

УТВЕРЖДЕН

РИВУ.05439-01 33 01-ЛУ

БИБЛИОТЕКА НПБК-ФП
Руководство программиста
РИВУ.05439-01 33 01

Листов 53

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

2026

Изм.	Лист	№ докум.	Подп.	Дата

Литера

АННОТАЦИЯ

В настоящем документе изложены основные сведения, необходимые программисту для практической работы с библиотекой НПБК-ФП (далее – библиотекой), адаптированной под ОС Debian 13 и имеющей также дополнительные библиотечные функции для обработки кодов доступа к ВСК.

Изм.	Лист	№ докум.	Подп.	Дата

СОДЕРЖАНИЕ

1.	Назначение и условия применения библиотеки	4
1.1.	Назначение	4
1.2.	Условия применения	5
2.	Характеристика библиотеки.....	6
3.	Обращение к библиотеке.....	7
3.1.	Способ загрузки	7
3.2.	Общая схема создания контейнера	7
3.3.	Общая схема восстановления кода доступа	8
4.	Функции программного интерфейса.....	9
4.1.	Функция nbfp_resultstring.....	9
4.2.	Функция nbfp_open.....	9
4.3.	Функция nbfp_close	10
4.4.	Функция nbfp_clear.....	10
4.5.	Функция nbfp_set_param.....	10
4.6.	Функция nbfp_create.....	11
4.7.	Функция nbfp_save_container	12
4.8.	Функция nbfp_load_container	13
4.9.	Функция nbfp_extract.....	13
4.10.	Функция nbfp_nfeat.....	14
4.11.	Функция nbfp_ncode.....	15
5.	Входные и выходные данные	16
5.1.	Описание входных и выходных данных.....	16
5.2.	Рекомендации по формированию обучающей выборки.....	17
5.3.	Рекомендации по прохождению процедуры восстановления кода	18
5.4.	Требования к векторам биометрических признаков, их формату.....	19
5.5.	Параметры ПБК	19
5.6.	Значения, возвращаемые функциями библиотеки	24
6.	Сообщения.....	25
	Сокращения	26
	Приложение 1. Заголовочный файл nbss_fp.h	27
	Приложение 2. Пример использования программного интерфейса.....	34
	Приложение 3. Контрольные данные.....	40

Изм.	Лист	№ докум.	Подп.	Дата

1. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ БИБЛИОТЕКИ

1.1. Назначение

1.1.1. Библиотека разработана для работы в ОС Debian 13 и предназначена для преобразования наборов векторов биометрических признаков, извлекаемых из отдельных изображений отпечатков пальцев в коды доступа к ВСК, а также для настройки параметров преобразования с использованием биометрических данных, предъявляемых пользователем во время регистрации, и дополнительного значения - соли.

1.1.2. Библиотека реализует преобразование биометрия-код, описанное в пакете стандартов ГОСТ Р 52633–2006.

1.1.3. Библиотека предназначена для решения следующих задач:

- повышение скорости и удобства подтверждения полномочий пользователей;
- повышение защищенности доступа к ВСК;
- защита от несанкционированной передачи полномочий пользователей при доступе к ВСК.

1.1.4. Место встраивания библиотеки, входные, выходные данные для создания ПИН-контейнера и восстановления кода доступа к ВСК показано на рис. 1, 2.

Создание ПИН-контейнера

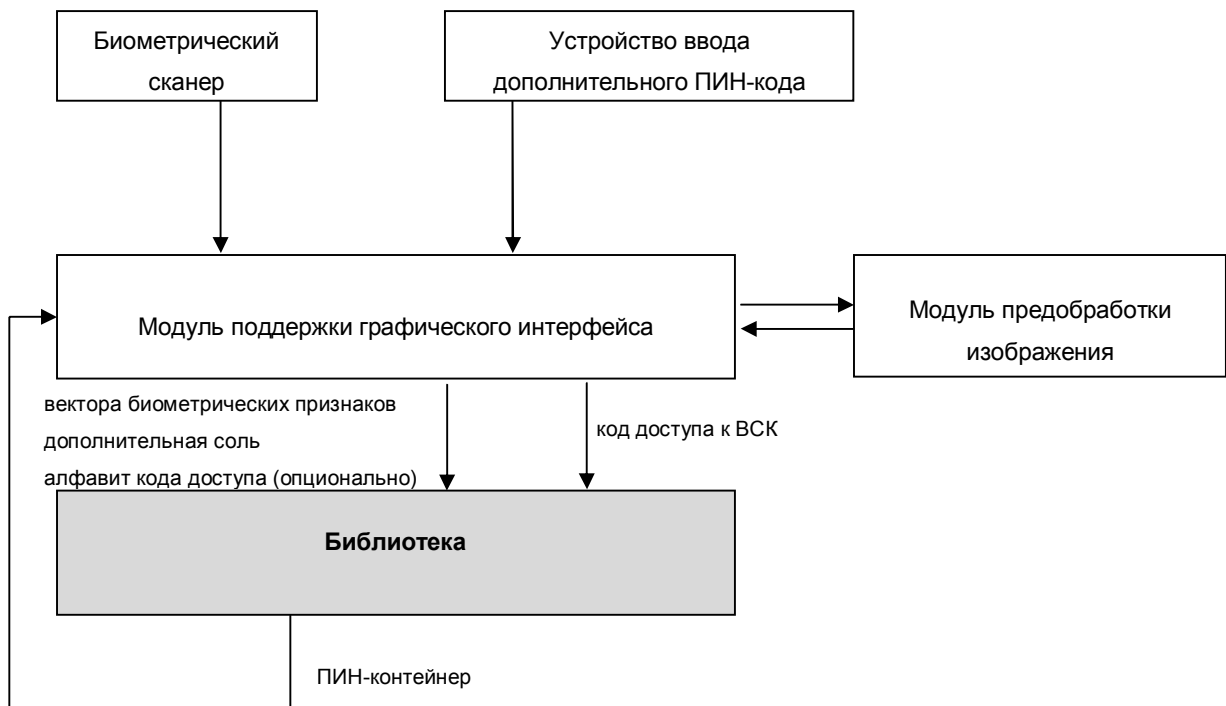


Рис. 1

Изм.	Лист	№ докум.	Подп.	Дата

Восстановление кода доступа к ВКС

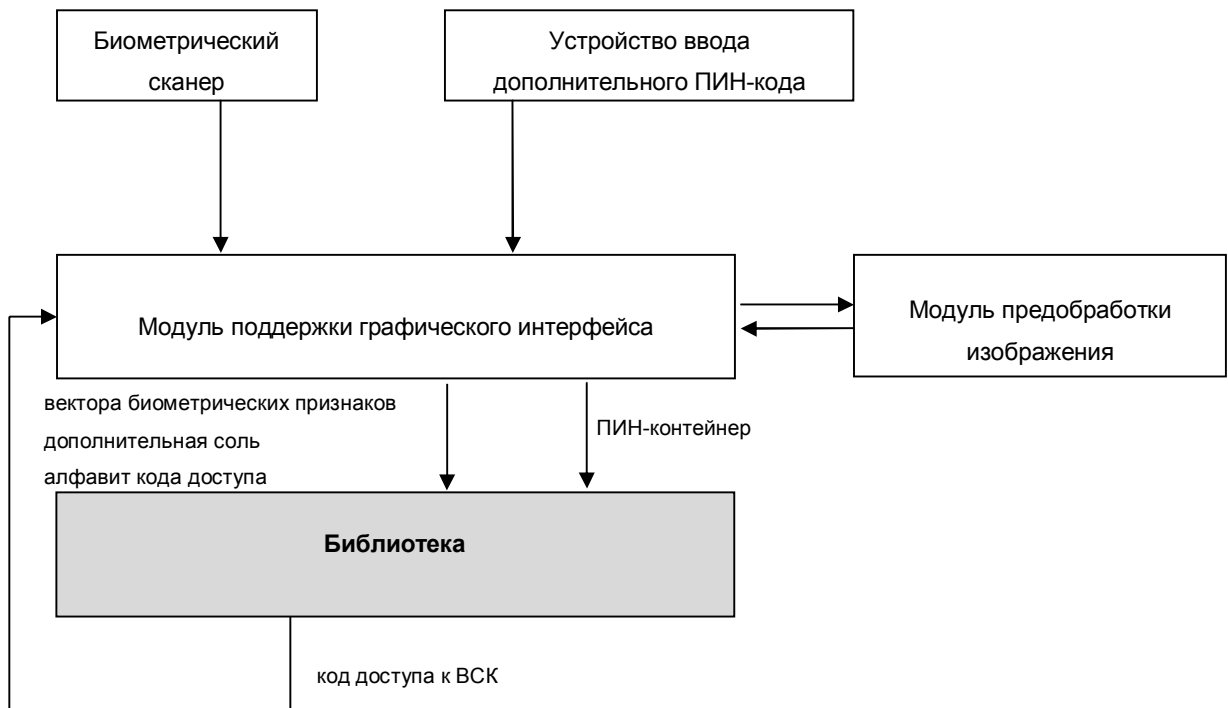


Рис. 2

1.1.5. Библиотека реализует следующие функции:

- инициализация ПБК;
- завершение работы с ПБК;
- создание ПИН-контейнера (далее – контейнера) с использованием переданных ПБК данных, в том числе ВБП пользователя;
- извлечение кода доступа из контейнера с использованием переданных ПБК данных, в том числе ВБП пользователя;
- сохранение контейнера в буфер;
- загрузка контейнера из буфера;
- установка дополнительных параметров работы ПБК, в том числе алфавита кода доступа;
- перекодирование символьного представления кода доступа во внутреннее представление для использования ПБК и обратно с использованием заданного алфавита.

1.2. Условия применения

1.2.1. Библиотека предназначена для использования на АРМ со следующими характеристиками:

Изм.	Лист	№ докум.	Подп.	Дата

- процессор: 11th Gen Intel Core i3 / Ryzen 5 или более производительный;
- свободная оперативная память: не менее 10 Мб;
- свободное место на диске: не менее 200 Кб.

1.2.2. Библиотека предназначена для работы в ОС Debian 13 в следующем программном окружении:

- ОС и ядро: Debian 13 (версия ядра Linux biodebian 6.12.73+deb13-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.12.73-1 (2026-02-17) x86_64 GNU/Linux);
- наличие установленных в системе библиотек в соответствии с таблицей 1.
- стандартные права доступа.

Таблица 1 – Список внешних зависимостей библиотеки

Имя	Версия	Лицензия
linux-vdso.so.1	1.	свободная
libstdc++.so.6	6.	свободная
libm.so.6	6.	свободная
libgcc_s.so.1	1.	свободная
libc.so.6	6.	свободная
ld-linux-x86-64.so.2	2.	свободная

2. ХАРАКТЕРИСТИКА БИБЛИОТЕКИ

2.1. Библиотека имеет следующие характеристики:

Примечание. Оценка указанных характеристик проводилась на базе данных, собранной и предоставленной заказчиком на момент оформления эксплуатационной документации; достижимость характеристик на других данных, собранных в других условиях, не гарантируется.

- длина одного ВБП – 128 вещественных значений;
- длина кода доступа – до 32 символов в кодировке UTF-8;

Примечание. Обеспечивается поддержка основного набора символов Unicode.

- длина соли – до 2048 байт;
- время создания контейнера – до 10 с;
- время одной попытки восстановления кода доступа – до 0,1 с;
- отсутствует ввод/вывод в стандартные потоки, файловый ввод/вывод, не применяются средства межпроцессорного взаимодействия;
- не используются механизмы выбрасывания исключений.

Изм.	Лист	№ докум.	Подп.	Дата

2.2. Для обеспечения максимального уровня безопасности и заявленных характеристик библиотеки при ее интеграции в средства аутентификации необходимо выполнять рекомендации раздела 5.

2.3. Контроль целостности библиотеки должен осуществляться внешним ПО.

3. ОБРАЩЕНИЕ К БИБЛИОТЕКЕ

3.1. Способ загрузки

3.1.1. Библиотека состоит из единственного файла *libnbcc_fp.so.1.0.0*.

3.1.2. Загрузку библиотеки выполняют с помощью стандартных средств ОС из каталога установки как общий объект (*so*). Например, в ОС Debian следует использовать функцию *dlopen*.

3.1.3. Обращение к библиотеке производят путем вызова функций ее программного интерфейса, описанных в разделе 4, после загрузки библиотеки. Текст заголовочного файла *nbcc_fp.h* библиотеки для программ, написанных на языке Си (C++), приведен в приложении 1.

3.2. Общая схема создания контейнера

3.2.1. Общая схема взаимодействия с библиотекой для создания контейнера, например, во время регистрации пользователя, может состоять из следующей последовательности действий:

- 1) для инициализации ПБК вызывают функцию *nbfp_open*;
- 2) с помощью функции *nbfp_set_param* устанавливают дополнительные параметры работы ПБК согласно подразделу 5.5, например, алфавит, если это необходимо;
- 3) задают уникальное значение соли, недоступное злоумышленникам;
- 4) задают эталонное значение кода доступа;
- 5) собирают необходимое число ВБП пользователя, полученных с разных изображений отпечатка пальца, например, 150, с учетом рекомендаций, изложенных в разделе 5;
- 6) вызывают функцию создания контейнера *nbfp_create*;
- 7) с помощью функции *nbfp_save_container* определяют размер буфера для контейнера и сохраняют его в буфер повторным вызовом этой функции;
- 8) завершают работу с ПБК вызовом функции *nbfp_close*;

Изм.	Лист	№ докум.	Подп.	Дата

3.2.2. На каждом шаге п. 3.2.1 необходимо анализировать возвращаемый функциями библиотеки код результата. Если получен код ошибки, то, в зависимости от ее значения, процедуру создания контейнера прерывают и используют альтернативный способ регистрации пользователя, либо запускают вновь после изменения тех или иных входных данных. При необходимости, код ошибки преобразуют в текстовый вид с помощью функции *nbfp_resultstring* и информируют о ней пользователя.

3.2.3. Создавать контейнер можно интерактивно. В этом случае, периодически, например, каждые 10 с вызывают функцию *nbfp_create*, передавая ей все накопленные к этому моменту ВБП, и анализируют код результата:

- при значении *NBFP_ERROR_INSUFFICIENT_DATA* сбор образов продолжают;
- при значении *NBFP_OK* сохраняют контейнера в буфер;
- решения по другим кодам принимают в соответствии с их описанием.

3.2.4. Пример кода на языке C++ для описанной схемы приведен в приложении 2.

3.3. Общая схема восстановления кода доступа

3.3.1. Общую схему взаимодействия с библиотекой для восстановления кода доступа во время одного сеанса аутентификации можно представить в виде последовательности шагов:

- 1) для инициализации ПБК вызывают функцию *nbfp_open*;
- 2) получают то же значение соли, которая использовалась во время регистрации;
- 3) с помощью функции *nbfp_load_container* загружают данные контейнера;
- 4) с помощью функции *nbfp_set_param* устанавливают дополнительные параметры работы ПБК согласно подразделу 5.5, например, алфавит, если это необходимо;

5) следующие действия выполняют в едином цикле до тех пор, пока не будет исчерпано допустимое число попыток, определенное внешним модулем или политикой безопасности подсистемы аутентификации, пока пользователь не отменит процедуру аутентификации, либо ПБК не вернет код ошибки, отличный от *NBFP_ERROR_INSUFFICIENT_DATA*:

- а) считывают очередное изображение отпечатка пальца;
- б) с помощью сторонних библиотек извлекают из него набор ВБП;
- в) вызывают функцию *nbfp_extract* для получения кода доступа;

Изм.	Лист	№ докум.	Подп.	Дата

г) если результат вызова функции не содержит ошибку, т.е. равен **NBFP_OK**, полученный код доступа передают внешнему модулю (TPM, серверу и т.д.) на проверку, в ходе которой подтверждают правильность восстановления кода доступа (ПИН-кода) или отбрасывают его как ошибочный;

б) завершают работу с ПБК вызовом функции **nbfp_close**.

3.3.2. Обратите внимание на то, что преобразователь биометрия-код не имеет механизмов проверки соответствия сформированного кода доступа эталону, поэтому значение **NBFP_OK**, возвращаемое функцией **nbfp_extract**, указывает только лишь на отсутствие ошибок при выполнении преобразования.

3.3.3. Пример кода на языке C++ для описанной схемы приведен в приложении 2.

4. ФУНКЦИИ ПРОГРАММНОГО ИНТЕРФЕЙСА

4.1. Функция **nbfp_resultstring**

4.1.1. Назначение. Получение текстового описания для кода результата, который возвращает функция программного интерфейса библиотеки.

Примечание. Не все функции библиотеки возвращают код результата.

4.1.2. Синтаксис – **const char* nbfp_resultstring(int res);**

4.1.3. Параметр – **res**, содержит код результата вызова функции интерфейса.

Примечание. Список возможных значений кода результата приведен в подразделе 5.6.

4.1.4. Возвращаемое значение. Текстовая строка с интерпретацией кода результата.

4.2. Функция **nbfp_open**

4.2.1. Назначение. Инициализация ПБК.

Примечание. Библиотека поддерживает работу только с одним экземпляром ПБК, поэтому повторный вызов функции до завершения работы с ПБК приведет к ошибке.

4.2.2. Синтаксис – **int nbfp_open();**

4.2.3. Возвращаемое значение:

- **NBFP_OK**, если ПБК создан и инициализирован;
- **NBFP_ERROR_INVALID_STATE**, если экземпляр уже создан;
- **NBFP_ERROR**, в случае ошибки создания или инициализации.

Изм.	Лист	№ докум.	Подп.	Дата

4.3. Функция `nbf_close`

4.3.1. Назначение. Завершение работы с ПБК. Освобождает и стирает используемую ПБК память.

Примечание. Функция должна обязательно вызываться до выгрузки библиотеки из оперативной памяти.

4.3.2. Синтаксис – `int nbf_close();`.

4.3.3. Возвращаемое значение – `NBFP_OK`.

4.4. Функция `nbf_clear`

4.4.1. Назначение. Сброс конфигурации и удаление хранимых значений. Переводит ПБК в начальное состояние.

4.4.2. Синтаксис функции – `int nbf_clear();`.

4.4.3. Возвращаемое значение – `NBFP_OK`.

4.5. Функция `nbf_set_param`

4.5.1. Назначение. Установка значения параметра работы ПБК.

Примечания:

1. Для правильного использования значения параметра в ПБК, он должен быть установлен после вызова функций `nbf_open`, `nbf_load_container` или `nbf_clear`, но перед вызовом функций `nbf_create` или первым вызовом `nbf_extract`.

2. Значения устанавливаемых параметров должны совпадать как во время создания контейнера, так и во время восстановления кода доступа.

4.5.2. Синтаксис – `int nbf_set_param(const char* name, int name_size, const char *value, int value_size);`.

4.5.3. Параметры:

- `name`, указатель на строку с именем параметра в кодировке UTF-8. Список допустимых имен параметров приведен в подразделе 5.5;

- `name_size`, длина строки с именем параметра без учета символа `'\0'` на конце, байт;

- `value`, указатель на буфер данных, содержащий значение параметра. Способ кодирования значения параметра, область допустимых значений описаны в подразделе 5.5;

- `value_size`, длина буфера данных, байт.

4.5.4. Возвращаемое значение:

Изм.	Лист	№ докум.	Подп.	Дата

- *NBFP_OK*, если значение параметра успешно установлено;
- *NBFP_ERROR_INVALID_STATE*, если ПБК не создан;
- *NBFP_ERROR_INVALID_ARG*, если параметр не установлен. Возможные причины отказа в установке значения параметра: неправильное имя параметра, неправильный способ кодирования значения параметра, выход за границы области допустимых значений.

4.6. Функция `nbfp_create`

4.6.1. Назначение. Выполняет обучение ПБК на обучающей выборке ВБП, эталонного кода доступа, дополнительного пароля и установленных параметров работы ПБК. В ходе обучения выполняется автоматическое тестирование качества обучения ПБК и проверка достижения целевых показателей качества. Результаты записываются в виде вектора.

4.6.2. Синтаксис – *int nbfp_create(int nfeat, const float* feats, int feats_size, const int* labels, const char* salt, int salt_size, const char* code, int code_size);*.

4.6.3. Параметры:

- *nfeat*, число биометрических признаков в одном примере биометрического образа. Допустимое значение – *128*;
- *feats*, указатель на вектора биометрических признаков обучающей выборки, объединенные путем конкатенации в один вектор;
- *feats_size*, общее число биометрических признаков в векторе, кратное *128*;
- *labels*, вектор целочисленных меток ВБП, которые интерпретируются в зависимости от параметра *label_type*: «*group*», если метка соответствует номеру группы, объединяющим ВБК, начиная с нуля; «*sample*», если метка соответствует порядковому номеру примера (изображения), начиная с нуля, из которого ВБП был получен. Вектора, не принадлежащие группам или примерам, должны иметь метку минус один;
- *salt*, указатель на дополнительную соль защиты контейнера, представленную в виде произвольной последовательности байт;
- *salt_size*, длина дополнительного пароля защиты, байт. Допустимые значения: от *0* до *65535*;

Изм.	Лист	№ докум.	Подп.	Дата

- *code*, указатель на блок данных, содержащий значение эталонного кода доступа. Допустимые значения блока данных: произвольная последовательность байт или строка в кодировке UTF-8, в случае, если задан алфавит преобразования;

- *code_size*, длина кода доступа, байт, без учета символа '\0' в конце UTF-8 строки.

4.6.4. Возвращаемое значение:

- *NBFP_OK*, если обучение ПБК завершено успешно и контейнер создан;

- *NBFP_ERROR*, если возникла системная ошибка или ошибка выделения памяти;

- *NBFP_ERROR_INVALID_DATA*, если предоставлены плохие данные для обучения и необходимо заменить обучающую выборку;

- *NBFP_ERROR_INSUFFICIENT_DATA*, если необходимо добавить ВБП в обучающую выборку;

- *NBFP_ERROR_INVALID_STATE*, если состояние ПБК не позволяет выполнять запрос на обучение;

- *NBFP_ERROR_INVALID_ARG*, если в функцию переданы недопустимые значения параметров;

- *NBFP_ERROR_TARGET*, если цели обучения не достигнуты. В этом случае рекомендуется заменить обучающую выборку или изменить дополнительные параметры работы ПБК;

- *NBFP_ERROR_PIN_ALPHABET*, если код доступа не может быть декодирован корректно с использованием установленного в ПБК алфавита.

4.7. Функция `nbfp_save_container`

4.7.1. Назначение. Сохранение параметров работы обученного ПБК во внешний буфер. Дополнительные параметры обучения или работы ПБК, одинаковые для всех пользователей системы и описанные в подразделе 5.5, например, алфавит, число потоков и т.д., кроме параметра *k*, в контейнере не сохраняются.

4.7.2. Синтаксис – `int nbfp_save_container(char* container, int csize);`.

4.7.3. Параметры:

- *container*, указатель на буфер для сохранения данных контейнера, выделенный вызывающей стороной;

- *csize*, размер буфера в байтах.

Изм.	Лист	№ докум.	Подп.	Дата

4.7.4. Возвращаемое значение:

- размер контейнера в байтах, если он успешно сохранен или *csize* содержит меньшее значение;

- *NBFP_ERROR_INVALID_STATE*, если ПБК не был обучен;
- *NBFP_ERROR_FORMAT*, если при сохранении в буфер возникла ошибка.

4.8. Функция `nbfp_load_container`

4.8.1. Назначение. Загрузка параметров работы обученного ПБК из внешнего буфера (контейнера).

4.8.2. Синтаксис – *int nbfp_load_container(const char* container, int csize);*.

4.8.3. Параметры:

- *container*, указатель на буфер, содержащий данные контейнера;
- *csize*, размер контейнера в байтах.

4.8.4. Возвращаемое значение:

- *NBFP_OK*, если данные контейнера успешно загружены в ПБК;

- *NBFP_ERROR_FORMAT*, если контейнер поврежден или имеет неподдерживаемый формат.

4.9. Функция `nbfp_extract`

4.9.1. Назначение. Восстановление кода доступа с помощью набора ВБП и дополнительной соли.

Примечания:

1. В текущей реализации предполагается, что число одновременно передаваемых на вход ВБП не меньше *min_parts* и не больше *64*.

2. При повторных вызовах предполагается, что возвращенное ПБК значением кода доступа оказалось неправильным.

4.9.2. Синтаксис – *int nbfp_extract(const float *feats, int feats_size, const char *salt, int salt_size, char *code, int *code_size);*.

4.9.3. Параметры:

- *feats*, указатель на набор ВБП одного примера отпечатка пальца;
- *feats_size*, длина ВБП. Должна быть кратна значению *128*;

Изм.	Лист	№ докум.	Подп.	Дата

- *salt*, указатель на значение соли, представленное в виде последовательности байт. Соль должна совпадать со значением, использованным во время создания контейнера. Значение соли должно быть неизвестно злоумышленнику;

- *salt_size*, длина значения соли, байт;

- *code*, указатель на буфер для извлечения кода доступа. Буфер должен быть целиком заполнен значениями '\0' перед вызовом, так как функция декодирования не добавляет '\0' в конце строки, чтобы обеспечить совместимость с вариантом, когда алфавит не задается;

- *code_size*, указатель на размер буфера, байт. Допустимое значение должно быть не меньше, чем $4 * nbfp_ncode()$. После корректного завершения работы функции *code_size* будет содержать фактическое число записанных в *code* байт.

4.9.4. Возвращаемое значение:

- *NBFP_OK*, если код доступа был восстановлен и записан в *code*. Прямая или косвенная проверка на соответствие кода эталону должна выполняться внешним ПО;

- *NBFP_ERROR_INSUFFICIENT_DATA*, если для выполнения преобразования недостаточно данных, например, если число векторов меньше *min_parts* или часть из них распознана как соответствующие одной и той же части;

- *NBFP_ERROR_ATTEMPT_LIMIT*, если лимит попыток восстановления исчерпан;

- *NBFP_ERROR_INVALID_STATE*, если состояние ПБК не позволяет выполнить восстановление, например, ПБК не обучен или контейнер не загружен из буфера;

- *NBFP_ERROR_INVALID_ARG*, если переданы недопустимые значения входных параметров (указатели, размерность, тип и т.д.);

- *NBFP_ERROR_PIN_ALPHABET*, если код не может быть декодирован с использованием установленного алфавита, либо размер буфера не позволяет записать результат;

- *NBFP_ERROR*, если возникла другая ошибка, например, ошибка выделения памяти, и нужно прервать процедуру восстановления.

4.10. Функция *nbfp_nfeat*

4.10.1. Назначение. Определение числа входных биометрических признаков, заданных во время обучения ПБК.

4.10.2. Синтаксис – *int nbfp_nfeat()* ;.

4.10.3. Возвращаемое значение – *128*.

Изм.	Лист	№ докум.	Подп.	Дата

4.11. Функция `nbfp_ncode`

4.11.1. Назначение. Определение длины кода доступа для обученного ПБК или после успешной загрузки контейнера.

4.11.2. Синтаксис – `int nbfp_ncode();`.

4.11.3. Возвращаемое значение:

- ноль, если длина кода не определена для текущего состояния ПБК;
- значение в диапазоне от **1** до **2048**, которое определяет:- число байт, занимаемое кодом доступа, если алфавит не задан; удвоенное число символов кода доступа, если алфавит задан.

Изм.	Лист	№ докум.	Подп.	Дата

5. ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

5.1. Описание входных и выходных данных

5.1.1. Библиотека предназначена для использования в составе средств аутентификации пользователя. Она не имеет собственных средств взаимодействия с пользователями, не взаимодействует с устройствами ввода данных, с биометрическими сканерами и подпрограммами предобработки сканируемых биометрических образов.

5.1.2. Входными данными для ПБК являются:

- эталонный код доступа (при создании контейнера);
- набор ВБП, формируемый из изображений одного и того же отпечатка пальца человека и предъявляемых им как во время создания контейнера, так и для восстановления кода доступа;

- дополнительная соль;

- контейнер, содержащий параметры ПБК (при восстановлении кода доступа).

5.1.3. Выходными данными для ПБК являются:

- код доступа (при восстановлении кода доступа);

- контейнер (при создании контейнера).

5.1.4. Эталонный код доступа должен формироваться вне библиотеки: пользователем, администратором или сторонними программно-аппаратными средствами таким образом, чтобы иметь достаточную стойкость к случайному угадыванию. Рекомендуется использовать коды доступа с длиной не менее шести символов.

5.1.5. Дополнительная соль должна формироваться вне библиотеки пользователем, администратором или сторонними программно-аппаратными средствами таким образом, чтобы иметь достаточную стойкость к случайному угадыванию. Рекомендуется использовать соль с эффективной длиной не менее **128** бит. Значение соли должно быть недоступно злоумышленнику.

5.1.6. ВБП и значение соли используются для связывания с кодом доступа. Для минимизации возможных векторов атаки на систему аутентификации ВБП и соль должны использоваться одновременно.

5.1.7. ВБП должны собираться и обрабатываться перед подачей в ПБК с учетом рекомендаций, изложенных в подразделах 5.2, 5.3. Выполнение приведенных рекомендаций является необходимым условием для достижения целевых характеристик работы ПБК.

Изм.	Лист	№ докум.	Подп.	Дата

5.1.8. Значение соли необходимо предъявлять в момент запроса на восстановление кода доступа и хранить в оперативной памяти до момента завершения запроса. Допустимо сохранять это значение в оперативной памяти на более длительный срок, например, в течение рабочего дня, но не дольше исчерпания лимита попыток на восстановление кода доступа.

5.1.9. При проектировании системы аутентификации должна быть предусмотрена возможность замены факторов аутентификации и контейнера пользователя. Замену контейнера необходимо выполнять после аутентификации пользователя с помощью библиотеки или альтернативным способом, например, вводом ПИН-кода. Так как сама процедура сбора биометрических данных занимает меньше 3 мин, то она не требует заметных усилий от пользователя. Рекомендуемая периодичность замены контейнера в типовом случае – один раз в 3 месяца.

5.1.10. Формирование контейнера должно происходить для конкретного АРМ. Контейнер должен храниться на АРМ, либо на доверенном сервере и загружаться на АРМ после идентификации АРМ сервером. В случае смены АРМ пользователем, контейнер должен быть удален с носителя информации (затерт произвольными последовательностями) и создан на новом АРМ заново.

5.2. Рекомендации по формированию обучающей выборки

Обучение ПБК допустимо выполнять с использованием наборов ВБП, полученных для одного пальца одного пользователя с помощью одного и того же метода предобработки примеров исходных биометрических данных. Для минимизации вероятности «отказа в регистрации» и достижения целевых показателей FAR, FRR разработчикам человеко-машинного интерфейса необходимо обеспечить поддержку перечисленных ниже возможностей ПО формирования обучающей выборки.

5.2.1. Сканирование биометрических данных необходимо выполнять на АРМ с тем же оборудованием и в тех же условиях, в которых планируется восстанавливать код доступа.

5.2.2. Рекомендуется осуществлять сбор отпечатков пальцев также, как это происходит в процессе аутентификации, т.е. последовательным прикладыванием и отрыванием пальца от поверхности сканера. Использование техники «проката» пальца по поверхности сканера не всегда гарантирует высокое качество связывания ВБП и кода доступа.

Изм.	Лист	№ докум.	Подп.	Дата

5.2.3. Минимальное число собираемых изображений, содержащих уникальные примеры ВБП, то есть отличающиеся друг от друга на расстояние не меньше *sq_point_dist* должно быть не менее **60** изображений. Общее число уникальных ВБП должно быть не меньше **80**.

5.2.4. Кластеризацию ВБП следует проводить автоматически ПБК или внешними средствами таким образом, чтобы число изображений, содержащих размеченные ВБП, было на уровне не менее **0.95**.

5.2.5. После завершения процедуры создания контейнера рекомендуется проверить качество распознавания и наработать навык предъявления биометрии, если он не был сформирован ранее. Для этого необходимо временно, на время от 3 до 5 с, покинуть место регистрации, а затем пройти процедуру тестового восстановления кода доступа из контейнера. Если несколько, например, пять следующих попыток будут успешными, то контейнер можно использовать по назначению.

5.2.6. Процедура регистрации занимает, в среднем, от 1 до 3 мин.

5.2.7. Если последующая эксплуатация созданного контейнера покажет, что он не обеспечивает приемлемый уровень FAR/FRR, должна быть предусмотрена возможность замены контейнера и смены предъявляемого пальца в любое удобное для пользователя время.

5.3. Рекомендации по прохождению процедуры восстановления кода

При реализации систем аутентификации, использующей библиотеку, разработчикам человеко-машинного интерфейса необходимо обеспечить поддержку перечисленных ниже возможностей ПО восстановления кода доступа.

5.3.1. Проходить процедуру восстановления кода доступа следует в условиях, близких к условиям регистрации кода доступа и с той же конфигурацией оборудования (на том же АРМ). Большинство попыток восстановления кода доступа должны быть успешными.

5.3.2. Ввод данных очередной попытки следует осуществлять явным отрывом и повторным касанием поверхности пальца.

5.3.3. При прикладывании пальца к поверхности сканера рекомендуется совмещать центральную часть («бугорок») отпечатка пальца и центр сканера.

Изм.	Лист	№ докум.	Подп.	Дата

5.3.4. Если несколько подряд идущих попыток восстановления кода доступа завершатся неудачно, следует предложить пользователю пройти процедуру аутентификации другим способом, например, с помощью пароля.

5.3.5. Общее число попыток доступное для пользователя должно быть не меньше суммы биометрических попыток и попыток аутентификации альтернативным способом.

5.3.6. Если фактор, влияющий на качество работы ПБК, не может быть устранен (отпечаток пальца стерт агрессивными средствами, травмирован или форма пальца изменена косметическими средствами/хирургическим путем), рекомендуется предложить пользователю пройти процедуру создания контейнера заново.

5.4. Требования к векторам биометрических признаков, их формату

5.4.1. Один ВБП должен представлять собой вектор вещественных чисел одинарной точности фиксированной длины (ГОСТ 52633.4–2011), например, **128**, полученный из одного примера биометрического образа «Свой» пользователя.

5.4.2. Для обеспечения стойкости контейнера к попыткам исследования каждый признак в ВБП должен быть нормирован и центрирован по базе «Все Чужие» по следующей формуле:

$$x'_i = (x_i - m_{all\ i})/s_{all\ i}, \quad (1)$$

где

x_i – значение i -го признака;

x'_i – значение i -го признака, модифицированного перед подачей в ПБК;

$m_{all\ i}$ – значение математического ожидания i -го признака, вычисленного на базе «Все Чужие»;

$s_{all\ i}$ – значение стандартного отклонения i -го признака, вычисленного на базе «Все Чужие».

5.4.3. Для ВБП, полученных со сверточных нейронных сетей, для которых уже выполнялась нормализация значений, допускается принимать значение стандартного отклонения признака за единицу.

5.5. Параметры ПБК

5.5.1. Библиотека позволяет задать ряд параметров работы ПБК в зависимости от потребностей использующих ее программ.

5.5.2. В таблице 2 приведен список параметров и допустимых значений. После таблицы дано развернутое описание для каждого дополнительного параметра.

Изм.	Лист	№ докум.	Подп.	Дата

Таблица 2 – Параметры работы ПБК и их допустимые значения

Имя	Когда	Тип, размер	Диапазон значений	По умолчанию	Назначение
<i>alphabet</i>	PB	char, до 2048	UTF-8 символы 0-й плоскости	null	Допустимый алфавит как последовательность символов или диапазонов в кодировке UTF-8
<i>k</i>	P	float, 1	[0, 255]	0	Коэффициент усложнения вычисления кода доступа
<i>max_parts</i>	P	float, 1	[1, 64]	64	Максимальное число групп (кластеров) в наборе ВБП, определяющих похожие ВБП
<i>min_parts</i>	P	float, 1	[1,max_parts]	64	Минимальное число ВБП пользователя, которые обеспечивают восстановление кода доступа
<i>min_group_size</i>	P	float, 1	[-1, 1000]	-1	Минимальный размер группы
<i>min_marked_part</i>	P	float, 1	[0, 1]	0.95	Минимальная часть ВБП, которая должна быть размечена
<i>label_type</i>	P	float, 1	sample, group	sample	Тип метки ВБП во время создания контейнера
<i>d_group_dist</i>	P	float, 1	[1,80]	22	Дискретный порог проверки схожести двух ВБП
<i>d_overlap</i>	P	float, 1	[0,1]	0.25	Минимальная часть ВБК одной группы, близкая к другой по <i>d_group_dist</i> , для их объединения
<i>sq_dist</i>	P	float, 1	[0, 1000]	0.95	Порог принятия решения Свой/Чужой при сравнении двух ВБП по расстоянию Евклида (квадрат значения)
<i>sq_point_dist</i>	P	float, 1	[0, 1000]	0.6	Квадрат расстояния объединения ВБП в группу
<i>sq_p2group_dist</i>	P	float, 1	[0, 1000]	0.6	Квадрат расстояния вторичного включения ВБП в группу

Изм.	Лист	№ докум.	Подп.	Дата

Продолжение таблицы 2

Имя	Когда	Тип, размер	Диапазон значений	По умолчанию	Назначение
<i>poly</i>	PВ	char, до 2048	UTF-8 символы 0-й плоскости	“80,7,5,4,3,2,1,0”	Строка, задающая степени примитивного полинома
Примечание. P – параметр может быть установлен во время регистрации; B – параметр может быть установлен во время восстановления; PВ – параметр может быть установлен во время регистрации, но это же значение должно использоваться во время восстановления.					

5.5.3. Параметр *alphabet*.

5.5.3.1. Параметр *alphabet* задает алфавит, необходимый для перекодирования кода доступа, если он представляет собой ПИН-код, состоящий из ограниченного набора символов. Эта операция обязательна для предотвращения атак «на открытый текст».

5.5.3.2. Алфавит должен представлять собой строку в кодировке UTF-8, содержащую допустимые символы или диапазоны допустимых символов, из которых может состоять код доступа. Строка может содержать любые разрешенные UTF-8 символы основной кодовой плоскости.

Примечание. Другие плоскости содержат неиспользуемые, на практике, для задания ПИН-кодов и паролей коды символов.

5.5.3.3. Для кодирования отдельных символов их нужно перечислить в строке алфавита в любом порядке. Важно учесть, что порядок перечисления имеет значение, поэтому одну и ту же строку нужно использовать как во время создания контейнера, так и во время восстановления кода доступа. В противном случае значение восстановленного кода будет отличаться от эталона. Примеры допустимых строк алфавита: «*0123456789*», «*9876543210*», «*abcdef1234*», «*эЮ-я!., [] {} ()*».

5.5.3.4. Для сокращения длины строки алфавита можно использовать диапазоны. Для кодирования диапазона значений в строку алфавита помещается последовательность из четырех символов вида «*[ab]*», где *a*, *b* - символы, которые окружены прямыми скобками, а их коды Unicode-символов расположены по возрастанию. Например, записи диапазонов «*[09]*», «*[az]*», «*[ая]*» являются допустимыми и соответствуют наборам «*0123456789*», «*abcdefghijklmnopqrstuvwxyz*», «*абвгдежзийклмнопрстуфхцщъзььэюя*». Записи «*[90]*», «*[a,b]*», «*[]*», «*[a]*», «*[]*»,

Изм.	Лист	№ докум.	Подп.	Дата

«**]]]]**», «**[!]**» не являются диапазонами и рассматриваются как последовательности символов.

5.5.3.5. При кодировании диапазонов следует обратить внимание на то, что символы «**ё**», «**Ё**» не включены в непрерывный диапазон символов «**[ая]**», «**[АЯ]**». Они должны быть добавлены в строку алфавита явно.

5.5.3.6. Примеры допустимых алфавитов, составленных из диапазонов: «**[09]**», «**[AZ][az][09]+/**», «**[ая]ё[АЯ]Ё**», «**[~]**».

5.5.4. Параметр `max_parts`.

5.5.4.1. Задаёт максимальное число групп ВБП, которое может использоваться для восстановления частей кода доступа.

5.5.4.2. При превышении этого значения, группы с минимальным числом элементов будут исключены из обучающей выборки.

5.5.5. Параметр `min_parts`.

Задаёт минимальное число ВБП из `max_parts` возможных частей, которые необходимы для корректного восстановления кода доступа.

5.5.6. Параметр `min_group_size`.

5.5.6.1. Задаёт минимальный размер группы для обучения. Рекомендуется использоваться не менее шести ВБП в каждой группе.

5.5.6.2. Группы, содержащие меньшее число элементов будут исключены из обучающей выборки.

5.5.7. Параметр `min_marked_part`.

5.5.7.1. Определяет минимальную часть примеров обучающей выборки, которая должна в обязательном порядке использоваться для настройки ПБК.

5.5.7.2. В случае, когда число ВБП задействованных для настройки ПБК групп, деленное на общее число ВБК окажется меньше значения параметра, ПБК вернет код `NBFP_ERROR_INSUFFICIENT_DATA`.

5.5.8. Параметр `label_type`.

5.5.8.1. Определяет тип меток ВБП, передаваемых в параметре `labels`.

5.5.8.2. Значение `"sample"` определяет вариант, когда каждому вектору обучающей выборки ставится в соответствие номер изображения (примера) из которого он получен. В этом случае кластеризация выполняется автоматически. Для ее настройки

Изм.	Лист	№ докум.	Подп.	Дата

следует использовать параметры *d_group_dist*, *d_overlap*, *sq_dist*, *sq_point_dist*, *sq_p2group_dist*, *min_group_size*, *max_parts*, *min_marked_part*.

5.5.8.3. Значение *"group"* определяет вариант, когда каждому ВБП обучающей выборки ставится в соответствие номер группы (значение минус 1, если ее нет). В этом случае преобразователь биометрия-код будет обучаться на указанных группах "как есть". В этом случае учитываются только значения параметров *min_group_size*, *max_parts*, *min_marked_part*.

5.5.9. Параметр *d_group_dist*.

Целочисленное значение, определяющее порог оценки близости ВБП одной группы к ВБП другой группы. Изменять не рекомендуется.

5.5.10. Параметр *d_overlap*

Вещественное значение, определяющее какая часть ВБП одной группы, которая должна находиться на расстоянии не больше *d_group_dist* для принятия решения об объединении этих групп.

5.5.11. Параметр *sq_point_dist*.

Параметр определяет квадрат расстояния между отдельными ВБП для принятия решения об их объединении в одну группу.

5.5.12. Параметр *sq_p2group_dist*.

Параметр определяет квадрат расстояния между неразмеченным ВБП и некоторой группой для принятия решения о включении ВБП в эту группу.

5.5.13. Параметр *poly*.

5.5.13.1. Задаёт строку с определением степеней примитивного полинома, который используется в схеме восстановления кода доступа по *min_parts* частям.

5.5.13.2. Строка должна представлять собой список целочисленных значений, разделённых запятыми, определяющих степени ненулевых членов полинома. Пробел между отдельными значениями использовать нельзя.

Пример: строка «80,7,5,4,3,2,1,0» задаёт полином $x^{80}+x^7+x^5+x^4+x^3+x^2+x^1+x^0$.

5.5.13.3. Максимальная степень полинома всегда **80**.

5.5.13.4. Значение по умолчанию: **"80,7,5,4,3,2,1,0"**.

Изм.	Лист	№ докум.	Подп.	Дата

5.6. Значения, возвращаемые функциями библиотеки

5.6.1. Все функции программного интерфейса библиотеки, если это не оговорено явно, возвращают код результата (ошибки) в соответствии с таблицей 3.

5.6.2. Возвращаемое значение каждой функции необходимо обработать, и, при необходимости, преобразовать в строковое представление с помощью функции *nbfp_resultstring*.

Таблица 3 – Допустимые значения кода результата

Значение	Сообщение
<i>NBFP_OK</i>	Операция выполнена успешно
<i>NBFP_ERROR</i>	Недостаточно памяти/прав на продолжение операции
<i>NBFP_ERROR_INVALID_STATE</i>	Недопустимый запрос для текущего состояния ПБК
<i>NBFP_ERROR_INVALID_ARG</i>	Неправильное значение аргументов функции или установленных параметров
<i>NBFP_ERROR_INVALID_ARG</i>	Неправильное значение аргументов функции или установленных параметров
<i>NBFP_ERROR_INVALID_DATA</i>	Недостаточное качество биометрических данных, рекомендуется заменить их другими
<i>NBFP_ERROR_INSUFFICIENT_DATA</i>	Недостаточно биометрических данных
<i>NBFP_ERROR_TARGET</i>	Целевые показатели ПБК не достигнуты. Измените обучающую выборку или параметры обучения
<i>NBFP_ERROR_FORMAT</i>	Неправильный формат данных, значение контрольной суммы
<i>NBFP_ERROR_ATTEMPT_LIMIT</i>	Превышено ограничение на число попыток извлечения кода
<i>NBFP_ERROR_PIN_ALPHABET</i>	Недопустимый символ алфавита, недостаточно места для кодирования символа

Изм.	Лист	№ докум.	Подп.	Дата

6. СООБЩЕНИЯ

6.1. Текстовые сообщения, соответствующие кодам результата работы функций библиотеки, приведены в таблице 4. Интерпретация кодов результата может меняться для отдельных функций, и указана в разделе 4.

Таблица 4 – Текстовые сообщения для кодов результата

Код результата	Сообщение	Тип
<i>NBFP_OK</i>	Операция выполнена успешно	Информация
<i>NBFP_ERROR</i>	Недостаточно памяти/прав на продолжение операции	Ошибка
<i>NBFP_ERROR_INVALID_STATE</i>	Недопустимый запрос для текущего состояния ПБК	Ошибка
<i>NBFP_ERROR_INVALID_ARG</i>	Неправильное значение аргументов функции или установленных параметров	Ошибка
<i>NBFP_ERROR_INVALID_DATA</i>	Недостаточное качество биометрических данных, рекомендуется заменить их другими	Ошибка
<i>NBFP_ERROR_INSUFFICIENT_DATA</i>	Недостаточно биометрических данных	Информация
<i>NBFP_ERROR_TARGET</i>	Целевые показатели ПБК не достигнуты. Измените обучающую выборку или параметры обучения	Ошибка
<i>NBFP_ERROR_FORMAT</i>	Неправильный формат данных, значение контрольной суммы	Ошибка
<i>NBFP_ERROR_ATTEMPT_LIMIT</i>	Превышено ограничение на число попыток извлечения кода	Ошибка
<i>NBFP_ERROR_PIN_ALPHABET</i>	Недопустимый символ алфавита, недостаточно места для кодирования символа	Ошибка

Изм.	Лист	№ докум.	Подп.	Дата

СОКРАЩЕНИЯ

АРМ	– автоматизированное рабочее место;
ВБП	– вектор биометрических признаков;
ВСК	– виртуальная смарт-карта;
ОС	– операционная система;
ПБК	– преобразователь биометрия-код;
ПИН	– персональный идентификационный номер;
ПО	– программное обеспечение;
FAR	– False Acceptance Rate;
FRR	– False Rejection Rate;
TPM	– Trusted Platform Module.

Изм.	Лист	№ докум.	Подп.	Дата

ЗАГОЛОВОЧНЫЙ ФАЙЛ NBCC_FP.H

```

#ifndef NBCC_FP_H
#define NBCC_FP_H

#ifdef __cplusplus
extern "C" {
#endif

//Список значений (кодов результатов), которые возвращают функции
//преобразователя биометрия-код, если это не оговорено явно:

#define NBFP_OK (0) //без ошибок
#define NBFP_ERROR (-1) //общая ошибка (системная
ошибка, память не выделена и т.д.)
#define NBFP_ERROR_INVALID_STATE (-2) //вызов функции невозможен для
текущего состояния
#define NBFP_ERROR_INVALID_ARG (-3) //неправильное или значение
аргументов (неправильная длина вектора, передан нулевой указатель)
#define NBFP_ERROR_INVALID_DATA (-4) //плохое качество биометрических
данных (нужно заменить другими)
#define NBFP_ERROR_INSUFFICIENT_DATA (-5) //требуется дополнительные
биометрические данные (недостаточно данных для обучения/восстановления)
#define NBFP_ERROR_TARGET (-6) //целевые значения характеристик
ПБК не достигнуты во время обучения
#define NBFP_ERROR_FORMAT (-7) //неправильный формат данных,
контрольная сумма и т.д.
#define NBFP_ERROR_ATTEMPT_LIMIT (-8) //превышено ограничение на число
попыток извлечения кода
#define NBFP_ERROR_PIN_ALPHABET (-9) //недопустимый символ алфавита,
либо недостаточно места для его кодирования

```

```

/*

```

Список параметров преобразователя биометрия-код для отпечатков пальцев

имя	когда*	тип	диапазон значений	по умолч.	назначение
alphabet	PB	char[]	UTF-8	null	допустимый алфавит
как набор символов или диапазонов			0 плоскость		в кодировке UTF-8 0-й
плоскости ("", "[09]", "[09][az][AZ]")					
k	P	float	[0,65535]	0	коэффициент
усложнения задачи вычисления ПИНа					
max_parts	P	float	[1,64]	64	максимальное число
групп,					которые
рассматриваются как независимые части секрета					

Изм.	Лист	№ докум.	Подп.	Дата

РИВУ.05439-01 33 01

Фактически, устанавливаемое значение не превышает число групп, найденных автоматически или заданных с помощью меток labels.

min_parts P float [1,maxparts] 0 минимальное число векторов, принадлежащих разным группам, которые необходимо предъявить для правильного восстановления

ПИНа

label_type P char[] {'sample','group'} 'sample' тип меток для векторов биометрических параметров
 Значение 'sample' определяет вариант, когда каждому вектору обучающей выборки ставится в соответствие номер изображения (примера) из которого он получен. В этом случае кластеризация выполняется автоматически. Для ее настройки учитываются параметры d_group_dist, d_group_overlap, sq_dist, sq_point_dist, sq_p2group_dist, min_group_size, max_parts, min_marked_part

Значение 'group' определяет вариант, когда каждому вектору обучающей выборки ставится в соответствие номер кластера (значение -1, если его нет). В этом случае преобразователь биометрия-код будет обучаться на указанных группах "как есть".
 Учитываются параметры min_group_size, max_parts, min_marked_part.

min_group_size P float [-1,1000] -1 минимальный размер группы
 Все найденные группы, содержащие меньшее число векторов, будут отброшены.

min_marked_part P float [0,1] 0.95 минимальная часть векторов, которая была сгруппирована и используется в обучении (иначе ошибка NBFP_ERROR_INSUFFICIENT_DATA)

d_group_dist P float [1,79] 22 значение, определяющая порог оценки близости векторов одной группы к векторами другого

d_group_overlap P float [0,1] 0.25 часть векторов группы, которая должна совпадать с другой группой для

принятия решения об их объединении
 sq_dist P float [0,1000] 0.95 порог принятия решения Свой/Чужой при сравнении

двух векторов по расстоянию Евклида
 sq_point_dist P float [0,1000] 0.6 квадрат расстояния объединения точек в группу
 При изменении значения sq_point_dist, значение sq_p2group_dist также устанавливается в sq_point_dist

sq_p2group_dist P float [0,1] 0.6 квадрат расстояния вторичного включения отдельной точки в группу

Изм.	Лист	№ докум.	Подп.	Дата

РИВУ.05439-01 33 01

poly P char[] UTF-8 определение
 собственного примитивного полинома в виде строки
 Строка представляет собой список целочисленных значений, через
 запятую, определяющих степени ненулевых членов.

Максимальная степень полинома всегда 80.

Пример: '80,7,5,4,3,2,1,0' задает полином
 $x^{80}+x^7+x^5+x^4+x^3+x^2+x^1+x^0$.

Обратите внимание, что пробел между числами не ставится!

```
-----
*когда можно/нужно устанавливать:
P - во время регистрации,
B - во время восстановления кода,
PB - в обоих случаях и должны иметь одинаковое значение
Все параметры являются опциональными и могут не устанавливаться.
Для сброса значений можно использовать nbfp_clear.
*/

#define NBCC_FP_EXPORT __attribute__((visibility ("default")))

//Получение текстового описания ошибки (результата)
NBCC_FP_EXPORT const char* nbfp_resultstring(int res);

//Создание и инициализация преобразователя биометрия-код (ПБК)
//примечание:
// Допускается работа только с одним экземпляром ПБК в программе
//возвращает:
// NBFP_OK успешно
// NBFP_ERROR_INVALID_STATE экземпляр уже создан
// NBFP_ERROR ошибка выделения памяти и инициализации
NBCC_FP_EXPORT int nbfp_open();

//Завершение работы с преобразователем биометрия-код
//возвращает: NBFP_OK
NBCC_FP_EXPORT int nbfp_close();

//Сброс конфигурации и удаление хранимых значений
//примечание:
// Преобразователь биометрия-код возвращается в начальное состояние
//возвращает: NBFP_OK
NBCC_FP_EXPORT int nbfp_clear();

//Установка значения параметра ПБК
//примечание:
// Параметр должен устанавливаться после после
nbfp_open/nbfp_load_container/nbfp_clear,
// но до вызова nbfp_create или первого вызова nbfp_extract.
// Значения параметров {PB} должны быть одинаковы как во время создания
контейнера,
```

Изм.	Лист	№ докум.	Подп.	Дата

РИВУ.05439-01 33 01

```

// так и во время восстановления кода.
//параметры:
// name      - строка с именем параметра
// name_size - длина строки параметра в байтах (без учета символа \0)
// value     - указатель на значение параметра (например на float, float[]
или строку utf-8)
// value_size- длина значения в байтах
//возвращает:
// NBFP_OK           параметр установлен
// NBFP_ERROR_INVALID_ARG  параметр не установлен (неправильное имя,
формат, значение, состояние ПБК)
NBCC_FP_EXPORT int nbfp_set_param(const char* name, int name_size, const char
*value, int value_size);

//Создание контейнера
//параметры:
// nfeat      - число биометрических признаков в одном примере биометрического
образа.
//           допустимые значения: 128
// feats     - плоский набор векторов биометрических признаков обучающей
выборки
// feats_size - общее число биометрических признаков в векторе (nsamp*128)
// labels    - вектор целочисленных меток (для каждого вектора ровно 1 метка)
(nsamp),
//           которые интерпретируются в зависимости от параметра labeltype:
//           "group" - метки соответствуют номерам групп, объединяющим вектора,
начиная с 0
//           "sample" - метки соответствуют порядковым номерам примеров
(изображений), начиная с 0, из которых они были получены
//           вектора, не принадлежащие группам или примерам, должны иметь метку -
1.
// salt     - дополнительная соль защиты, произвольная последовательность
байт.
// salt_size - длина дополнительной соли защиты, 0..65536.
// code     - выходной код, задается во время обучения
//           Может представлять собой строку в кодировке UTF-8, если задан
алфавит преобразования.
// code_size - длина выходного кода, байтах.
//возвращает:
// NBFP_OK           контейнер создан
// NBFP_ERROR        ошибка выделения памяти, внешняя ошибка
// NBFP_ERROR_INVALID_DATA  плохие данные для обучения, необходимо
выполнить повторный сбор данных
// NBFP_ERROR_INSUFFICIENT_DATA  нужно больше данных для обучения (добавить
в обучающую выборку и повторить запрос)
// NBFP_ERROR_INVALID_STATE  состояние не позволяет выполнять обучение
// NBFP_ERROR_INVALID_ARG    недопустимые значения параметров, число,
порядок, формат
// NBFP_ERROR_TARGET        цели обучения не достигнуты
//                           (можно снизить требования к качеству
обучения или заменить обучающую выборку)
// NBFP_ERROR_PIN_ALPHABET   выходной код code не может быть декодирован
с использованием установленного алфавита

```

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

```

NBCC_FP_EXPORT int nbfp_create(int nfeat,          //число признаков в одном
векторе
    const float* feats, int feats_size,          //плоский набор всех примеров
    const int* labels,                          //метки примеров
    const char* salt, int salt_size,            //соль
    const char* code, int code_size);          //выходной код

//Восстановление выходного кода
// feats - плоский набор векторов признаков
// feats_size - общее число признаков (например, 128, 128*3)
// salt - дополнительная соль защиты, произвольная последовательность
байт.
// salt_size - длина дополнительной соли защиты, 0..65536.
// code - буфер для извлечения выходного кода
// code_size - [вх] размер буфера в байтах, [вых] фактическая длина кода
//возвращает:
// NBFP_OK код был восстановлен и записан в code
// (проверка на соответствие кода эталону должна
выполняться во внешнем ПО)
// NBFP_ERROR_INVALID_STATE состояние ПБК не позволяет выполнить
восстановление
// (ПБК не обучен или контейнер не загружен из
буфера)
// NBFP_ERROR_INSUFFICIENT_DATA недостаточно данных (допускается следующая
попытка вызова)
// NBFP_ERROR_ATTEMPT_LIMIT лимит попыток восстановления кода исчерпан
// NBFP_ERROR_INVALID_ARG недопустимые входные параметры (указатели,
размерность, тип и т.д.)
// NBFP_ERROR другая ошибка, нужно прервать процедуру
восстановления
// NBFP_ERROR_PIN_ALФАВЕТ код не может быть декодирован с
использованием установленного алфавита
// или размер буфера не позволяет записать
результат
NBCC_FP_EXPORT int nbfp_extract(
    const float *feats, int feats_size,
    const char *salt, int salt_size,
    char *code, int *code_size);

//Сохранение контейнера в буфер
//примечание: данные контейнера копируются в буфер только при условии
достаточной его длины
//возвращает:
// длина контейнера >=0
// NBFP_ERROR_INVALID_STATE преобразователь биометрия-код должен быть
сначала успешно обучен
// NBFP_ERROR_FORMAT неправильный формат или неподдерживаемые
параметры для сохранения
NBCC_FP_EXPORT int nbfp_save_container(char* container, int csize);

```

Изм.	Лист	№ докум.	Подп.	Дата

РИВУ.05439-01 33 01

```
//Загрузка контейнера из буфера
//возвращает:
//  NBFP_OK
//  NBFP_ERROR_FORMAT      контейнер поврежден или имеет неподдерживаемый
формат
NBCC_FP_EXPORT int nbfp_load_container(const char* container, int csize);

//Число входных биометрических признаков
//возвращает:
//  128
NBCC_FP_EXPORT int nbfp_nfeat();

//Определение длины выходного кода для загруженного контейнера/обученного ПБК
//примечание: размер буфера для code должен быть не меньше этого значения * 4
//возвращает:
//  >0  длина в байтах;
//  0   если значение не определено для текущего состояния ПБК
NBCC_FP_EXPORT int nbfp_ncode();

#ifdef __cplusplus
}
#endif

#endif // NBCC_FP_H
```

Изм.	Лист	№ докум.	Подп.	Дата

ПРИМЕР ИСПОЛЬЗОВАНИЯ ПРОГРАММНОГО ИНТЕРФЕЙСА

1. Пример написан на языке C++ и состоит из файлов:

- *mtypes.h*;
- *mtypes.cpp*;
- *nbcc_fp_emulator.h*;
- *main.cpp*.

2. Пример демонстрирует следующие варианты использования программного интерфейса библиотеки:

- инициализация и завершение работы библиотеки;
- установка параметров обучения ПБК;
- создание контейнера с использованием обучающей выборки, загружаемой из файла *finger* (приложение 3);
- сохранение параметров преобразования из ПБК в контейнер;
- загрузка параметров преобразования из контейнера в ПБК;
- восстановление ПИН для всех примеров обучающей выборки и оценка FRR.

3. Для запуска примера скопируйте тексты примера в одну папку, добавьте к ним заголовочный файл библиотеки *nbcc_fp.h* и соберите его с помощью стандартного компилятора g++. Затем скопируйте в каталог программы файл *finger* с диска с документацией или используйте собственный файл с образцами ВБП одного пальца. Запустите программу.

4. Содержимое файла *mtypes.h*:

```
#ifndef MTYPES_H
#define MTYPES_H

#include <iostream>
#include <iomanip>
#include <chrono>
#include <string>
#include <sstream>
#include <vector>
#include <chrono>
#include <algorithm>

#define m_unused(X) (void)X

typedef std::vector<unsigned char> bytes;
typedef std::vector<float> features;
```

Изм.	Лист	№ докум.	Подп.	Дата

```

typedef std::vector<int>          vec_int;
typedef std::vector<float>      vec_float;

namespace m {
    using namespace std;

    template<typename Ta, typename Tb>
    ostream& operator<<(ostream& f, const pair<Ta,Tb> &v){
        return f<<'('<<v.first<<' '<<v.second<<')';
    }

    template<typename T>
    ostream& operator<<(ostream& f, const vector<T> &v){
        for (size_t i=0; i<v.size(); ++i) f<<v[i]<<' ';
        return f;
    }

    //Проверка нахождения значения среди списка значений
    //возвращает: 0 (false), если значение не найдено
    //            i, если элемент найден в (i-1) позиции
    template<typename T>
    inline int contain(const std::initializer_list<T> &vals, const T &val){
        auto it = std::find(vals.begin(),vals.end(),val);
        return (it==vals.end())? 0 : (1+(it-vals.begin()));
    }

    template<typename T>
    inline int contain(const std::vector<T> &vals, const T &val){
        auto it = std::find(vals.begin(),vals.end(),val);
        return (it==vals.end())? 0 : (1+(it-vals.begin()));
    }

    //Проверка нахождения значения в интервале [a,b)
    //возвращает: 0 (false), если значение находится вне диапазона
    //            1 (true), если значение входит в диапазон
    template<typename T>
    inline bool contain(const T &a, const T &b, const T &x){
        return (a<=x) && (x<b);
    }

    typedef unsigned char uchar;
    typedef vector<string> stringlist;

    //Разделение строки на несколько с помощью делителя
    stringlist split_string(const string &s, char ch);

    struct mstimer {
        std::chrono::system_clock::time_point _tp;
        uint64_t _ms;

        mstimer(){ start(); }

        void start(){ _tp = std::chrono::system_clock::now(); }
        void stop() {
            _ms = std::chrono::duration_cast<std::chrono::milliseconds>(

```

Изм.	Лист	№ докум.	Подп.	Дата

```

        std::chrono::system_clock::now().time_since_epoch() -
        _tp.time_since_epoch()
        ).count();
    }

    operator uint64_t() const { return _ms; }
};
} //end namespace
#endif // MYPES_H

```

5. Содержимое файла *mtypes.cpp*:

```

#include "mtypes.h"

namespace m {

stringlist split_string(const string &s, char ch) {
    stringlist v;
    size_t pred = 0, pos = 0;
    for (; (pos=s.find(ch,pos))!=string::npos; ++pos, pred=pos){
        v.push_back(s.substr(pred,pos-pred)); //не пустая ||||
        if (v.back().empty()) v.pop_back();
    };
    v.push_back(s.substr(pred,s.size()-pred));
    if (v.back().empty()) v.pop_back();
    return v;
}

}

```

6. Содержимое файла *nbcc_fp_emulator.h*:

```

#ifndef NBCC_FP_EMULATOR_H
#define NBCC_FP_EMULATOR_H

#include "mtypes.h"
#include "nbcc_fp.h"

using namespace std;

//Интерфейс класса, выполняющего сравнение сразу по нескольким векторам,
//независимо полученных из частей биометрического образа (фреймов).
class cluster_matcher {
protected:
    int    _nfeat;
    string _fit_rep;
    string _match_rep;
public:
    cluster_matcher(int nfeat):_nfeat(nfeat){};
    virtual ~cluster_matcher(){}

    virtual bool fit(const features &feats, const vector<int> &labels, const
bytes &salt, const bytes &pin)=0;
    virtual bool extract(const features &feats, const bytes &salt, bytes
&pin)=0;

```

Изм.	Лист	№ докум.	Подп.	Дата

PIBY.05439-01 33 01

```

virtual bool set_param(const string &name, const char* value, int
value_size)=0;

bool set_param(const string &name, const float &value) { return
set_param(name, (const char*)&value, sizeof(value)); }
bool set_param(const string &name, const string &value) { return
set_param(name, value.data(), value.size()); }

virtual void clear()=0;

virtual bool match(const features &feats, const bytes &salt, const bytes
&pin){
    bytes pin2;
    return (extract(feats, salt, pin2) && (pin==pin2));
}

virtual string fit_report() const { return _fit_rep; }
virtual string match_report() const { return _match_rep; }
};

//Эмулятор библиотеки для отпечатка пальца
class nbcc_fp_emulator : public cluster_matcher {
protected:
    int _res;
public:
    enum {
        FeatsCount=128
    };

    nbcc_fp_emulator():cluster_matcher(FeatsCount){
        _res = nbfp_open();
    }
    ~nbcc_fp_emulator(){
        _res = nbfp_close();
    }

    virtual bool fit(const features &feats, const vector<int> &labels, const
bytes &salt, const bytes &pin){
        _res = nbfp_create(FeatsCount, feats.data(), feats.size()
, labels.data()
, (const char*)&salt.data(), salt.size()
, (const char*)&pin.data(), pin.size());
        return (_res==NBFP_OK);
    }

    virtual bool extract(const features &feats, const bytes &salt, bytes
&pin){
        //Выделить достаточный буфер для кода (4 символа на код максимум)
        pin.resize(nbfp_ncode()*4);
        int pin_size = pin.size();

        _res = nbfp_extract(feats.data(), feats.size()
, (const char*)&salt.data(), salt.size()
, (char*)&pin.data(), &pin_size);
    }
};

```

Изм.	Лист	№ докум.	Подп.	Дата

```

        if (_res==NBFP_OK) pin.resize(pin_size); //Изменить длину по
фактической коду
        return (_res==NBFP_OK);
    }

    bool save(bytes &container){
        container.clear();
        _res = nbfp_save_container(0,0);
        if (_res>0){
            container.resize(_res);
            _res = nbfp_save_container((char*)container.data(),
container.size());
        }
        return (_res>=0);
    }

    bool load(const bytes &container){
        _res = nbfp_load_container((const char*)container.data(),
container.size());
        return (_res==NBFP_OK);
    }

    virtual void clear(){
        _res = nbfp_clear();
    }

    virtual bool set_param(const string &name, const char* value, int
value_size){
        return _set_param(name,value,value_size);
    }

    bool set_param(const string &name, const string &value){
        return _set_param(name,value.data(),value.size());
    }
    bool set_param(const string &name, const float &value){
        return _set_param(name,(const char*)&value,sizeof(value));
    }

    int error() const { return _res; }
    int nfeat() const { return nbfp_nfeat(); }
    int ncode() const { return nbfp_ncode(); } //общая длина дополнения
(длина кода может быть меньше)
    int nresp() const { return 80; }

protected:
    inline bool _set_param(const string &name, const char* value, int
value_size){
        _res = nbfp_set_param(name.data(),name.size(),value,value_size);
        return (_res==NBFP_OK);
    }
};

#endif // NBCC_FP_EMULATOR_H

```

Изм.	Лист	№ докум.	Подп.	Дата

7. Содержимое файла *main.cpp*:

```

#include <iostream>
#include <fstream>
#include <vector>
#include <memory>
#include "nbcc_fp_emulator.h"

using namespace std;
using namespace m;

long size_of_file(const string &filename){
    ifstream f(filename, ios::binary | ios::ate);
    return f.tellg();
}

template<typename T>
static bool read_file(const string &filename, vector<T> &v){
    size_t size = size_of_file(filename);
    if (!size || (size % sizeof(T)) || size>0x7FFFFFFF) return false;
    ifstream f(filename,ios_base::binary|ios_base::in);
    v.resize(size/sizeof(T));
    f.read((char*)v.data(),size);
    return !f.eof();
}

bytes string_to_bytes(const string& value){
    return bytes(value.begin(),value.end());
}

struct new_metrics {
    int tst_match;
    int tst_total;
    int all_match;
    int all_total;

    float frr() const { return (tst_total) ? (1-(float)tst_match/tst_total) : -1.0f; }
    float far() const { return (all_total) ? (float)all_match/all_total : -1.0f; }
};

int report_error(const shared_ptr<nbcc_fp_emulator> &nb, const string &msg){
    cout<<"error:"<<msg;
    if (nb){ cout<<" ("<<nb->error()<<")"; }
    cout<<endl;
    return -1;
}

int main(){
    shared_ptr<nbcc_fp_emulator> nb = make_shared<nbcc_fp_emulator>();
    if (!nb) return report_error(nb,"create share");

    vector<float> samples;

```

Изм.	Лист	№ докум.	Подп.	Дата

PIBY.05439-01 33 01

```

    if (!read_file("finger",samples)) return report_error(nb,"load finger
data");

    size_t feat_count    = 128;
    size_t sample_count  = samples.size()/feat_count;
    cout<<"sample_count="<<sample_count<<endl;
    if (sample_count<10) return report_error(nullptr,"sample count");

    cout<<string(30,'-')<<"FIT"<<string(30,'-')<<endl;

    bytes salt= string_to_bytes("21w9-!kdur8yFf");
    bytes pin  = string_to_bytes("123456");
    bytes container;
    vector<int> labels(sample_count,-1); //разметка не требуется

    nb->set_param("alphabet","[09]");
    nb->set_param("min_parts",1);
    nb->set_param("min_marked_part",0.8f);
    nb->set_param("label_type","sample");
    nb->set_param("sq_point_dist",0.6f);
    nb->set_param("sq_p2group_dist",0.6f);
    if (!nb->fit(samples,labels,salt,pin)) return report_error(nb,"fit");
    if (!nb->save(container)) return report_error(nb,"save");

    cout<<string(30,'-')<<"EXTRACT"<<string(30,'-')<<endl;

    if (!nb->load(container)) return report_error(nb,"load");
    if (!nb->set_param("alphabet","[09]")) return
report_error(nb,"alphabet");

    new_metrics met = {0,0,0,0};
    mstimer tic;
    for (size_t i=0; i<sample_count; ++i){
        cout<<setfill('0')<<setw(5)<<i<<" : ";
        bytes pin2;
        tic.start();
        vector<float>
sample(samples.begin()+i*feat_count,samples.begin()+(i+1)*feat_count);
        if (!nb->extract(sample,salt,pin2)) return
report_error(nb,"extract");
        tic.stop();
        cout<<pin2<<' ' <<tic<<"ms ";
        if (pin2==pin) met.tst_match++;
        else          cout<<"FRR";
        cout<<endl;
        met.tst_total++;
    }

    cout<<string(80,'-')<<endl;
    cout<<"TOTAL: count="<<met.tst_total<<" frr="<<met.frr()<<endl;

    return 0;
}

```

Изм.	Лист	№ докум.	Подп.	Дата

КОНТРОЛЬНЫЕ ДАННЫЕ

Данные файла **finger** (размер 20 992), содержащего обучающую выборку из 41 ВБП по 128 признаков в шестнадцатеричном формате:

```
d7f6fe3c40e1823d229319bbfaf911bab9e470bd5b5b18bc70b17a3dda18
d03ca24bb0bd99bfb23dae893cbdf7f3f4bbc9d364bd1dd7ac3d201c5a3d
0263f2bc748800bccd748ebef6482dbc8770903cabf62e3da0575f3b85ea
92bd31a5eebc89e85dbb397ee83d08ae9dbd8da3d73da1f5353e8812b33d
77ef71bdaddc5fbd331e0a3eef5b19bd0c1d42bd986cea3de83c0abdc54d
033e25e41d3ee2be6abd6cda963dff2364bdbc8025bdac66aa393ab44abd
0244883dc45e0f3e33ddbdbc039372bd415cc7bd66b0df3d264ae43d7dfc
743d0b64983da0961fbd2085323d1819d2bdb1da06bd5d65fe3db2fa143c
1320f6bb6e2ce9bda137a33b5c7b06bed4c09a3d64c001bd522f87bc7ed1
16be16f9fab5223833c6c0f0e3e780b58bd45eceebede095973df0ec393e
bb006e3d97ef51bd922adc3d2baa90bdc7cfe1bdef97d6bc8140a0bdcd71
673d3d29c93ded6381bdbc26043d6f33503d347fb43ce3c8abbd00b931be
5bc9d1bdc8a53ebd03d581bb77fa083df1f3523d35a357be5f80aa3d8c3c
733e4a66e23c717f6cbd5f3554bd8ace1d3eb7c7083d37edb5bda8cc8bbc
30ab403e2fd6793c231c533e706f9ebded7d71bc48285abd82cfebfcd244
043eb67712bdb53309be9a2a95bdd37edf3c8b90a6bd26fdf2bd2a70e53d
d9e627bb876cd2bdbe079dbdd3b9363d6fba f83cb5990e3d00234e3dc3db
033eeaa2ed3d82c52f3c7138d13d99c594bdae86f73ceae9ed3d100c6dbd
1553043ef2c756bd51080a3dd70c683d1f4234bd26e492bdd24bbe3dbea9
373e872f123e82e74c3da5df883a5da685bc60bfbfd3ccb2b6cbd2296a93d
6710e23da70c5abc429ede3dea2fd2bdf49378bdc449cb3cf09a06bc83c4
2d3db4ecdf3c0e8a34bd0bc8d3bda55e913cb2791fbef2f7ae3cfc21883d
0dbaf7bc41f7a73d962a9e3cb4402b3ecfafcb3bf4ecd03dfd17273d4464
3fbebab16ebdeda525be8345713df828ff3d848dedbcf674cb3dfb15c6bb
43a8533dcc0b7c3d9f9c40bbac46bd3d0f72a4bc3735cc3d9d95e53da959
9a3dea989bbddc4387bc120976bda77958bda4c1ea3d6488f8bd9d192bbe
bfd56a3df635983d914065bd88f6d53d969a6ebcdcccfbc42cefd3d2482
963dca93063d0627463dd081f3bdbc92ac3d1f92533dcfafeebd4213fd3d
5fa313be00f4113def030d3e3744d33d29b1a5bd16e0173da73b08be1e38
95bd7e1b94bd74dea1bd8f4df33c592c70bc188d073e56e3bc3c9f752b3d
23bb1a3e14427abd370a393ec65184bcf31f53bd7f339dbd983e77bd9ca1
b83de0a68e3ca10222bd5b64863c959da4bc24ac0f3d436853bccca33343c
53c449bdeb3c48bd51cfa53c6a2161bccd0207be23f5b83da2a5fd3cd285
793e5081fb3d58124d3cd89c5abd1e7acebd123b923d9c95a83d0f6b063e
d6c37dbe0244adb66deccbd8131213d54f06bbd6e1b383cd356fbbdc599
59bcf6df6c3d4acf12be3fb9b43c8c1ddfb9a4819bd97a55e3d0a02a53c
3d0810be885decdbd840a9a3d0393f23d3b59273deb181dbe238ac83d7690
c23dfd5176bd7f3eb73dd03a063ea71aafbd616d7cbe7690a23b8e532d3e
5e428dbd718a21bd5f1dd9bc127d13bdc7ee873de1db3e3df626943cedee
91bdf4fec0bd638fc83d5ae402bdbfdda93c6d163c3cf2fe1dbe6c3bf13d
67fc0dbdcc3def3de8d803be725c5a3eb1e925bd573cd4bcef49f2bc724e
403e506c813d8c6f423e487deebc70989d3d67f2b2bdbc192fbd640dea3b
120f91bd21c27ebde7af193d8e6b05bdefbf7abd34e29bbd45e3ba3d283b
5cbe0128033d25e58abdfb77ecbd5a2842bdc23c0c3d2a50873d0663473e
17bd0d3d553234bd67bf2bbcef0a8d3d09be2abc8925593d59b344bdd673
623de0ee27bd7723c2bc496fa6bdfba1dabc5327a4bd6f0e0d3c416cf13c
```

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

183140bd5a5369bcf89bc6bdba99b53d27bf343db372183d00a88dbd623c
9bbe1c1b933d596c81bd04511d3edb292abed1e394bc2c5aabbc0ad97c3d
238989bdb0a3543b6e1c8abb900df3bc360f783b456e77bcc40e2f3de4f0
4cbb3d5cbb3c7905b8bc21ecf83ca527a83ccac80b3db8e7f5bccedc433b
4df241be240f87bdbb76033aa7e852bd1e9102be7159683c234c003d324c
853da13c26be1f2f1f3e2ba15abd88cffbbcb1670ebe8983f9bb2121ad3d
209671bd72ee6a3dc42ee7bd2f81a43df26f44bd6b58a7bd7ff82bbe1290
5b3ea8bedbbc6d8ca33dcd50163dbd1cc9bc571bab3dee3b8b3d7b67df3c
aa6a453dfdd7c93a121c1c3d9da4553e8da432bd381bd4bc81f7043c33f4
213cc074af3bb90c063d0edd3fbd25c11fbc31de033e1532433d7df6173d
5ab681bc9a28533d8a81a23da756053eca98e73da75be33d1bf8d1bd1d73
903c3f538a3dabbfb5e3c2418093c3e2bebbcb163d83df32ad03c6fc6ad3d
a201013ef1c1b73d4382463edf317c3d6dec083de8e9ecbc14a0c9bd3d69
143ed081cfbd99b303bd963d9ebd825305be2733333d5f39343d5cc9e6bd
aa7fa7bdc4dcc0bb34e5aa3c1d2ac6bcae21783eb231c03d74b096bcc2f3
0f3ee257e53d1b9c12bead44f13d9e34a2bcad3c923da22bb43b583c5b3d
822113bde5b6c83ddc6c963d1a26a9bcf3be813d6536cfbdadfc23c63bd
683d7b532bbd9551393e0f989c3dc8c36b3d7e0596bd8e74203c1257243c
d0814fbdcef0603d5878fbbc6f62eb3d71dc0c3da49b56bdc11e79bc3589
4c3eae6f753c8a5e5c3d046d603dd0698cbc0c1e793d2bf909be6defcd3d
62b40abd8598ce3db67eaf3d039d5fbd17b31f3d778d0f3ebfccd4bc9208
543d3f7370bc81a2093ea3bf023e388e43be6f5b1a3d15bb35bd00a9403e
274c69bd5362cbbd93257d3d96812ebd6a93e8bcf27fbc3c7f7c2abd5ba0
bdbde016923da21ae3bc33a523bd8b6f8a3d04ab31be565d9bbdcd1f66bd
f6df33bc3c98df3df1d291bddd7261bd80ea43be8ff3fd3b01d6c63d1309
193ee6dfd2bd7e17cdbc7a8916bc280c793bd346323e4f4d96bc667cdf3c
9bb5173e3932b63d842c6a3c7657c8bde9ed033e8c3e9a3b0cd2efbcd86e
613d87abc0bd275f113e5ed5ed3ddc9413bd97b5073de69e52bd352d403d
3faa15bddd84aabce4659a3dddabc93ddb72c8bc9a8f423df7586cbd00a4
d63d763126bdb390323d4445f3bc3e5ad3bdc997853d5e256fbd5e629bd
e6d98a3d7526743daab867bdd6c0eabd18c1123e709f04be98e2be3dd87d
503d09db4abdd2181cbe08c255bdd51a833d4a68333edf6b05be55850fbe
d469c83d2b80233d547d533dca4beabbf1810d3e4f507ebd399dc0bd99a0
60bda23f413cd4c90f3e9db89b3ca730bbbd83daa93d2e0506bdbca3963c
4aa517be88a012bed42ca9bdbc7ba4bd090c953d59e2283dea260e3e78d0
dbbc6ec8103e138f043ed71ebc3d31df1fbd8645a7bd2092a23d7889ed3d
8c72e0bc79f44f3d5a10073e6ae5be3cfe8a553e03b52bbef00c69bda824
c0bd6571a5bd8a9f793d8f31b2bc0982f4bd405b99bc25acd73bf57982bc
982e13bc25afe93d2d04a2bc482027befa1d86bdf828a53d9778be3d9760
e83ca361053ee77b533c4a821abcf5e11b3d4be06ebd9b64d7bd470a50bd
f6e2b3bcfd6b12be464fa43d6f6ac9bcc288753c99551c3d22cd98bcbf4b
a8bd75e9f1bd5cdd2a3ede4f93bc25d3773d87e816bd1100c33d5af7823d
4925653c779ff43b9eab073efc380b3ddbeef3d560d22bec85b10be17c1
0b3e62e5b43d617e00bd330be63ca6f398bd4b697ebd161042bde6ccbcb
e268e63de862913dcd479fbd4a72a83dbd9bb2bb3e35373ebf8b833d58a6
b7bcffbc343cd5198bbdd683f3b9cdd35cbe85d2d83d33cff63da6e0fc3b
f71cf83dc566e03c3d484ebd4b8fa43d93bb29bd62ea313e26fd1ebeatd1
7c3c082d9f3cdf5803df93f363d5def003df7148fbd77e13bea4d7363e
c6ec33bd0a8c99bdc7de603d2632cd3d462ea5bde7e38abcd5fa893d105f
5d3b9676e63c80825d3d2f57233c4295133d8c9f8ebd6ed0e73da83362bc
32ba3cbe6b00413d01a721bdf4bfd5bcb2581bdc986ad3dfb3c883d558b
7e3d0c6094bd28b593bd339500be0852aebd35c689bdde35fabcac08a83d
9b1677bdf986cb3dd905573e45882e3b4bb9383dd378febdb43cd33cdf8e
9ebde90b19be5f931e3da4c252bd93a66abdb2d2e33c32f0debb4e7355bd
956d8b3de07aa8bcb57488bc8becb0ba98ae1ebd4f707dbb4b8a08be6c4b

Изм.	Лист	№ докум.	Подп.	Дата

c93de02ac23c9ed44c3e187a123ebcd12d3d5993bbbc8b0a0cbe2538bc3d
1bafabbba4a7163e4d0b62be01d10b3cc47984bd565e023dc3320f9c0
cbbcf6241bd1a7b693bc65e803d6de6cbbc7e98dc3d223d403de242cf3a
c9d91c3d7c16643cce01a43cdc7691bc8c2f423ef045d33db094d73d2926
a0bd8eedb73dee9c59bdc9fa5a3d6752213edc34283cfc7b8c3b8beb40be
dbe652be9d20193ccb32fbd4a780f9e594682bdd3b2b0bc28e4383e43ad
ea3c59cf10be0ad6263d008077bdc2302d3e980974bd6e0610be293ae2bd
fd382f3c3fea433ca71e0bbe956106bddfb8ee3c39e7b43d9f31e83ddb51
e5bd13904dbc0ecbc63de8ac263e6a3a513e7d2dc93d44f1603b273decdb
47622bbbe7361ebcde22603d08b4dabd53c7963d3996a33bc34601bdc336
973dba9b383ef85d1abe24d68b3be46279bd50c716bea92653bda492c33d
4fedf5bc4733443e090d123d7ed4863de7c7c2bd0d740e3e0d1de8bc35ac
1c3c926b1dbe7999ed3cccdeff3c9f4211be2e87f1bd5cbb9e3b6260f0bd
f77c183d52072e3d783ffa3d9630fbbb976384bd220f83ba9672d53c0695
8cbcef9eca3c060507beb1d3993da7dd55bdb692a53d183fb13d6d99a3bd
45522ebddf3f023ebab031bdb59baabdf73013eb1b3903c457cd13d9d69
bcbadfed533deffa033e10e5693c7aed563c64b6b3bb6228343d3429873d
2b9fbebddd18331be3fb613bdd407bd3d07fe07bd49d521bd765c623bc712
903cd36706ba0cae8bbc242621bd8dfe883dd74bb1bd6c66cebc024a0bc
6a9fc23c65cc27bee390c73d60ad9e3cb5e09f3c7eeb4f3d8b76323d7cf3
72bdd762ab3dba58153edbacc463d14bbe23abd060d3eb16cc63dbfb02a3e
28b61cbdd3833f3b4e71b53cd57818bb1ff6013e06128f3d94facc3cefa9
11be755f46be532b333ce5c42e3d63f1c5bdf0555cbdc5bf9ebddc08123e
211c083d6a1a3dbc1be6473da117b0393dd6fc3d9043bfb8dc0a6bd3406
38bd0753af3d0def9c3d6455f0bd9ca9c6bd03e993bc16009e3d3076a43d
d3022cbd41f0113bd5e7933d5bb9423ef4ef123e89c4ca3d2cde64381040
65bdd6080c3be672eb38d6ee8f3c37a143befc04b43db0d922be3aa132be
4c29c53df331f93d2c0ae5bc8b3da3bdeedddeebdafa32ebef2d09cbb1859
fa3d148372bc9681e03d9c10823de9aa2d3dc210ecbd2572673ddef4eabc
3e38afbc0fa115be547f9c3bc90e053d75968fbd2647ebbd2f1797bddd97
06bd935c113d06ac81bc5ee2083ea59080bd6226adbde352fb3b0392e9ba
6c3574bcef2b203e09c5c9bd2190013ebfc3e6bdd1b6123e1aaff73d1f67
d2bdc54bc9bc6f41563d3cea8dbd1d46c7bc6c8cbd3cbbc2fd3c2a642a3e
3caa243c7ff7ac3d51e4e8bbd1d7f4bce01101bb173c2ebd157d783d65a9
223edf909bbd64a36dbe3cb622bd510512bcade0bbbda6df5cbde977b63d
e5154fbca98bfbcb8739ebbc1da3b03bfae5953d24f3bcbd7d711fbba9be
25bdd1e5a43cd6dd11be988cd33dc051cf3b712c93bcebfa643db6a5333d
dbeb58bd1c5ea23d45ec103e6182413d14780d3d09f30c3ea668c33ddf20
373e5d30ecbc5426423cf363f83c7cdf2abb5f27163ea20d9c3dd150d83c
4d5c14be23922bbe9534373c74a48c3c85819dbdf493ecbc7ea59cbdbf49
1f3ec77b123d7a9f1ebd4e58803c05b72bbb70ee0d3eaa6c85bd0ee0e1bd
c5d60bbdc2fa823dd429373d308e17be23c0bbbd956261bbad09833d27c9
9d3d1288e5bca5e7653c6f5a8d3d5b32343e46d8253e7ef3c63dad69ff3c
523b80bd2e4d323a70cb22bdfea38d3c2b051cbea9c2863d724033bed0e0
19beffd49a3da1f1c93db24a5bbdc8b541bd8aeef1bd484e30be51f7b63b
828af53de037f4bce6c51e3e1acb643df52c5d3d122f25be606fb03db958
4fbd4d60c9bc8a0c15bea363813c8ef4d43c489c74bd28f1d9bdd9b557bd
66bc3abd1c3107bc26518b3c249a0b3e28dd54bddf99c2bd776c1abac7b0
10bdf7e1c8bcc05c0e3ea09354bdac95c03dd620dfbd3185ff3dc9a1ff3d
5ac215bef5e0c9bcee8c3c3d4156b7bda8f671bc24ca6a3db07d8e3dba6d
383eb91b96bcbdd6cd43d4b3042bcb84738bdf0744a3c782af6bc19a99e3d
e5250e3e272b99bd0f0276bedebb0fbd3670d0bb4dd730bd3e199abdfae
b93dd2a493bc3c50f6bca7c785bd0552e5bb25f8f3bbb50cae3d92af73bd
ec57f63d4f75543d0da2283d81de903d88d403beef2db3bd3c43313dd14e
5fbdfb17363caba2143eb968aebd44f2903d16e6f0bda9b92c3d5b8336be

Изм.	Лист	№ докум.	Подп.	Дата

ПИБУ.05439-01 33 01

5f19b1bd58c5bf3dbaa07d3d7bfe22bdc4c64b3dc5b4713a9cb5bc3cae73
a4bc7889093d8d7ae63d341be83db9f118be91873fbde746223eb48fedbb
a9d8063e39eed7bca72599bdeed13cfbd93ea913d9a3502bed3e39c3d5f58
72bd631a093e53ffc93baa9407bc3509073d8a71453d459c773d21f259be
ed0b2abea1e2d2bde15ee7bdd99a4abddd50523c6135463d8afbdabcf904
f03cca26003d32ce583c9725e43da44c65bbb740803d23d8e6bb7cc404be
e5e093bd3ff39abd7c5dbd3b8224e3bdc92031bd26c19ebdba8434bd9d2c
1a3d73f8c63d529ced3daba2d53ded2181bdd1f63ebda192ea3dad7f113c
931e873db8a9a13dfe481ebbd48615bce142f4bd9414933d8228493d9fc3
033df5239f3d2879c03d088c05bd843ca0bd18f7c0bd0d0a363e6de7a43d
d55d353e661197bc808b7abd9b2d083cca70063e018f84bd7bfe34bd6063
29bc3a1deabd8dabbabdab16fe3c89686f3e4380353e6e0c783c936986bd
742e9e3dd7895fbd5ca3063dc3bc19bee6f25f3cf30127bed8b487bdc975
fe3d6e9bc23c1da6f43aee6e40bef918d03a4d4bb8bd07ae96bcafa38e3c
e72414be290d1e3d1c8277bb2726ef3d66f595bd169bb4bc0904103e73de
d3bd8138303d624ca83c824fb43dd18a033e4174d83d113fa43d13c01ebd
4df494bd55a9b83c5d4db83d0ce50dbcc11d8f3da467073edaa0d1bdb45c
a1bd6aedb53ce708e03ce44c9f3df0e197bc340ac5bdb2d0323dfa6cac3c
92426bbd9a4b2dbeb35340bdf891c4bc5db2c63df9b448bd42cef3db450
bebdd1ff093d916b423cd2de1abe8b8aba3d33bf493d883f093dba17963c
fac66e3dd09fad3d4d9316bea80a343ec0328bbd0d96e83d0ff8b83c52b5
b73dce3470bd4746cebd130cbabc79a016be979659bcb211353d1b30d5bd
83f59abc14d802bea712a7bd142f38bd4622223e1aae003e70ce143d9d5f
c4bd2140c63dbb57aebd083546bd6a681abdc5422bbb8aad5fbd2ae5e9bc
f9b08ebd99f0c63be4419d3d5b0b1e3db45724bedbf326be24ec12bafabd
1fbce3468c3c14c5223e4483933dae5ecbbcb8888bd549375bd0f393abe
e36c86bd4b7545bd3eec1c3deaf203bc80fb8a3d896644bb3d7722bc69d4
c43babdf0a3e269da43d8de0f43c9841653c85ccdf3dd65ac3bcc98668bd
e08ce13d8996313cc647073e4455e73ca420fb3d1bf2ffbc0f8d3bd73a1
88bdd9ad493e905513be2e1b163e7d1e47bd011953bd86ffcebd1f05c73d
aa86e63dc2ba70bdcc9e243e50ba7bbdb5ea4c3ecee9a53ddee255bdc3a9
1a3c5d0c5abec25a2ebd65ce5bbdfb54f5bca47dfcbc841bfebdd119873e
63c893bd837c3ebe8717563de01cab3d2d2d0c3e180ca93dd14f1c3d4fb3
c8bc24fc1c3d0c157f3dae62f03d5c1486bb096b323c8dd90a3dea3262bc
155313beef07b83df9ac203e5820153e1c2751bbba7296bd8db64fbd8498
bcbdef3de03dda7c39be1f3b0bdbb6a815be99efb1ba1bdc6c3d8d213a3d
dc99153db66b143d79d39c3d989590bac94bdd3d55d650bc3ea03c3d8a8a
6abaa304363dc4591e3eea791cbdec8cbc3dcc5b00bd2e91d43d2844e2bd
2d31f83d741c21bdc9aa8d3b6205003e20c0aebd9eeb23bd7bd8863d7f33
14bec6c2d73d7f57ddbd53c629bd17a911be0e0f0c3ea89df23d22e9503d
c7a81d3c27424dbd477ddcbceedf783d0f0238bdf2ead93d26e628bba5a4
6b3daa1124bec6493abc697d60bcc80b113d2e6890bd471f04be9ea37d3d
6fdf863d7592993d3284bbbc20d183bd89a9d3bcf27664bc111fe1bd4b6e
0abe6d0406be9d9207be7ef7b7bdd8c69c3df5dcf93cf379833c65871bbe
bace1fbc3124143efb1248bdd4525abdb6f42d3cc1af173d5631403da5eb
693ca6a5803dc9bbb43ca3564c3a5fe0b53dd5f31f3ddad8993de03b033d
5e221d3e8575323d8b4504be31bd7d3dbd3bbbde78aa3bbd4d8abdde90
3d3e20a20c3d692c8abde06a6b3d0c2d1bba600e1f3e33c8ca3cd9f093bd
ef5d5bbd0f0d20be0d8ff23bba5f5b6bc5765c8bd0bfac93d7ac6053dad7d
1e3eacfa9dbdc945bb3cccfeb8bcf40369bd9d2542bdf09ee5bc7224d4bd
3ee9903a1d1c0fbe43fdb83be8698e3e2f2e0cbeca346bbdb8bcf5bdd9e4
56bcfe6e50bde9e3aabdfb23fa3daa3a1a3e297434bd9f87e8bca67114bd
679e5ebca54c843b0f3ad43dc52c1e3ef5ac873d46182cbda32c113db13a
3d3eef650f3d4d2ce03d3492b4bd95ddf3ceeebc83dd789d83ce3248c3c
2d0b8c3d756f973c3a523b3e349f213e13bf403e4b14a2bc98e08d3d8bc9

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

ba3b4a4918be46f52f3dccfc703c85e37dbc158e203baed97b3dd594193e
e3dc4cbc8b243d3c412934bee34ce4bd0443e13ac5dc353c26a0c23cda6e
bd3ce713cebda347e7bd825115bda494ccb036079bdac3020bce92656bd
e73c3dbe28b4963d6014713d3a34a73d682f123db0ecbfbd5b4645be9e23
a63c20e5fcbd5814b33dc36f093ddf228b3d2a8c49bcfb6900be0d9463bd
5e963d3d0171223dfcbd88bb3d08473d1e6d75bddd3196bba95ccc3b6a25
14bd1b5e3d3d5d25f43bae1f10bef5a608be7a99713dcd572a3e284bc8bd
7d1632bcfcb9503c50b414bd7aef1abc5b81bb3dd66b583d26ae9b3d930c
d1bd5046843ddf1a2a3ed4f0173d7c4e9bbceb4614bef2cca4bde09564bc
06b781bd0a62163e50efee3d47f9cebc4b141abe4eff02bd3c4ccdbd02
a9bd40c5c8bca7b6b0bd34c128bd2fa2b4bcea2fc93de03f6abbc765553d
04cfa13e16c6acbd95cc93bdde46723d89b0b13d3a3a033e42a62c3e97ef
743d6d2582bbd76713bd0980ff3c5c97ca3d88c39e3a06a421bd47155c3c
626008bd5cc6cbbdf2b3e53d777df23cb614e13d78532f3e9141acbd60eb
ba3cd464113db880f43d227d6dbd4d873dc002d4bc8094903d87c4993c
4610a2bcb32a92bdf1a9e03d01390c3ee3aaaa3cde761e3d56829f3d1fa4
9e3dc0fa85bd7a4463bd61d2f03da39b43bd3a7c813d5f4ec63c585f873d
2c855ebe5a71b93d5052ce3d9e3b9d3be2ad2b3d76fccfbd84fb46bd6863
463dc9595ebe20f26c3d30e738be8fd2abbd973c59bd7cd79b3d60dcaa3d
b78f333d63afc4bb46750c3ca368203d4b6f06bcc31f25bd41f8d53daabd
82bccfe48d3d245f65be4072abbcfc1e8cbdb20c37bca7fffebd13ecc7bd
27cf093c1db6973dba0a8dbc6e992bbbe37881bdfd11b43d9190d93c2e90
61bd30deb1bd0b0b3bbd316fdfbc8307fbbd1c39243db35a083e93c45b3d
bb84c6bd1a7a823db12bf13d2782ec3b68f314be5586243d8aeb823d5730
783c88ec9f3cecb9c5bcfa180e3d7722993db90b9b3d8558de3dd3586a3c
4dfc823c08a1a3bd9407053e4f37b3b8b9ca8fbdc29aa03d0ed8933d69b3
b6bcfdca9bd0314413d4f2532bed796273eb527aebcb65f083e5374883d
c121a1bc94a6b0bd174fa4bd0bcb6a3dd7ac96bdb3966fbd34af85bd8e1e
fc3c00e2043ed9438dbdbd17843cde2d6b3c8d26f03b32efc4bc4e4b3e3d
382bfebdfe1e9c3de155fbbd0f83573bbd60213eb36f80bd78538cbd95aa
08be60ef0bbc455241bd904069bd733a633d2ac4813dd8cb503a1b57babc
2844d8bc2e9fa8bd36b023bc5c83b03d4a9d373e8cda25bc903b4cbd1fbf
f43d8572bb3d81a2a03d0220173e5cb2fe3ca63f7c3d50e1dc3d46f5e33d
6dd3fab9dd46523dbc9f1c3ceb1f333ebc1dc23ddf58963d6f362cbd489b
3d3ec2d07a3d4196f2bdc9be5fbd05dbb63cc9416fbbfba154bda178753d
63deff3d9cc84f3de7532f3ca4734f3be80c65fbae04a6bd146d863defee
a03d70e8c23d9f7108be17cd93bd2a3255bcfc4198bd961eb0bd8e8908be
ec3e53bb1f0e8b3a8fb26e3c18da193ebe74aebb8ac6373db98309be36ba
04be3e65c83dd62905befcf5bf3c7a3edbbda0d48c3d737e9abd2b7d02be
1fb99bbdba43743d4c9549bcc7a0673d7d1aa73d1331093de668afbc8dca
4fbbc8d207bdba6ed8bdb805ef3d1e8551bd71e4c9bcd8999d3c3f2e2b3e
81bc75bd60d601bd779ca5bdc91ba9bdb1e946bcdcc4263e6c80d63d0a2b
223dcbfdcfbdde05a8bd0b7a903d47c2943d2fe0cabdba6ccbbd5a53e4bd
92b3f2bd3b704cbd3a5c2e3ea27bd63d94aff5bb5c2120be19d1aebd3c5f
bebd6e0296bd1a73c6bc76eee7bd84c54e3c54a1d13ceb8daf3dac90473d
bdafa8bcbf6f693e5f2453bdc4eba9bd50aca83c5614173e3f581c3eb124
3b3d8ee7a43d326e02be68f22c3dc761bf3de6c2c63d5b67abbb7ceb943b
1c4bf83c1691a7bdd3b7d6bdd2562f3e8a69d53d465e893da8d68a3dead1
c6bd39f6dbbcbc3a343c4d64693dfa99c8bd7bdce8bcb8870fbed28831bd
57931d3df44e063d2f7b91bdc738943c96c9533de42fccbd21d57a3d3c1c
423dcd19ac3d821ec5bd27ff24bb3e62d43d233ecabd93b9f23df4d22d3d
65c32d3e827edabc1d97243ec4d484bcebe70abb03d8eb3c7d38dbbda4a8
1bbdeb592dbda0d321beb84827bd73a911bebdbd0dbe9c530abe4753d53d
4eb02c3e70bed23c415f39be73e5613d744fb8bb2495b0bca741acbd22e3
ad3df1e56cbdea89ac3c114509be049285bc0864903d4a11883d61001bbe

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

be600cbe0668b83c512bb63d7150ae3ca99e6b3da1428dbd55abc7bc3e82
 95bd3b0088bd0ffd12beadfe2bbd9998a7bddd1705bd8243623d27a7943d
 2f200c3d76fa29be8a132d3d5fedec3dbb9a5aba947d54bd7f55723d6397
 753dc035063ca44961bd10bd9c3dd815a5bcf0bd813d9f31933d668d863d
 4069a93d1312d7bc98ebb83d0ae0ca3d40c314bec7ea1b3e39b70ebd76f0
 4dbc59880cbd96cb903d6d66623d3ef3e5bbfd4b243e9f70adbcc04b1f3e
 35a6ccbbdfa6a6bdc1905fbd8da25abecda3483dae7506bd210e4ebdd9b4
 303d09fa9e3cc1534e3c54cfe6bd2d5b363d11ba283d3836ef3c6f0bf33c
 d59967bd61f207be1f242d3db9449fbd8da9f5bc6f70323ea7562ebe4b89
 0cbd67583cbefeb9583dac3500bebc320cbe20220f3e33778f3d5535fbbba
 1b20e33cf4b2593c05a0273dd67aa3bd5c22a43defc03c3efa25ac3cb148
 54be22d77ebd90b5203ef78c27bbaf2bfe3d4d033ebd2dd605bdf3dc88bd
 6494503d0c2ee4bd500fe73dc8f48fbd03c53a3ed3c2993c95ee063ef61f
 f5bc412c953d837b613da6bb76be1c0557bd9fe107bc19b4aab7ca769bd
 0ccc833d8fe44bbcd7207bd6098e13c5d068bbd8ede6cbd0b00b13d7724
 edbcacf8b643d786ae2bbbe660dbe6c7837bdd95dc6bc5f4bd8bc3608cabd
 d5f767bd706ae6bd90baa4bdebd0353b807db13d3235a33d7b1caa3c3088
 14bec49efabdf261d33d90ed4abd39ee833d18a8123d04e5bcbc457df6ba
 db6af7bdd0c28d3d2636473d9276da3c162a3d3d988d503d81af21bad5bb
 b3bb337cc5bd1397b93d366789bcf7945f3da5fbb1bd680fd5bd86fe6bbb
 de8a833d83abc0bd17480bbdc98946bdeedfc22be3918febde11b8a3d1389
 083e6d19023e00fa5bdce2ebfbde535713d0b5627bc798ff93b8c6317be
 57522ebdfad101bedb2d3ebded58ea3d9c7b563d816f083cc2e411be16cb
 65bb3275d4bdca1a26bd37430fbd52f965be104779bd4d33d6ba3edad83d
 71d43d3ba354093e1286513c28aef33ba6b291bdc2be66bd5fb90c3ef631
 09bdc507ea3c841f19be3bd7963d4ad215be62389fbd5136ccb8b079c3d
 cc0dd4bd3887393d3359adbc4924a63dd9834a3e412729bd8b8cb1bd2bf1
 91bb41e791bd7f55cbbc4458003e16843dbd8003cf3c61aa57bd5ecf8e3c
 87fe0f3cf762acbc9bf5a4bd27c0a7bce07e2a3e37a2243cf7542f3d0009
 a83ce6887a3d57f6c53d3e88203ed1c0183ec3efb83d2356d1bd79ee023c
 21e4623db35b04bd2f74273cc2478ebd950b973dc1d02abd2a358e3d7555
 083e9990973d8db1603e149f053dcb53853d07c6b4bccccfdabd6624033e
 97247ebd3a21e8bcd7198fbc6d6824be4d0351bd1ff9ce3d0e32ec3c08da
 22be7254ec3d2fb7cb3d53f78ebd45bdbb3da45af33d7969c3bc9a5ab73d
 30a9c03d458f49be77914d3e37958fbd295e043ee73903bd154dc33a4d89
 50bd72d5c73de6d83f3bacd774bd6eb4573dc0e6db3c67b7b83dc3d8173c
 8196213d7616093e94f1713cc927ad3d644354bd7f61d5bc653bc93ddaa2
 06bd5ff5163e0c30283dfb49103ef328a4bd44ef29bd253e6fb77a07ee3d
 f9c2903d337aeebc22cbf2bc0e354d3d79d785bb895c7fbd0e1804bc6ac8
 06bd54e95c3bb2aefeb3ba88617be5ea415bdf4d3113e37218bbd801a38bd
 4a9fe73c0a771e3e4650b33d5ac02ebe9e6ac43d1dcf2b3df5032b3e075e
 60bd229277bd49cd053da541833e6bd60abd614323bd57f071bc11e8163e
 edc7263ee868cf3dcc0dfe3c11143fbd5c14843c70ae323dcc6bce3df2cb
 86bdb2d149bcd871abdfca9e4bd52de2bbe2694c33d962b4d3d03f8863d
 40b0ef3d91fc73bd7b13c73c48e9483d926c243d3e010dbd909e0d3da8e0
 b6bc74e3853d6001043d6e1c743c778ed9bd9db5313dbaa17d3d691aa4bc
 fa19013d9193973d2d7aa73d7f5992bc7e3eb5bdf1e0103e05b4b6bd4405
 0f3e3292cfbcc532163ee295f2bdd99e213ecf05913d991fedbc66dd513d
 e9c2f8bdf7dfb6bd1c3a3e3c9bdec2bd4c3c5a3d86fd2dbef6929fbd8a8e
 88bd49fda53dcbf7b83d4001473d4df1febde569a83db16abd3ce9925bd
 dc8cd2bd02a7103d2467a2bd965d623da4de29be96dd41bdc916bcbccd73
 06bb8f091abe2ce50cbe644ca23c9a4e223e218e36bd170ff0bb896810be
 60e3783d7ec712bd1713b5bd91550abe27d083bd55d00cbddbb6cabde8ef
 783cf996b83dd874a83c989e07be7fa8acbbf4923d9bad3dbd033f46be
 b7ba093e2fd5cd3dac0f5bbd5f7edb3be42f5b3d5aa0663c6553793d6a43

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

693d14444f3e9aa1a7bc4c38883ca8a134bb8c19d43d1f4f57bd7fd39dbc
5e852f3d326fcd36e1debc382e03bd8da1513d67fba fbda495583e4ccd
09bdd5562e3ef65a533d8e856dbdb91f82bd10fe2cbcd39d2cbd1c0c45bc
b5fb8dbdf19950bd1eb43abc633fb63d1ed07abd9ddd3d3db2d83d86ab
953d01a5213dd2062fbd233090bd0542a93de84fcbbd6bcc303c550d693e
255d05be6c5112bda4f40bbe85d4283d417af4bdbe54dbbdf85ba3d6571
003d1b7e773d5b98ed3baecc933c19d3ef3cc0d772bd096b9c3d16693a3e
97742d3d598becbd4dfcc5bc1cd1093eac468fbc7c96073eb422a9bc8276
bdbc62fd01bdec7a9a3dbd7fd1bd6653ee3ddaa21bbd36010a3e6029253d
236b0c3e07bf23bd2c4a0a3e0e45003dc74e9eb08ee1bd69f51dbd5fbc
cabd59149dbd97c79e3dce7d7d3c4f04abbc572a873de1f0e9bd031696bd
49b90e3ddcde013c46ebcf3d4c37a8bb8fbd10be918f38bc79aa4abd3384
c6bdd58ca8bd67dfccbdb864bbbd0b5c4bbd30b0003dbd13123edad8c03d
ee014a3d8eddeabdde06a8bdb472233ea31248bdf3963cbcd932e3c4c52
86bdc231f83c8104adbdca5d023bc8875f3d13185abc340ed43d0d16393d
ebc2033da7c7febc5450cebd7ef6603d46098abc1a208f3d28cc02bedba8
eabd088b133cf9ead73d1d29b0bd97c8e43b407b90bcb76f16beed87f2bd
b31a923db2b2123e3252a33d8ce6bfbd4c3f83bd1fcdfd3cbc06833c54ae
4abd2cd0d1bde22536bd040af8bd746ce4bdd009e53d030b843da2d9133d
25c6b7bd314e26bd3d0505be648bc73c2b4869bd69a641be8dc8a7bc5056
423cb31cbf3d9d72ec3c5247ccbc99586e3e28f1a7bc60e627bd08accabc
33c0123e26082e3ee29abd3d406a243c97e537bd2737923cf599a43dae7d
c63d125587bda2d0d0bcec5938bd581108be609329beb9dbef3de34bf73c
9c5a9b3ddb18ff3d1127a7bd267497bccf5e833db421063dea743bbd7445
7d3d2f92b8bc333e7d3d5de1923cc6dc0a3dd906d9bd0b59813dfe1cbd3d
d22fd3bc7d9edb3cb5ea903dd08dd13df50351bc84a3bfbdea31063eb2fb
abbd0393f33d29411fbd0c4a1e3e8811ccbdbb5b243e40b2813d7a0b1fbc
562d403d87e79bbd14a1cbbd2b26d33b0d1ff2bda8c8e63c5e8c1ebe5843
c0bd845578bd1c51db3d3452d73d3c654a3dc09e23be0adaaf3d9dba313d
a3ed5fbd5accd2bdc832ec3cfd95ccbdc682323dc17d32be7fd583bd1163
89bb29c2053c50062dbeecd127be6e60443cc025213ec4608ebce38a96bc
6ad4ecbd0feb3d3d324514bd6f7baabdf9cecfbd721dcbdd85e82bbd1a5f
9cbdf3cdae3b91b6ad3dae3a323c665317bea8cb913c5b2b573d8c2001bd
51a71dbe82ac033e5bf9bb3d191728bdec6459bbddde233d2515c93c6382
903d9312303de34a413e6e9a2cbdf36a783c8086673c3f2ac33da29162bd
b48d88bc8cdd8a3d289ea6bc048fb3bb4f5804bd7465ce3c849cd3bd396d
443e993327bdbb941c3ef326be3dc207cab8cbe73bdd4893ebe872f84bc
934c83bc8bc577bd0a0394bd9a1d3d3c9f9d863db5ba02bd122a543ddf93
a63d854c1f3d7ea03e3d5e1ef6bc311e99bd19fc063d77e7e8bdf0f4dc3a
10d4473ee72309bed7be57bca5e002be3273973d92d800be855330bdef14
cd3d15b5fe3c89effd3c6dfae9bc0d8b83bb9dc640bc869964bda5a3d43d
7bb9263e33971a3c1447eebde2de3abca19a123e7e0e74bc1e7dc73ddfcc
0d3c600f21bd7d33eabb99a2bf3dc299f5bd468bbd3dbff079bdeae31e3e
e083c93c1ba4de3dd699cd39b653a53d9f50c73b9547aabe0a5cefbde1bf
cb3c4ef094bd005674bdcddf803d5889f23c37bd1b3d6b4e933d41b703be
266574bd8d61933df784f7bb303f903dd6fc95bc64f512be048a1dbd739d
bebd828622bd1a7d93bd98fc06be671b8ebdfb9290bd4597fc3c396bfa3d
91e94e3dcb11973dea5418be86a2cbbd0dcd0e3ecf347fbdaa4e3b3d1a8d
ab3cf453a7bdf512503cd646b3bd57b303bdc191923d2242fcbbe5cf993d
fe12863df742ea3c142090bd217a03beac39973dbd88a4bcecaade3d8539
0ebe2251d6bdc3f2aa3c4b4ad03de8c917be5f9ecabc97398abd1ed811be
8a3592bd3e2c703d4441c83dc8b3c43d1a0eb1bd51409cbdc8ab713a8233
d53caf06bbdb99015be0a1147bdb24114be292b95bde2bf173eac81443d
52f788bbda5ccbdc67071bd9b1737bef8554a3c562b77bd679e0fbe1571
04bd2356333d0a7efe3de539a43cbfe4ba3c1ad5883d4f02d5bcd0e1df3d

Изм.	Лист	№ докум.	Подп.	Дата

e77e3e3d6173b03dab816d3d056ea6bdd204313db13e1ebcaa6aacbd0913
a3bc5cb4e33d92588fbd04bd3a3dbfb117be74205b3d128806beecaaccbd
b018903d6d00a93d807d103c50d91d3d229b99bd49ab6fbad41b92bd78e5
1a3d6524b13d7166713def2e18be4e5870bdc87ee03cb94982bd0213183e
df2f2c3c6e7ba4bc1e8daebd53085f3dbc32c0bddf2f5c3d76345bbd225a
203ef2fa233beffdc1bc08efa8bb9aad27bdfbec5d3df38724be962b50be
bf369bbdf12447befb1879bd663ee93b7811e13dac4e57bd4d6deb3ba8eb
19bc5eec30b9f77dce3d218f15bc6e650d3d0156a83c1cdef9bda0397dbd
69f8d8bd4708c9bd276064bdcbf56abd5ae2aabc8742c8bd02da18bde2c0
093e73beef3daac9b13d8d0bb6bde35b5ebdf66a933d4c3158bdb20eb23d
db2e353d4478d33c6664783d44921ebe6cf8a63d7f221fbc83f22e3d18bb
f63d3a93083d2aa7673d2d678ebd1bdd15bdf3ff33d1717c03cd9c82d3e
71fd79bda6eceeccc1cd053d7e42ce3d729411bdc5a5c8bdd8f4b9bcb8d0
33be99ff11be8295cf3d93c39a3ee77afa3dc5c53bbb9fcd25bdbd61b5bc
83a52abb8ed547bbdd36e1bdc54085390d7b48be44b126bed3cd9a3d1303
33bc4e21293dab1d22bd16909ebda3e838bd9705eb3ceb0683bcfc8116be
a483aa3d3f52013e71b7303e2e8cf63a0acad3d40a6b13c74babe3b0a4f
df3c9659cf3ce3cbf0bceef9003dbd4b68bcb5d0dabdfafcb3d9df49fbd
29c9dd3c68a90bbe76d0a33cc7dd3f3cbe2a6cbcd588aabcd4a56bbec6b4
833c07f23c3da1b7f83d4f9b1c3d39d036bd189f8ebd20eca93c1bd9ce3d
c03b1ebd11007e3db763e63dc28f1a3e574dd73c547aefbd8044d23da2a8
513b6f4986bd4880043d6b485ebc9f14063e5105173e3535b4bcad59373c
e1657fbbf55bbe3d587c9f3d01b7aebd6f83843d5946873df7ae7a3d49af
313d673016bd72e2f53d2976473de96688bc1a98b23da80ddabde9ce4d3d
2589efbd4dcfd6bb00668a3d41ddedbcd86eabbcff28a7bc3848823d6f9e
22be5d44ba3d7406673d8e8912bd85dffabc75b6adbd8a647f3d1be7003e
87d8a2bde593aebdd46f003eeef0f03c9bf4f9bbc08c213ca886003eb196
e9bda02c8bbdab2f13bda1a58ebd8fb8183d9bb87b3d4336ebbda47dad3d
134880bc2cac873d8158a4bd1c57bcbdc34492bdbe9a7dbd1a71c33d0e76
b73c3f221e3e6dde2cbec4ff293eb45c7e3e6b89d53c162bbcbdddaab7bd
4403e73d5640fc3c146ecbbd6d4c293da5de453e0747273d55934a3ed7bb
08be469f87bd9dff5dbdff8340bdd81ec03d7d31cabd286c07be75d465bc
345c233d032fb3bd8014e8bd917c093e7f29bddd1d2e3ebe1ebc01bd6297
223d0a4795bca103023d04ba133db7901c3efe12f4bcf69e0abdee22453d
99bac2bdc499a53c2ca201be211c11bc9d3e0b3cbf7b9fbddbb9783daca1
213c7ff841bd8524b53dde1a943d5414d8bc999217bd8caa423eb9a1983d
94b3933dc3800ebeaa0a793d56c5d33c5d084e3e22bac43da3d7523c0509
b1bcc05596be697a15be44ceac3d5ebd95bd6755b5bd131d8fbda5861c3d
1577b73d3fcbe83cdb72043deb0b613d85510bbd8962233e71df34bd97c3
02bd158f51bd0bacf5bde8a1da3d8d072cbe9547c4bc442105bda98ba13d
e963d33b7f61bebdbae4aebdb7cb353ebb44373ef055563e76701e3c3299
f2bb8bf639bdf144273d82d100bdc566ddbdcbe08abb542921bc7a518abd
e52e12be4ce4023da3bc433e31f549be9fb9a8bdea2e72bd794332be1995
c9bd65b10dbcb237003c88ae1a3e4b5ac53db868373d8e39c8bd0f8fc33d
158591bc60be21bd90f8c5bcc3a0a83d354c0abd6d29f5bd230af3bd20dc
edbc414a06be07c2843da96e223d3daea33d5c2953bcb34ca4bd3c0ee33b
4a80b13d331de13ca01962bc60fe3abe11fd0e3d8bd8febd291f943d45d2
47bd0246323dec32d73d5dbb043e558426bdacd9be3cd710d33c1baffc3b
2c2f7c3d03c628bd70b9413c5355d03d1ecf99bc4bc0f7bc29462e3dcdca
313cab6eb03d2d7f833aca78aebd4b932abdd5ae833b403224bd1f0889bd
06e05cbccc4fd73db83ca83c09fc8b3dc36f78bda042393eccb170bd0aaa
17beebfd453dd893ebbc5f8a6b3dd6f71f3d23aa02be44a59abdedbc953d
e0a64bbe88d583d5d9b6abc355b283d418d9bbcf9cdd3bdd6a125be83f3
bbbdda09223d79bdab3dc272693da05d243c4d1e9abd0b25cfbc2ed4813d
8b84183e76a793bd5e16583d6f55a13dcfcdb83d61d1743de3e40bbe23af

Изм.	Лист	№ докум.	Подп.	Дата

PIBY.05439-01 33 01

afbceaacfa3c3d2406be8791aebded641ebd424f8e3d5dc8a43de66ccb3d
 2b45573d15afa13df47b7abc6d008e3db4ca6bbda20b673cd02a1cbde36c
 51bd7f0699ba263590bd521fc53d9a8783bd8a9c2c3d494aae3d2af31bbe
 3677da3b0df67ebd2963e03df105253ea057573da08f473d928ab1bd40bc
 153e172105be65505f3d762f8c3d2725ecbceee632bda070e8bdd9db193c
 16b97d3e575adfbfd196948bd7a72283e9865223d51d2c53d25e88c3d8740
 303d7a56fdbdb19881bccc8597bd74d9ab3dfa4cc73d600f463c72b267bc
 c8bfcc3c027adcbc14ac2b3d37b0b4bdc2239ebce3a0d3bdf1ca1dbd4e98
 163d32964e3ccce2de3d5df7e1bca704443e28664f3ed5ac273d6c023ebd
 8a4bb4bdfcf9b03c49720b3e61293bbd175e2c3d3a6ca43dd38507bc3e1f
 e83dec2501be4817f43d27023bbe2ae5c5bd1b90763d00a2babdd42aa6bd
 f5b05a3c3b7534bd6a8c1dbd5b4089bc8235293e551aacbbf3639dbdbcb7
 93bd101ffdbdec41b03ccfc2803d8d6eb03d12645d3d41dcc73d66208a3c
 7ee6fe3cf5fac1bd77a1583d6ec5923c8575e8bdfc5f33dfd1c08bdf2c3
 84bdd7a91b3db3361dbd300222bdc29a03bddd47143ed65c823ca1506c3d
 0be683ba774cbf3d0e59003d60db52bc80e88e3da0a4a93df184573d21a7
 9b3dddea69beb660e4bcb435a2bbfb8608bcee2555bc10c297bdaa4295bd
 eeade4bc08cc90bd02de6dbd57cca33c57c1843ce03ce2bdd6700a3df90e
 95bca6285f3e704e10bdd060983df62fe63c692f84bd2cd40cbdb06a6dbe
 7c32cd3cabb52e3c7ed261bdf84b833d1dbbd9bd0185bdbd63a1ab3df54d
 ebbdb934563e37932fbd91028b3ddc34f13df979a33d9069583bfa1415bd
 606b17be07bb0bbc844be63d4605b0bd509532be523fd33bc57bf13d9780
 edbc428110bd012411b54d1fe0bb41973cbdb6fde73d377cc6bd0a3c813c
 b3f2d1bd75de273e4d11bfbc5798e5bdd7d0b63d1dd9fbbc935ff03c9192
 3e3cc170243eb06020bce21a2b3d36baa7bdf9d6d1bdbea7f1bd2688d5bd
 9e0783bc3c10acbd278fb53d24e00dbd2f76023ee3ec513e1d5c29bd5f45
 263e6a5c13be832f59bd4405a8bd598de5bd6e73d2bd5638afbccc299fc3b
 4382a1bd79590c3da5c079bd6e27fa3cda14c03bf194c13b4cb61b3c917d
 0ebdd99aa53ca55d0dbe250a6f3dd650d13dd8e84a3e68de0e3e803fba3c
 f68487bc9b102d3c4b63a53de8eea83d3ff6883d92ed4abe74783e3e5a8b
 20bdc62119be1d65863da48648bd60ed383d9bd16bbc7719c1bdedf409be
 69979f3c05273f8e84f7a53ce37453bd97eb733de67a0c3c05a9b3bd2148
 11be5317cbdd3803193d276ccd3ce6081f3bfad09c3d859963bdfab247bd
 23819f3db64e263eee9ea5bd193bb03d629a993d40ac8c3dcc92943d79f3
 0fbe9e33a6bba96db7bc812607bea0c20bbec8f943bdb1da673de1db923c
 3590243efff79d3d19329b3d109644bc7ed2373d33f312bda0b4ad3ddf5f
 31bcba7d91bc08b0463cb9e815bdf78173e28a9cabdfeaa993d005bfff3d
 91fa50bec58d493d011859bdddce863daad3103e34e28d3dcb18ff3b76b6
 70bda2a5f83dc70a16be38bc103d754fb13b532bd1bccd5c24bd5f2a09be
 d715163c1976603ea47a89bdd644b0bd42a45b3e5a8e2d3dcb4bfb3c7996
 0f3d7ba3253c127dcebdbb8a95bc3099aabd966bbb3c5222bd3dd6373abd
 5bc2cc3c1bb54b3c7dda50bd645a9d3d9eafb5bd134d423c003a32bd9f79
 1ebc76c0783d1299edbba5fde23d4b4a163d58f91a3ea304533e75ed3e3d
 91b617bdc76700be33721bbd6bcbd43dd2302f3cc069b63c6607ec3c7100
 b83abedf103e3e9961bd0deddf3d7a0213bef47f6abd1f1b3d9dfeaedbd
 dcad99bdf26edabcb02c8bbcef6915bdb906743c2308713e39c191bcb252
 7ebc804defbcfd532f1152533d6bf6843d0943aa3d9a3f323df4386fbc
 49dc6cbdec34e6bcf864adbd14b1e7bb1109ccbd8c3789bcb920623d2527
 81bd709aa83d79df07b78fdc243cb77821bc0c58483db48ff2bd1c50afbc
 f383293e0685ba3ca542173d2b1726bebd1d063eb085883c9574cc3c0c82
 803ddc9f373d40b317bd365052be2f5a46bd3d04a93dab99b1bdfac1abd
 a38e17bd1c55a23df66aa13dc88bb53db16f03bbc3d00abd7a189dbd7094
 f73d266bc8bd1865513c0d551bbd7a6f23be3011ad3da92e2dbe5a27113c
 bfdc2fbc16c71f3e8f36813d3e0185bd5b7ddc3a95691e3eb9bb1c3e02da
 503ef841833df63f183d43050fbeb5cc69bde34e04bcdfad333d75dd4bbd

Изм.	Лист	№ докум.	Подп.	Дата

585cba3c1687223d79cbd2bd33d233bd59840b3e1ecc86be4e928ebda3ba
44bdaa1f0f8be404772bd01c34e3d367a303baf592d3e3743dabb94efdb3c
c8914bbd25b91a3c17dd893d9a1f1c3d51a133bd198e393ef95563bdc608
40be9a6c14bde6f311bdef06f1bdc4139e3c3abb39bd23bea83db66492bc
0fd8debde90f203cb254b53c3ba1c23d4937a4bd34d93cbe3d5ff43c6f6a
01bc45a3153ee7b33cbd445a21bd9d0886bdaffae93d784f44bd1b74553c
40d78e3dbf60f33b985cb53d795e0bbec3c9b23d3ede8f3dbf500abcd8e2
bb3bfaade93d2cdad1bc0eeb6e3d7aed21beab17b7bd6ed5893c27211cbc
97abf33bb7ed52bc3687efbd17514bbce335af3ac697b83dad714bbe2db2
dc3d705b76bb73eeb0bb711e023cd69743bda70ec5b93868433cc009033d
879ff1bd9e8f9f3ae235c6bdbdeb9abcf0c213bd8f585e3d03bb7a3d0d29
67bd6ccfc6bdb9f781be8589d4bb6d29c53c142a793c110e063d2c13e5bd
a05284bd5b499bbcebfd233ed78dbfbde330ee3d66c3013e604dfe3d5bac
8f3cd552a4bde0d1533d60b7babdc9248abd4f2f9bbce4a149bd9c81543e
a32aa43d6f43483d86cbe03cbdea213ce1b1803d282401bcac6084bda8d2
aa3d64d57f3d9f50f33db671c3bc45bb49bdb773e3e08d8f03c9b1d963d
bfbcb263e24ea11be3d18c93af335dfbdfcb75cbd0a57ef3db75f193c1bbd
063dca9005bdab1ae93d8fb230be18681b3d109761bdd71872bd8661cfbd
fc0e15bebbffdf3d88dca3deca4edbc062afcbd53d3273e51ee673db745
f53c74f51c3d4a7add3dad8c6bd86b814be382aa43cedc762bd36a942bc
2a402fbb201e25bd40202b3d6c504b3cc19d943d882574bd85bfff8bd071b
aabc42e1ecbbf86aaf3dbcb7c0bb83ace53d8ab471bd7a6acd3d156c2b3e
0b39ec3ddd084bdc65083bdcb69033e08b53f3d0f080fbc6456b93d817b
083efb5bce3de48a383e75e101be796f343d4523edbd9e4ea43c8013d03d
bbe18abc90dcc7bd89af8fbdf5149abc28da79bcacf10a0bd02dd363edc99
19bdca4edabddea92d3d20fd2d3d01dc873dd0b7223d5684603d13e6273e
f0dce23d731ba0bc9089053b7ee2a43b906c66bd1410adbc4ad415bc573e
3d3ca50dedbd3cd6703c9ec18abda64b31bd9cfe84bd64e1b43dc89b6e3d
c1e1a5bd6620cbcb9b7e75be71cddc3ba2c94e3d4b23363dc7bc343db674
cabdda7aafbd60a21dbc43b7043e1a0499bd5666003e6b6fe33dd30d083e
7228c03b6007ccbde2a12a3dfb1f6abde5f89dbd6e4d2fbc7aa21fbdbb7b
313e4b7bc63dd397823dd2b0333d0fef80bcd35c863d03e2423bddfdb8bd
b753c33db3dc993df841cc3d4656b5bcd4f285bd58ca233eb418613cc338
733d5cc2113efa0f16bea829fb3ce300e4bdd993c8bc6d46ea3d8ff2f13c
cfeaf03b7b3b1bbd0810063e680939beaf59253d9f41953ceaae69bd98c5
93bd5a1220be90afd13daf8d063e18e13fbbdf7ffbfdf2be1d3ebb906a3d
4e07093d6ecd853cdf2e0a3eef55f6bd7082d2bd45c7e03b549e72bd7976
313daa510a3dac122bbd8cb2073d9ec2b13b2dfa293c1d0172bde610e5bd
79818abdef2711bdbd0bb03d3051b0bcf2cee63d53b0adbd60f0b63d40bb
3b3ef1dad73dff1dd8bd9acb97bd03390f3e1d2ccb3ce5953fbd997ec53d
610fe53d4e53df3d2537393eb09414bea458dc3c96c3cbbd78f74dbc367d
bf3d7cb650bd7994c8bd62e268bcd46c363b092e38bde1b5f1bd1da7273e
7f38c1bdb740ffbd946b3c3d949a933c8fdb3a3d905e213d2f49db3cb5b1
283e7aeed83deaf067bc4039893c030c353be1dd98bd54d1d0bca70bd2bb
80ce36bb8284e0bd3817ba3c55c3a0bdbd1867bdda89a2bd4b86d43d54ef
043d7e40c1bd07164fbd2f7b64be88a7afb9ce6a593d2faf643cff22313d
88f9e9bda1feacbd8aa996b9c694de3df47b61bdc401043ec17dee3dde7b
043ecc65d3ba582dabbde0de3f3df525b8bdbf1ba3bd751cb9bc2ebaaafd
0d45253e7dede73d990a7f3daa56823dab1499bc9260823d6a0018bd782a
c4bdc0b2dc3d2dbeaa3de025b83d30311bbd111f90bd28c7463efd72cab7
ae00463dd9cd0c3eb48629bee787ca3acba9ebbd0556cdcb452fa83dc29a
883c8f5c9bbcc401b2bc883d013e2ffd45beab724a3d2c41553c6d83e5bc
882c74bdca2c1cbe762a013eadc1ee3d5c425bbd6393abbdcc17273e11b1
8d3dd0175e3d24bd9d3cebe8e23dbc5fbbbd75aeb5bd635983bc180c4bbd
c76bee3c2bcff03c25d648bd585a143ddd5673b8cb9cab150a7bbd9fd9

Изм.	Лист	№ докум.	Подп.	Дата

d7bd1b1db4bd1b2ce1bc5bb4a63ddb2841bd824ed93d2ef56bbdb74fd63d
0b2c4c3e7cadfb3dfd8dccbdce3fa8bd51bcde3d6f210abcb5c398bc0fab
003e4222c13d27edef3d109a533e4fad22be00310c3d40349ebdcbe50a3c
ed43ce3d91be87bd80fae5bdb86129bd10a5183b09d4ccbc22c0aabdc1a0
183e991eb5bd3089d7bd6ba4113d83aca13cb3e2a23d1718ea3cff27103d
51d9173e9b98fe3d3473f2bcb0d0b73c964bfa3bdc0fa2bd54c43bbc6867
093cad7453bc4cbcd1bd9bca983b01e395bd153a85bd1b1e5ebdec3dbf3d
6b0d293d29bfc7bd52a668bd248569be757ca4bb8ce68c3db1e5073cd20c
203dd666ddbdbcb7bebd67fb46bc2b4bf13d33b485bdcb2fc63d296ed43d
62f5f23df346eb3ba489c4bd513a3d3d8aaecabd6779bbbd625ce4bb9308
85bd974a1c3ed217df3d4f56593d2a07133dc005a4bc288e853d5c0489bc
2c82cfbdd134cc3d4416883d8666c33d9b820ebd444f75bdf2c9433e48b5
22bc9cfa5d3dcde2023e1c5c28be2742d93c839603be8f9f1abdf0db33d
4f69bd3c036f3fbc676bf6bc3e06013e9a2929bea9f31a3d7a35a33c94b4
44bdce6c6ebdbd8119be7503053e0a3a063e319587bd7c98c4bdbfad103e
93228a3d2fae1b3deb78283d10f4d13d02a1babd2c4598bd191c99bc6982
73bd1968493d1af3cd3cb31e45bd1cc4c93ce60f48bc8c3cb1bc8fbe63bd
9326f2bd76a6a8bdb4dccbbcf913c83dad9040bd6a55053e9b113bbd77d8
b43dd090433e58b3e63d6125d2bd9222b8bdda6bd63d6c2f213b5dc4ccb
21fa0f3e0be5ad3d178b073ec5bc593e6b2918be3565163dc6179ebd4741
d1bbb1a7ca3d93478abdfb9ce0bd3869b1bc03578b3b2151d7bc860aa2bd
67cc2b3e5d86d7bd2a6aedbd2649c43c4b522c3dcd63cc3dfb8a4d3c67b3
373dc699263eb34d223d56d1823ef097213c454824bd3cbc29bded3d1b3e
91111a3efd1a053e04bc113ceb74abbd8b5015bc48a8cd3d6d5cbd3dec17
62bddcd31dbd1c1e133d4d8595bd5888f4bd8a6e853d9948ca3c781ad33d
aac0863dc12b9cbd496a7fbbdb3c97f3b8432363dde95b9bda33fe23c6ebb
61bd75cca13d66d88f3d7f37673de3ccc6bd2fd5b63da92c9f3d39482fbd
ac8ca13dc5f391ba699f103efb3b593cf71927bd88e9d73d11f7b1bdeae
7c3ddeac1f3ccffa113e326cfabda869243e037d4e3ddb3ece3cfac8233d
38d9a7bd631899bd5290363de6fedfbd1048103de8d32bbe2a33eabdf296
97bdcbb5123d3c260c3e66658a3d42038abd98e31f3da09044bd3f61fbb
3f6c9bbded2fa73da55904ba80d0983d63b866be6eb0073cf13d43bdad9b
193df1f216be5bab13be4256063d86d6dd3d4c7c113cfbbddfbcce2abcbd
e4430d3df7ada1bcbd9516beb6db27be9222a3bd74d815bdcc0899bdc12a
5a3decba003e4cace73da07bd8bdaa9a203cf378013e7ea7703ce78bdfbd
d4b7e83dbe2c4a3d93e145bd9cf13b3d4b5b703d93e05d3cc516853de025
a73dcccfa13dcc2b473b82087f3c8d3d8dbc7f9ffa3d81210bbbb96e1abc
a3e6843da4bd263bd3d7373cfa33acbcad17ad3ce70e75bd0e17413e5eb1
72bd17ff693e5fe9bf3d183fe63bf2fa7fbd867362bec94a303db16748bd
b3ec9cbd2e5744bd550aa53c15920b3e6811f5bcd5f6543d23c9853d2c77
693cc41c9d3bea51a6bbfec7aabd521c1f3d6e9afdbd858f53bd2ae8773e
af2117be52312ebc685e60bdf9756c3cb19fdebd53f2fabceb03db3ce8a0
ae3d977b0a3d172ef2bc6592c7bc9de618bda7e6eed3e38203d38fc533e
1cd5003ddb4cf5bdf1d10a3cc7621c3ed5e717bd3ea30b3e3a2929bca9f7
8cbd5ae005bd6975b33dc79f7bbdf358693d2525cabca46fb143e9e879f3d
7679183e915794bd71cbbd3d0deac33ce68f56bef494afbdba76d83cb012
d1bd1bba4cbd8afaa93dd1ebed3d9086963d5359f43d07740ebeaeedabd
85af683c1d5d393d314d163d742459bca01301be8f0b99bdbd4398bd4e64
cfbb04dfaabd981811be0f2228bdd00fb7bd7a73f9bb71bb163efe814a3d
bd82a33d1cff48beba2d0ebe95b2213ddaa934bd86226b3dbb5554bd401a
a83cc198a0bdf5bc19be5c905c3bc1d9b03d30b7f53cd9d4713d71868c3d
a6f75bbc8f1e40bd4782adbd6465df3c29039c3cd4ec523d99b5eed3eef
f0bd5e2dddcbad70d63dd3daddbd3e476c3b39d1ca3ad40d0abe4bbba4bd
4146813dd66e023edeb6fe3d8dbad7bdf5ad25bcffcc873d11e45b3d3182
06bdd464b2bd43abc2bd212f0bbefc78e4bc88bb203ebd4cc53de328873c

Изм.	Лист	№ докум.	Подп.	Дата

РИБУ.05439-01 33 01

e509fbbdbaa6e2bc71e41fbe63455d3cb8beb1bd6b89c2bdc940fffbcebf4
56bd2b000a3edf59ebbcbd2fed3c7b18803e6d16823ba77006bd7a8b1fbd
805e343e3acf163e947c0a3e21da8f3be93c75bd422ec6bbcbded03dd58e
d43d8cb856bd01ed18bd2eae0f3d182ea5bd368908be3846683daa060f3d
e831d33de716563d2e6d8fbda88a8bbdb6ba633ce15a743d3455a7bd4bec
173df0c245bdf6e6bf3dc4deb43dc4503a3d72ebd5bd561c9c3daa609d3d
686f2abd192e9f3dd750503cea77223e0ea48a3b5f2236bd2186ee3d2a32
a0bd78908f3d89cc563c7cd6073e309103be823f243e3e1cef3c68ddfe3c
f6ca163d8f66acbdad9d90bd2b18743dae26febd35302c3d60512ebe25c5
e5bd29f095bdf767323d5b0e023e254e8d3d79dd89bd64d30d3d532f03bd
02d621bb739a9ebd6912963d0ed2973aae7f883d931156be6b6f073cb023
11bd973c733de2d71fbe90d406be128f303d7670f73dd461e33c0907a6bc
0f52cfbd5712823c57ee0abdcd4b12bef9842dbee1c8abbddacd20bd8cce
87bdc6f9323d3bc1ff3dc24dd33d8e22b6bd5d29953c1352053ec77db03c
020cb8bd709ec93d47d3473d4d6b26bd042a243da410743d0ac80e3db55e
693d2ef56f3db631e93d27b156ba095f793cc3291cbcd8f13de2d3843b
1cb4be38bcde653dd9588d3c25875c3c9d4db8bcf942ef3c944170bd6865
443e4e2086bd1f80633e9a9ce33d109a57bc3b2b91bdf05469be31083b3d
c4a51abde210aabd789222bd5a16cd3c3a4b0c3e886c8abad2974f3d2964
883d41a09f3b9575053d97d6bdbca3e2abbd9a0a3c3dfc6efbd59e562bd
4e667d3ed6a90ebe96e6bebcc1d432bd20a7683a0b01d7bd30c7b4bcab3c
1a3dd280ac3dd3b1fe3c3c4645bd44fa7abcb56e15bd96edd1bd43082a3d
9dce5a3e969dec3cf6b3e2bddb84c93b8274263e6efbbdbca681003e79e4
11bc581e8dbd77db18bd90eca63db0a057bdde625e3dce282cbcf2780a3e
c01b9d3d0b92083e7dbc85bde480d13d9d1db53ce5935cbeafc7a1bda112
1d3d3cc2c3bd388286bda1abc53d99eafa3d1d69843d693b003e76c705be
2064b3bde8d46e3c8184753dc9eeda3c153dc5bcbbcbe2bdcbf68ebd0e74
a8bde4b4f6bbbbb9397bdb59417bea6a828bd4009b5bd769db1bb4c8e183e
e5d68a3d080ab13d1ef44dbe8a5df4bd93e7303de95073bdfе884a3db6cc
65bdc01a63c0630b0bdd52616be73da2f3c2d70ab3dc81d683c3426893d
702d8a3d27d47b3a2cd31bbd6966bdbd0c09003d7653153d4704793dec9f
f4bde59ce8bd1f94d5bc9facf33d8e75dabdfc1bd0ba8c388fb9a6211bbe
915eb9bd3dd07f3d3996003e45d1fc3d4c48d8bddf118dbbd518823d476d
583dd349b0bc96fec8bd17cddabd26c90cbecf9c9bbc6bf72a3ec2a0b23d
7355853cc1bef5bd157bb3bc665b22bed162a23ca2aba4bdfе77a2bd87bc
cbbcf9758bdec0eed3d6a35b0bc47250e3d3bc77a3e57ec103c53bd1dbd
762336bd41c00c3e643a153e259bf83d3413793c33cf5dbdb4c148bcceba
d93d5ff2bc3dda043fbd0c3ed8bc7fc4463ddbc89abd4ae802bef967603d
758fd53c0d17dc3d7d04513d4db0b1bda8d085bd25ec373c57c94a3d2359
9abdd2fc2e3de6ac5dbd69639c3dd1f4993df587783db56ac8bdb7c2bd3d
47f3db3d967ad6bc7780ae3dad7bc038e8d5f83d4a1164bb326a21bdd50a
c53d24a8b8bdb5a843d19b29a3c4186063e6a57f9bd5fd4343e3be05a3d
8043183dbf58183df3e7d3bdacca95bddcca7c3dfdcfebbd01beee3c4fd5
34bec78e00be120fa9bd98a91b3d3660f63d7ee9a83d970289bd7d2d7c3d
7ca323bd8d39ff3bbb6ac3bd6dcc8b3d373f42bcbe267c3db4875fbe2d95
4fbc77d34bbd147e2b3d1c4ee3bd903416befe73273dc58bee3daa76af3c
3d530cbd8951d2bdb85c883c5931e7bcb4ae27bef5e61ebe3dabc1bdb7a5
1abd3e36c0bdcf777b3d7913df3d82a6b23d0015b0bd2e9c893cec3c003e
74f0a13c533c03be3bc3ed3d079d3a3d22c139bd9a87403ded662c3d64e5
de3c26cc1b3d56f2953d095ded3d89c9d93cea806e3c044310bc33140b3e
ea73a4bb87ae2bbc9dc88c3d2312083ddc11e33b605c09bdcab5023d7894
52bd97544e3e20358cbd0e955f3e37b9bf3da92fcbb83fa660bde24467be
b7e80d3d2d354fbdс7a2a4bd4efe50bdf765153d6728273e7535cdbc0b54
363d7d558a3d3c64823ca3dbd43cd935f0bc478d95bd76a0903d9d77d9bd
09df20bd43f2733ec57a11beb4c808bd96577dbdded6603c9125f1bd22c1

Изм.	Лист	№ докум.	Подп.	Дата

ПИБУ.05439-01 33 01

90bc37f32f3dfe6ca13d51505a3d7a6ebabccda4bebc8587fab6756c6bd
ea1efe3c993b5e3eb4d01e3de57ceabd72aaed3b2a78253e3d2907bd049e
053e9f569abc392351bdf0006bdb553793d20b1a6bd31a29b3d35ff05bd
2b90153e0e998f3d5a071c3e747291bddc9ed93df3d6ff3b0b3072be10a4
b3bdec76633c4311ddbd40d099bdc352a73d020bfe3dd3c6923d06c3c43d
60980fbe167589bd81e3833c2a303d3df60e373de90719bd8881f3bdf72f
89bd72b6b5bd77b3fbbbc0ee8cbd4efcf3bd5b6b16bdd1f2b8bd96476d3c
ffc4133e1df0593dcb3cc43dbd6743beec2aecbd58115d3d4c8e5cbd571c
813d61c405bdcb2b27bc30bc94bd814c11be488153bbd6d8b73dfea86b3c
4691933d00f8693d9d5528bca32234bd4dd897bd689ea83c753a563cbb0e
7a3d61dc05be7cd8e9bdb580843b1423d93d0276cebfd9a61abade05273c
a01e11be69c59abd0e417a3d90e0d03d6fe9013e2417c4bd9407cdbbcb3a
9e3d20526a3d40b307bd4439cabd1a13e7bd5c6916bef1f021bddb3e253e
13b5d93d199a873c89d8d9bd84784ebd0a791dbefac6353cdb0a9bdba5fa
a4bdf9debdbc575518bd49d2f43d10f0babc02ad053d8578813ef58e113c
842327bd372ffdbc5c55133eae3e1d3ec82c033e1d6de4ba7c84a8bd4722
80bcc8d5c33d1f6fb93dc05241bd31f425bddcae2f3d25f2a4bd51e1f0bd
5211953df5c9903c763ef03d63b38c3d6b82aabde77472bd4cff513cae82
193d25babcbd0eff2e3de36079bdd920953daf82883d1542533da71be4bd
80f7c63de92cac3d42e71abd5b6ead3dde1b373bc5cc0d3e5e3c9a3c7a86
43bdf62cb3dfa16c1bd513b643d6494663cb975093e6fcee3bd24ec253e
1250573d61f90b3d7856e83c30f9a4bdd3c2bfbdb0d4553d436feabd367a
0e3dbab532be6bd8c8bdca419ebdcd73573dbab8133ed6f1803db55e79bd
bc71133d5ce550bd57c2fa3beac094bd1a46af3dedfd8dbbaef8af3d45eb
5abe5ce8d23cb64d22bd01503f3dc84a1cbe500418beac02cb3ca5a8f63d
492d263cde94abbca0f8d3bdc42a183d917901bdd76a0bbef19f18beeb59
a3bd301751bd9572a4bddbd8483d7f38033ec591df3d47e0d8bd0c54073a
493af63d318b473c40b4d8bdd18ff43d291a413d42f10ebd3702303d4d80
5f3d03078c3c04aaa23d1997843d4eb1ea3da52c6e3cf1085d3cbb34b3bc
2ef8003ee3cb81bcc9956bbc9a06663d08f7bf3993c5323cf8ed2bbdf705
873c950a86bd7426473e326073bdfb355d3eddebb83d1975913b568393bd
b7005dbea9095f3d67d23bbd1c8993bdaf5558bdc296ab3c421c0c3e7a9e
b5bcbd41453dcb5f8e3d3d87e43c86b9ab3b287e4abc64ffc1bd85ad503d
1063f9bda3a564bdeaec6a3ecccd1dbe9464b8bc442c82bdf3e6e03b739c
ecbdf3491ebd54b8f43c39d3b13ddbd7ec3c939c4fbc044b04bd2deb3dbd
713bdbbd7ea3ef3cc426573e8df3ed3c61690dbe3e50553c6ea11a3ee79f
0fbd17b8043e543bddbb94397cbde02806bda16ead3d15ad93bdc42b953d
1d0fe5bcc8741d3ebc8d9e3d70d5183e120eadbd2ad7b13d256cc23c5f56
61be2c0cb7bd44a1e33c46c3e0bdde7d54bdc6a49f3d7b7e093e85f3813d
121be93d712d16bee32b90bd21a1153cfd3d063d439f333dfa441ebc5586
04bea9f583bd2b089ebd859e37bbc24a9ebdeae607be624d12bd8bdcb6bd
e87fecbbc84b1e3e21934c3d643db73daad646be6e89febd0c73093dcc68
07bdf0026c3d2af536bd1c73443bcfd596bd30d719be43ef8d3c4073ba3d
82b8133d9c6c823d56ef893dec5bb3bc5b8e28bdba1da0bd6d24083cc2da
813c896e503dd857f9bdf84ee1bd89b5c6bb6de7c53d178aeabd69fdad39
db8dbe3c9ec30ebe5e979cbd32d1513df582fa3dcab9023ef031d7bd48db
9ebc47a6913dafc82f3d434e27bd649e9cbddca0c3bdc89e0dbea107c8bc
4797223e965ab03d00d68b3c921bf2bde1a51cbd2d4522be79f73e3c9e9a
a1bd18eebfbd70a5bc98a515bd99f8013e5666fdbc

Изм.	Лист	№ докум.	Подп.	Дата

