].

Более того, получение характеристик относительно фиксированных структур позволяет не только характеризовать их при разделении на классы различных участков, но и предложить их автоматическое объединение в так называемые совокупные фрагменты по значению разброса между минимальным и максимальным значением с кратностью 2^n. При этом можно задать сразу несколько таких параметров, которые для удобства в дальнейшем именуются «калибрами». Тогда весь поток можно разбить на совокупные фрагменты со своим значением параметра калибра и далее при наличии достаточно большого такого фрагмента, представить его в оптимальном представлении, экономя число используемых разрядов. Безусловно, требуются дополнительные описатели, хотя бы для указания способа преобразования от обычной структуры к оптимальной для совокупного фрагмента. Но такие дополнительные издержки существенно малы по отношению к экономии представления структур в оптимальной форме. Такое преобразование мы именуем оптимизацией представления количественных значений, которое применимо и к другим полям исходной структуры.

Структурой называется модель данных, которая описывает характер поведения конфигурации или формы данных на основе некото-

рого множества устойчивых по содержанию элементов. Защитой же в случае с используемой звуковой информацией является использование КоФ, как было упомянуто ранее, со способом реализации этой КоФ, который не является общеизвестным.

Формальный способ такого представления соответствует схемам-формулам (3) и (4).

С помощью такого способа представления данных становится возможным, в том числе, реализация преобразованного варианта представления, названного уплотнением, что представлено ранее в рамках оператора-процедуры Q и G. Кроме того, также становится доступным еще один вариант преобразования, названный оптимизация, в рамках которого форма представления информации основывается на упомянутых характерных элементах – характерных точках, или паттернах, названных универсальный примитив (УНИПРИМ) [3]. Благодаря такому представлению информации становится реализуемым способ описания всех передаваемых звуковых данных через множество УНИПРИМов.

Защита данных через процесс совокупной фрагментации

В предложенном выше варианте использования характеристик при оптимизации выделялась количественная характеристика, указывающая на разброс значений в выделенных полях. Но по совокупности характеристик можно характеризовать и информационное содержание потока данных, которое, безусловно, является назначаемым субъектом-экспериментатором [4]. Однако при установлении параметров, которые могут быть объединением определенного набора характеристик, становится возможным автоматически назначать информационное содержание. При наличии обратной процедуры получения совокупного звукового фрагмента в соответствии с параметрами открывается перспектива восстановления речевых единиц. Тогда кодом, раскрывающим конкретное информационное содержание, будет соответствующий набор параметров. Ключом к вскрытию речи в таком случае будет программа, восстанавливающая речевые единицы. При абсолютно адекватном распознавании речевых единиц возможно также их хранение в символьной форме.

В данный момент используется вариант с выделением совокупных фрагментов с так называемым минимальным информационным содержанием. То есть при наличии достаточно больших участков звуковой записи, когда ничего не произносится, а также когда присутствует пауза или промежуток между словами, их можно заменить любой последовательностью значений амплитуд в заданном диапазоне значений. Все это никак не повлияет на то содержание, которое присутствует в речи.

Указанные участки можно маркировать короткой записью, которая в какой-то мере является кодом маркируемого фрагмента.

Таким образом, в соответствии с предложенной схемой будет оптимизирован как объем данных, так и в то же время закодирована речевая информация.

Описанные методы защиты звуковых данных реализуются в соответствии со схемой, представленной на рис. 1.

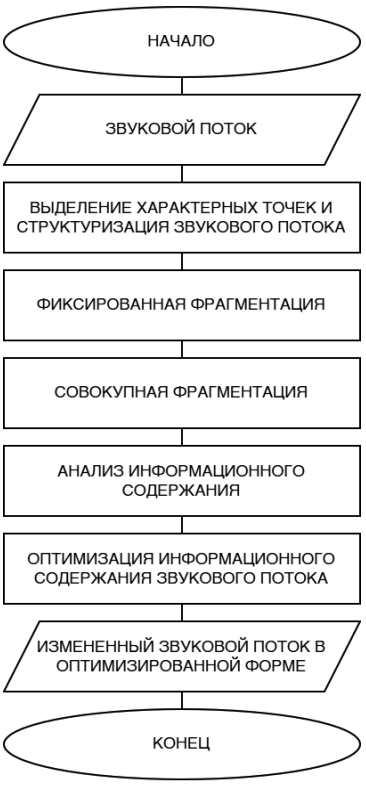


Рис. 1. Схема обработки звукового потока в рамках качественного подхода

Примечание: составлено авторами на основании данных, полученных в исследовании.

Защита данных через обфускацию программного кода

Как следует из предыдущего раздела защиты данных, можно понять, что не менее важным моментом при таком обеспечении является защита и программных процедур.

Программные средства «Голосовая почта» и «Акустический оптимизатор», как уже было сказано ранее, реализованы как прикладное программное обеспечение для операционных систем стационарных компьютеров и предоставляются в виде исполняемого файла и нескольких файлов подключаемых библиотек.

Современные платформы разработки, такие как.NET и JVM, предполагают использование байт-кода, который легче поддается анализу и дизассемблированию. Это повышает риск раскрытия логики работы программ, особенно с применением инструментов типа IDA, что требует дополнительных мер по защите исходного кода.

Обфускация применяется для затруднения анализа исходного кода программ, особенно при использовании платформ.NET и JVM, где байт-код легко поддается декомпиляции. Основные приемы включают переименование функций и переменных, добавление «мусорных» конструкций, усложнение арифметики, маскирование литералов и строк. Эти меры усложняют понимание логики без изменения функциональности.

На примере программного средства «Акустический оптимизатор» проведена обфускация экземпляра программного средства, используя инструмент обфускации для.NET [17]. Фрагмент кода после обфускации представлен ниже на рис. 2.

```
// CompressionTool, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// <PrivateImplementationDetails>{302B7DB6-35E9-4728-A0ED-6F35C076BEA6}.A7D8999C-9C76-4AEB-837A-B923EA1AFBF9
using ...

[StructLayout(LayoutKind.Auto, CharSet = CharSet.Auto)]
internal class A7D8999C-9C76-4AEB-837A-B923EA1AFBF9
{
    [StructLayout(LayoutKind.Explicit, Pack = 1, Size = 17024)]
    private struct 2
    ...

    internal static 2 3/* Not supported: data(EE D9 C9 D3 C3 C4 E9 C1 C7 CE C5 CF D2 87 DF 85 FB DF DC CB DB CC CF 9D 8F 93
     66 F7 0E 7B 0C 6D 02 6A 00 6A 07 5C F4 04 AB FA AA F8 AE FE AF FD 98 F2 A2 F0 A5 F6 A4 05 12 11 16 4C 50 56 15 41 18 1D
     DC 89 EE 8F E5 8D E9 82 DB 71 7E 7D 7A 7F 83 D7 82 87 84 7F 1B 7D 10 72 23 71 1C 77 1A 75 08 6A 33 69 0B 9F 6D 39 62 03
     08 67 0E 6D 0C 68 02 67 00 64 06 6A 04 6B 0A FA AB F8 AF FF 93 FD 97 F2 A8 F1 9B F7 9A 0B 1A EB 98 E9 8E EF 8B ED 8E E3
     88 E9 64 7F 31 3C 39 36 13 3E 3B 31 26 34 84 C8 C7 C4 C8 DD CD 7D 06 73 10 71 1F 77 1F 74 31 9B 69 38 6E 01 6C 0F 62 0D
     64 09 6B 0E 5E 0D 68 03 68 F1 06 68 04 6B 0A FB 92 F9 9E FF 97 FD 9A F3 91 F0 A6 F6 A7 1F 57 5A 53 5C 7A 4D 5D 47 5F 58

    internal static byte[] 4 = new byte[17024]
    ...;

    internal static string[] 5 = new string[419];

    private static string 6(int P_0, int P_1, int P_2)
    ...

    public static string A()
    {
        return 5[0] ?? 6(0, 0, 13);
    }

    public static string a()
    ...

    public static string B()
```

Рис. 2. Фрагмент текста программы «Акустический оптимизатор» после применения обфускации
Примечание: составлено авторами на основании данных, полученных в исследовании.

Следует учитывать минусы обфускации, такие как увеличение времени на выполнение кода или увеличение размера исполняемого файла.

Защита данных через индивидуализацию экземпляров программы

Ключевым механизмом стратегии защиты, вокруг которого реализуются остальные, является индивидуализация экземпляров программного средства [10]. Данный механизм предлагается для организации взаимодействия участников обмена информации в рамках указанных программных средств таким образом, чтобы охватить задачи по своей реализации за счет механизмов использования различающихся вариантов кода программы

и структур передаваемой информации и их периодического изменения. Или, другими словами, механизм создания уникального экземпляра программного средства под каждого пользователя.

Теперь сформулируем требования к организации создаваемых программных средств через введенные ранее понятия:

предлагается распределение механизма защиты на всем пути работы с информацией;

вводится механизм контроля программного средства через его индивидуализацию.

Модель программного средства «Голосовая почта» показана на рис. 3.

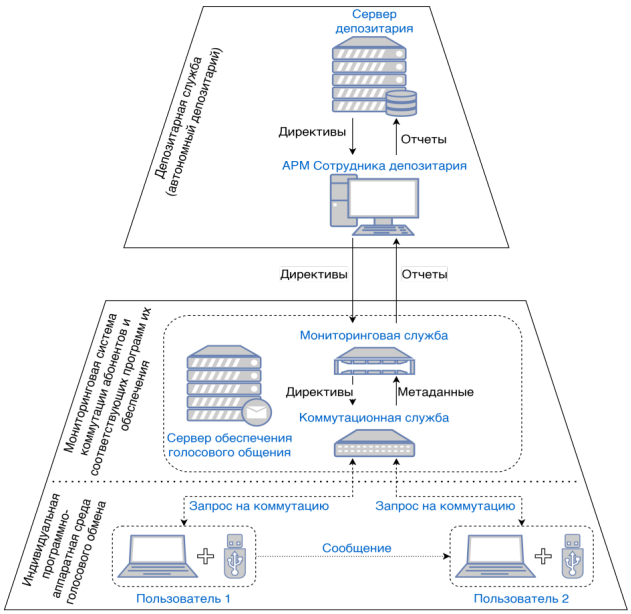


Рис. 3. Модель программного средства «Голосовая почта»

Примечание: составлено авторами на основании данных, полученных в исследовании.

Определим основные используемые далее понятия:

1) время вскрытия ($T_{\text{В}}$ от «time»), характеризующее временной интервал, требуемый для вскрытия информации;

2) ресурсы злоумышленника ($R_{\text{З}}$ от «resources»), характеризующее программные и аппаратные возможности, требуемые для вскрытия информации;

3) квалификация злоумышленника ($S_{\text{З}}$ от «skill»), определяющее его уровень компетенций, знаний, умений.

Чтобы программное средство можно было охарактеризовать как защищенное, следует реализовать ее таким образом, чтобы:

1) время вскрытия было больше времени существования конкретного защитного механизма ($T_{\text{В}} > T_{\text{З}}$);

2) ресурсы злоумышленника были меньше ресурсов, требуемых для вскрытия ($R_{\text{В}} > R_{\text{З}}$);

3) квалификация злоумышленника оказалась меньшей требуемой для вскрытия ($S_{\text{В}} < S_{\text{З}}$).

Поскольку злоумышленник ставит своей целью получение информации из локальных данных работающего программного обеспечения или передаваемого потока, он анализирует, как уже было сказано ранее, входные и выходные данные какого-то программного обеспечения или какие-то компоненты программно-аппаратного решения, если смог получить к нему физический доступ. Чтобы усложнить злоумышленнику указанные действия, далее будут описаны предлагаемые способы программной и физической защиты.

Чтобы механизм индивидуализации мог работать, был также введен механизм учета и контроля как набор дополнительных действий (формула (12)). Данные механизмы помогают значительно увеличить требуемое для вскрытия время со стороны злоумышленника, т. е. параметр $T_{\text{В}}$. Благодаря этому механизму стало возможным контролировать сменяемость версий программного средства при функциональном обновлении, что регулирует временные интервалы актуальности конкретного варианта защиты.

После безуспешной попытки вскрытия информации будет попытка обратиться к алгоритмам программных средств. Меры, предпринятые на этом этапе, характеризуются усложнением анализа кода за счет следующих действий:

1) использование ассемблера при написании ключевых частей для минимизации повторяющихся участков кода;

2) создание индивидуальных версий, меняя части кода, не влияя на функционал, с применением внешнего аппаратного модуля (аппаратного ключа защиты);

3) ввод контроля использования индивидуальных версий;

4) применение обфускации кода индивидуальных версий, о чем будет дополнительно сказано далее.

Данный комплекс мер реализует:

1) вариабельность кода через индивидуализацию, в том числе перенося часть кода на внешний аппаратный модуль (увеличивая параметр $R_{В}$);

2) индивидуализацию структур пакета через внешний аппаратный модуль, без изменения функциональности (увеличивая параметр $S_{В}$);

3) возможность периодического изменения кода и структур (увеличивая параметр $T_{В}$).

Добавляемый внешний аппаратный модуль призван обеспечить вынесение критичного функционала с целью снижения влияния на рабочий процесс со стороны основного компьютера, когда подвергнуться воздействию могут рабочие процессы и данные в оперативной памяти. Процесс и обрабатываемые в этом процессе данные, находящиеся во внешнем аппаратном модуле, не видны для основной работающей системы, поэтому находятся в большей сохранности. С другой стороны, просто украв и использовав у себя, злоумышленник не сможет получить доступ к содержимому внешнего аппаратного модуля, т. к. имеется контроль его запуска на компьютерах, т. е. на неразрешенном он не запустится. Для наглядности индивидуализация с применением внешнего аппаратного модуля показана на рис. 4.

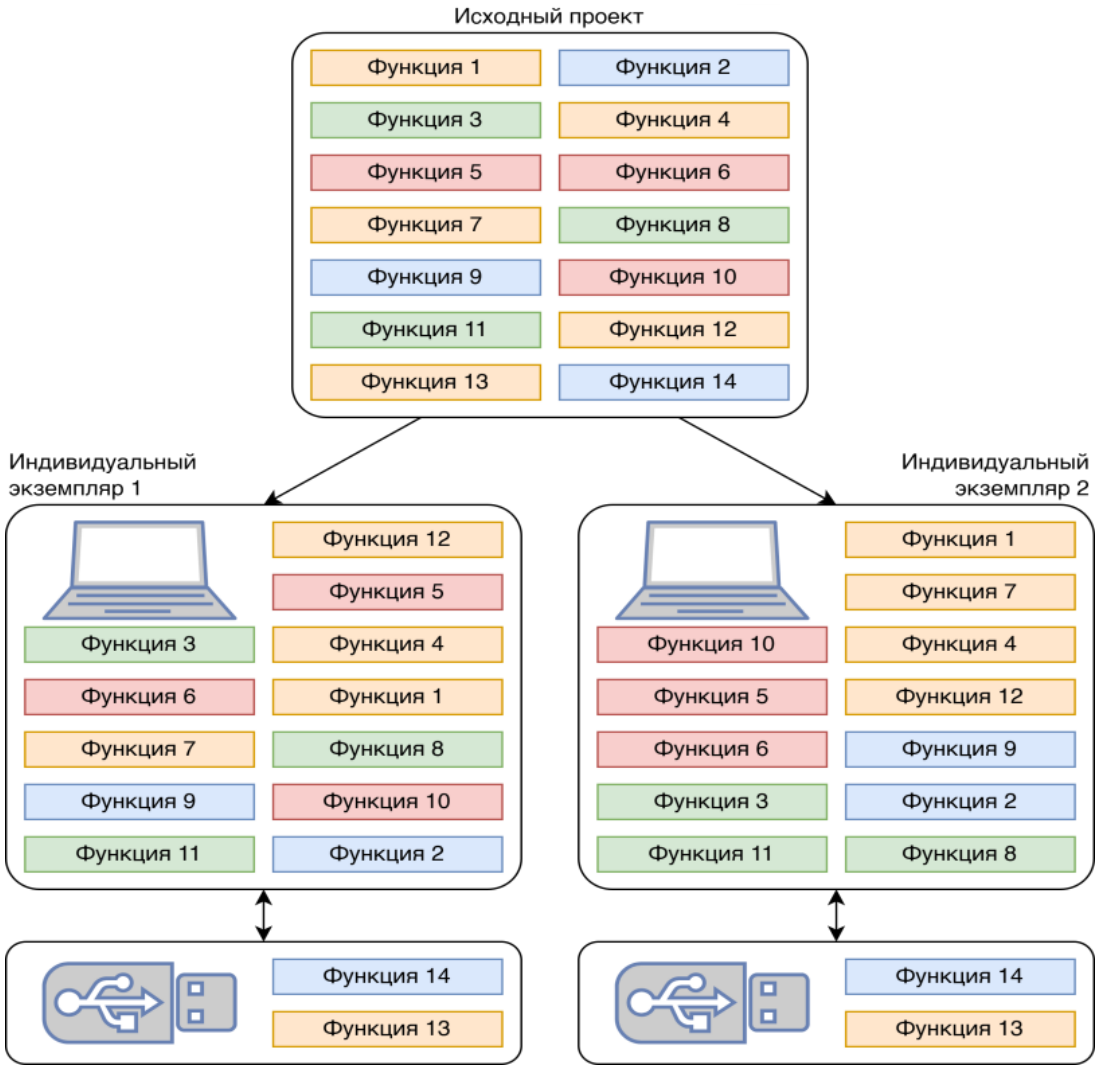


Рис. 4. Модель организации вариабельности кода и индивидуализации структур экземпляров программ

Примечание: составлено авторами на основании данных, полученных в исследовании.

Таким образом, достигаемые цели от введения индивидуализации с применением внешнего аппаратного модуля: бесполезность использования результатов вскрытия алгоритмов программного средства, т. к. к завершению вскрытия код программного средства уже может быть изменен. При этом периодичность изменения индивидуальных экземпляров программного средства предлагается отработать имитацией действий злоумышленника, осуществляя сменяемость в рамках правил выбранной политики, т. е. опытным путем корректируя $T_{В}$ через $T_{ЗМ}$ (моделируемый $T_{З}$).

Организационное обеспечение защиты

Организационная составляющая стратегии защиты направлена на повышение безопасности программных средств за счет индивидуализации экземпляров и использования внешнего аппаратного модуля. В рамках предложенного подхода часть алгоритмов переносится во внешний модуль, что снижает риск копирования, анализа и модификации программного кода на целевом устройстве.

В качестве внешнего модуля используется Guardant Code от ООО «ГК-АКТИВ» [17], позволяющий разместить уникальный код обработки звукового потока вне исполняемого файла рис. 4. Потеря или компрометация модуля делает невозможным доступ к критическим частям кода.

Программный код записывается в виде прошивки, недоступной для чтения стандартными средствами [18]. Экземпляр программы функционирует только при наличии корректного аппаратного модуля, с которым ведется постоянный обмен. При извлечении устройства исполнение прекращается. Таким образом, копирование экземпляра без соответствующего модуля бессмысленно.

Из-за ограничений модуля используется компактная реализация функций, обрабатывающих переданные данные и возвращающих результат. Для повышения надежности введена проверка UID, что позволяет различать «правильные» и «неправильные» сценарии работы рис. 5, не уведомляя пользователя напрямую. Это затрудняет несанкционированное применение программного средства даже при попытке подмены.

Особенностью предлагаемого способа является хранение UID на внешнем аппаратном модуле, по которому можно идентифицировать экземпляр программного средства и обеспечить в рамках алгоритма работы «правильное» или «неправильное» выполнение этапа структуризации, от которого зависят последующие этапы сжатия и восстановления звуковых файлов.

При стандартном «правильном» сценарии выполнения происходит:

– оптимизация объема на хранение значений амплитуд без изменений;

– аппроксимация значений амплитуд на фрагментах минимального информационного содержания к исходным значениям;

– выполнение алгоритмов оптимального кодирования без изменений.

Все эти действия позволяют уплотнить и восстановить звуковой файл, не потеряв качество сообщений, однако в случае несанкционированного использования индивидуального экземпляра программного средства произойдет отработка в алгоритме структуризации сценария «неправильной» структуризации, что придет к:

1) ложно-корректной проверке объема на хранение значения амплитуды (например, в двухбайтной записи будет утерян второй байт);

2) расчет диапазонов калибра для определения минимального информационного содержания с выдачей ложно-корректного значения (часть информации будет меняться во *внешнем аппаратном модуле* на не соответствующее по отношению к варианту с легитимным использованием значение, т. е. выдаваться подложное и подпадать во фрагмент с предполагаемым минимальным информационным содержанием).

Результатом сценария, где происходит ложно-корректная «неправильная» оптимизация и восстановление, является, например, проигрываемый звуковой файл, который содержит неразборчивый шум вместо исходного звукового сообщения.

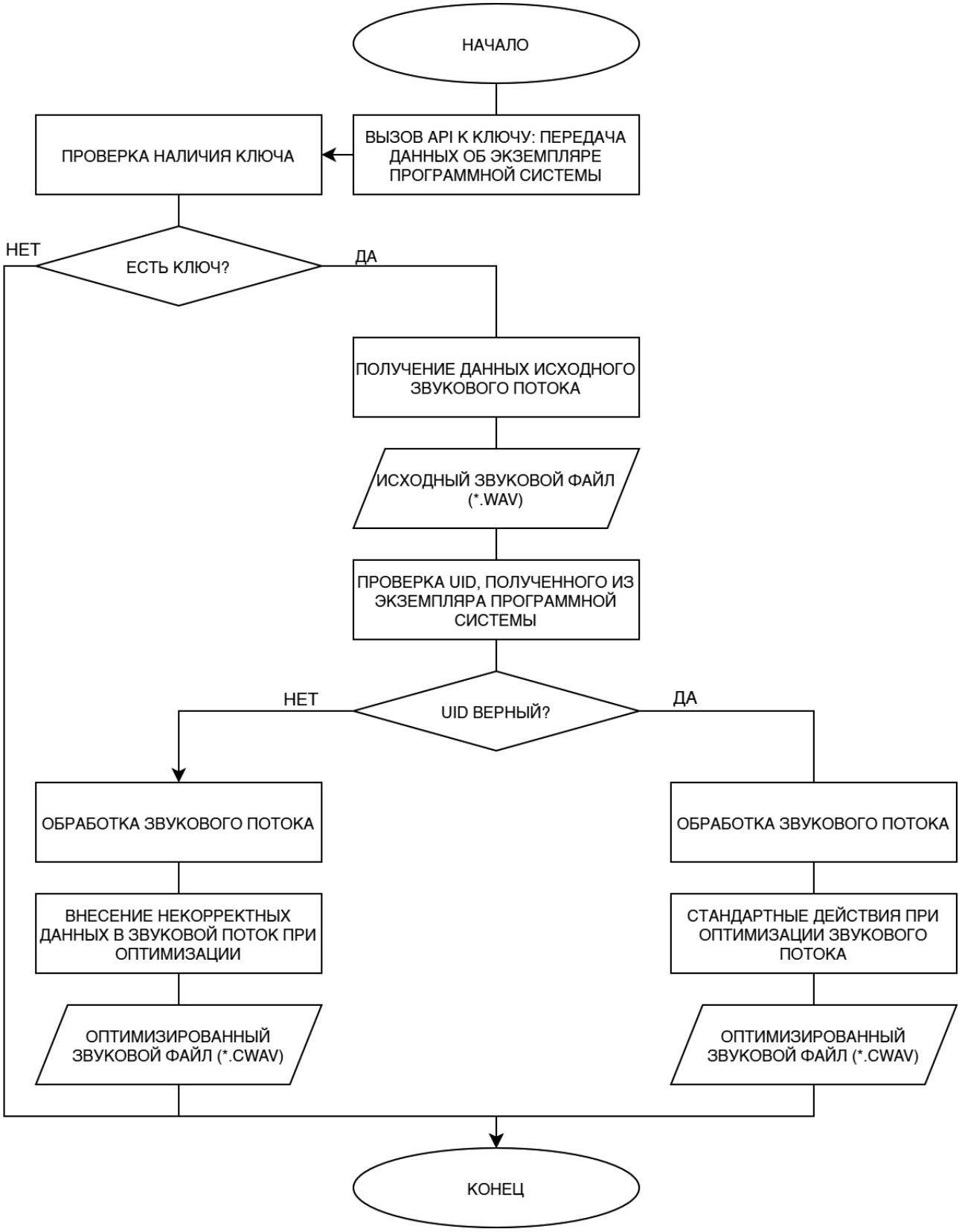


Рис. 5. Схема алгоритма определения сценариев взаимодействия индивидуального экземпляра программного средства и физического ключа

Примечание: составлено авторами на основании данных, полученных в исследовании.

Этот пример позволяет отразить возможность дальнейшего использования экземпляра программного средства, когда по внешним признакам злоумышленнику не будет сообщаться о неправомерности использования, а инсценироваться внешне полностью корректная работа, но с отработкой алгоритма на выдачу ложно-корректных данных.

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

В представленной работе разработана и обоснована многоуровневая стратегия защиты программных средств обработки акустической информации и генерируемых ими данных. Предложенное решение базируется на объединении структурных, алгоритмических, программных и аппаратных методов защиты, позволяющих обеспечить комплексную устойчивость как самих программных компонентов, так и обрабатываемой звуковой информации к несанкционированному доступу и анализу.

В качестве базовых компонентов стратегии рассмотрены:

– структуризация и фрагментация звукового потока с использованием нестандартных методов преобразования и уплотнения;

– индивидуализация экземпляров программных решений с возможностью динамического изменения кода и структуры;

– применение механизмов обфускации программного кода для усложнения анализа логики работы;

– перенос критически важного функционала на внешний аппаратный модуль (аппаратный ключ), обеспечивающий изоляцию части вычислений и контроля запуска;

– формализация процедур защиты в виде операторных моделей, позволяющих описывать варианты реализации и взаимодействия алгоритмов защиты.

Практическая реализация указанных подходов продемонстрирована на примере программных комплексов «Голосовая почта» и «Акустический оптимизатор», в которых реализованы предложенные принципы защиты. Апробация решений показала, что комбинированное применение программных и аппаратных методов при поддержке организационных мер позволяет значительно повысить стойкость программных решений к анализу, копированию, подмене и декомпиляции.

Разработанная стратегия может быть адаптирована и масштабирована для других классов информационно-технических систем, ориентированных на безопасную обработку, хранение и передачу акустической информации в условиях потенциальных угроз.

ЗАКЛЮЧЕНИЕ

В статье представлена комплексная стратегия защиты программных средств обработки акустической информации, объединяющая структуризацию звуковых данных, программно-аппаратные методы, включающие обфускацию и индивидуализацию экземпляров программы с применением аппаратных ключей.

Формализованная в виде операторной модели, стратегия обеспечивает устойчивость к анализу, декомпиляции и подмене. Апробация на примере систем «Голосовая почта» и «Акустический оптимизатор» подтвердила ее практическую эффективность.

Список источников

1. Шеннон К. Теория связи в секретных системах // Проблемы передачи информации / пер. с англ. 1965. Т. 1, № 1. С. 3–18. https://doi.org/10.1002/j.1538-7305.1949.tb00928.x.
2. Nilsson M. The audio/mpeg Media Type. Internet Engineering Task Force. 2000. URL: https://www.rfc-editor.org/rfc/rfc3003 (дата обращения: 29.08.2025).
3. Балакирев Н. Е., Фадеев М. М., Родионов В. С. Качественный подход в раскрытии информационного содержания волновых данных // Труды МАИ. 2024. № 136. С. 1–21.

References

1. Shannon C. E. Communication theory of secrecy systems. *The Bell System Technical Journal*. 1949;28(4):656–715. htpps://doi.org/10.1002/j.1538-7305.1949.tb00928.x.
2. Nilsson M. The audio/mpeg Media Type. Internet Engineering Task Force. 2000. URL: https://www.rfc-editor.org/rfc/rfc3003 (accessed: 29.08.2025).
3. Balakirev N. E., Fadeev M. M., Rodionov V. S. Qualitative approach in extracting the information content of wave data. *Trudy MAI*. 2024;(136):1–21. (In Russ.).
4. Balakirev N. E., Nguyen H. D., Malkov M. A. et al. Structuring and qualitative consideration of a audio

4. Балакирев Н. Е., Нгуен Х. З., Малков М. А. Структуризация и качественное рассмотрение звукового потока в системе синтеза и анализа речи // Программные продукты и системы. 2018. Т. 31, № 4. С. 768–776.
5. Чижов И. И., Балабанова Т. Н. Сжатие аудиоданных на основе психоакустических принципов восприятия звука человеком // Вестник ВГУ. Серия: Системный анализ и информационные технологии. 2024. № 3. С. 127–137.
6. ГОСТ Р 34.10-2012. Национальный стандарт Российской Федерации. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи. М. : Стандартинформ, 2012. 33 с.
7. ГОСТ Р 34.12-2015. Информационная технология. Криптографическая защита информации. Блочные шифры. М. : Стандартинформ, 2015. 25 с.
8. Курганов Е. А. О глубине аппаратной реализации блочного шифра Кузнечик // Интеллектуальные системы. Теория и приложения. 2016. Т. 20, № 1. С. 61–78.
9. Сергеев И. С., Коновалов К. А., Балакирев Н. Е. Методы применения аппаратных способов защиты программных решений «Голосовая почта» и «Оптимизация объема акустической информации» для реализации механизма индивидуализации // Авиация и космонавтика : сб. 23-ей Междунар. конф., 18–22 ноября 2024 г., г. Москва. М. : Изд-во «Перо», 2024. С. 159.
10. Коновалов К. А. Обеспечение защиты информации в соответствии со схемой взаимодействия компонентов // Управление развитием крупномасштабных систем MLSD’2024 : труды XVII Междунар. конф., 24–26 сентября 2024 г., г. Москва. М. : ИПУ РАН, 2024. С. 1106–1111.
11. Фадеев М. М. Обеспечение компьютерных исследований и физических экспериментов средствами воспроизведения получаемых результатов // Информатика: проблемы, методы, технологии : материалы XXIII Междунар. науч.-практич. конф. им. Э. К. Алгазинова, 15–17 февраля 2023 г., г. Воронеж. Воронеж : ВГУ, 2023. С. 730–738.
12. Оппенгейм А., Шафер Р. Цифровая обработка сигналов. 3-е изд., испр. М. : Техносфера, 2019. С. 1048.
13. Загуменнов А. П. Компьютерная обработка звука. М. : ДМК Пресс, 2011. С. 582.
14. Mohammed A. A., Mohammed D. M. Modified phase coding audio watermarking resistant to signal attacks // International Journal of Computer Applications. 2014. Vol. 92, no. 2. P. 1–6.
15. Музычук Д. С., Медведев М. С. Сегментация, шумоподавление и фонетический анализ в задаче распознавания речи // Молодой ученый. 2013. № 6. С. 86–96.

stream in a speech synthesis and analysis system. *Software & Systems*. 2018;31(4):768–776. htpps://doi.org/10.15827/0236-235X.031.4.768-776. (In Russ.).
5. Chizhov I. I., Balabanova T. N. Audio data compression based on psychoacoustic principles of human sound perception. *Proceedings of Voronezh State University. Series: Systems Analysis and Information Technologies*. 2024;(3):127–137. https://doi.org/10.17308/sait/1995-5499/2024/3/127-137. (In Russ.).
6. GOST (State Standard) R 34.10-2012. National Standard of the Russian Federation. Information technology. Cryptographic data security. Generation and verification processes of electronic digital signature. Moscow: Standartinform; 2012. 33 p. (In Russ.).
7. GOST (State Standard) R 34.12-2015. Information technology. Cryptographic data security. Block ciphers. Moscow: Standartinform; 2015. 25 p. (In Russ.).
8. Kurganov E. A. O glubine apparatnoy realizatsii blochnogo shifra “Kuznechik”. *Intellektualnye Sistemy. Teoriya i Prilozheniya*. 2016;20(1):61–78. (In Russ.).
9. Sergeev I. S., Konovalov K. A., Balakirev N. E. Metody primeneniya apparatnykh sposobov zashchity programmnykh resheniy “Golosovaya pochta” i “Optimizatsiya obema akusticheskoy informatsii” dlya realizatsii mekhanizma individualizatsii. In: *Proceedings of the 23rd International Conference: “Aviatsiya i kosmonavtika”*, November 18–22, 2024, Moscow. Moscow: Publishing the Pero; 2024. 159 p. (In Russ.).
10. Konovalov K. A. Obespechenie zashchity informatsii v sootvetstvii so skhemoy vzaimodeystviya komponentov. In: *Proceedings of the 17th International Conference: “Large-Scale System Development Management (MLSD’2024)”*, September 24–26, 2024, Moscow. Moscow: ICS RAS; 2024. p. 1106–1111. (In Russ.).
11. Fadeev M. M. Obespechenie kompyuternykh issledovaniy i fizicheskikh eksperimentov sredstvami vosproizvedeniya poluchaemykh resultatov. In: *Proceedings of the 23rd International Scientific-Practical Conference named after E. K. Algazinov: “Information Systems and Computer Modeling”*, February 15–17, 2023, Voronezh. Voronezh: Voronezh State University; 2023. p. 730–738. (In Russ.).
12. Oppenheim A., Schafer R. Digital Signal Processing. Rev. 3rd ed. Moscow: Tekhnosfera; 2019.1048 p. (In Russ.).
13. Zagumennov A. P. Kompyuternaya obrabotka zvuka. Moscow: DMK Press; 2011. 582 p. (In Russ.).
14. Mohammed A. A., Mohammed D. M. Modified phase coding audio watermarking resistant to signal attacks. *International Journal of Computer Applications*. 2014;92(2):1–6.
15. Muzychuk D. S., Medvedev M. S. Segmentatsyya, shumopodavlenie i foneticheskiy analiz v zadache raspoznavaniya rechi. *Young Scientist*. 2013;(6):86–96. (In Russ.).

16. Документация Guardant. Работа с Guardant Code. URL: https://dev.guardant.ru/pages/viewpage.action?pageId=1278008 (дата обращения: 29.08.2025).
17. Obfuscar Documentation. Obfuscar. URL: https://www.obfuscar.com/ (дата обращения: 29.08.2025).
18. Актив. Схема работы ключа. Характеристики // Guardant Code. URL: https://www.guardant.ru/products/hardware-keys/guardant-code/ (дата обращения: 29.08.2025).

Информация об авторах

Н. Е. Балакирев – кандидат технических наук, доцент;
https://orcid.org/0009-0002-3252-0687,
balakirev1949@yandex.ru
И. С. Сергеев – аспирант;
https://orcid.org/0009-0001-9862-4684,
noctisik76@gmail.com✉
К. А. Коновалов – старший преподаватель;
https://orcid.org/0009-0008-7530-0039,
kkonov@gmail.com

16. Guardant Documentation. Working with Guardant Code. URL: https://dev.guardant.ru/pages/viewpage.action?pageId=1278008 (accessed: 29.08.2025). (In Russ.).
17. Obfuscar Documentation. Obfuscar. URL: https://www.obfuscar.com (accessed: 29.08.2025).
18. Guardant Code. Aktiv. Key operation scheme. Specifications. URL: https://www.guardant.ru/products/hardware-keys/guardant-code/ (In Russ.)

About the authors

N. E. Balakirev – Candidate of Sciences (Engineering), Docent;
https://orcid.org/0009-0002-3252-0687,
balakirev1949@yandex.ru
I. S. Sergeev – Postgraduate;
https://orcid.org/0009-0001-9862-4684,
noctisik76@gmail.com✉
K. A. Konovalov – Senior Lecturer;
https://orcid.org/0009-0008-7530-0039,
kkonov@gmail.com