REDS:

Телекоммуникационные устройства и системы

№2

2025

СОДЕРЖАНИЕ

АЛГОРИТМ КОМПЛЕКСНОГО МОНИТОРИНГА ДАННЫХ

Гадасин Денис Вадимович
МТУСИ, заместитель заведующего кафедры СИТиС, к.т.н., доцент
dengadiplom@mail.ru

Бессолицын Антон Дмитриевич
МТУСИ, Магистрант, гр. М092401(75)
besstoni2100@gmail.com

Серов Никита Романович
МТУСИ, студент, гр.БСТ2104
nikita-serov-2003@mail.ru

Гадасин Даниил Денисович
МТУСИ, студент, гр.БСТ2103

Аннотация
В данной работе представлен алгоритм мониторинга СУБД, позволяющий в реальном времени фиксировать изменения в сущностях, проводить автоматизированные проверки качества данных и разрешать конфликты. Предлагается принцип работы системы, формальная (математическая) модель взаимодействия транзакций и данных, а также анализируются временные характеристики (алгоритмическая сложность). Приведена блок-схема, демонстрирующая основные этапы работы алгоритма.

Ключевые слова
Мониторинг СУБД, проверка качества данных, разрешение конфликтов, асинхронная обработка, алгоритмы в СУБД

Введение

В современном мире базы данных (БД) являются фундаментом для хранения и обработки различных видов информации. От корректности и согласованности данных, хранящихся в СУБД, часто зависят стабильность и эффективность работы многих бизнес-процессов [1]. При этом рост объёмов данных и возрастание числа параллельных запросов приводят к необходимости постоянно контролировать целостность информации и быстро реагировать на любые нарушения или конфликты.

Для решения этих проблем важен комплексный подход к мониторингу СУБД, который выходит за рамки традиционного наблюдения за производительностью и предоставляет инструменты для автоматической проверки качества данных, выявления аномалий и управления конфликтами

Целью данной работы является разработка алгоритма комплексного мониторинга СУБД, который объединяет несколько ключевых функций: непрерывный контроль изменений в базе данных, проверку качества данных, управление конфликтами, сбор метрик и логирование [2].

Основной задачей является разработка алгоритма и архитектуры системы, которые одновременно обеспечивают баланс между производительностью транзакций и глубиной проверок, способны масштабироваться для обработки большого объема данных и растущего числа запросов, а также поддерживают гибкость настроек, позволяющую адаптироваться к специфическим требованиям различных систем.

Принцип работы алгоритма

Перед началом работы важно определить, что подразумевается под понятием «комплексный мониторинг» [3]. Комплексный мониторинг в системах управления базами данных представляет собой подход, направленный на всесторонний контроль состояния базы данных и хранимых в ней данных с целью обеспечения их качества и согласованности в режиме реального времени. Ключевыми аспектами комплексного мониторинга являются: многоуровневый анализ данных, реакция в реальном времени, управление конкурентным доступом и логирование и сбор метрик. Рассмотрим каждый аспект немного подробнее.

Многоуровневый анализ данных включает контроль за структурными изменениями [4], такими как удаление колонок или таблиц, а также оценку соответствия данных установленным правилам и ограничениям. Среди них можно выделить соблюдение уникальных индексов и корректность внешних ключей. При реакции в реальном времени система оперативно фиксирует изменения в базе данных, выполняя необходимые проверки и корректировки. Такой подход позволяет быстро обнаруживать и устранять ошибки, не дожидаясь их накопления, что минимизирует влияние сбоев на функционирование системы [5].

Рассматривая такой аспект, как управление конкурентным доступом, стоит отметить, что для предотвращения взаимных блокировок и конфликтов при одновременной работе нескольких пользователей или систем осуществляется мониторинг транзакций. Применяются стратегии разрешения конфликтов, например, «Последний записавший выигрывает» или приоритетность на основе источника."

Для выполнения логирования [6] и сбора метрик система фиксирует все события, включая успешные операции, выявленные ошибки и конфликты [7]. Собранные метрики используются для аналитики, определения трендов и предотвращения будущих проблем, что позволяет принимать обоснованные решения для улучшения работы базы данных.

Комплексный мониторинг в СУБД является важным инструментом управления данными. Он не только позволяет контролировать производительность системы, но и активно влияет на качество данных, минимизируя влияние ошибок и сбоев на бизнес-процессы и обеспечивая надежность и устойчивость работы.

Для выполнения комплексного мониторинга необходимо разработать алгоритм. На рисунке 1 представлена схема работы создаваемого алгоритма, позволяющая наглядно понять последовательность действий.

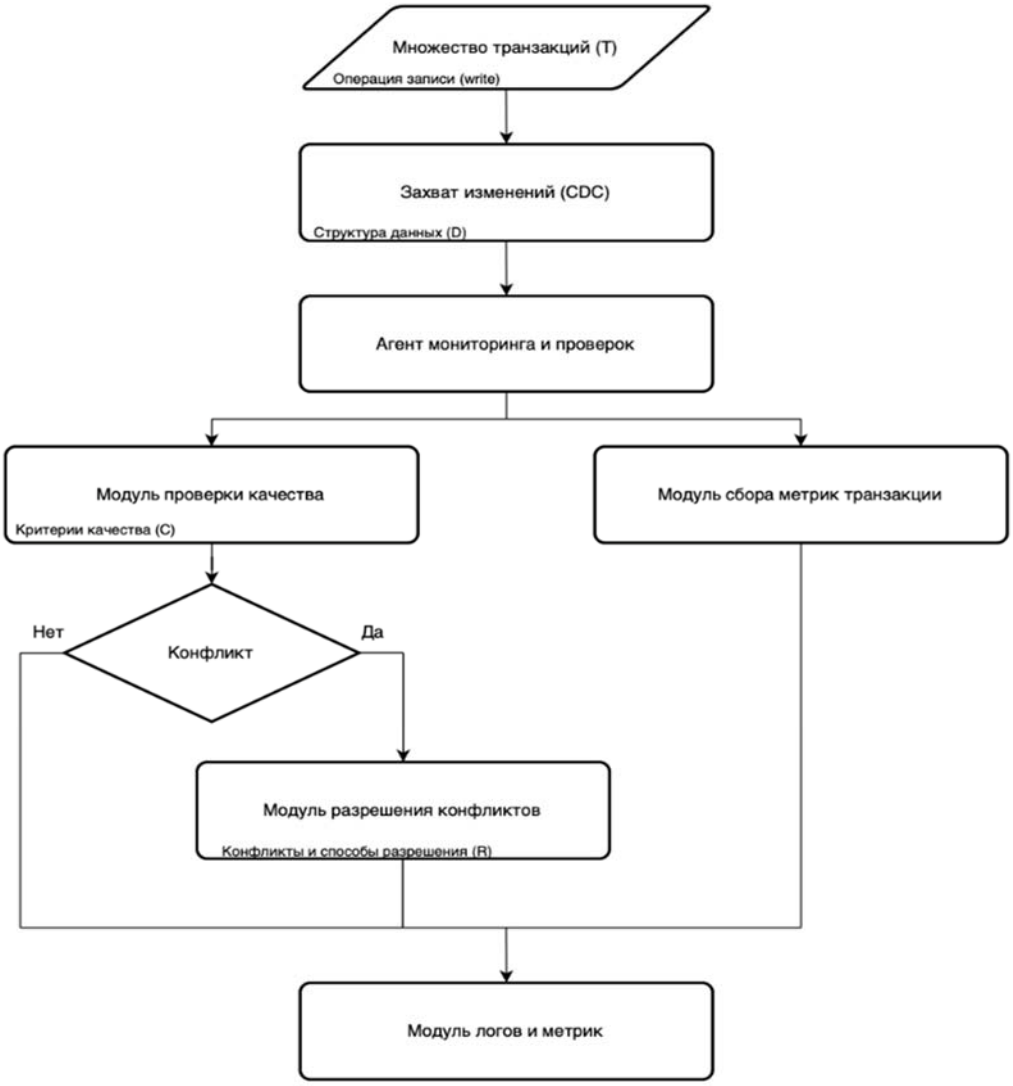

Рис. 1. Схема разрабатываемого алгоритма

Тригером для запуска алгоритма является транзакция, затрагивающая одну или несколько сущностей, таких как строки, документы и другие объекты данных [8, 25-32]. При совершении обновления, вставки или удаления механизмы CDC (Change Data Capture) формируют событие, фиксирующее произошедшие изменения, и передают его агенту мониторинга.

Агент мониторинга принимает это событие, проводит предварительный анализ данных и инициирует вызов модуля проверок для дальнейшей диагностики и модуля сбора метрик транзакции для будущего анализа и предсказания поведения транзакций. При необходимости модуль проверок обращается к базе данных с дополнительными SQL-запросами для проверки ссылочной целостности или получения контекстных данных, чтобы убедиться в корректности изменений.

Если в процессе проверки обнаруживаются конфликты или нарушения, инициируется переход к модулю разрешения конфликтов. В случае отсутствия проблем транзакция признается корректной и продолжается без вмешательства. Все результаты проверки, включая успешные операции без ошибок, фиксируются в хранилище логов и метрик. Эти данные используются для исторического анализа, построения отчетов и графического отображения различных показателей.

При обнаружении нештатных ситуаций, таких как грубые ошибки данных, массовые конфликты или технические сбои, формируются уведомления для операторов системы. Дополнительно соответствующие сигналы отображаются на дашбордах для оперативного реагирования и устранения проблем. Таким образом, система обеспечивает непрерывный контроль и поддержание целостности данных.

Чтобы понять, как разработанная система функционирует на практике, важно последовательно представить основные идеи и логику работы основных модулей [9].

1) Непрерывный захват изменений (Change Data Capture)

В основе алгоритма лежит постоянный контроль за любыми операциями над сущностями – будь то добавление новых записей, обновление существующих или их удаление.

Для этого в СУБД настраиваются триггеры или механизмы чтения журнала транзакций, позволяющие «подслушивать» процесс внесения изменений.

В ряде случаев используют специализированные инструменты (например, Debezium), которые автоматически переводят события о внесении изменений в удобный формат и передают их в брокер сообщений (Kafka, RabbitMQ) или непосредственно специальному модулю мониторинга.

2) Агент мониторинга и проверок

События о модификации передаются агенту мониторинга [10], который получает информацию о том, какая сущность затронута, какие поля были изменены и когда именно это произошло.

Агент анализирует тип операции (INSERT, UPDATE, DELETE) и исходя из этого запускает соответствующие проверки.

Важно, чтобы подобная логика работала асинхронно, не создавая дополнительных блокировок в основной базе: цель – быстро обнаружить возможные проблемы, не влияя на производительность транзакций [11].

3) Модуль проверки качества данных

После поступления сведений об изменении, система активирует серию проверок по заранее определённым правилам:

Проверяются форматы (например, поля-строки на соответствие регулярным выражениям или маскам);

Отслеживается наличие обязательных полей, которые не должны быть пустыми;

Проводятся корреляционные проверки на уровне целостности (внешние ключи, ссылки на связанные сущности) и логической непротиворечивости;

При необходимости учитываются временные метки, бизнес-ограничения, уникальные индексы и др.

Проверки могут выполняться в двух режимах: синхронном (до фиксации транзакции) и асинхронном. Синхронная схема даёт более высокую гарантию, что некорректные изменения вообще не попадут в базу, но при этом задерживает выполнение транзакции, что не всегда допустимо. Асинхронная схема позволяет системе работать быстрее, однако может означать, что некоторые «плохие» данные всё же временно окажутся в СУБД, но при этом будут оперативно выявлены.

4) Выявление и разрешение конфликтов

Конфликты возникают тогда, когда несколько транзакций стараются изменить одну и ту же строку или когда различные копии данных (например, при репликации) начинают расходиться между собой.

Большинство современных СУБД (таких как PostgreSQL, Oracle, MS SQL Server) имеют встроенные механизмы управления конкурентным доступом и дедлоками, но иногда требуется и дополнительная логика – например, для сложных сценариев распределённой базы или когда бизнес-правила требуют особых методов разрешения (Last Writer Wins, Source Priority).

Алгоритм организует хранение версий строк (MVCC) или ведёт отдельный журнал изменений, позволяя при необходимости откатывать неверные записи, сливать версии или отправлять уведомления администраторам для ручной проверки [12].

5) Сбор метрик, логирование и уведомления

Все выявленные события – будь то корректное прохождение проверок, нарушения или конфликты – протоколируются в специальном логе, который может храниться в отдельном индексе (Elasticsearch, Splunk) или системах вроде Prometheus.

При обнаружении серьёзных проблем система генерирует уведомления (e-mail, Slack, Teams и др.), а также формирует алерты в общие дашборды, чтобы администраторы могли оперативно реагировать [13].

Метрики, собираемые при этом, позволяют анализировать общее состояние базы, частоту и тип ошибок, время отклика, долю отклонённых изменений, динамику изменений за определённые интервалы.

Главная идея принципа работы – поддерживать баланс между скоростью обработки операций и качеством данных. Система должна быть достаточно «легковесной», чтобы не затормаживать транзакции, но при этом обеспечивать быстрый отклик на нарушение консистентности [14].

Математическая модель

Для более глубокого понимания описанного подхода полезно взглянуть на него через призму формальной модели, которая иллюстрирует основные взаимодействия транзакций, данных и набора проверок.

1) Множество транзакций

Пусть существует набор транзакций $T = \{T_1, T_2, \ldots, T_n\}$, где каждая T_j инициируется пользовательским или системным процессом и содержит конкретный набор операций чтения-записи над базовыми сущностями.

2) Структура данных

Определим множество сущностей $D = \{d_1, d_2, \ldots, d_m\}$ Под сущностями можно понимать строки в таблицах, документы в хранилищах NoSQL или набор полей в сервисах типа Redis. Каждая сущность d_i описывается набором атрибутов $A(d_i) = \{a_1, a_2, \ldots, a_k\}$ где a_x – наименование конкретного поля и его значение.

3) Операции чтения-записи

Read $\mathrm{Read}(d_i, a_x)$ обозначает операцию чтения значения атрибута a_x сущности d_i.

Write $\mathrm{Write}(d_i, a_x, v)$ означает установку (запись) значения v в атрибут a_x сущности d_i

Эти операции формируют основу любой транзакции: от простого SQL-запроса до распределённого изменения в кластерной среде.

4) Критерии качества (функция проверок)

Далее рассматривается набор булевых предикатов $C = \{c_1, c_2, \ldots, c_r\}$, каждый из которых отражает определённое бизнес-правило или требование к корректности данных. Например:

$c_1(d_i)$ может проверять, что поле «email» действительно содержит валидный адрес;

$c_2(d_i)$ проверяет, что поле «количество» не отрицательно;

$c_3(d_i)$ контролирует, что между таблицами (например, «заказы» и «клиенты») соблюдаются ссылочные ограничения.

Совокупная проверка для сущности d_i будет: $Q(d_i) = k = 1rc_k(d_i)$, где $\wedge$ означает логическое «И». Если хотя бы один предикат возвращает false, значит, данные не прошли проверку.

5) Конфликты и функция разрешения

Конкурентный конфликт: если две транзакции T_j и $T_j^{'}$ выполняют $\mathrm{Write}(d_i, a_x)$ одновременно или в пересекающихся интервалах, возникает возможность дедлока или нарушения целостности.

Логический конфликт: когда несколько независимых процессов (или реплик) пришли к противоречивым значениям для одного и того же атрибута.

Введём функцию разрешения: $R : (T, D) \rightarrow (T, D)$ которая анализирует текущее состояние транзакций и данных, выявляет конфликты и выбирает стратегию (например, отменить одну из транзакций, разрешить по схеме «последний записал – победил» и т. д.).

Таким образом, мы можем описать систему в каждый момент времени как пару $(T_{active}, D_{current})$ Под T_{active} понимается множество активных (ещё не завершённых) транзакций, а $D_{current}$ – текущее состояние всех сущностей с учётом сделанных изменений. При срабатывании проверки Q выявляются несоответствия, и если они критические или пересекаются с конфликтами, то вызывается R, возвращающий новое согласованное состояние системы.

Алгоритмическая сложность

Рассмотрим основные этапы алгоритма и оценим, как растут затраты времени при увеличении числа операций и правил проверки.

1) Фиксация изменений

Встроенные триггеры или механизмы журналирования СУБД обычно работают за $O(1)$ или O(log n) на операцию, так как регистрируют факт изменения (операцию INSERT/UPDATE/DELETE) и вносят информацию в очередь или лог [15-16].

Если используется внешний брокер (Kafka, RabbitMQ), добавление события в очередь, как правило, имеет сложность порядка $O(\log \textit{размерочереди})$, но в реальных системах почти всегда это эффективно «замаскировано» под константу за счёт внутренних оптимизаций.

2) Проверки качества данных

Если на один объект d_i приходится r правил и каждое проверяется в худшем случае за константное время (например, сопоставление с регулярным выражением, доступ по индексу и т.д.), то общая сложность — O(r) для одного изменения.

Для более сложных видов проверки (корреляция между разными таблицами, агрегированные данные) время может увеличиться до $O(r \log m)$ или даже $O(r * m)$, если требуется полный обход. Однако на практике стараются использовать индексы и денормализацию, чтобы удержаться ближе к логарифмической сложности по числу записей [17-18].

3) Обработка конфликтов

Выявление взаимоблокировок (deadlock detection) внутри СУБД обычно выполняется за O(n) или O(nlogn) где n — число активных транзакций. Но поскольку механизм встроен в систему, разработчикам зачастую не приходится реализовывать это самостоятельно [19-20].

Логические конфликты могут потребовать дополнительного сравнения версий или обращения к распределённым узлам. Предположим, что за счёт индексирования и распределённых журналов это также занимает порядка O(logm).

4) Запись результатов и метрик

Сохранение результатов проверок и конфликтов в служебной таблице или внешней системе журналирования обычно происходит за O. С ростом системы это может заметно увеличиваться, но на практике применяют шардирование, ротацию и архивацию журналов.

В целом, при правильном подходе можно удерживать алгоритмическую сложность на уровне от O® до O(rlogm), для единичной операции изменения, что даёт приемлемую производительность даже при больших нагрузках, особенно если проверки выполняются параллельно или асинхронно [21-22].

Реализация основных модулей на Python для PostgreSQL

В рамках данной работы были разработаны два основных модуля алгоритма: модуль проверки качества данных и модуль обработки конфликтов. Эти модули представляют собой базовый функционал, обрабатывающий основные сценарии, и служат первичной основой для дальнейшего расширения алгоритма.

Модуль проверки качества данных отвечает за обеспечение целостности и корректности данных. Он выполняет проверки форматов полей (например, валидация email-адресов), контролирует наличие обязательных полей и проверяет целостность внешних ключей [23-24]. Эти функции помогают идентифицировать возможные проблемы на уровне данных и минимизировать риск их некорректного использования.

Модуль разрешения конфликтов нацелен на обработку ситуаций, связанных с нарушением уникальности данных, таких как дублирование значений уникальных индексов. Основной стратегией модуля является использование подхода "Last Writer Wins", где конфликты разрешаются на основе временных меток, что позволяет сохранять наиболее актуальные данные. Также модуль фиксирует нарушения для последующего анализа.

На текущем этапе разработанный код охватывает только самые основные сценарии и нуждается в дальнейшем развитии для обработки более сложных случаев. В будущем эти модули будут доработаны и интегрированы в состав основного алгоритма.

Ниже на рисунках 2-5 представлены фрагменты кода реализованных функций.

```python
import psycopg2
import re
from psycopg2 import sql

class ConflictResolution:
    """
    Модуль разрешения конфликтов.
    Обрабатывает основные виды конфликтов: дедлоки, логические расхождения, конкуренцию за ресурс.
    """

    def __init__(self, db_conn):
        self.conn = db_conn

    def resolve_conflict(self, table, record, operation):
        """
        Основной метод для разрешения конфликтов.
        :param table: Таблица, в которой произошел конфликт
        :param record: Данные записи, вызвавшей конфликт
        :param operation: Тип операции (INSERT, UPDATE, DELETE)
        :return: bool
        """
        try:
            if operation in ("INSERT", "UPDATE"):
                # Проверка на уникальность значения уникальных индексов
                if not self._check_unique_constraints(table, record):
                    self._handle_violation("Уникальный индекс нарушен", table, record)
                    return False

                # Проверка временных меток для определения Last Writer Wins
                if not self._resolve_last_writer_wins(table, record):
                    self._handle_violation("Конфликт временных меток", table, record)
                    return False

                return True

            except Exception as e:
                print(f"Ошибка при разрешении конфликта: {e}")
                return False
```

Рис. 2. Код модуля обработки конфликтов

```python
def _check_unique_constraints(self, table, record):
    """
    Проверяет уникальные индексы на конфликт.
    """
    unique_fields = ["id", "email"]  # Поля, которые должны быть уникальными
    for field in unique_fields:
        if field in record:
            query = sql.SQL("SELECT 1 FROM {table} WHERE {field} = %s LIMIT 1;").format(
                table=sql.Identifier(table),
                field=sql.Identifier(field)
            )
            with self.conn.cursor() as cur:
                cur.execute(query, (record[field],))
                if cur.fetchone():
                    return False
    return True

def _resolve_last_writer_wins(self, table, record):
    """
    Реализует стратегию Last Writer Wins на основе временных меток.
    """
    timestamp_field = "updated_at"
    if timestamp_field in record:
        query = sql.SQL("SELECT {timestamp_field} FROM {table} WHERE id = %s;").format(
            table=sql.Identifier(table),
            timestamp_field=sql.Identifier(timestamp_field)
        )
        with self.conn.cursor() as cur:
            cur.execute(query, (record["id"],))
            existing_record = cur.fetchone()
            if existing_record and existing_record[0] >= record[timestamp_field]:
                return False
    return True

def _handle_violation(self, reason, table, record):
    """
    Обрабатывает нарушение, записывая лог.
    """
    print(f"Конфликт в таблице {table}: {reason}. Данные: {record}")
```

Рис. 3. Код модуля обработки конфликтов

```python
class DataQuality:
    """
    Модуль проверки качества данных.
    Выполняет проверки: целостность данных, форматы, уникальность.
    """

    def __init__(self, db_conn):
        self.conn = db_conn

    def check_data_quality(self, table, record, operation):
        """
        Проверяет качество данных для указанной записи.
        :param table: Таблица, где проверяются данные
        :param record: Проверяемая запись
        :param operation: Тип операции (INSERT, UPDATE, DELETE)
        :return: bool
        """
        try:
            # Проверка форматов данных
            if not self._check_formats(record):
                return False

            # Проверка наличия обязательных полей
            if not self._check_required_fields(table, record):
                return False

            # Проверка ссылочной целостности
            if not self._check_foreign_keys(table, record):
                return False

            return True

        except Exception as e:
            print(f"Ошибка при проверке качества данных: {e}")
            return False

    def _check_formats(self, record):
        """
        Проверяет форматы полей.
        """
        if "email" in record:
            email_regex = r"^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$"
            if not re.match(email_regex, record["email"]):
                print("Некорректный формат email")
                return False

        return True
```

Рис. 4. Код модуля проверки качества данных

```python
def _check_required_fields(self, table, record):
    """
    Проверяет наличие обязательных полей.
    """
    required_fields = {
        "users": ["id", "name", "email"],
        "orders": ["id", "user_id", "amount"]
    }
    for field in required_fields.get(table, []):
        if field not in record or record[field] is None:
            print(f"Отсутствует обязательное поле: {field}")
            return False

    return True

def _check_foreign_keys(self, table, record):
    """
    Проверяет целостность внешних ключей.
    """
    foreign_keys = {
        "orders": {"user_id": ("users", "id")}
    }
    if table in foreign_keys:
        for fk, (ref_table, ref_field) in foreign_keys[table].items():
            if fk in record:
                query = sql.SQL("SELECT 1 FROM {ref_table} WHERE {ref_field} = %s;").format(
                    ref_table=sql.Identifier(ref_table),
                    ref_field=sql.Identifier(ref_field)
                )
                with self.conn.cursor() as cur:
                    cur.execute(query, (record[fk],))
                    if not cur.fetchone():
                        print(f"Нарушение ссылочной целостности: {fk}={record[fk]}")
                        return False
    return True
```

Рис. 5. Код модуля проверки качества данных

Заключение

Представленный подход к мониторингу СУБД сочетает в себе несколько важных свойств:

Реакция в режиме реального времени. Система быстро реагирует на любые операции изменения данных, практически не откладывая проверку в «долгий ящик». Это даёт возможность вовремя отсеять или исправить некорректные значения, предотвращая их широкое распространение.

Гибкость механизма проверок. Набор правил и предикатов легко адаптируется под потребности конкретного проекта. Одним системам важны проверки форматов и непротиворечивость справочников, другим – сложные межтабличные согласования и контроль временных меток.

Полуавтоматическое и автоматическое разрешение конфликтов. За счёт встроенных в СУБД механизмов блокировок и дедлок-детектора многие конфликтные ситуации решаются «на лету». Дополнительная логика (например, Last Writer Wins) позволяет гибко настраивать процесс в распределённых сценариях.

Минимизация нагрузки на производительные контуры. Большая часть проверок и анализ аномалий вынесена в асинхронные процессы, что даёт возможность не оказывать существенного влияния на основные операции записи и чтения. При желании в критически важных случаях можно «подключить» синхронный режим, но это должно решаться на уровне бизнес-требований.

Масштабирование. Использование брокеров сообщений, распределённых журналов и внешних систем хранения метрик помогает обслуживать растущее количество транзакций. Система легко масштабируется горизонтально, если необходимо обрабатывать всё большие и большие объёмы данных.

С точки зрения формальной модели, базовые операции чтения/записи, набор проверок Q и функция разрешения конфликтов R создают фундамент, который можно развивать и усложнять – например, добавляя механизм машинного обучения для прогнозирования коллизий или динамического анализа качества (Data Drift Detection). Однако уже в текущем виде алгоритм решает ряд практических задач, связанных с обеспечением целостности и актуальности данных в условиях высокой нагрузки.

Таким образом, интеграция данного алгоритма в корпоративные системы даёт организации уверенность в том, что данные остаются корректными, согласованными и не приводят к сбоям в самых ответственных узлах инфраструктуры. С учётом бурного роста объёмов информации и усложнения требований к её качеству, подобные решения становятся всё более востребованными и значимыми для успеха любого бизнеса и для стабильной работы критически важных сервисов.

Литература

1. *Гадасин Д.В., Шведов А.В., Пантелеева К.А., Гадасин Д.Д.* Определение порога количества информации для возможности структурирования данных // Телекоммуникационные и вычислительные системы : Юбилейный сборник трудов тридцатого международного научно-технического форума, Москва, 12-15 декабря 2022 г. М.: Издательство МБА, 2022. С. 125-130. EDN MYMHUP

2. *Гадасин Д.В., Бессолицын А.Д.* Виды и методы структурирования данных из различных информационных систем: анализ и применение // Актуальные проблемы и перспективы развития экономики, Симферополь – Гурзуф, 12-14 октября 2023 года. Симферополь: ИП Зуева Т. В., 2023. С. 202-204. EDN UGZRXL.

3. *Гадасин Д.В., Каледина А.В.* Использование современных средств мониторинга для анализа состояния IT-систем // Технологии информационного общества : Сборник трудов XIV Международной отраслевой научно-технической конференции, Москва, 18-19 марта 2020 г. М.: Издательский дом Медиа Паблишер, 2020. С. 267-269. EDN XAPWGD

4. *Гадасин Д.В., Шведов А.В., Мелькова Е.К.* Структурирование данных исходя из центра масс // Актуальные проблемы и перспективы развития экономики : Труды XXI Международной научно-практической конференции, Симферополь-Гурзуф, 20-22 октября 2022 г. Симферополь: Крымский федеральный университет им. В.И. Вернадского, 2022. С. 266-268. EDN RFCCST

5. *Шульпина П.Д., Гадасин Д.В., Тремасова Л.А.* Взвешивание признаков как Предварительная обработка исходных наборов данных // Системы синхронизации, формирования и обработки сигналов. 2024. Т. 15, № 3. С. 40-47. EDN BIOWRB

6. *Гадасин Д.В., Шведов А.В., Каледина А.В., Юдина А.А.* Мониторинг и анализ log-файлов инфокоммуникационной сети с помощью комплексного инструментария elk Stack // Актуальные проблемы и перспективы развития экономики : Труды XVIII Всероссийской с международным участием научно-практической конференции, Симферополь-Гурзуф, 24-26 октября 2019 г. / Под редакцией Н.В. Апатовой. Симферополь-Гурзуф: ИП Зуева Т.В., 2019. С. 297-298. EDN SAOLRI

7. *Гадасин Д.В., Бессолицын А.Д., Гадасин Д.Д.* Оценка качества данных информационных систем // DSPA: Вопросы применения цифровой обработки сигналов. 2024. Т. 14, № 2. С. 4-12. EDN GYIWJU

8. *Пантелеева К.А., Палибза С.А., Гадасин Д.В.* Принципы построения системы управления при возникновении сбоев в ит-инфраструктуре // REDS: Телекоммуникационные устройства и системы. 2024. Т. 14, № 2. С. 24-34. EDN MOYCNG

9. *Гадасин Д.В., Шведов А.В., Савкин Д.И.* Проектирование единого информационного пространства в рамках курса лабораторных работ по дисциплине «Принципы построения систем управления базами данных и знаний» // Методические вопросы преподавания инфокоммуникаций в высшей школе. 2023. Т. 12, № 2. С. 14-21. EDN FLOPUM

10. Свидетельство о государственной регистрации программы для ЭВМ № 2022662724 Российская Федерация. Программное приложение "Анализатор текстовых данных" ("Text Data Analyzer" - на английском языке) для выполнения лабораторных работ студентами вузов по дисциплине "Мультимедийные информационные системы" : № 2022661351 : заявл. 21.06.2022 : опубл. 07.07.2022 / В. А. Докучаев, В. В. Маклачкова, Д. В. Гадасин

и др. ; заявитель Общество с ограниченной ответственностью Фирма «ТЕЛЕСОФТ». EDN DVURCM

11. *Марченко Д.О., Клыгина О.Г., Гадасин Д.В., Шведов А.В*. Обеспечение механизмов балансировки нагрузки в сетях с сегментной маршрутизацией на основе данных мониторинга // Перспективные технологии в средствах передачи информации : материалы 14-ой международной научно-технической конференции, Владимир, 06-07 октября 2021 года. Владимир: Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых, 2021. С. 419-422. EDN ZSCNIR

12. *Золотарева П.Ю., Гадасин Д.В., Маклачков К.А.* Методы обработки информации в распределенных информационных системах // Тенденции развития Интернет и цифровой экономики : Труды VI Международной научно-практической конференции, Симферополь-Алушта, 01-03 июня 2023 г. Симферополь: ИП Зуева, 2023. С. 187-189. EDN LGONZK

13. *Яковенко Н.В., Гадасин Д.В., Коцич Л.* Повышение точности коэффициента влияния ошибок в информационных системах с применением метода обратного распространения ошибки // Системы синхронизации, формирования и обработки сигналов. 2024. Т. 15, № 4. С. 35-42. EDN CMFVNH

14. *Гадасин Д.В., Гадасин В.А*. Система комплексной оценки информационных технологий // Технологии информационного общества : X Международная отраслевая научно-техническая конференция: сборник трудов, Москва, 16-17 марта 2016 г. М: Издательский дом Медиа Паблишер, 2016. С. 18-21. EDN VPDXOX

15. *Гадасин Д.В., Шведов А.В*. Применение транспортной задачи для балансировки нагрузки в условиях нечеткости исходных данных // T-Comm: Телекоммуникации и транспорт. 2024. Т. 18, № 1. С. 13-20. DOI 10.36724/2072-8735-2024-18-1-13-20. DN WKNPIX

16. *Гадасин Д.В.* Построение бинарного дерева минимальной цены // T-Comm: Телекоммуникации и транспорт. 2024. Т. 18, № 11. С. 38-44. DOI 10.36724/2072-8735-2024-18-11-38-44. EDN GMCEWG

17. *Гадасин Д.В., Вакурин И.С., Тремасова Л.А*. Алгоритм распределения данных между системами хранения на основе свойства самоподобия // Электросвязь. 2024. № 4. С. 44-50. DOI 10.34832/ELSV.2024.53.4.015. EDN BRSLCL

18. *Тремасова Л.А., Первухина А.А., Гадасин Д.В.* Использование методов Косарайю и k-средних для формирования кластеров // Электросвязь. 2024. № 9. С. 47-55. DOI 10.34832/ELSV.2024.58.9.007. EDN DOZTZK

19. *Гадасин Д.В., Назаренко С.С., Тремасова Л.А*. Особенности проведения практических занятий по дисциплине «Принципы построения систем управления базами данных и знания» // Методические вопросы преподавания инфокоммуникаций в высшей школе. 2023. Т. 12, № 1. С. 21-31. EDN FGSGBK

20. *Аль Мусави О.А.Р., Кравец О.Я*. Исследование алгоритмов повторной оптимизации запросов в облачных базах данных // Решение. 2022. Т. 1. С. 168-171.

21. *Gadasin D.V., Melnikova E.M., Palibza S.A., Gadasin D.D.* An Algorithm for Distributing Data between Storage Systems Based on the Property of Self-Similarity // 2024 Systems of Signals Generating and Processing in the Field of on Board Communications, Moscow, Russian Federation, 2024, pp. 1-7, doi: 10.1109/IEEECONF60226.2024.10496761.

22. *Salloum S.* et al. Big data analytics on Apache Spark //International Journal of Data Science and Analytics. 2016. Т. 1. С. 145-164.

23. *Кузнецов С.Д.* Основы современных баз данных. Информационно-аналитические материалы: [Электрон.ресурс] – Режим доступа: http://www.citforum.ru/database/osbd/contents.shtml. Дата доступа: 20.12.2024.

24. *Роб П., Коронелл К*. Системы баз данных: проектирование, реализация и управление : Пер. с англ., 5-е изд. СПб.: БХВ-Петербург, 2004. 1024 с.

25. *Гадасин Д.В., Шведов А.В., Кузин И.А.* Трехмерная реконструкции объекта по одному изображению с использованием глубоких свёрточных нейронных сетей // T-Comm: Телекоммуникации и транспорт. 2022. Т. 16, № 7. С. 29-35. DOI: 10.36724/2072-8735-2022-16-7-29-35 EDN: YTLCNW

26. *Shvedov A.V., Gadasin D.V., Alyoshintsev A.V.* Segment routing in data transmission networks // T-Comm. 2022. Vol. 16. No. 5, pp. 56-62. DOI: 10.36724/2072-8735-2022-16-5-56-62 EDN: VAYLJQ

27. *Назаров М.Д., Шведов А.В.* Корреляция атрибутов соглашения об уровне обслуживания с основными параметрами QoS в корпоративных сетях // Телекоммуникации и информационные технологии. 2020. Т. 7. № 2. С. 73-79. EDN: VQHDTJ

28. *Kalmykov N.S., Dokuchaev V.A.* Segment routing as a basis for software defined network // T-Comm. 2021. Т. 15. № 7. С. 50-54. EDN: LYVZCV

29. *Dokuchaev V.A., Maklachkova V.V., Statev V.Yu.* Classification of personal data security threats in information systems // T-Comm. 2020. Т. 14. № 1. С. 56-60. EDN: QOGYHH

30. *Докучаев В.А., Маклачкова В.В., Статьев В.Ю.* Цифровизация субъекта персональных данных // T-Comm: Телекоммуникации и транспорт. 2020. Т. 14. № 6. С. 27-32. EDN: XVWYJP

31. *Pavlov S.V., Dokuchaev V.A., Mytenkov S.S.* Model of a fuzzy dynamic decision support system // T-Comm. 2020. Т. 14. № 9. С. 43-47. EDN: VYFNLB

32. *Кузин И.А., Гадасин Д.В.* Модель контейнера данных для минимизации трафика при передаче субъективных характеристик объектов на изображении трехмерной сцены // Телекоммуникации и информационные технологии. 2021. Т. 8. № 2. С. 96-100. EDN: TYFFBH

ИССЛЕДОВАНИЕ МЕТОДОВ ПРОГНОЗИРОВАНИЯ ПОКАЗАТЕЛЕЙ БИЗНЕС-ПРОЦЕССОВ

Фатхулин Тимур Джалилевич
Московский технический университет связи и информатики, доцент кафедры МК и ИТ, к.т.н., Москва, Россия
t.d.fatkhulin@mtuci.ru

Зозуля Ирина Сергеевна
Московский технический университет связи и информатики, студентка группы МБД2331, Москва, Россия

Аннотация
Данная работа представляет анализ существующих методов прогнозирования показателей бизнес-процессов. Показано, что прогноз показателей стал мощным инструментом для создания успешной деятельности компаний и организаций. В статье рассматриваются различные аспекты использования методов прогнозирования показателей бизнес-процессов: экстраполяция временных рядов, регрессионный анализ, метод экспертных оценок, метод сценариев и другие.

Ключевые слова
Прогнозирование, показатели, бизнес-процессы, качественные методы, количественные методы, машинное обучение

Введение

В условиях рыночной экономики, когда информация о потребностях в услугах ограничена, а действия конкурентов и партнёров трудно предугадать, возрастает важность прогнозирования.

Главная задача прогноза – разработать стратегии для достижения целей компании и инструменты управления, которые помогут эффективно вести деятельность компании от текущего положения к желаемому будущему [6-10, 19]. Исследования в данной области имеют давнюю историю, а с возросшими объемами данных и появлением методов машинного обучения задача повышения эффективности прогноза стала особенно актуальной.

На текущий момент разработано множество алгоритмов и готовых продуктов, а крупные компании разрабатывают собственные алгоритмы, учитывающие все особенности в применяемой ими области. В данном исследовании рассмотрим основные методы прогнозирования показателей бизнес-процессов и их применение на практике.

Особенности прогнозирования показателей бизнес-процессов

Основная цель анализа и прогнозирования показателей бизнес-процессов заключается в получении результатов для последующего совершенствования этих процессов [11-15, 19]. Измерение и анализ показателей процесса являются ключевыми инструментами для определения способов улучшения процессов.

Каждый бизнес-процесс характеризуется следующими показателями:

1. показатели процесса;
2. показатели продукта (услуги);
3. показатели удовлетворённости потребителя.

Показатели процесса представляют собой численные значения, описывающие ход бизнес-процесса и связанные с ним затраты (время, финансы, ресурсы и т.д.)

Показатели продукта – это численные значения, характеризующие продукт как результат выполнения процесса (объём услуг, количество ошибок, номенклатура услуг и т.д.).

Показатели удовлетворённости потребителя – это численные значения, отражающие степень удовлетворённости клиента результатами процесса (выход, услуга и т. д.).

Рисунок 1 показывает разделение показателей процессов на *качественные* и *количественные*. Обычно качественные оценки не применяются, поскольку они затрудняют принятие обоснованных решений. Количественные показатели делятся на два типа: *абсолютные* и *относительные*. *Абсолютные*

оценки содержат следующие показатели: время выполнения какого-либо процесса, технические показатели, оценка качества и стоимости. *Относительные оценки* могут формироваться на основе абсолютных путём создания различных связей между ними [1, 15-19].

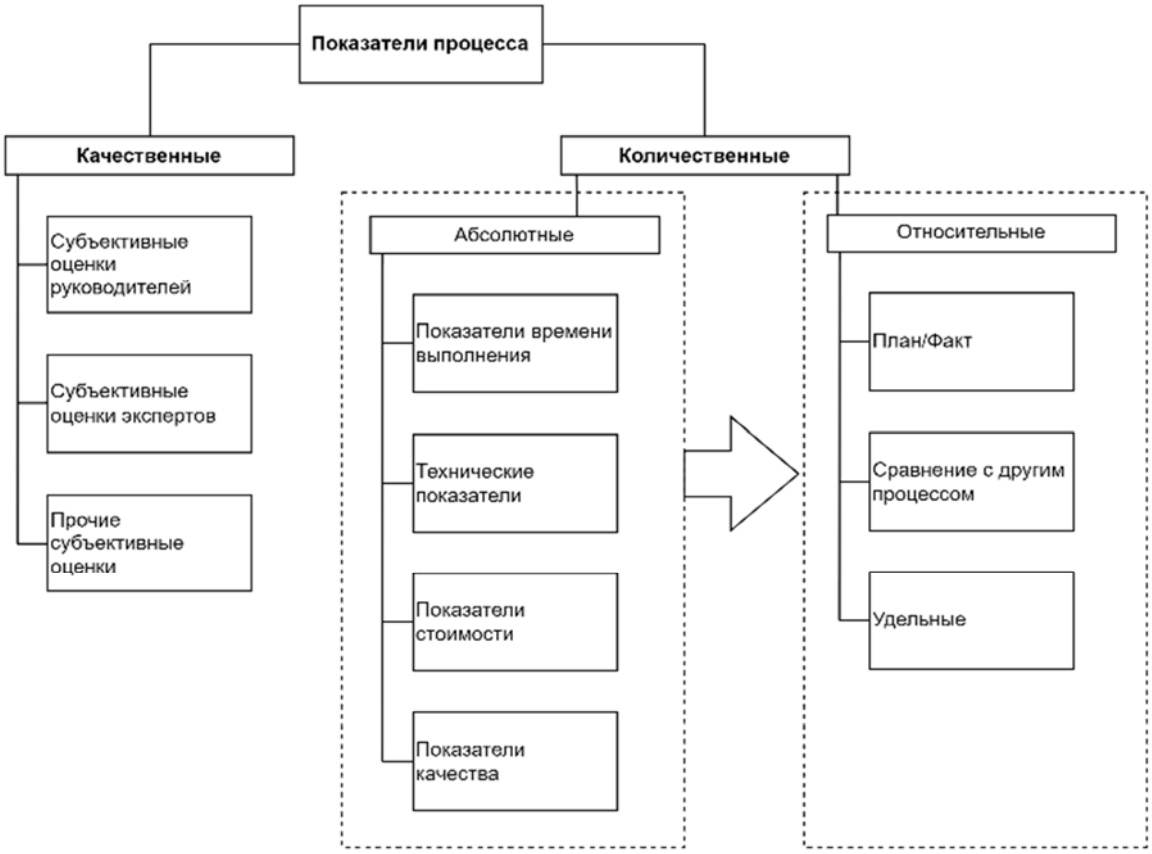

Рис. 1. Классификация процессов

Методы анализа и прогнозирования показателей бизнес-процессов

Анализ бизнес-процессов направлен на определение фактической эффективности конкретного бизнес-процесса [15, 19]. Главная задача этого анализа – представить текущую систему и найти способы её оптимизации.

Обычно анализ бизнес-процессов включает следующие задачи:

1) повышение прозрачности системы и эффективности работы сотрудников;
2) оптимизация работы системы и внедрение автоматизации;
3) разработка новых подходов для создания эффективной структуры организации;
4) распространение и масштабирование бизнеса.
5) улучшение качества работы с персоналом и финансового положения компании.
6) увеличение рыночной стоимости и возможность выхода на другие рынки.

В зависимости от специфики предприятия анализ и оценка бизнес-процессов могут быть направлены на решение одной или нескольких задач одновременно.

К методам качественного анализа относят:

1. Качественный анализ на основе экспертного мнения:

a. *SWOT-анализ*, направленный на выявление сильный и слабых сторон организации, а также возможностей улучшения и угроз ухудшения деятельности данного предприятия [2]: метод стратегического планирования, который помогает проанализировать сильные и слабые стороны компании, а также возможности и угрозы внешней среды.

b. *Анализ проблем процесса*: метод, который используется для выявления и устранения проблем в бизнес-процессах компании. Он помогает определить слабые места и разработать стратегии для их улучшения.

c. *Ранжирование процессов*: процесс классификации бизнес-процессов компании по степени их важности и приоритетности. Данный метод помогает определить, какие процессы требуют

наибольшего внимания и ресурсов для оптимизации и улучшения.

2. Визуальный качественный анализ графических схем:

a. *Анализ входов/выходов*: метод, который позволяет определить достаточность или избыточность входных данных и результатов процесса, а также структурировать бизнес-процесс и устранить «узкие места».

b. *Анализ функций:* метод, который позволяет определить и структурировать основные функции бизнес-процесса, а также выявить возможные проблемы и недостатки.

c. *Анализ ресурсов*: метод, который позволяет оценить достаточность и эффективность использования ресурсов, необходимых для выполнения бизнес-процесса, и определить возможные пути оптимизации их использования.

3. Анализ состояния бизнес-процессов по отношению к требованиям

a. *По отношению к типовым требованиям* включает в себя сравнение текущих бизнес-процессов с регламентами компании, международными стандартами и законодательными положениями. Этот анализ направлен на выявление сильных и слабых сторон бизнес-процессов, поиск возможностей для улучшения и соответствия установленным требованиям.

b. *По отношению к нормативным актам* предполагает сравнение текущих бизнес-процессов с типовыми требованиями и законодательством. Этот анализ помогает выявить несоответствия и определить необходимость корректировки бизнес-процессов для их приведения в соответствие с нормативными актами.

К методам количественного анализа относят:

1. Анализ показателей эффективности процессов: сбор и обработка информации о процессах, происходящих в организации, с целью определения степени их результативности и эффективности. Он включает в себя изучение входных и выходных данных, времени выполнения, ресурсов, используемых для выполнения процессов, и других факторов, влияющих на их успешность.

2. Анализ удовлетворенности клиентов процесса: оценка восприятия клиентами качества, сервиса и уровня обслуживания, предоставляемого в рамках определённого процесса или услуги. Он проводится с целью выявления сильных и слабых сторон процесса, определения областей для улучшения и повышения общей удовлетворённости клиентов.

3. Сравнительный анализ процессов: систематическое сравнение двух или более процессов с целью выявления их сходств и различий. Он используется для оценки и сравнения различных аспектов процессов, например, продукции, бизнес-процессов, финансовых показателей или маркетинговых стратегий. Сравнительный анализ помогает компаниям принимать обоснованные решения, минимизировать риски, повышать эффективность и улучшать результаты деятельности.

4. Имитационное моделирование процесса: метод исследования, при котором изучаемая система заменяется моделью, с достаточной точностью описывающей реальную систему. С этой моделью проводятся эксперименты с целью получения информации об этой системе.

5. ABC – анализ процесса: метод классификации и ранжирования ресурсов, таких как товары, контрагенты или клиенты, в зависимости от их вклада в формирование прибыли компании.

Этот анализ предполагает разделение ресурсов на три группы:

1) группа A – самые ценные активы, которые составляют 20% ассортимента и обеспечивают 80% дохода;

2) группа B – промежуточные позиции, занимают около трети ассортимента и дают 15% прибыли;

3) группа C – аутсайдеры, составляют примерно половину ассортимента и имеют вклад в продажи не более 5%.

Также выделяют методы планирования и прогнозирования по степени формализации.

Методы машинного обучения в задаче прогнозирования показателей бизнес-процессов

Одним из главных преимуществ является способность алгоритмов машинного обучения обрабатывать сложные и большие наборы данных [3], автоматически корректировать модель при изменении ключевых факторов и получать быстрые результаты.

Алгоритмы машинного обучения все еще имеют ряд ограничений [4]. Они лучше всего работают при обучении на больших объёмах данных, и качество собранных данных напрямую влияет на создаваемую модель. Чтобы повысить достоверность, данные должны соответствовать критериям полноты, согласованности и отсутствия ошибок.

Машинное обучение можно определить как класс методов в области интеллектуальной обработки данных, характеризующийся обучением на примерах решения множества сходных задач. Этот

инструмент основан на статистическом анализе существующих решений и последующем решении задач на основе проведённого анализа.

Статистические модели, используемые в машинном обучении, были известны задолго до его появления, но возможность их полноценного использования стала возможной только недавно благодаря появлению доступных программных и технических ресурсов.

Линейная регрессия – это базовая модель машинного обучения, основанная на анализе одной независимой переменной и взаимосвязи между зависимыми и независимыми переменными. Она используется давно и является основой для применения машинного обучения.

Другие количественные методы, такие как регрессия и анализ временных рядов, являются более систематичными и надёжными. Методы регрессии связаны с определением причинно-следственных связей между независимыми и зависимыми переменными.

Модели «Дерева решений» и *«Случайный лес»* используются для прогнозирования при принятии последовательных альтернативных решений с выбором оптимальной цепочки решений.

Метод опорных векторов (SVM) – популярный метод обучения, применяемый для решения задач классификации и регрессии. Он основан на построении гиперплоскости, разделяющей объекты выборки оптимальным образом.

Градиентный бустинг (Gradient Boosting) – современная и эффективная техника машинного обучения, используемая для решения широкого круга задач классификации и регрессии. Она строит предсказательные модели в форме ансамбля слабых предсказывающих моделей – «деревьев» решений.

Цель алгоритмов машинного обучения – определение функции потерь и её минимизация. Математические основы градиентного бустинга сводятся к типовому алгоритму, где среднеквадратическая ошибка прогноза (MSE) минимизируется с помощью обновления прогноза и градиентного спуска.

Нейросетевое моделирование создаёт модели, схожие с работой человеческого мозга, и позволяет не тратить время на подготовку данных для обучаемой модели, используя их в исходном виде. Для обеспечения высокой скорости и точности, кроме регрессора экстремального повышения градиента (XGBoost), крупные корпорации разработали интересные ансамблевые алгоритмы, основанные на повышении градиента, такие как LightGBM, регрессор повышения категории (CatBoost – Яндекс) и усиление естественного градиента (NGBoost). XGBoost – масштабируемый алгоритм, сложный для решения задач машинного обучения. LightGBM использует выборочную выборку для учёта экземпляров с наибольшим градиентом и обеспечивает высокую производительность обучения. CatBoost обеспечивает высокую точность, избегая смещения порядка предсказания путём изменения градиентов. NGBoost предоставляет вероятностный прогноз, используя повышение градиента с обработкой ковариатов.

Для прогнозирования используются только основные модели машинного обучения, а алгоритмы бустинга – это путь к достижению скорости и эффективности. В исследовании, проведённом авторами, XGBoost, LightGBM, CatBoost и X-NGBoost применяются для обучения большого объёма онлайн-данных с высокой скоростью и точностью. XGBoost считается основной моделью обучения для NGBoost, и на каждой итерации лучший градиент оценивается на основе оценки, указывающей ожидаемый результат на основе наблюдаемых признаков. XGBoost строит деревья последовательно, минимизируя ошибки, созданные в предыдущем дереве, но не может оценить функции. Таким образом, NGBoost учитывает результаты XGBoost и предоставляет лучшую оценку на основе функций, что дополнительно ускоряет обучение и повышает точность.

Основной целью работы исследователей по данной проблематике является разработка новой методики, основанной на искусственном интеллекте, для прогнозирования на основе модели XGBoost и алгоритма NGBoost, называемой техникой X-NGBoost [5].

Алгоритм XGBoost не поддерживает категориальные данные, поэтому настройка выполняется вручную. Алгоритм также может обучаться на больших наборах данных, так как он распараллеливается и использует возможности многоядерных процессоров. Внутренние параметры включают регуляризацию, пропущенные значения и параметры дерева. Ансамбли деревьев генерируют деревья последовательно, пытаясь уменьшить ошибки классификации на каждой итерации.

Недостаток модели XGBoost заключается в том, что результаты взвешиваются на основе результатов предыдущего момента времени n-1. Правильным прогнозам присваиваются меньшие веса, а неправильным – большие. Обрезка дерева – это метод, при котором переоснащённое дерево удаляется согласно выбранным критериям. Перекрёстная проверка сравнивает разделение и неразделение переобученного дерева, исключая узлы без лучших результатов.

В предлагаемой модели предварительно обработанные данные обрабатываются с использованием XGBoost и алгоритма прогнозирования естественной вероятности (как базовой модели обучения).

Исходные данные подгоняются к XGBoost для запуска модели. Гиперпараметры XGBoost выбираются методом проб и ошибок и оптимизируются с помощью байесовской оптимизации. Оптимизированные прогностические модели эффективно повышают точность.

Алгоритмы XGBoost, LightBoost и CatBoost обучаются и реализуются на одном наборе данных для сравнения. Сравнение алгоритмов проводится на основе среднеквадратической ошибки (RMSE), которая является важным показателем для вычисления чувствительности ошибки.

Заключение

В процессе анализа были определены особенности прогнозирования показателей бизнес-процессов, проблемы эффективности существующих моделей и доступности информации. Каждая модель зависит от данных, горизонта прогноза и настроек параметров модели. Различные исследования показывают разные результаты.

Традиционные методы прогнозирования полностью полагаются на оценку человека и экспертное мнение, которые могут учитывать только ограниченное количество факторов для определённого количества товаров.

Методы машинного обучения для прогнозирования показателей бизнес-процессов позволяют эффективно работать с данными, динамически их изменять и настраивать для более точного прогноза. Активное развитие алгоритмов машинного обучения, таких как градиентный бустинг, обеспечивает достаточную точность работы с разными признаками. В отличие от нейронных сетей, градиентный бустинг обладает большей интерпретируемостью результатов.

Доработка существующих моделей на основе градиентного бустинга путём комбинирования разных алгоритмов позволяет устранить недостатки каждого из них и делает дальнейшее использование перспективным.

Литература

1. *Минка К.Е.* Методы анализа и прогнозирования бизнес-процессов // Методы и средства обработки и хранения информации. Рязань: Издательство ИП Коняхин А.В. (BookJet), 2020. С. 101-105.

2. *Неверова М.Д.* Методы качественного анализа бизнес-процессов в практической деятельности предприятия // Инженерные кадры – будущее инновационной экономики России: материалы III Всероссийской студенческой конференции (Йошкар-Ола, 21-24 ноября 2017 г.): в 9 ч. Часть 6. Экономическое, финансовое и учетно-аналитическое обеспечение инженерных решений. Йошкар-Ола: Поволжский государственный технологический университет, 2017. С. 107-110.

3. *Рогулин Р.С.* Использование методов анализа данных и машинного обучения для прогнозирования и планирования спроса при управлении цепочками поставок // Теоретическая экономика. 2023. Т. 104. № 8. С. 35-35.

4. *Алаудинов Б.Р., Шахбазова М.С.* Ограничения машинного обучения // Развитие современной науки и технологий в условиях трансформационных процессов. 2022. С. 26-29.

5. *Namburu A., Selvaraj P., Varsha M.* Product pricing solutions using hybrid machine learning algorithm. Innovations Syst Softw Eng. 2022.

6. *Вострикова П.В., Рыбка С.О., Рыжкова У.С., Фатхулин Т.Д.* Анализ нейросетевых технологий, использующихся для улучшения качества изображений // REDS: Телекоммуникационные устройства и системы. 2024. Т. 14, № 1. С. 57-65. EDN WVBDRN

7. *Фатхулин Т.Д., Смирнов Д.А., Разумов И.В.* и др. Анализ влияния составляемых текстовых запросов (промптов) на качество изображений, генерируемых нейросетевыми технологиями // Системы синхронизации, формирования и обработки сигналов. 2024. Т. 15, № 2. С. 52-57. EDN TSVMSK

8. *Леохин Ю.Л., Фатхулин Т.Д.* Разработка методов и алгоритма формализации текстового запроса к онлайн-сервисам, генерирующим изображения посредством нейросетевых технологий // Вестник Рязанского государственного радиотехнического университета. 2023. № 85. С. 82-95. DOI 10.21667/1995-4565-2023-85-82-95. EDN PZWYZV

9. *Леохин Ю.Л., Фатхулин Т.Д., Кожанов М.С.* Анализ и исследование применения нейросетевых технологий для генерации программного кода // Вестник Рязанского государственного радиотехнического университета. 2024. № 87. С. 41-53. DOI 10.21667/1995-4565-2024-87-41-53. EDN HKEOFX

10. *Леохин Ю.Л., Фатхулин Т.Д., Ментус М.В.* Разработка и применение методов распознавания зашумленных аудиофайлов посредством нейросетевых технологий // Вестник Рязанского государственного радиотехнического университета. 2024. № 88. С. 65-73. DOI 10.21667/1995-4565-2024-88-65-73. EDN NMXASI

11. *Маслов К.В., Фатхулин Т.Д., Иванов Д.А.* Анализ технологий автоматизации бизнес-процессов и разработки программного обеспечения с использованием low-code платформ // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 1. С. 6-11. EDN HDBOYM

12. *Фатхулин Т.Д., Бойцов К.В.* Анализ функционала программного обеспечения, применяемого для классификации труб на предприятии методами компьютерного зрения // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 2. С. 6-12. EDN FVQYQA

13. *Мяличева А.А., Фатхулин Т.Д.* Анализ методов машинного обучения для прогнозирования дефектов в исходном коде // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 2. С. 16-19. EDN IVJCZF

14. *Фатхулин Т.Д., Леонова В.О., Тремасова Л.А.* Анализ нейросетевых технологий, применяемых для web-разработки // REDS: Телекоммуникационные устройства и системы. 2024. Т. 14, № 2. С. 35-41. EDN SDCNKM

15. *Фатхулин Т.Д., Исаев А.В.* Анализ моделей ARIMA и LSTM, используемых для прогнозирования криптовалют и определения портфеля инвестиций // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 2. С. 20-25. EDN ODWOPA

16. *Фатхулин Т.Д., Фатхулина Г.Г., Ментус М.В.* Разработка методики формирования запроса к нейросети с целью генерации изображений с учетом рекомендаций компьютерной лингвистики // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2023. № 1. С. 133-139. EDN PPRTOM

17. *Фатхулин Т.Д., Исаев А.В.* Анализ эффективности использования моделей ARIMA для прогнозирования котировок и определения портфеля инвестиций в области криптовалюты // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 2. С. 26-31. EDN CESRTK

18. *Фатхулин Т.Д., Бойцов К.В.* Оценка эффективности алгоритма на основе YOLO v.8 для классификации труб на предприятии по фото в зависимости от различных условий // Труды Северо-Кавказского филиала Московского технического университета связи и информатики. 2024. № 2. С. 51-55. EDN KWLMYA

19. *Вишневский В.М., Леохин Ю.Л., Фатхулин Т.Д., Занегин А.В.* Методы машинного обучения в решении задачи прогнозирования спроса на отдельные виды товаров // T-Comm: Телекоммуникации и транспорт. 2024. Т. 18. №10. С. 34-43.

АНАЛИЗ ОСОБЕННОСТЕЙ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ КРИТИЧЕСКОЙ ИНФОРМАЦИОННОЙ ИНФРАСТРУКТУРЫ РЕСПУБЛИКИ АНГОЛА

Михалевич Игорь Феодосьевич
Российский университет транспорта, доцент, к.т.н., старший научны й сотрудник, Москва, Россия
mif-orel@mail.ru

Франсишко Нелсон Адемар Мануэл
Российский университет транспорта, аспирант, Москва, Россия
admarnelson@hotmail.com

Аннотация
В данной работе представлены результаты анализа условий и факторов, оказывающих существенное влияние на организацию и защиту критической информационной инфраструктуры Республики Ангола. Изложены проблемы обеспечения безопасности и связанные с ними причины. Предложена применимая терминология и сформулированы задачи, решение которых позволит повысить безопасность рассматриваемой инфраструктуры

Ключевые слова
Информационная безопасность, критическая инфраструктура, компьютерная атака, компьютерный инцидент, угроза

Введение

В современных условиях безопасность критической инфраструктуры (КИ) каждой страны неразрывно связана с безопасностью ее критической информационной инфраструктуры (КИИ), нарушение безопасности которой может повлиять на социальную и экономическую стабильность страны, снизить уровень национальной безопасности в целом [1, 2].

Объекты КИИ включают в себя как физические, так и виртуальные компоненты, которые взаимосвязаны, взаимозависимы и сталкиваются с многочисленными актами незаконного вмешательства. Среди них в настоящее время наиболее распространены компьютерные атаки [3, 4].

Изложенное в полной мере относится к африканскому континенту в целом и Республике Ангола (далее – РА, Ангола), в частности.

Условия и факторы, требующие повышения безопасности КИИ Республики Ангола

На конец 2023 г. в РА Интернетом было обеспечено более 11 миллионов абонентов, что составило 33% от общей численности населения страны [5]. Рост этих показателей был достигнут в 2021-2023 годы за счет расширения телекоммуникационных сетей, разработки и внедрения доступных тарифных планов, улучшения инфраструктуры. Увеличение проникновения мобильной связи и интернета облегчило и ускорило доступ к информации, что способствует образованию, развитию финансовых услуг, улучшает возможности для бизнеса. Утверждается, что правительство страны продолжит инвестировать в развитие инфраструктуры и цифровых сервисов в целях преобразования национальной экономики и улучшения качества жизни граждан.

Ангола вошла в тройку стран, в которых реализуется первый этап программы «Инклюзивная цифровизация в Восточной и Южной Африке» (IDEA) Международного банка реконструкции и развития (МБРР) [6]. Программа рассчитана на 8 лет. По ней РА с 2024 годам получит кредит на 300 миллионов долларов США и дополнительно привлечет около 80 млн долл. частных инвестиций. Одной из целей программы заявлен переход Анголы от нефтяной экономики к более диверсифицированной. Проект включает три технических компонента:

– недорогая широкополосная связь и инклюзивность. Это предусматривает расширение доступа к недорогим и качественным интернет-услугам, создание цифровых общественных пространств, реализацию программ цифровой грамотности, поддержку реформ правительства РА в этом секторе;

– масштабирование инклюзивной и безопасной цифровой общественной инфраструктуры. Это предполагает раскрытие правительственных и частных инноваций, расширение онлайн-доступа к государственным и частным услугам, что создаст основу и благоприятную среду для роста цифровой экономики РА;

– продуктивное использование цифровых технологий для расширения экономических возможностей.

Особенности и проблемы обеспечения безопасности КИИ РА

Реализация столь масштабных проектов требует принятия мер по обеспечению безопасности КИИ РА с учетом следующих особенностей. В РА:

– недостаточно развита система национальных институтов, осуществляющих разработку и внедрение норм, правил, стандартов обеспечения безопасности КИИ;

– не создана общегосударственная система реагирования на компьютерные атаки;

– отсутствует единый центр по компьютерным инцидентам (ЕЦКИ) на объектах КИИ;

– наблюдается существенный дефицит квалифицированных кадров в области обеспечения безопасности КИИ;

– наблюдается общий низкий уровень культуры и образования населения;

– наблюдается низкий уровень цифровой гигиены населения, работников коммерческих организаций и государственных служащих;

– недостаточно исследованы вопросы обеспечения безопасности КИИ;

– КИИ полностью зависит от иностранных решений;

– взаимодействие с правоохранительными органами других стран недостаточно эффективно.

Рассмотрим некоторые особенности более подробно.

Технологическая инфраструктура РА развита пока слабо, что создает трудности для защиты объектов КИИ.

Недостаток финансовых средств ограничивает возможности для создания и развития институтов, нормативной правовой и методической базы в области безопасности КИ и КИИ, отвечающих современным угрозам. Существующие органы, такие как Министерство телекоммуникаций и информационных технологий, не обладают достаточными ресурсами для полного охвата существующих проблем.

В обществе и даже среди государственных служащих не всегда осознается важность защиты КИИ, что приводит к низкому приоритету этих вопросов на государственном уровне. Ограниченное понимание необходимости разработки стандартов и законодательной базы в этой области препятствует выделению ресурсов и принятию мер.

Создание ЕЦКИ РА и обеспечение его деятельности требует значительных финансовых вложений. В развивающихся странах, таких как Ангола, бюджетные средства часто перераспределяются в пользу более приоритетных для государства нужд, таких как здравоохранение, образование и инфраструктура.

Для функционирования ЕЦКИ РА требуются квалифицированные кадры, численность которых в РА недостаточна.

Для создания ЕЦКИ РА необходимо специальное законодательство, которое в РА отсутствует.

Уровень цифровизации на объектах КИ РА остается низким, что снижает приоритетность создания ЕЦКИ РА.

Не имея собственного центра, РА может принять участие в международных или региональных системах реагирования на компьютерные инциденты, например, в рамках Сообщества развития юга Африки (SADC).

Нехватка квалифицированных кадров в области обеспечения безопасности КИИ РА обусловлена рядом причин, характерных для развивающихся стран. Среди основных причин можно выделить следующие.

1.Недостаток образовательных учреждений и программ подготовки. В РА ограниченное количество университетов и учебных заведений, предлагающих программы по информационной безопасности. Многие образовательные учреждения не имеют достаточных ресурсов, чтобы обеспечивать современное обучение в этой области, включая доступ к специализированным лабораториям, программному обеспечению и учебным материалам.

2. Низкий уровень подготовки кадров в ИТ-сфере. Образование в области информационных технологий в целом остается недостаточно развитым. Основное внимание уделяется базовым навыкам, таким как программирование, администрирование сетей и систем, а специализированные направления, например, компьютерная безопасность или защита КИИ, остаются на периферии.

3. Отсутствие национальной стратегии подготовки специалистов. На уровне государства отсутствует четкая стратегия и программа по подготовке кадров вышеуказанных специальностей. Это связано с недостаточным пониманием важности этой проблемы или с приоритетом других национальных задач.

4. Миграция специалистов. Квалифицированные ИТ-специалисты из Анголы часто мигрируют в более развитые страны, такие как Португалия, Бразилия или страны Европы, где возможности для профессионального роста, заработная плата и условия труда значительно лучше. Это приводит к «утечке мозгов», лишая страну критически важного кадрового ресурса.

5. Нехватка финансирования на подготовку кадров. Недостаточное финансирование образовательных и профессиональных программ ограничивает возможности подготовки специалистов. Это касается как государственных, так и частных инициатив в области образования и повышения квалификации. Без финансовых стимулов сложно привлечь молодежь к освоению сложных профессий.

6. Слабая осведомленность о проблемах обеспечения безопасности КИИ. В Анголе наблюдается низкий уровень осведомленности среди общественности и руководителей о важности защиты КИИ. Это снижает интерес к требуемым в данной области профессиям, так как они могут восприниматься как менее важные по сравнению с другими направлениями.

7. Недостаток стажировок и практического опыта. Даже если базовые знания в области ИТ и информационной безопасности получены, студентам и специалистам не хватает возможностей для стажировок или реального опыта работы с КИИ. Это приводит к разрыву между теоретической подготовкой и практическими навыками.

8. Маленькое число местных инициатив и центров компетенции. В Анголе отсутствует достаточное количество исследовательских центров, сертификационных программ и специализированных курсов, направленных на развитие компетенций в области защиты КИИ. Это ограничивает возможности для специалистов совершенствовать свои навыки.

9. Зависимость от зарубежных ресурсов. Вместо развития местных кадров Ангола полагается на услуги международных компаний и специалистов для решения задач в области безопасности. Это снижает стимулы для инвестиций в подготовку местных кадров.

10. Общий низкий уровень культуры, образования и цифровой гигиены. Это обусловлено историческими факторами и рядом структурных, экономических и социальных факторов, которые препятствуют развитию данной области. Уровень грамотности населения, включая базовые знания о цифровой гигиене, остается крайне низким. Отметим, что недостаточный уровень цифровой гигиены является вызовом для многих стран [7].

В РА ощущается нехватка доступной образовательной литературы, материалов и онлайн-ресурсов на местных языках или португальском. Ограниченный доступ к интернету в ряде регионов страны также препятствует возможности получения необходимых знаний через онлайн-курсы и материалы.

11. Полная технологическая зависимость КИИ от иностранных решений обусловлена рядом факторов, связанных с историческими, экономическими, технологическими и кадровыми аспектами. Ниже представлены основные причины этой зависимости. Это:

– отсутствие местных технологических разработок. В Анголе недостаточно развит сектор ИТ, что делает страну неспособной самостоятельно разрабатывать решения для обеспечения безопасности КИИ. Отсутствие местных компаний, специализирующихся на создании программного обеспечения, оборудования или технологий для управления и защиты КИИ, вынуждает правительство и частные организации полагаться на иностранные продукты;

– использование готовых иностранных продуктов (как программных, так и аппаратных) зачастую более экономически выгодно, чем разработка собственных;

– иностранные компании предлагают готовые комплексные решения, которые проще интегрировать и сопровождать, что делает их более привлекательными для ангольских организаций;

– Ангола полагается на импортное оборудование и программное обеспечение от крупных международных корпораций, таких как Cisco, Huawei, Microsoft и др. Это обусловлено отсутствием собственного производства высокотехнологичных компонентов, необходимых для функционирования КИИ, включая серверы, сети, системы управления данными и средства кибербезопасности.

Иностранные решения требуют значительных финансовых затрат на лицензирование, сопровождение, обновление и техническую поддержку. Это увеличивает экономическую нагрузку на страну. Многолетняя зависимость от иностранных поставщиков делает РА уязвимой перед ростом цен, санкциями или перебоями в поставках.

Терминология в области обеспечения безопасности КИИ РА

Решение проблем обеспечения безопасности КИИ РА требует согласованных действий всех заинтересованных сторон, что невозможно в условиях недоговоренностей о базовых терминах и их содержании. Учитывая опыт Российской Федерации, имеет смысл рассмотреть некоторые термины и их толкование:

– критическую информационную инфраструктуру образуют информационные системы, информационно-телекоммуникационные сети, автоматизированные системы управления (далее – объекты) КИИ и связывающие их каналы (линии, сети) связи [8];

– к информационно-телекоммуникационным сетям относятся системы передачи данных по каналам и линиям связи [9];

– безопасность КИИ обеспечивается если при проведении компьютерных атак сохраняется устойчивое функционирование вышеуказанных объектов [8];

– под компьютерной атакой понимается любое преднамеренное воздействие (или совокупность таких воздействий), осуществляемое (-ые) с использованием программного обеспечения (в том числе, программно-аппаратных средств и/или комплексов) на объекты КИИ и (или) связывающие их каналы (линии, сети) связи, направленное (-ые) на прекращение (нарушение) их функционирования или их информационной безопасности [8];

– под компьютерным инцидентом понимается факт наступления вышеуказанных последствий на объектах КИИ и (или) соответствующих каналах (линии, сети) связи [8].

Созданная в Российской Федерации нормативная правовая [8, 10-16] и методическая база [17, 18] обеспечивают полноту и непротиворечивость понятий, используемых в сфере обеспечения безопасности КИИ, что позволяет рекомендовать их к использованию в РА.

Так, например, по аналогии с [19, 20], на ЕЦКИ РА могут быть возложены следующие основные функции:

– разработка рекомендаций по предупреждению компьютерных атак;

– планирование и контроль мероприятий реагирования на компьютерные инциденты;

– мониторинг компьютерных атак;

– разработка и реализация рекомендаций по ликвидации последствий компьютерных атак;

– ведение баз и банков данных о компьютерных инцидентах и компьютерных атаках, способах и средствах предупреждения, реагирования и ликвидации последствий.

Заключение

Обеспечение безопасности критической информационной инфраструктуры Республики Ангола требует решения комплекса задач, включая усиление законодательства, увеличение объемов инвестиций на модернизацию и защиту объектов, внедрение новых технологий и улучшение подготовки кадров. Своевременное решение этих задач позволит повысить безопасность критической инфраструктуры страны, ее устойчивость к существующим и новым типам угроз.

Литература

1. *Kalashnikov A.O., Mikhalevich I.F.* About the single system of protection classes elements of critical information infrastructure by the criteria of importance and information security // International Journal of Engineering and Technology (UAE). 2018. Vol. 7. № 2. C. 247-250. DOI:10.14419/ijet.v7i2.23.11952.

2. *Juan E. Rubio, Rodrigo Roman, Javier Lopez.* Analysis of Cybersecurity Threats in Industry 4.0: The Case of Intrusion Detection, 12th International Conference “Critical Information Infrastructures Security”, G. D’Agostino and A. Scala (Eds.): CRITIS 2017, LNCS 10707, pp. 119-130, 2018.

3. Complete Guide to OT Threat Detection and Response By Sectrio | November 20th, 2023. URL: https://sectrio.com/ot-threat-detection-and-response/#what-is-threat-detection-investigation-and-response (дата обращения 23.01.2025).

4. *Mikhalevich I.F.* Critical Infrastructure Security: alignment of views. 2019 International Conference “Systems of Signals Generating and Processing in the Field of on Board Communications”. doi.org/10.1109/SOSG.2019.8706821.

5. Acesso à internet em Angola atinge mais de 11 milhões de subscritores entre 2021 e 2023 (ANGOTIC-2024), Forbes, 13.06.2024. https://www.forbesafricalusofona.com/acesso-a-internet-em-angola-atinge-mais-de-11-milhoes-de-subscritores-entre-2021-e-2023/ (дата обращения 21.01.2025).

6. Inclusive Digitalization in Eastern and Southern Africa Program: Angola. Word bank Groupe, 27.06.2024. https://www.worldbank.org/en/news/factsheet/2024/06/27/inclusive-digitalization-in-eastern-and-southern-africa-program-afe-angola (дата обращения 11.12.2024).

7. *Михалевич И.Ф.* Цифровая гигиена информационного общества: влияние пандемии COVID-19 // REDS: Телекоммуникационные устройства и системы, № 3-2022. С. 10-17.

8. Федеральный закон «О безопасности критической информационной инфраструктуры Российской Федерации» от 12.07.2017 г. № 187-ФЗ.

9. Федеральный закон «Об информации, информационных технологиях и о защите информации» от 27.07.2006 г. № 149-ФЗ.

10. Указ Президента Российской Федерации от 30.03.2022г. № 166 «О мерах по обеспечению технологической независимости и безопасности критической информационной инфраструктуры Российской Федерации».

11. Правила категорирования объектов критической информационной инфраструктуры Российской Федерации (утв. постановлением Правительства Российской Федерации от 08.02.2018 г. № 127).

12. Перечень показателей критериев значимости объектов критической информационной инфраструктуры Российской Федерации и их значений (утв. постановлением Правительства Российской Федерации от 08.02.2018 г.№ 127).

13. Требования по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации (утв. приказом ФСТЭК России от 25.12.2017 г. № 239).

14. Требования к созданию систем безопасности значимых объектов критической информационной инфраструктуры Российской Федерации и обеспечению их функционирования (утв. приказом ФСТЭК России от 21.12.2017 г. № 235).

15. Правила перехода субъектов критической информационной инфраструктуры Российской Федерации на преимущественное применение доверенных программно-аппаратных комплексов на принадлежащих им значимых объектах критической информационной инфраструктуры Российской Федерации (утв. постановлением Правительства Российской Федерации от 14.11.2023 № 1912).

16. Постановление Правительства Российской Федерации от 14.11.2023 г. № 1912 «О порядке перехода субъектов критической информационной инфраструктуры Российской Федерации на преимущественное применение доверенных программно-аппаратных комплексов на принадлежащих им значимых объектах критической информационной инфраструктуры Российской Федерации».

17. Методические рекомендации по переходу на использование российского программного обеспечения, в том числе на значимых объектах критической информационной инфраструктуры Российской Федерации (утв. приказом Минцифры России от 18.01.2023 г. № 21).

18. Методические рекомендации по категорированию объектов критической информационной инфраструктуры, функционирующих в сфере транспорта (утверждены Министерством транспорта Российской Федерации Российской Федерации 24.01.2024). https://mintrans.gov.ru/documents/10/13201?ysclid=lu2cx9gfse525544140. (дата обращения 10.01.2025).

19. Положение о Национальном координационном центре по компьютерным инцидентам (утв. приказом ФСБ России от 24.07.2018 г.№ 366).

20. Национальный координационный центр по компьютерным инцидентам. https://www.cert.gov.ru/index.html (дата обращения 23.01.2025).

ФУНКЦИОНАЛЬНО-РЕЛЯЦИОННЫЙ МЕТОД РАЗРАБОТКИ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ АВТОМАТИЗИРОВАННЫХ СИСТЕМ ОТЕЧЕСТВЕННЫХ КРУИЗНЫХ КОМПАНИЙ

Ковтун Игорь Иванович
МТУСИ, к.т.н., доцент, доцент кафедры «Системное программирование», Москва, Россия
i.i.kovtun@mtuci.ru

Козлова Яна Васильевна
МТУСИ, студент группы БФИ2203, Москва, Россия
yana-kozlova-04@inbox.ru

Мячина Лада Андреевна
МТУСИ, студент группы БФИ2203, Москва, Россия
ladaamy@yandex.ru

Аннотация
Проводится адаптация принципов построения интерфейса взаимодействия с конечным пользователем АС, положенных в основу разработанного ранее функционально–реляционного метода к специфике деятельности отечественных круизных компаний. Представленный подход аккумулирует в себе достоинства как реляционного, так и функционального моделирования.

Ключевые слова
Круизная компания, автоматизированная система, пользовательский интерфейс, язык Python, реляционная модель, модель IDEF0

Введение

В последние годы в Российской Федерации отмечена тенденция к активному развитию внутреннего туризма [1]. Из данных Российского союза туриндустрии следует, что по итогам 2024 года количество внутренних туристических поездок достигнет 96 млн., что на четверть больше, чем в 2023 году. Данная тенденция связана, главным образом, с тем, что развитие внутреннего туризма – один из приоритетов долгосрочной стратегии экономического развития России. При этом, целесообразно отметить, что на фоне роста внутренних туристических поездок уровень российских речных круизов у флагманов данного направления заметно возрастает.

В условиях острой конкуренции в сфере туристических услуг вопрос эффективного управления туристической компанией приобретает первостепенное значение. В свою очередь, обеспечение механизмов такого управления без средств автоматизации становится практически невозможным. Важной составляющей процесса комплексной автоматизации, пожалуй, любой компании является формализация описания ее бизнес-процессов и потоков обрабатываемой при этом информации. При этом, как правило, применяется реляционная модель данных [2], [3], одним из ограничений которой является отсутствие выразительных возможностей для описания функциональной структуры проектируемой автоматизированной системы (АС) [4]. С другой стороны, процессе проектирования АС используют функциональные модели материальных, финансовых и информационных потоков, например, IDEF0 [5], [6], которые, в свою очередь не дают полного представления о структуре информации проектируемой АС [7]. Преодоление выразительных ограничений модели IDEF0 и реляционной модели простым объединением диаграмм не дает однозначного понимания элементов пользовательского интерфейса, проектируемых в составе такой системы АРМ [8].

Таким образом, целью настоящего исследования является развитие предложенного в [9] подхода к формализации и интеграции результатов обследования отечественных круизных компаний, сочетающего в себе достоинства реляционного и функционального моделирования, а также учитывающего общие закономерности и особенности функционирования таких компаний и позволяющего системным образом разработать для их автоматизации пользовательские интерфейсы прикладных программ.

Достижение поставленной цели выливается в осуществление следующих практических шагов:

– построение комплексной, интегрированной модели деятельности круизной компании, позволяющей принять весь спектр управленческих решений по ее автоматизации;

– систематизация применения построенной модели в процессе разработки пользовательских интерфейсов автоматизированных систем отечественных круизных компаний;

– разработка программного обеспечения формирования пользовательских интерфейсов;

– подведение результатов исследовательской работы.

Моделирование общих закономерностей туристического бизнеса

Анализ бизнес-процессов деятельности отечественной круизной компании [10] позволил спроектировать модель IDEF0 по принципу AS-IS, которая отражает состояние такой компании, представленное на рисунке 1.

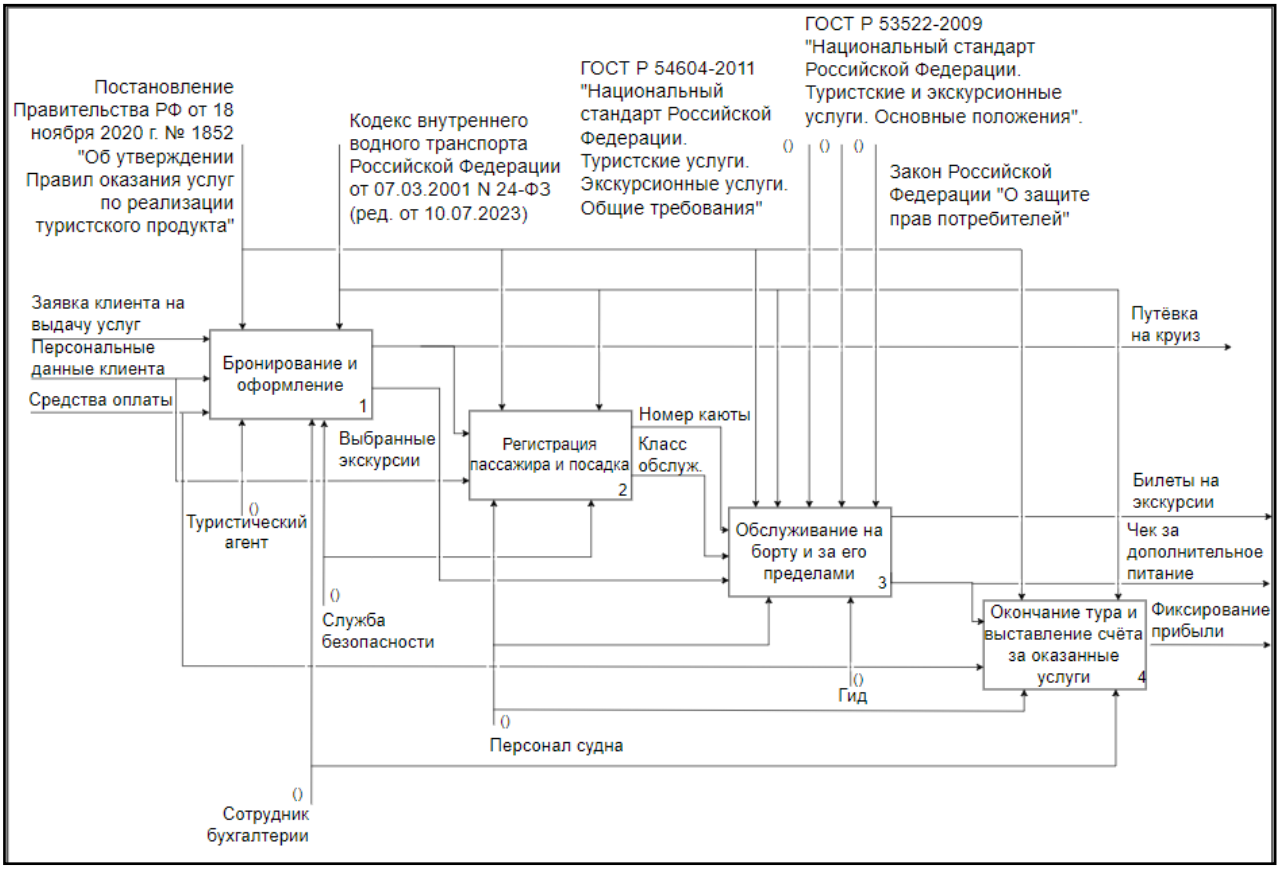

Рис. 1. Состояние AS-IS деятельности отечественной круизной компании в нотации IDEF0

Проблемы, выявленные в процессе анализа деятельности отечественной круизной компании:

– отсутствие достаточного количества времени для принятия эффективных управленческих решений в сфере круизного бизнеса вследствие большого количества монотонной и ручной работы;

– большое количество ошибок в процессе подготовки договорной документации;

– отсутствие оперативного доступа к информации из структурных подразделений и филиалов.

Для решения выявленных проблем предложено внести изменения в организацию процесса управления круизной компанией, см. [10]. Предложенные нововведения формализованы и представлены в виде состояния TO-BE функциональной модели IDEF0, фрагмент модели отображен на рисунке 2.

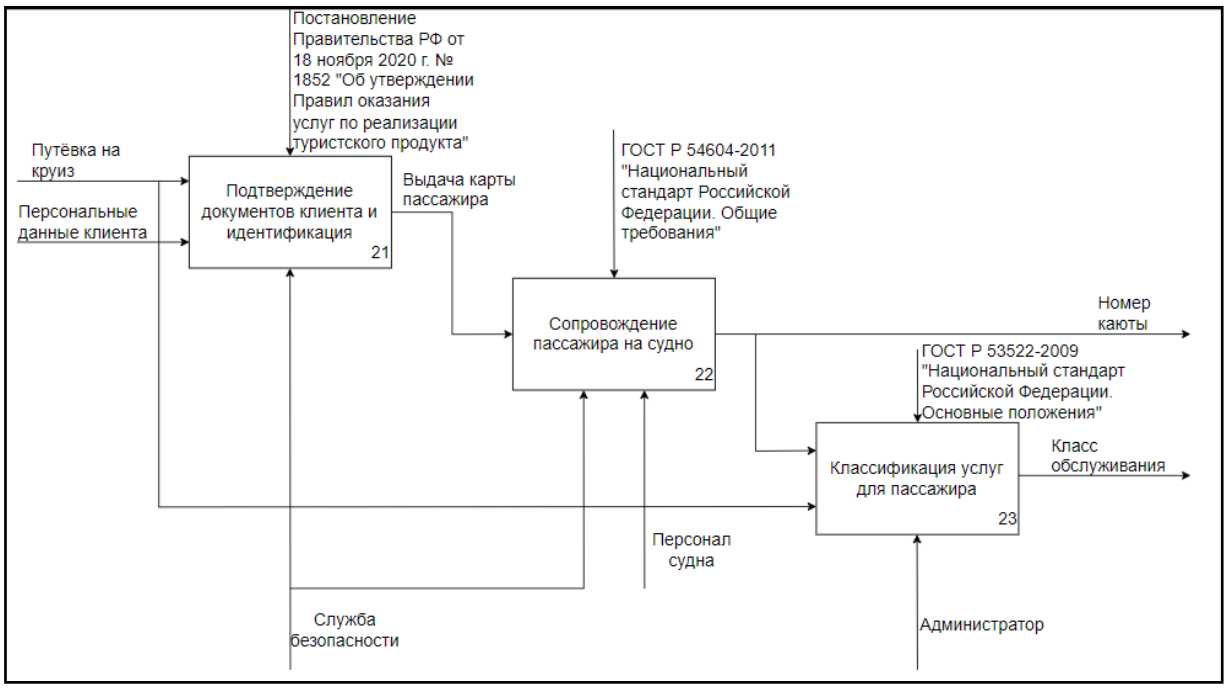

Рис. 2. Состояние TO-BE деятельности отечественной круизной компании в нотации IDEF0

С учетом предложенного решения регистрация пассажира и посадка происходит в следующем порядке:

– пассажир предоставляет путевку и документ, подтверждающий личность;

– служба безопасности проверяет документы и идентифицирует пассажира, сопровождает его на борт;

– определяется класс обслуживания для пассажира.

Ресурсами необходимыми для выполнения являются служба безопасности, персонал судна, администратор. Результатом процесса является фиксирование класса обслуживания и номера каюты**.**

Проведенный анализ туристических процессов позволил также спроектировать реляционную модель данных, фрагмент физической составляющей которой представлен на рисцнке 3.

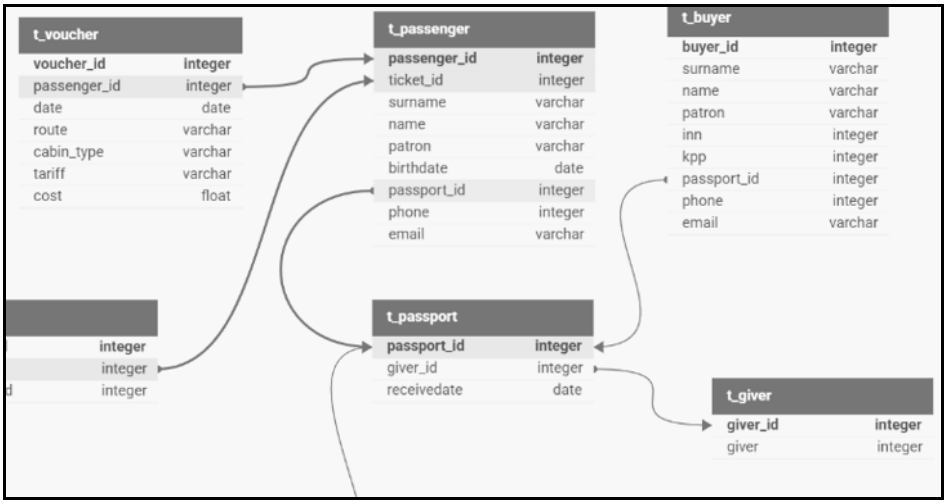

Рис. 3. Фрагмент реляционной модели данных круизной компании

При этом детально описана структура каждой таблицы. В качестве примера приведем таблицу t_passenger (табл. 1).

Таблица 1

Структура таблицы t_passenger

Атрибут	Тип данных	Признак объекта	Ключ
a_passenger_id	Numder	Ключ	Первичный ключ
a_ticket_id	Varchar2	Номер билета	Альтернативный ключ 1.1
a_surname	Varchar2	Фамилия	Альтернативный ключ 1.2
a_name	Varchar2	Имя	Альтернативный ключ 1.3
a_patron	Number	Отчество	Внешний ключ
a_birthdate	Number	Дата рождения	Альтернативный ключ 2.1
a_passport_id	Varchar2	Паспорт	
a_phone	Number	Телефон	Альтернативный ключ 2.1
a_email	Varchar2	Электронная почта	

Принцип систематизации пользовательского интерфейса АС отечественной круизной компании на основе интеграции реляционной и функциональной моделей

Общий принцип к построению интерфейса взаимодействия с конечным пользователем АС, положенный в основу функционально–реляционного метода представлен в [11] и доработан с учетом специфики отечественных круизных компаний следующим образом – для каждого реляционного отношения проектируется некоторый режим работы. Характерной особенностью такого режима является главное окно, в котором пользователю предоставляется возможность просмотреть все кортежи отношения.

С помощью ряда средств, а именно, меню и кнопок обеспечивается мощная навигация по кортежам и доменам отношения. Также с помощью меню и кнопок обеспечивается вызов других окон, которые обеспечивают модификацию отношения, и отношений, наиболее тесно связанных с ним.

Структура программного кода определяется, в свою очередь, пользовательским интерфейсом следующим образом – каждому режиму соответствует некоторая структурная единица системы – подсистема, функция или модуль. При открытии режима происходит вызов такой структурной единицы, а при закрытии – выход из нее и возврат системы в первоначальное состояние.

Практическая реализация предложенного метода на языке Python

Применим данный подход к базе данных «Система управления круизами». Придем к необходимости создания трех режимов работы – «Теплоходы», «Туры» и «Пассажиры». Указанные режимы представлены кнопками «Добавить пассажира», «Добавить тур», «Добавить экскурсию» на стартовой форме (рис. 4).

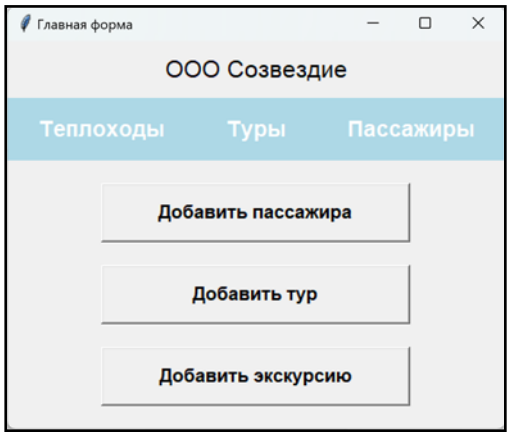

Рис. 4. Стартовая форма программного обеспечения

При этом, программный код формирования такой стартовой формы целесообразно реализовать на языке Python. Принципиальный фрагмент кода представлен в листинге 1.

Листинг 1. Фрагмент программного кода формирования главной формы

```
def create_form():
    root = tk.Tk()
    root.title("Главная форма")

    title_label = tk.Label(root, text="ООО Созвездие", font=("Arial", 16))
    title_label.pack(pady=10)

    button_frame = tk.Frame(root, bg='lightblue', height=50)
    button_frame.pack(fill='x')

    button_style = {
        'font': ('Arial', 14, 'bold'),
        'fg': 'white',
        'bg': 'lightblue',
        'activebackground': 'lightblue',
        'activeforeground': 'white',
        'relief': 'flat',
        'borderwidth': 0
    }

    button_style_bottom = {
        'font': ('Arial', 12, 'bold'),
        'width': 26,  # Размер в текстовых единицах для соответствия 290px
        'height': 2
    }

    btn1 = tk.Button(button_frame, text="Теплоходы", **button_style)
    btn1.pack(side='left', padx=20, pady=10)

    btn2 = tk.Button(button_frame, text="Туры", **button_style)
    btn2.pack(side='left', padx=20, pady=10)

    btn3 = tk.Button(button_frame, text="Пассажиры", command=open_passenger_list_form, **button_style)
    btn3.pack(side='left', padx=20, pady=10)

    button_column_frame = tk.Frame(root)
    button_column_frame.pack(pady=10)

    add_passenger_btn = tk.Button(button_column_frame, text="Добавить пассажира",
command=open_add_passenger_form, **button_style_bottom)
    add_passenger_btn.pack(pady=10)

    add_tour_btn = tk.Button(button_column_frame, text="Добавить тур", command=open_add_tour_form,
**button_style_bottom)
    add_tour_btn.pack(pady=10)

    add_excursion_btn = tk.Button(button_column_frame, text="Добавить экскурсию",
command=open_add_excursion_form, **button_style_bottom)
    add_excursion_btn.pack(pady=10)

    root.mainloop()
```

Типовой подход для формирования формы просмотра

В составе режима *Пассажиры* разработано главное окно *Пассажиры*, см. рис. 5. В данном окне обеспечена возможность просмотра всех кортежей и полей таблицы t_passenger и поиска по номеру билета определенных полей, а также возможность вызова с помощью кнопок окон, позволяющих осуществлять внесение новых записей, редактирование или удаление существующих. Возврат системы в некоторое исходное состояние осуществляется по нажатию кнопки *Назад*.

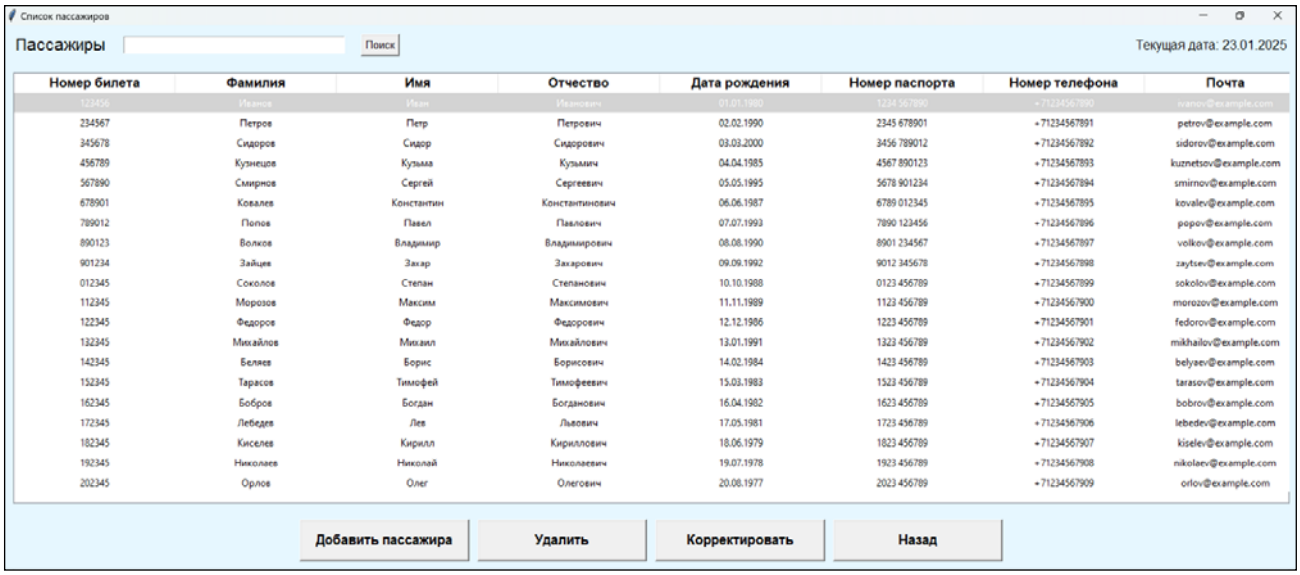

Рис. 5. Примерный вид главного окна Пассажиры

Принципиальный фрагмент программного кода формирования главного окна *Пассажиры* на языке Python представлен в листинге 2.

Листинг 2. Фрагмент программного кода формирования вида главного окна Пассажиры

```
def open_passenger_list_form():
    passenger_list_window = tk.Toplevel()
    passenger_list_window.title("Список пассажиров")
    passenger_list_window.configure(bg='#E6F7FF')
    passenger_list_window.geometry('800x600')

    label_style = {'bg': '#E6F7FF', 'font': ('Arial', 12)}

    top_frame = tk.Frame(passenger_list_window, bg='#E6F7FF')
    top_frame.pack(fill='x', pady=10)

    tk.Label(top_frame, text="Пассажиры", font=('Arial', 16), bg='#E6F7FF').pack(side='left', padx=10)

    search_var = tk.StringVar()
    search_entry = tk.Entry(top_frame, textvariable=search_var, font=('Arial', 12), width=30)
    search_entry.pack(side='left', padx=10)

    search_button = tk.Button(top_frame, text="Поиск", font=('Arial', 10))
    search_button.pack(side='left', padx=10)

    tk.Label(top_frame, text=f"Текущая дата: {datetime.now().strftime('%d.%m.%Y')}", **la-
bel_style).pack(side='right',
                                                                padx=10)
    columns = ("ticket_no", "last_name", "first_name", "middle_name", "birth_date", "passport_no", "phone_no",
"email")
    tree = ttk.Treeview(passenger_list_window, columns=columns, show='headings', style='Treeview')
    tree.heading("ticket_no", text="Номер билета")
    …
    tree.heading("email", text="Почта")

    for col in columns:
        tree.column(col, anchor='center')

    # Add vertical and horizontal lines
    style = ttk.Style()
```

```
style.configure("Treeview.Heading", font=('Arial', 12, 'bold'))
style.configure("Treeview", rowheight=25, bordercolor='#E6E6E6', borderwidth=1)
style.map("Treeview", background=[('selected', '#D3D3D3')])

style.configure("Treeview", highlightthickness=1, highlightbackground="#E6E6E6", relief='solid')
for passenger in passengers:
   tree.insert("", "end", values=(
   passenger["ticket_no"], passenger["last_name"], passenger["first_name"], passenger["middle_name"],
   passenger["birth_date"], passenger["passport_no"], passenger["phone_no"], passenger["email"]))

tree.pack(fill='both', expand=True, padx=10, pady=10)

button_frame = tk.Frame(passenger_list_window, bg='#E6F7FF')
button_frame.pack(pady=10)

button_style = {
   'font': ('Arial', 12, 'bold'),
   'width': 20,
   'height': 2
}

tk.Button(button_frame, text="Добавить пассажира", command=open_add_passenger_form, **but-
ton_style).pack(side='left', padx=5)
…

tk.Button(button_frame, text="Назад", **button_style, command=passenger_list_window.de-
stroy).pack(side='left', padx=5)
```

Типовой подход для формирования формы ввода информации

Типовой подход для формирования формы ввода информации целесообразно рассмотреть на примере окна *Добавить пассажира.*

Окно *Добавить пассажира*, см. рис. 6 появляется по нажатию на кнопку *Добавить пассажира* в окне *Пассажиры*.

Данное окно спроектировано таким образом, что для регистрации нового пассажира необходимо внести сведения о нем и нажать на кнопку *Добавить*, а для закрытия окна без ввода набранной информации – на кнопку *Отмена*. Предусмотрен порядок работы, при котором после нажатия на кнопку *Ввести* происходит запуск программного кода, проводящего анализ базы данных на соответствие ограничениям целостности и непротиворечивости с учетом вновь набранной информации. Если такие требования соблюдаются, запрограммировано внесение информации в базу, если нет – выводится сообщение о причинах отказа во вводе информации в базу.

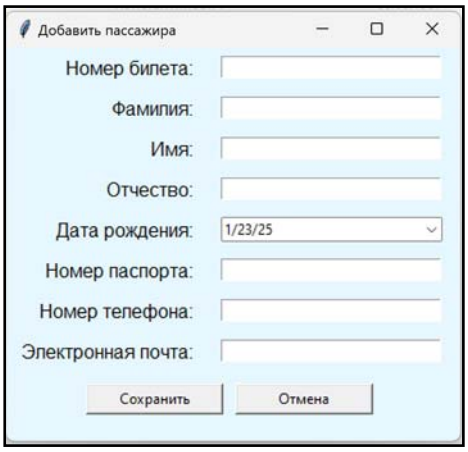

Рис. 6. Форма для добавления нового пассажира

Программный код формирования окна *Добавить пассажира* на языке Python представлен в листинге 3.

Листинг 3. Фрагмент программного кода формы добавления нового пассажира

```
def open_add_passenger_form():
    add_passenger_window = tk.Toplevel()
    add_passenger_window.title("Добавить пассажира")
    add_passenger_window.configure(bg='#E6F7FF')
    add_passenger_window.geometry('380x330')  # Увеличение ширины до 500 пикселей

    label_style = {'anchor': 'e', 'bg': '#E6F7FF', 'font': ('Arial', 12)}

    entry_style = {'width': 30}

    tk.Label(add_passenger_window, text="Номер билета:", **label_style).grid(row=0, column=0, padx=10,
pady=5, sticky='e')
    ticket_number_entry = tk.Entry(add_passenger_window, **entry_style)
    ticket_number_entry.grid(row=0, column=1, padx=10, pady=5)

    tk.Label(add_passenger_window, text="Фамилия:", **label_style).grid(row=1, column=0, padx=10, pady=5,
sticky='e')
    surname_entry = tk.Entry(add_passenger_window, **entry_style)
    surname_entry.grid(row=1, column=1, padx=10, pady=5)

    …

    tk.Label(add_passenger_window, text="Отчество:", **label_style).grid(row=3, column=0, padx=10, pady=5,
sticky='e')
    patronymic_entry = tk.Entry(add_passenger_window, **entry_style)
    patronymic_entry.grid(row=3, column=1, padx=10, pady=5)

    button_frame = tk.Frame(add_passenger_window, bg='#E6F7FF')
    button_frame.grid(row=8, column=0, columnspan=2, pady=10)

    button_width = 15
    tk.Button(button_frame, text="Сохранить", width=button_width).pack(side='left', padx=5)
    tk.Button(button_frame, text="Отмена", width=button_width, command=add_passenger_window.de-
stroy).pack(side='left', padx=5)
```

Типовой подход для формирования формы удаления

Типовой подход для формирования формы удаления информации целесообразно рассмотреть на примере окна *Предупреждения об удалении пассажира.*

Удаление записи о пассажире организовано в следующем порядке – обеспечена возможность выделения требуемой записи в окне *Пассажиры* и нажатия на кнопку *Удалит*ь данного окна. После этого появляется окно предупреждения, см. рис. 7, с помощью которого пользователь еще раз опрашивается о необходимости удаления.

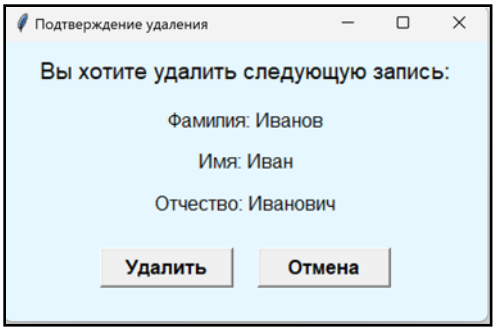

Рис. 7. Форма для подтверждения удаления

По нажатию на кнопку *Удалить* инициируется программный код, производящий анализ базы данных на соответствие ограничениям целостности и непротиворечивости с учетом отсутствия удаляемой записи. Если они соблюдаются, происходит удаление информации из базы, если нет – выводится сообщение о причинах отказа в удалении. По нажатию на кнопку *Отмена* происходит закрытие окна без удаления информации.

Заключение

Результаты проведенного исследования:

1. Проведен системный анализ туристической деятельности отечественной круизной компании. Выявлены узкие и проблемные места в управлении такой деятельностью.
2. Предложен оригинальный функционально-реляционный метод разработки пользовательских интерфейсов для систем комплексной автоматизации отечественных круизных компаний, позволяющий устранить выявленные проблемы.
3. Проанализировано и выбрано программное обеспечение, позволяющее реализовать предложенное решение.
4. Разработаны пользовательские интерфейсы автоматизированных систем отечественных круизных компаний с помощью выбранного программного обеспечения.

Литература

1. *Мухер Н.* Особенности развития круизного рынка в современных условиях // Актуальные исследования. 2023. № 41 (120). С. 70-75.

2. *Codd E.F.* A Relational Model of Data for Large Shared Data Banks. CACM 13, No. 6 (June 1970). Republished in Milestones of Research - Selected Papers 1958-1982 (CACM 25th Anniversary Issue), CACM 26, No. 1 (January 1983).

3. *Мейер Д.* Теория реляционных баз данных: Пер. с англ. М.: Мир, 1987. 608 с.

4. *Ковтун И.И.* Проблемы моделирования проектных решений в процессе проектирования автоматизированных информационных систем // *Информатизация и связь.* 2012. № 8. С. 145-151.

5. РД 50.1.028-2001. Методология функционального моделирования IDEF0. Руководящий документ. Издание официальное. – М.: ИПК Издательство стандартов, 2000. 75 с.

6. Integration Definition for Function Modeling (IDEF0). Draft Federal Information Processing Standards Publication 183, 1993 December 21.

7. *Ковтун И.И.* Функционально-реляционный анализ вариантов автоматизации сложных организационно-технических систем. Монография. СПб.: Изд-во НИЦ АРТ, 2024. 214 с.

8. *Ковтун И.И., Козлова Я.В., Мячина Л.А.* Интеграция функциональной и реляционной модели в процессе разработки автоматизированных информационных систем отечественных круизных компаний // REDS: Телекоммуникационные устройства и системы. 2024. № 1. С. 33-39.

9. *Ковтун И.И., Козлова Я.В., Мячина Л.А.* Функционально-реляционный метод в процессе разработки автоматизированных информационных систем отечественных круизных компаний // Телекоммуникации и информационные технологии. 2024. № 1. С. 41-47.

10. *Ковтун И.И., Козлова Я.В., Мячина Л.А.* Функционально-реляционный анализ в процессе автоматизации отечественных круизных компаний // Научно-практический, теоретический журнал «Экономика и управление: проблемы, решения». 2024. Т.4. № 6. С. 87-94.

11. *Ковтун И.И.* Теория и практика проектирования государственных информационных систем. Учебное пособие. СПб.: Изд-во НИЦ АРТ, 2023. 194 с.

ДОСТИЖЕНИЕ ОТКАЗОУСТОЙЧИВОСТИ В КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ С ПОМОЩЬЮ АЛГОРИТМОВ РАСПРЕДЕЛЕННОГО КОНСЕНСУСА

Заблудин Герман Юрьевич
Московский технический университет связи и информатики, магистрант, Москва, Россия
g.zabludin@gmail.com

Тутова Наталья Владимировна
Московский технический университет связи и информатики,
зав. кафедрой «Бизнес-информатика», к.т.н., доцент, Москва, Россия
n.v.tutova@mtuci.ru

Слядников Павел Евгеньевич
Московский технический университет связи и информатики,
старший преподаватель кафедры «Бизнес-информатика», Москва, Россия
p.e.slyadnikov@mtuci.ru

Аннотация
В работе рассмотрена роль алгоритмов распределенного консенсуса, таких как Paxos, Raft и Practical Byzantine Fault Tolerance, в обеспечении отказоустойчивости информационных систем. Проанализированы требования корпоративных систем, проблемы, возникающие при реализации механизмов консенсуса, и компромиссы между масштабируемостью, производительностью и безопасностью.

Ключевые слова
Распределенные системы, CAP-теорема, согласованность данных, консенсус, Paxos, Raft, pBFT

Введение

В традиционных бизнес-инфраструктурах для обеспечения отказоустойчивости использовались централизованные архитектуры с механизмами обхода отказа и репликации данных для достижения избыточности данных. Однако по мере масштабирования систем с учетом межрегиональных глобальных операций и растущих объемов распределенных транзакций ранее применяемые подходы часто оказываются недостаточными.

Централизованные системы по своей сути ограничены зависимостью от узлов в составе решений, которые являются узкими местами или даже потенциальными точками отказа, особенно при высоких нагрузках или во время неожиданных сбоев.

В качестве решения появились распределенные архитектуры, обеспечивающие повышенную масштабируемость, доступность и отказоустойчивость. Однако применение таких архитектур сопряжено с появлением других проблем, в частности, в необходимости поддержания согласованности между распределенными компонентами и данными.

Алгоритмы распределенного консенсуса стали краеугольным камнем достижения отказоустойчивости в информационных системах. Они особенно важны в распределенных средах, где компоненты могут выходить из строя, вести себя непредсказуемо или работать асинхронно.

Теорема CAP

Погружение в проблематику стоит начать с освещения фундаментальной теоремы, лежащей в основе распределенных систем и их архитектур. Теорема CAP, также известная как теорема Брюера [1], является основополагающим принципом распределенных систем, определяющим компромисс между тремя ключевыми свойствами: согласованностью, доступностью и устойчивостью к разделению (рис. 1). Сформулированная Эриком Брюером и впоследствии формально доказанная, теорема утверждает, что распределенная система может одновременно гарантировать не более двух из этих трех свойств.

Согласованность (Consistency) в контексте теоремы CAP означает, что все узлы распределенной системы имеют одинаковое представление о данных в любой момент времени. Это означает, что каждая операция чтения отражает последнюю операцию записи, независимо от того, какой узел обрабатывает

запрос. Доступность (Availability), с другой стороны, гарантирует, что каждый запрос к системе получит ответ, даже при наличии сбоев. Под устойчивостью к разделению (Partition Tolerance) понимается способность системы продолжать корректно функционировать даже при наличии разрывов связи или разделов между узлами сети [2].

Теорема утверждает, что в случае разделения сети распределенная система должна выбирать между согласованностью и доступностью. Системы, для которых приоритетом является согласованность, гарантируют, что все узлы согласны с одним и тем же состоянием, но они могут пожертвовать доступностью, отклоняя запросы, когда происходит разделение. И наоборот, системы, для которых приоритетом является доступность, гарантируют, что запросы всегда будут обработаны, даже если это означает, что узлы могут временно иметь разные представления о данных.

Рис. 1. CAP-теорема в формате диаграммы Эйлера

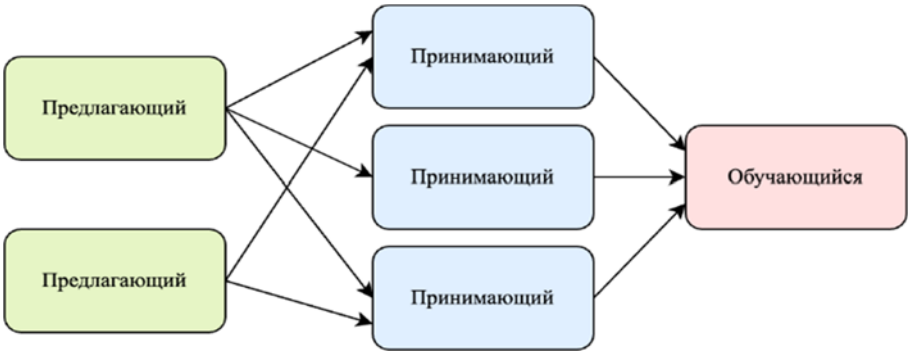

Рис. 2. Взаимодействие узлов в алгоритме Paxos

Отказоустойивость – одна из основных характеристик корпоративных систем, обеспечивающая сохранение работоспособности системы, несмотря на наличие сбоев. Поскольку информационные системы все больше полагаются на распределенные архитектуры для выполнения межрегиональных глобальных операций в разных зонах доступности, важность отказоустойчивости становится еще более очевидной. Сбои в распределенной среде могут быть вызваны неисправностями оборудования, ошибками в программном обеспечении или сбоями сети, а в некоторых случаях и злонамеренными действиями. Эта способность является ключевой для обеспечения точной обработки транзакций и сохранения согласованности данных на всех узлах.

Уникальные требования корпоративных систем, в частности в финансовом и банковском секторах, усиливают потребность в отказоустойчивости. Эти требования включают обработку больших объемов транзакций, работу в режиме реального времени и соблюдение строгих нормативных стандартов точности и безопасности данных. Любое отклонение от этих требований может иметь негативные последствия, нарушая работу сервисов, подрывая доверие клиентов и подвергая организации юридическим и финансовым обязательствам, репутационным рискам. Для достижения отказоустойчивости в этом контексте требуются сложные механизмы, обеспечивающие согласование и синхронизацию между распределенными компонентами.

Алгоритмы распределенного консенсуса

Алгоритмы распределенного консенсуса играют ключевую роль в решении этих проблем, предоставляя средства для достижения согласия по одному состоянию или значению в сети распределенных узлов. Позволяя узлам координировать свои действия и восстанавливаться после сбоев без ущерба для согласованности и доступности, эти алгоритмы составляют основу отказоустойчивых распределенных корпоративных систем. Их способность переносить сбои, включая аварийные сбои и византийское поведение, освещенное в криптологической задаче «О византийских генералах» [3], делает их незаменимыми в средах, где ставки на отказ исключительно высоки.

В настоящее время широкую известность имеют алгоритмы Paxos, Raft и PBFT. Здесь уместно будет остановиться подробнее и рассмотреть принципы работы алгоритмов более детально.

Paxos – один из самых ранних и наиболее широко изученных протоколов распределенного консенсуса. Алгоритм был разработан Лесли Лэмпортом и предназначался для гарантированной согласованности данных в распределенной среде при потенциальных отказах узлов. Сам алгоритм реализует процесс, в рамках которого участники взаимодействия общаются друг с другом по установленным правилам для достижения общего и согласованного решения. Алгоритм включает в себя три основные роли участников: предлагающие (Proposer), принимающие (Acceptor) и обучающиеся (Learner). Предлагающие инициируют предложения, принимающие участвуют в процессе голосования, а обучающиеся наблюдают за результатом, чтобы узнать согласованное значение (рис. 2).

Алгоритм состоит из двух основных фаз: фазы предложения и фазы подтверждения [4]. На этапе предложения предлагающий формирует предложение с тем значением, которое он хочет согласованно утвердить для системы. В качестве идентификатора предложения зачастую используется последовательно возрастающее число. Затем предлагающий отправляет сообщение с этим идентификатором предложения большинству принимающих. В рамках фазы подтверждения каждый принимающий, получив сообщение, сравнивает идентификатор предложения с самым высоким числом, которое он видел в прошлых этапах согласования. Если номер нового предложения выше, принимающий обещает не принимать в будущем предложения с меньшими номерами. Это гарантирует, что более ранние предложения не будут мешать более новым и что система сохранит последовательность. Консенсус достигается, когда большинство принимающих согласны принять предложение. В этот момент принимающие уведомляют обучающихся о результате, гарантируя, что значение с этого момента известно всей системе. Несмотря на то, что алгоритм гарантирует, что не может быть выбрано два разных значения, его практическая реализация может быть сложной из-за накладных расходов на несколько раундов связи между участниками взаимодействия. Тем не менее, несмотря на сложность реализации стоит отметить его практическое применение в таких продуктах как Google Spanner и Apache Cassandra.

Raft был создан как более простая технология достижения консенсуса, учитывая недостатки более старого Paxos [5]. Алгоритм основан на концепции лидера (Leader), где один узел избирается в его качестве для управления взаимодействием и поддержания согласованного состояния всей системы, а остальные узлы выступают в роли последователей (Follower), причем некоторые из них могут стать кандидатами во время следующих выборов лидера.

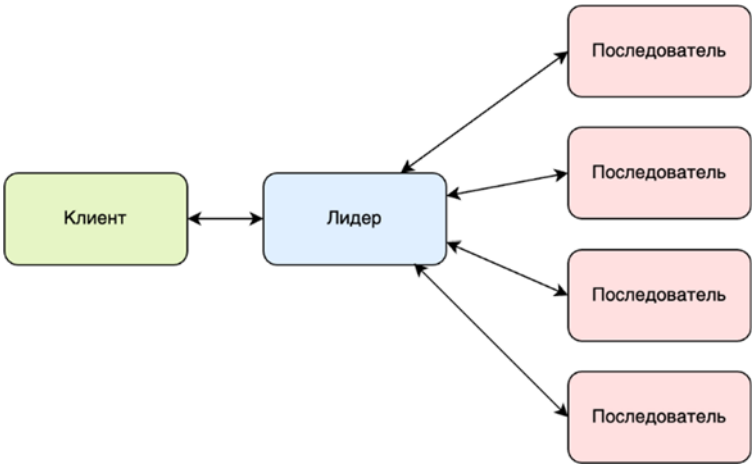

Рис. 3. Взаимодействие узлов в алгоритме Raft

Выборы инициализируются, когда последователи не получают heartbeat-сообщений от лидера в рамках установленного тайм-аута [6]. Последователь переходит в разряд кандидатов на лидерство и начинает выборы, отправляя запросы на голосование другим узлам. Кандидат становится лидером, если получает большинство голосов (рис. 3), таким образом процесс гарантирует, что одновременно существует только один лидер в кластере. После избрания лидер берет на себя ответственность за управление журналом записей, реплицированных на всех узлах, добавляя новые записи в свой журнал, реплицируясь с последователями и формируя heartbeat-ы, необходимые ему для поддержания статуса лидера в кластере. Если журнал последователя не совпадает с журналом лидера, алгоритм перезаписывает противоречивые записи и добавляет недостающие. Таким образом гарантируется, что лидером не станет кандидат, чей журнал не соответствует остальным узлам в кластере.

Алгоритм нашел практическое применение во многих продуктах, применяющихся при разработке высоконагруженных и отказоустойчивы корпоративных приложений, таких как Apache Kafka, Clickhouse, Etcd [7].

Practical Byzantine Fault Tolerance (pBFT) – это алгоритм, разработанный Мигелем Кастро и Барбарой Лисков, который предназначен для работы в распределенных системах, где узлы могут проявлять так называемые «византийские ошибки» [8].

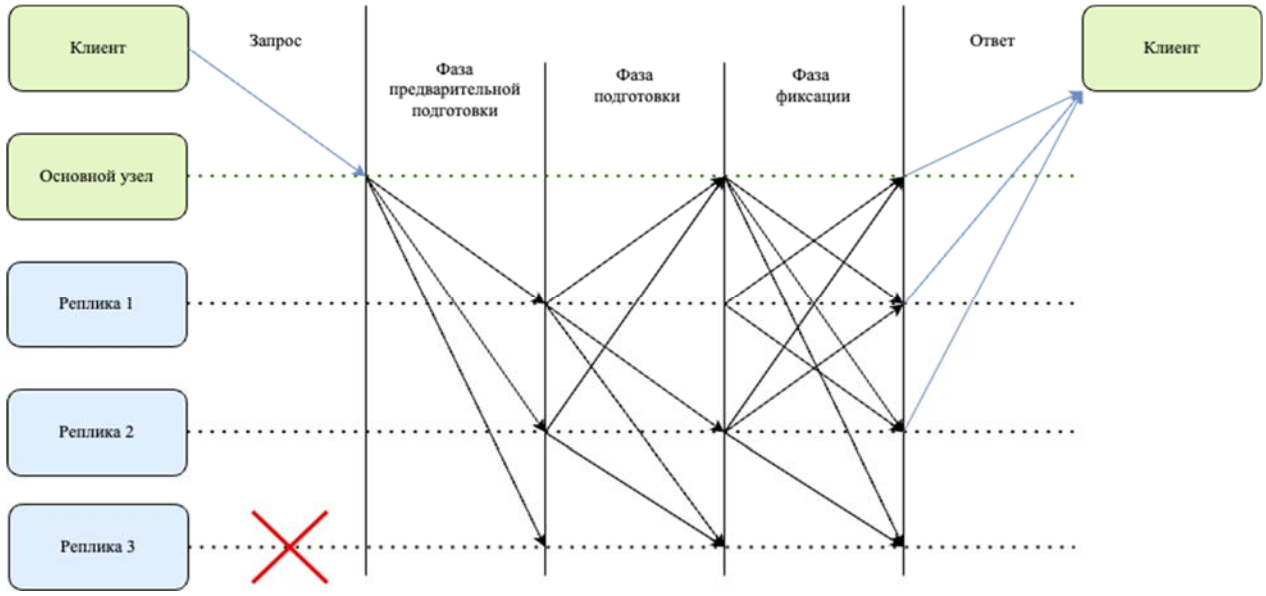

Рис. 4. Фазы алгоритма pBFT

Процесс начинается с того, что клиент посылает запрос основному узлу, который перенаправляет запрос репликам. Таким образом достигается наличие единой отправной точки принятия решений. Затем реплики последовательно вступают в цикл фаз для достижения консенсуса (рис. 4).

В рамках фазы предварительной подготовки основной узел отправляет предложение репликам, которые, проверяют его достоверность и переходят в фазу подготовки, передавая свое согласие с предложением всем остальным узлам. Если реплика получает согласие от большинства узлов, она переходит в фазу фиксации, отправляя сообщение остальным репликам о своей готовности к выполнению операции. Фаза предотвращает ситуацию, когда часть узлов уже выполнила ситуацию, а часть – нет. Таким образом достигается конечная согласованность всех узлов.

Несмотря на то, что описанный процесс подразумевает большое количество коммуникации между узлами, что в свою очередь может привезти к накладным расходам в сети, и может стать ограничением на внедрение, pBFT остается оптимальным для систем, работающих в недоверенных и асинхронных средах.

Применение алгоритмов консенсуса в корпоративных средах

Рассмотрев алгоритмы, закономерным будет рассмотреть их практическое применение в корпоративных средах и, в частности, в финансовом секторе. Финансовая индустрия, характеризующаяся большими объемами транзакций, строгими нормативными требованиями и потребностью в глобальных

операциях реального времени, получает значительные преимущества от использования алгоритмов консенсуса.

Ярким примером являются платежные системы, в которых обеспечение согласованности записей о транзакциях имеет первостепенное значение для предотвращения двойного расходования средств. Глобальность присутствия платежных систем также обусловлена использованием алгоритмов для репликации данных между разными ЦОДами и регионами.

Финансовые торговые платформы – еще одна область, где алгоритмы используются, обеспечивая единое представление рыночных данных для всех участников. Отказоустойчивость, обеспечиваемая рассмотренными алгоритмами, гарантирует бесперебойную работу торговых платформ даже в условиях сбоев глобальных сетей.

Системы на основе блокчейна, в свою очередь используют алгоритмы для формирования децентрализованных бухгалтерских записей. Возможность достижения единого состояния между географически разнесенными узлами позволяет системам работать прозрачно, что делает их подходящими для трансграничных платежей, бирж цифровых активов и токенизированных ценных бумаг.

Говоря о финансовом секторе и применимости алгоритмов нельзя не упомянуть банки. Репликация банковских данных в географически распределенных ЦОДах требует наличия механизмов поддержания согласованности и отказоустойчивости. Алгоритмы консенсуса позволяют синхронизировать базы данных, обеспечивая точность и актуальность клиентских данных, остатках на их счетах и истории операций. Банковские процессы скоринга и андеррайтинга также используют алгоритмы распределенного консенсуса. Системы, принимающие участие в этих процессах, должны агрегировать данные из множества источников для расчета величины риска. Алгоритмы обеспечивают согласованность и отсутствие фальсификации собранных данных, позволяя принимать взвешенные решения на основе достоверной информации.

Алгоритмы в данный момент переживают бурное развитие, вызванное ростом сложности и масштабностью глобальных финансовых систем. Одной из важнейших тенденций развития является разработка гибридных алгоритмов консенсуса, которые объединяют сильные стороны сразу нескольких подходов. Например, алгоритмы, сочетающие аспекты Paxos и Raft с механизмами византийской отказоустойчивости (BFT), обеспечивают повышенную отказоустойчивость при сохранении эффективности в средах со смесью доброкачественных и злонамеренных сбоев. Эти гибридные модели хорошо подходят для финансовых систем, требующих одновременно высокой безопасности и низкой задержки, таких как децентрализованные биржи и платежные системы реального времени. В то же время развитие технологии блокчейн стало катализатором инноваций в механизмах консенсуса, адаптированных для распределенных записей. Такие алгоритмы, как Proof of Stake (PoS), Delegated Proof of Stake (DPoS) и Delegated Byzantine Fault Tolerance (dBFT), предлагают альтернативу традиционному Proof of Work (PoW) [9], снижая энергопотребление и увеличивая пропускную способность транзакций. Эти специфические для блокчейна механизмы консенсуса находят применение в децентрализованных финансах (DeFi), управлении токенизированными активами и трансграничных платежах, где очень важен компромисс между безопасностью, масштабируемостью и эффективностью.

Заключение

Алгоритмы распределенного консенсуса являются основой для обеспечения отказоустойчивости, согласованности и безопасности в современных корпоративных системах [10-13], за счет возможности поддерживать операционную целостность и надежность даже в неблагоприятных условиях. Они находят широкое применнение в современных финансовых, банковских и торговых системах. Однако, несмотря на свой потенциал такие алгоритмы характеризуются сложностью реализации и дополнительными коммуникационными расходами в целях достижения согласованности между узлами.

В заключение можно отметить, что в условиях все более взаимосвязанного и цифрового мира, их постоянное развитие и совершенствование, в том числе в синергии с технологиями блокчейн, будут иметь глобальное значение для решения сложных задач современных корпоративных систем и секторов экономики.

Литература

1. *Лисовский А.Т.* Теорема CAP в распределенных хранилищах данных // Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях: Материалы XXII Республиканской научной конференции студентов и аспирантов, Гомель, 25-27 марта 2019 года. Гомель: Гомельский государственный университет им. Франциска Скорины, 2019. – С. 79-80.

2. *Бурмистров А.В., Белов Ю.С.* Недостатки реляционных баз данных // Электронный журнал: наука, техника и образование. 2015. № 3(3). С. 25-34.

3. *Черняк Л.* Облака и византийские генералы // Открытые системы. СУБД. 2012. № 1. С. 44.

4. *Жиленков А.А., Черный С.Г*. Существование и достижимость консенсуса как проблема обеспечения надёжности в распределённых приложениях и киберфизических системах // Электротехнические и информационные комплексы и системы. 2020. Т. 16, № 2. С. 92-104. DOI 10.17122/1999-5458-2020-16-2-92-104

5. *Полунин А.А.* Применение протоколов решения задач консенсуса в распределенных системах вычислений на базе микросервисной архитектуры // Сборник избранных статей научной сессии ТУСУР. 2022. № 1-2. С. 170-174.

6. *Насибуллин А.Р., Новиков Б.А.* Репликация в распределенных системах: модели, методы и протоколы // Программирование. 2020. № 5. С. 60-71. DOI 10.31857/S0132347420050064

7. *Шуляк А.В*. Анализ производительности хранилища etcd // Научные горизонты. 2021. № 1(41). С. 132-141.

8. *Мельников М.О*. Консенсус-алгоритмы в блокчейн-системах // Современные методы прикладной математики, теории управления и компьютерных технологий (ПМТУКТ-2023) : сборник трудов Международной научной конференции, Воронеж, 04-06 декабря 2023 г. Воронеж: Воронежский государственный педагогический университет, 2023. С. 56-58.

9. *Захаров Д.А., Пелипас В.О*. Основы алгоритмов консенсуса // Мир компьютерных технологий : Сборник статей всероссийской научно-технической конференции студентов, аспирантов и молодых ученых,, Севастополь, 06-10 апреля 2020 г. / Науч. редактор Е.Н. Мащенко. Севастополь: Федеральное государственное автономное образовательное учреждение высшего образования "Севастопольский государственный университет", 2020. С. 163-168.

10. *Kirov D.E., Toutova N.V., Vorozhtsov A.S., Andreev I.A.* Feature selection for predicting live migration characteristics of virtual machines // T-Comm. 2021. Т. 15. № 7. С. 62-70. EDN: AGGBDW

11. *Тутов А.В., Тутова Н.В., Ворожцов А.С., Андреев И.А.* Многокритериальная оптимизация размещения виртуальных машин по физическим серверам в облачных центрах обработки данных // T-Comm: Телекоммуникации и транспорт. 2021. Т. 15. № 1. С. 28-34. EDN: IOFQSS

12. *Губин А.С., Тутова Н.В.* Анализ подхода к разработке приложений с "чистой" архитектурой // Телекоммуникации и информационные технологии. 2022. Т. 9. № 1. С. 28-37. EDN: NOZMKG

13. *Дубельщиков А.А., Тутова Н.В.* Навыки яндекс.алиса: от идеи до реализации // Телекоммуникации и информационные технологии. 2020. Т. 7. № 2. С. 92-97. EDN: ZIFDYG

АВТОМАТИЗАЦИЯ РАЗВЁРТЫВАНИЯ И УПРАВЛЕНИЯ ASTERISK В МИКРОСЕРВИСНОЙ АРХИТЕКТУРЕ С ПРИМЕНЕНИЕМ KUBERNETES

Маликова Елена Егоровна
Московский технический университет связи и информатики,
кафедра Сетей связи и систем коммутации, к.т.н., доцент, Москва, Россия
e.e.malikova@mtuci.ru

Фартышев Михаил Александрович
Московский технический университет связи и информатики, студент, Москва, Россия
fartyshevmihail@gmail.com

Рогач Иван Сергеевич
Московский технический университет связи и информатики,
кафедра Сетей связи и систем коммутации, ассистент, Москва, Россия
i.s.rogach@mtuci.ru

Аннотация
В данной работе рассматриваются современные тенденции и задачи, с которыми встречаются многие VoIP-сервисы в условиях быстро меняющейся технологической среды. Цель исследования заключается в анализе новых инструментов для реализации виртуальной АТС, а именно виртуализации и контейнеризации, влияющих на эффективную работу VoIP- сервисов. Рассматриваются ключевые аспекты интеграции системы IP-телефонии в контейнеризированную среду. Работа направлена на исследование возможностей автоматизации и оптимизации использования виртуальной АТС для повышения качества и надежности IP-телефонии в контейнеризированных средах.

Ключевые слова
Виртуализация, контейнер, контейнерная оркестрация, IP-телефония, VoIP, Kubernetes Cluster, K8s, манифест, Asterisk, Kamailio

Введение

В современных телекоммуникационных системах программные решения на базе IP-телефонии играют ключевую роль, обеспечивая гибкость, масштабируемость и экономическую эффективность. Одним из наиболее популярных свободных решений с открытым исходным кодом является программная автоматическая телефонная станция (АТС) типа Asterisk, которая широко используется для организации корпоративных и глобальных Voice over IP – сетей (VoIP) [1].

С развитием облачных технологий и контейнеризации возрастает необходимость в эффективном развёртывании и управлении Asterisk в среде, обеспечивающей высокую отказоустойчивость, автоматическое масштабирование и удобство эксплуатации. Kubernetes (K8s), является стандартом контейнерной оркестрации, который предоставляет все необходимые инструменты для построения гибкой и отказоустойчивой инфраструктуры IP-телефонии.

В данной статье рассматриваются подходы к автоматизации развёртывания и управления Asterisk в микросервисной архитектуре с применение K8s. Основное внимание уделяется вопросам интеграции Asterisk в контейнерную среду, взаимодействия с протоколами SIP и RTP [2], обеспечению масштабируемости и отказоустойчивости АТС.

Обзор технологий для реализации виртуальной АТС

В процессе развёртывания сервиса IP – телефонии с Asterisk в K8s кластере будут использованы несколько ключевых технологий:

Kubernetes (K8s) – платформа для автоматизации развёртывания, масштабирования и управления контейнерными приложениями [3]. Kubernetes позволяет эффективно управлять компонентами системы, обеспечивая балансировку нагрузки внутри кластера, поддерживать достаточную степень отказоустойчивости системы и автоматическое восстановление сервисов. Все настройки описываются декларативно с помощью манифестов в формате YAML или JSON, это определённый формат файла,

воспринимаемый K8s. *Подами* называются запущенные контейнеры с определёнными характеристиками, внутри которых запущены приложения. *Ноды кластера* – управляющие и вычислительные узлы. *Мастер нода (Master Node)* отвечает за постановку задач, принятых от API сервера, *воркер нода (Worker Node)* отвечает за запуск контейнера на своих вычислительных ресурсах.

Asterisk – программная телефонная станция (IP-PBX), обеспечивающая обработку голосовых вызовов, маршрутизацию, голосовую почту, конференции и другие функции IP-телефонии. Asterisk поддерживает протоколы SIP, RTP и интеграцию с различными внешними системами, что делает его гибким решением для построения телефонных сетей [4].

Kamailio – SIP-прокси-сервер, который используется для обработки и маршрутизации SIP-трафика. Kamailio помогает распределять нагрузку между несколькими подами с Asterisk, улучшает безопасность за счёт аутентификации и авторизации трафика и повышает масштабируемость системы.

PostgreSQL – реляционная база данных, используемая для хранения информации о пользователях, детализации вызовов (CDR) и конфигурационных данных. PostgreSQL обеспечивает надежное хранение и обработку данных, а также легко интегрируется с Asterisk и другими сервисами.

Helm – пакетный менеджер, для управления Kubernetes – приложениями, упрощающий развёртывание сложных систем за счёт шаблонных манифестов. С помощью менеджера Helm можно автоматизировать установку Asterisk, Kamailio и PostgreSQL, управляя настройками через конфигурационные файлы.

Архитектура развёртывания Asterisk в Kubernetes

Развёртывание Asterisk в Kubernetes требует декомпозиции монолитного приложения на отдельные микросервисы, что позволяет добиться гибкости, масштабируемости и управляемости системы [5-9].

Основными компонентами такой архитектуры являются:

1. Asterisk Core – основной сервис, обрабатывающий голосовые вызовы, маршрутизацию и обработку медиапотоков. Он работает в контейнере и требует гибкой конфигурации для взаимодействия с внешними системами.
2. SIP-прокси (Kamailio) – выполняет функции маршрутизации и балансировки SIP-сообщений перед передачей в Asterisk, разгружая основную нагрузку и повышая отказоустойчивость.
3. База данных (PostgreSQL, MySQL) – используется для хранения конфигурации, данных CDR (Call Detail Record) и авторизации пользователей.
4. Настройка и управление (ConfigMaps, Secrets) – Kubernetes предоставляет инструменты для централизованного управления конфигурациями и секретами (логины, пароли, ключи шифрования).
5. Балансировщик нагрузки (Ingress Controller, LoadBalancer) – обеспечивает распределение трафика между репликами сервисов и предоставляет точку входа в кластер.
6. Мониторинг и логирование (Prometheus, Grafana) – используются для отслеживания метрик, производительности и анализа логов работы с Asterisk.

Развёртывание Asterisk в Kubernetes кластере можно организовать несколькими способами, каждый из которых имеет свои особенности и области применения.

Монолитное развёртывания предполагает запуск Asterisk в одном контейнере со всеми необходимыми модулями. Это самый простой вариант, подходящий для тестирования или небольших систем. Однако такой подход затрудняет масштабирование и обновление приложения.

Микросервисный подход предусматривает разделение функций Asterisk на несколько контейнеров: SIP-прокси (Kamailio), база данных (PostgreSQL), RTP-прокси (RTP Proxy) и т.д. Такой вариант обеспечивает гибкость, отказоустойчивость и удобство обновления, но требует более сложной настройки.

Гибридный подход сочетает в себе элементы монолитного и микросервисного развёртывания, позволяя выделить критические компоненты в отдельные сервисы, а основные функции оставить в одном контейнере. Это хороший компромисс для постепенной миграции.

Практическая реализация и тестирование

Была разработана структурная схема взаимодействия с виртуальной АТС на базе Asterisk, которую мы развернём в Kubernetes кластере. Схема представлена на рисунке 1.

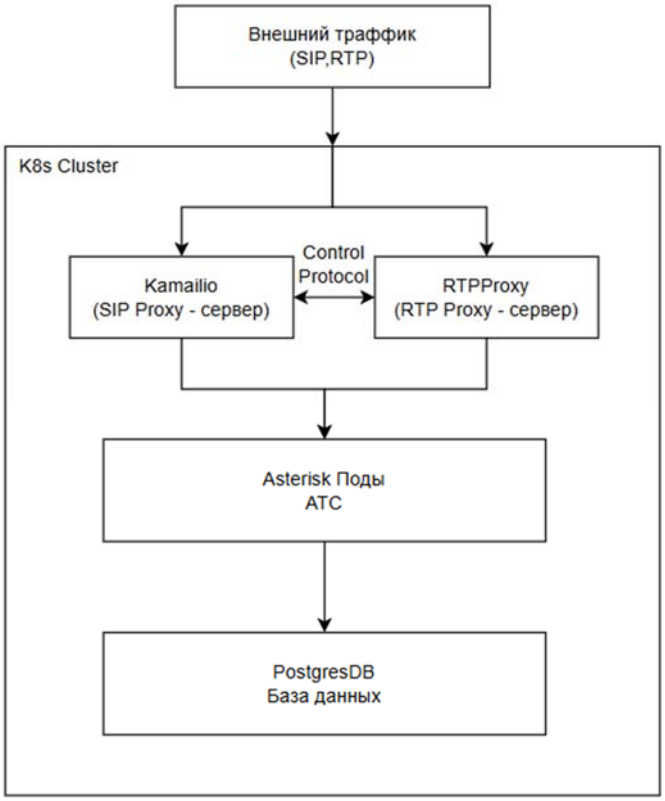

Рис. 1. Структурная схема взаимодействия компонентов системы

Для начала нам нужно развернуть K8s кластер. Одна из минимальных конфигураций подразумевает под собой один Master – узел и один Worker – узел. В нашем случае будут использованы виртуальные машины (ВМ) со следующими характеристиками:

– Master Node с двумя виртуальными процессорами и 2 ГБ оперативной памяти;

– Worker Node с четырьмя виртуальными процессорами и 8 ГБ оперативной памяти.

Выбор мощностей ВМ не случаен: минимальные параметры узла в K8s – кластере – 2 CPU (Central Processing Unit) и 2 ГБ оперативной памяти. Для Master Node этого вполне достаточно, т.к. при наличии хотя бы одного Worker Node запуск контейнеров производится на Worker Node, в то время как Master Node занимается отслеживанием жизнеспособности подов, перезапуском и т.д.

Для того, чтобы инициализировать K8s кластер с помощью kubeadm необходимо ввести следующую команду на Master Node:

**kubeadm init **

--apiserver-advertise-address =192.168.255.136

Данная команда инициализирует запуск K8s кластера, с адресом API-сервера по адресу 192.168.255.136.

В результате выполнения команды получим ответ, о том, что кластер инициализирован. Дальнейшие инструкции по импорту файла конфигурации для взаимодействия с кластером. Указания о том, как лучше развернуть плагин для управления IP адресацией подов, а также команда для добавления первого Worker Node в кластер со сгенерированным токеном. *Токены Kubernetes* – это криптографические учетные данные, которые используются для аутентификации пользователей и учетных записей служб при взаимодействии с сервером API Kubernetes. Вывод данных команд представлен на рисунке 2.

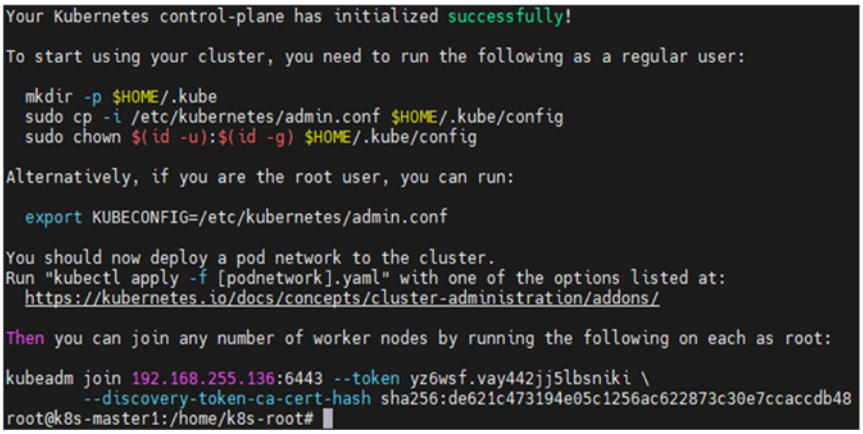

Рис. 2. Результат выполнения команды

Далее необходимо экспортировать файл конфигурации созданного кластера, для взаимодействия с ним. Чтобы это сделать нужно ввести команду:

export KUBECONFIG=/etc/Kubernetes/admin.conf

Следующим этапом будет добавление узлов Worker Node в кластер. Для этого на Master Node необходимо ввести следующие команды:

kubeadm token generate

kubeadm token create >generated-token> --print-join-commnd –ttl=0

Результат вывода команд представлен на рисунке 3.

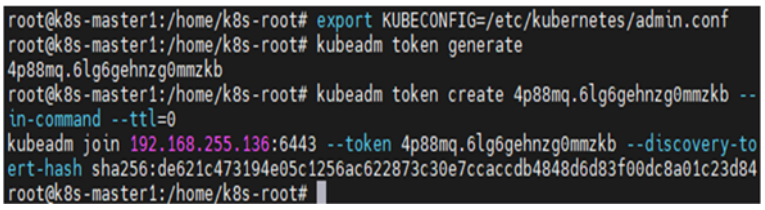

Рис. 3. Экспорт файла конфигурации и генерация токена для подключения Worker Node

В результате мы получим команду, которую нужно ввести на Worker Node для присоединения к кластеру. Команда имеет следующий вид:

kubeadm join --discovery-token abcdef.1234567890abcdef --discovery-token-ca-cert-hash sha256:1234..cdef 1.2.3.4:6443

Далее нам необходимо ввести данную команду на Worker node. Вывод команды представлен на рисунке 4.

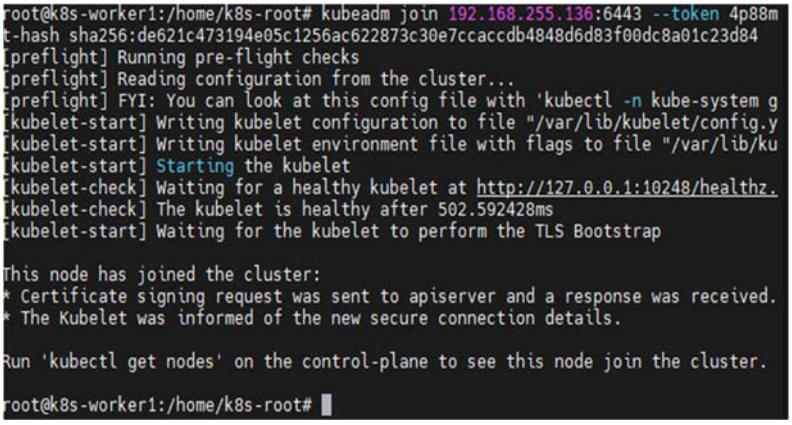

Рис. 4. Результат подключения Worker node

Проверим состояние узлов и подов на созданном кластере с помощью команд:

kubectl get nodes

kubectl get pod -A -o wide

На рисунке 5 можно увидеть все запущенные поды в кластере и общую информацию о них. Так же представлен вывод команды состояния мастер и Worker Node.

```
root@k8s-master1:/home/k8s-root# kubectl get pod -A -o wide
NAMESPACE     NAME                                 READY   STATUS
kube-system   cilium-c2clw                         1/1     Running
kube-system   cilium-envoy-5wplh                   1/1     Running
kube-system   cilium-envoy-vzx6m                   1/1     Running
kube-system   cilium-n45nd                         1/1     Running
kube-system   cilium-operator-59f8db7bf5-z89k5     1/1     Running
kube-system   coredns-55cb58b774-9p7ps             1/1     Running
kube-system   coredns-55cb58b774-bwvxl             1/1     Running
kube-system   etcd-k8s-master1                     1/1     Running
kube-system   kube-apiserver-k8s-master1           1/1     Running
kube-system   kube-controller-manager-k8s-master1  1/1     Running
kube-system   kube-proxy-4v844                     1/1     Running
kube-system   kube-proxy-8pgkt                     1/1     Running
kube-system   kube-scheduler-k8s-master1           0/1     Running
root@k8s-master1:/home/k8s-root# kubectl get nodes
NAME          STATUS   ROLES           AGE     VERSION
k8s-master1   Ready    control-plane   13m     v1.30.9
k8s-worker1   Ready    <none>          9m43s   v1.30.9
root@k8s-master1:/home/k8s-root#
```

Рис. 5. Сведения о состоянии узлов и подов

Теперь можно приступать к запуску подов Asterisk, Kamailio, RTPProxy, PostgreSQL.
Создадим пространство имён VoIP, и добавим нашу Worker – ноду для Kamailio:

kubectl create namespace voip
kubectl label nodes k8s-worker1 kamailio=true

Далее необходимо создать манифесты и файлы конфигурации, которые описывают параметры создания подов в кластере. Создадим файлы в формате yaml для всех сервисов и опишем их параметры:

kamailio.cfg (основной файл конфигурации)

```
listen=udp:0.0.0.0:5060

loadmodule "dispatcher.so"
loadmodule "rtpproxy.so"

modparam("rtpproxy", "rtpproxy_sock", "udp:rtpproxy-service.voip.svc.cluster.
modparam("dispatcher", "list_file", "/etc/kamailio/dispatcher.list")

route {
    if (!ds_select_dst("0", "4")) {
        sl_send_reply("500", "Server error");
        exit;
    }
    t_relay();
```

Рис. 6. Файл конфигурации kamailio.cfg

dispatcher.list (список Asterisk – подов)

```
1 sip:asterisk-0.asterisk.voip.svc.cluster.local:5060
```

Рис. 7. Список Asterisk – подов dispatcher.list

kamailio-daemonset.yaml (Kamailio manifest)

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: kamailio
  namespace: voip
spec:
  selector:
    matchLabels:
      app: kamailio
  template:
    metadata:
      labels:
        app: kamailio
    spec:
      nodeSelector:
        kamailio: "true"
      containers:
      - name: kamailio
        image: kamailio/kamailio:5.6.4-alpine
        ports:
          - containerPort: 5060
            protocol: UDP
        volumeMounts:
          - name: kamailio-config
            mountPath: /etc/kamailio
      volumes:
        - name: kamailio-config
          configMap:
            name: kamailio-config
```

Рис. 8. Manifest kamailio-daemonset.yaml

rtpproxy-deployment.yaml (RTP Proxy manifest)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: rtpproxy
  namespace: voip
spec:
  replicas: 1
  selector:
    matchLabels:
      app: rtpproxy
  template:
    metadata:
      labels:
        app: rtpproxy
    spec:
      containers:
      - name: rtpproxy
        image:  cgrindel/rtpproxy:latest
        args: "-l", "0.0.0.0", "-s", "udp:localhost:7722"]
        ports:
          - containerPort: 7722
            protocol: UDP
```

Рис. 9. Manifest rtpproxy-deployment.yaml

rtpproxy-service.yaml (Сервис для RTP Proxy)

```yaml
apiVersion: v1
kind: Service
metadata:
  name: rtpproxy-service
  namespace: voip
spec:
  selector:
    app: rtpproxy
  ports:
    - protocol: UDP
      port: 7722
      targetPort: 7722
  type: NodePort
```

Рис. 10. Manifest rtpproxy-service.yaml

asterisk-statefulset.yaml (Asterisk manifest)

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: asterisk
  namespace: voip
spec:
  serviceName: asterisk
  replicas: 1  # Одна реплика из-за ограничений ноды
  selector:
    matchLabels:
      app: asterisk
  template:
    metadata:
      labels:
        app: asterisk
    spec:
      containers:
      - name: asterisk
        image: asterisk/asterisk:18.11.1
        ports:
          - containerPort: 5060
            protocol: UDP
        volumeMounts:
          - name: config
            mountPath: /etc/asterisk
        resources:
          limits:
            cpu: "2"
            memory: "2Gi"
      volumes:
        - name: config
          secret:
            secretName: asterisk-config
```

Рис. 11. manifest asterisk-statefulset.yaml

Запустим поды по созданным манифестам с помощью следующих команд:
kubectl apply -f kamailio-daemonset.yaml
kubectl apply -f rtpproxy-deployment.yaml
kubectl apply -f rtpproxy-service.yaml
kubectl apply -f asterisk-statefulset.yaml
Проверим состояние подов в K8s кластере с помощью команды
Kubectl get pod -A -o wide

```
root@k8s-master1:/home/k8s-root# kubectl get pod -A -o wide
NAMESPACE     NAME                                 READY   STATUS
kube-system   cilium-c2clw                         1/1     Running
kube-system   cilium-envoy-5wplh                   1/1     Running
kube-system   cilium-envoy-vzx6m                   1/1     Running
kube-system   cilium-n45nd                         1/1     Running
kube-system   cilium-operator-59f8db7bf5-z89k5     1/1     Running
kube-system   coredns-55cb58b774-9p7ps             1/1     Running
kube-system   coredns-55cb58b774-bwvxl             1/1     Running
kube-system   etcd-k8s-master1                     1/1     Running
kube-system   kube-apiserver-k8s-master1           1/1     Running
kube-system   kube-controller-manager-k8s-master1  1/1     Running
kube-system   kube-proxy-4v844                     1/1     Running
kube-system   kube-proxy-8pgkt                     1/1     Running
kube-system   kube-scheduler-k8s-master1           1/1     Running
voip          asterisk-0                           1/1     Running
voip          kamailio-l6m2x                       1/1     Running
voip          postgres-75f7f9c87f-g9j66            1/1     Running
voip          rtpproxy-65bf8b657-zv5t9             1/1     Running
```

Рис. 12. Состояние подов в K8s кластере

Как видно из состояния подов, все наши приложения успешно запустились и находятся в рабочем состоянии.

Произведём тестовый звонок и убедимся, что АТС функционирует нормально:

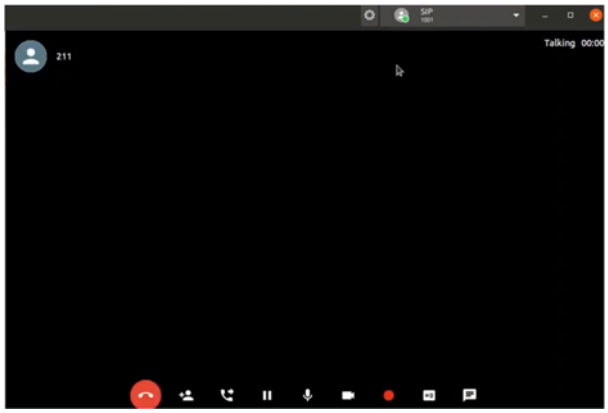

Рис. 13. Результат тестового звонка

Масштабируемость и отказоустойчивость

При развёртывании Asterisk в Kubernetes важно обеспечить стабильную работу системы при увеличении нагрузки на VoIP сервисы.

Проблемы масштабируемости решаются за счёт автоматического создания дополнительных подов из образа Asterisk. Kubernetes позволяет динамически масштабировать с помощью Horizontal Pod Autoscaler, масштабирования имеющихся ресурсов для ваших контейнеров, который отслеживает нагрузку процессора и оперативной памяти пода и в случае необходимости создаёт дополнительные поды. Kamailio в свою очередь выполняет роль балансировщика, распределяя нагрузку SIP – трафика между репликами Asterisk. База данных PostgreSQL также масштабируется за счёт создания подов-реплик.

Отказоустойчивость обеспечивается за счёт механизмов автоматического перезапуска контейнеров в случае возникновения ошибок. Также Kubernetes обеспечивает балансировку нагрузки внутри кластера, равномерно распределяя её между узлами кластера.

Перспективы развития

В процесс развития технологий и повышения требований к сервисам связи Asterisk в паре с Kubernetes имеет хорошие перспективы для дальнейшего использования и улучшения качества предоставляемых услуг. В дальнейшем можно ожидать значительные улучшения не только в производительности системы, но и в её гибкости, безопасности и отказоустойчивости.

Современные подходы к автоматизации, активно использующие в разработке DevOps инженерами, позволяют упростить управление инфраструктурой за счёт внедрения автоматизированных процессов, включающих в себя сборку, тестирование и развёртывание программного обеспечения. Использования пакетов для запуска готовых приложений в Kubernetes делает процесс управления сетью у операторов более удобным и предсказуемым.

Инструменты контейнеризации и виртуализации в совокупности существенно оптимизируют использование ресурсов, гибко распределяя их, в то время как IP – телефония имеет динамическую нагрузку. Дальнейшая оптимизация K8s для Asterisk может дать значительный прирост в оптимизации нагрузки на АТС, что позволит повысить качество обслуживания и обеспечить высокий уровень отказоустойчивости.

Заключение

В ходе работы над развертыванием АТС Asterisk в среде Kubernetes были успешно реализованы ключевые этапы, включая настройку инфраструктуры, автоматизацию развертывания и конфигурацию сервисов. Проведенные тесты, включая тестовый звонок, показали стабильную работу системы, что подтверждает возможность создания высокоэффективных и масштабируемых решений для IP-телефонии в контейнеризированной среде.

Использование Kubernetes для управления Asterisk позволяет обеспечить гибкость, отказоустойчивость и масштабируемость системы, что критично для современных требований к сервисам связи. Возможности автоматизации процессов, интеграции с облачными решениями и сетевыми технологиями

открывают широкие перспективы для дальнейшего развития и оптимизации работы таких сервисов.

Таким образом, проведенная практика и успешный тестовый звонок подтверждают жизнеспособность и эффективность выбранного подхода, а также создают основу для дальнейшего развития и совершенствования инфраструктуры на базе IP-телефонии с использованием виртуальной АТС Asterisk в платформе Kubernetes.

Литература

1. *Канищева М.Г., Маликова Е.Е., Пелевин И.И., Пшеничников А.П.* Основы работы с виртуальной телефонной станцией IP-АТС ASTERISK. М.: Издательский дом Медиа Паблишер, 2021.

2. *Маликов Е.Е., Пшеничников А.П.* Системы сигнализации в инфокоммуникационных сетях : учебное пособие. М.: МТУСИ, 2024. 160 с. Лань: электронно-библиотечная система. https://e.lanbook.com/book/439199 (дата обращения: 25.01.2025).

3. K8s. Официальный сайт: Программное обеспечение для контейнерной оркестрации: https://kubernetes.io/ru/

4. *Grushko S.A., Pshenichnikov A.P., Malikov A.Y., Malikov E.E.* Virtual Asterisk IP-PBX operation studying and exploring at the university. 2022 Systems of Signals Generating and Processing in the Field of on Board Communications, pp. 15-21.

5. Asterisk. Официальный сайт: Программное обеспечение АТС. https://www.asterisk.org/. (дата обращения 25.01.2025).

6. *Антонова В.М., Богомолова В.Е., Маликова Е.Е.* Методика преподавания дисциплины "инженерная и компьютерная графика" в современных условиях // Методические вопросы преподавания инфокоммуникаций в высшей школе. 2023. Т. 12. № 1. С. 76-81.

7. *Антонова В.М., Богомолов В.Е., Бужин И.Г., Маликова Е.Е.* Особенности интегрирования программы Anylogic в систему образования университета // Методические вопросы преподавания инфокоммуникаций в высшей школе. 2023. Т. 12. № 3. С. 4-12.

8. *Antonova V.M., Malikova E.E., Panov A.E., Spichek I.V., Malikov A.Y.* Implementation of IOT technology for data monitoring via cloud services // T-Comm. 2021. Т. 15. № 2. С. 46-53.

9. *Antonova V.M., Malikova E.E., Stepanov M.S.* Ways of the Asterisk software PBX protection // T-Comm. 2023. Т. 17. № 10. С. 52-58.

АНАЛИЗ МЕТОДОВ КОРРЕЛЯЦИИ СОБЫТИЙ В SIEM-СИСТЕМАХ ДЛЯ ВЫЯВЛЕНИЯ ИНЦИДЕНТОВ БЕЗОПАСНОСТИ В ФИНАНСОВЫХ ОРГАНИЗАЦИЯХ

Глебова Елизавета Михайловна
Московский технический университет связи и информатики, студентка магистратуры, Москва, Россия
e.m.glebova@edu.mtuci.ru

Аннотация
В статье рассматривается важность процесса корреляции данных в системах SIEM для выявления инцидентов и анализа угроз в условиях растущих объемов данных и усложняющихся атак. Подчеркивается, что корреляция важна для выявления причинно-следственных связей между событиями, а методы варьируются от правило-ориентированных д интеллектуальных подходов. Исследование подчеркивает необходимость разработки новых эффективных алгоритмов, способных автоматически обрабатывать и анализировать события.

Ключевые слова
Управление информационной безопасностью, SIEM, корреляция данных, автоматизация, эвристические методы, правило-ориентированный подход

Введение

На современном рынке имеется множество решений для обеспечения безопасности финансовых организаций, которые направлены на предупреждение и обнаружение кибератак, а также на мониторинг безопасности. Одной из таких систем являются SIEM, которые помогают выявлять инциденты и предупреждают о нарушениях безопасности. Эти системы используют методы нормализации, агрегации, фильтрации и корреляции событий для достижения своей цели. Однако усложнение атак и рост объема данных усложняют процесс обработки, что требует значительных ресурсов и разработки эффективных алгоритмов [1].

Корреляция данных в SIEM является ключевой, так как позволяет устанавливать причинно-следственные связи между событиями, что критично для выявления вредоносной активности и источников атак. Процесс корреляции сильно зависит от реализации конкретной системы, хотя наиболее популярным остается правило-ориентированный подход, который основывается на формулировании правил для связывания событий [2].

В статье рассматривается важность процессов корреляции, их функции и основные принципы, а также модель сбора данных для корреляции событий, связанных с информационной безопасностью. Обсуждаются различные этапы реализации подхода и использование собранных данных для функционального и поведенческого анализа, а также формулируются условия для эффективного применения SIEM.

Результаты исследований

Правильное понимание корреляции в системах управления информационной безопасностью (SIEM) критически важно, так как она служит фундаментом для решения множества стратегических задач. Благодаря корреляции можно не только обнаруживать взаимосвязи между событиями, происходящими на разных уровнях и в различных сферах безопасности, но и эффективно оценивать их значимость, формировать точные оповещения и оперативно выявлять потенциальные угрозы.

Изучение взаимосвязей в сфере безопасности подразумевает сбор данных из разнообразных источников и завершается составлением отчета, отражающего текущую защищенность рассматриваемой инфраструктуры [5]. Важно понимать, что выявление корреляций – это непрерывная работа, требующая постоянной обработки поступающей информации в режиме онлайн. Автоматизированная методика оценки разносторонней безопасности разрабатывается с использованием синтезированной информации, полученной при анализе защищаемых систем с разных точек зрения: структурной, функциональной, поведенческой и эволюционной. Такой комплексный подход обусловлен необходимостью глубокого понимания сложных и постоянно меняющихся объектов, которые представляют собой многоуровневые системы.

Процесс корреляции использует для своей работы широкий спектр данных, полученных как из внутренних, так и из внешних источников. К таким источникам относятся сенсоры, осуществляющие измерения, агенты, ответственные за сбор информации, логи событий и конфигурационные файлы объектов, составляющих инфраструктуру [6]. Представлена схема, иллюстрирующая взаимодействие различных источников данных, где ключевым элементом выступают базы данных, содержащие сведения о программном и аппаратном обеспечении, установленных в данной инфраструктуре. В данной модели исходные данные, именуемые "сырыми", интерпретируются как динамические внутренние сведения, а также как статичные внутренние и внешние сведения, что демонстрируется на рисунке 1.

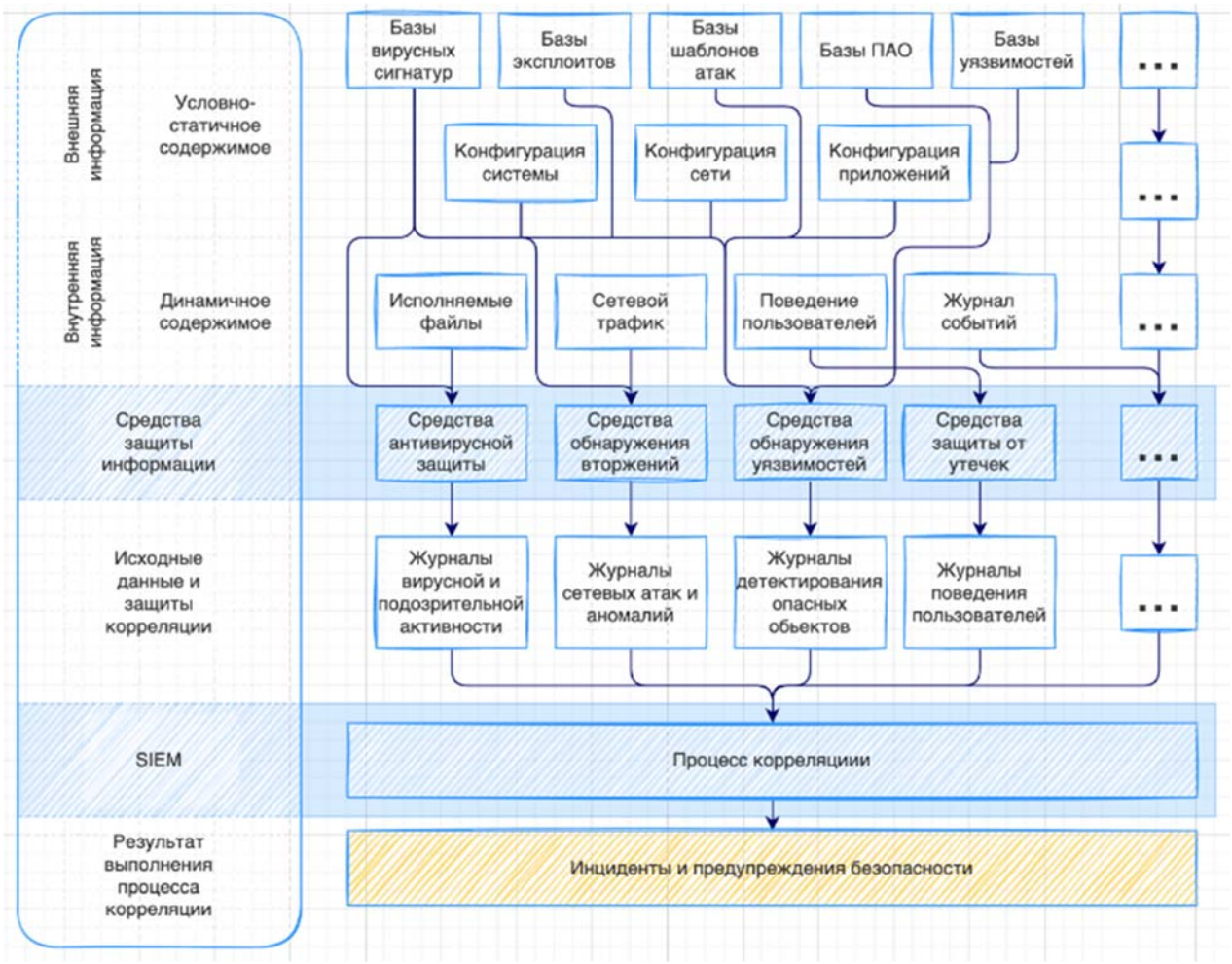

Рис. 1. Схема использования входных данных для процесса корреляции

Разделение данных обусловлено сложностями в объединении информации из разных категорий, ключевым фактором разницы между которыми является временная маркировка [7]. В рамках текущего этапа разработки, основной фокус уделяется динамическим данным, так как любая модификация статической информации может быть расценена как значимое событие. Хотя статические данные не являются ключевым фактором при определении уровня безопасности, их использование всё же возможно. Безопасность в системе достигается за счёт многоуровневой защиты, которая строится на предварительной анализе поступающей информации и обобщении её в более крупные события. При этом, нет единого соответствия между источниками данных и применяемыми методами обработки, и выбор подходящего инструмента определяется спецификой самого источника. Применение такого подхода влечет за собой использование событий безопасности различной природы и сложности при их корреляции, что создает сложности в обеспечении комплексной безопасности.

Независимо от исхода - успеха, отказа или ошибки - любое действие генерирует событие, которое можно трактовать как следствие его выполнения или же как отражение самой попытки его совершения. Источником такого события может быть как источник действия, так и система, обрабатывающая его. Каждый зафиксированный случай события имеет строго определённую структуру описания, понятную системе обработки, и снабжён уникальными атрибутами, отражающими суть произошедшего.

Исследовательские задачи опираются на события, запечатленные в журнале, которые можно рассматривать как наборы событий и соответствующих им типов журнала [8].

Изучение типов событий должно быть основано на фактических данных, например, журналах событий, что гарантирует точность и исключает погрешности, которые могут возникнуть при изменении формата событий, так как все модификации фиксируются. Анализ журналов событий позволяет составить профиль для каждого типа событий, выявляя общие черты и особенности различных категорий событий. Данный анализ можно интерпретировать как трансформацию набора событий в систему категорий и характеристик, присущих каждой категории, что является ключевым шагом для всестороннего понимания сущности событий и их взаимосвязей [10].

Данные, собранные в одном журнале, отражают разнообразие событий и служат основой для научных исследований, их можно представить в формате

$$\{e_1, e_2, ..., e_k\} = E_L, \{t_1, t_2, ..., t_n\} = T_L$$

где множество E, которое содержит все события из журнала L, и множество T, состоящее из всех типов событий, зафиксированных в журнале L.

Изучение типов событий должно быть основано на фактических данных, например, журналах событий. Такой подход исключает ошибки, которые могут возникнуть при изменении формата типов событий, так как любые трансформации будут отражены в журналах [11]. Анализ журналов событий позволяет выделить ключевые особенности для каждого типа событий:

$$\{p_1, p_2, ..., p_m\} = P_T$$

где P обозначает совокупность характеристик, присущих множеству типов T.

В результате, изучение журнала событий с целью выявления повторяющихся шаблонов и их особенностей можно охарактеризовать как трансформацию набора событий в систему типов и свойств, присущих этим типам:

$$E \rightarrow T \times P$$

Изначально концепция корреляции данных нашла применение в системах обнаружения вторжений, где ее задачей было выявлять взаимосвязи между сетевыми действиями и группировать их для более эффективного выявления кибератак. Позднее, алгоритмы корреляции, первоначально созданные для этих систем, были модифицированы и успешно внедрены в SIEM-системы для обработки более широкого спектра информации. Для проведения корреляции необходимо пройти ряд важных шагов: нормализацию данных, агрегацию, фильтрацию и, наконец, саму корреляцию. Каждый из этих этапов играет существенную роль в получении достоверных результатов [13].

В сфере безопасности существует целый арсенал методов корреляции событий, каждый из которых обладает уникальными сильными и слабыми сторонами. Выбор конкретного метода зависит от специфики решаемой задачи и этапа процесса. Чаще всего в современных системах безопасности для комплексной корреляции событий применяются комбинированные подходы, объединяющие сигнатурные и эвристические методы. Статистические методы корреляции определяют уровень взаимосвязи между переменными, в то время как методы, работающие с правилами, устанавливают связи между событиями на основе заранее определенных инструкций. Существуют и другие подходы, например, графовые методы, использующие теорию графов, и нейронные сети, которые применяются для выявления зависимостей и выявления нетипичных паттернов в последовательностях событий [15].

Оценить эффективность методов корреляции данных в SIEM-системах сложно, ведь разработчики не всегда предоставляют исчерпывающую информацию о своей разработке, что создает барьер для понимания используемых технологий. Изучение модуля корреляции, даже при использовании готового решения, может вызвать сложности, так как пользователь сталкивается с настройкой правил и исключений, но не с пониманием лежащих в основе алгоритмов. Для тех же, кто желает изучить суть корреляции событий безопасности в SIEM, доступны решения с открытым исходным кодом и обширные научные исследования, раскрывающие различные подходы и стратегии [14].

Хотя правило-ориентированный подход к выявлению корреляций, опирающийся на заранее определенные связи между событиями, и обладает простотой и широкой популярностью, он всё же сталкивается с серьезными трудностями. Ключевым фактором, ограничивающим его эффективность использования метода, является сложность и продолжительность процесса создания правил, успех которого во многом определяется профессионализмом разработчика и иными человеческими факторами [4].

Современные технологии интеллектуального анализа данных, такие как байесовские сети, иммунные сети и нейронные сети, демонстрируют впечатляющий прогресс в повышении точности корреляции. Автоматизация корреляции событий с помощью этих инструментов значительно упрощается, освобождая специалистов от рутинных настроек. Однако, создание предсказательных моделей, опирающихся на интеллектуальные алгоритмы, по-прежнему требует предварительной обработки данных, которая, несмотря на все достижения автоматизации, не всегда может быть полностью решена в автоматическом режиме. Важно также отметить, что применение интеллектуальных методов подразумевает оценку точности и надежности моделей, а для обучения алгоритмов нужны полные, разнообразные и, что немаловажно, достоверные данные. Разработчики должны обладать глубоким пониманием причин возникновения инцидентов для успешной работы с данными.

Определение причин инцидентов представляет собой непростую, но решаемую задачу, которую успешно выполняют лишь ограниченное число SIEM-систем. Понимание причин инцидента часто является ключевым фактором для адекватного реагирования и обычно возникает непосредственно во время события, например, при попытке понять мотивы пользователя, внесшего изменения в настройки службы. Недостаточно вендоров применяют накопленные базы данных для анализа и взаимосвязи причин инцидентов.

Кибератаки нередко сопровождаются цепочкой действий, таких как сканирование компьютерных сетей, попытки проникновения в защищенные системы, блокировка вложений в электронной почте и создание учетных записей с повышенными привилегиями. Важно помнить, что каждое из этих событий может иметь несколько различных мотивов, которые могут быть как взаимосвязанными, так и совершенно независимыми, что усложняет определение точной взаимосвязи между происходящими событиями и самим инцидентом. Порядок событий при поведенческом анализе можно определять по силе связей, существующих между разными категориями событий и их конкретными примерами [16].

В журнале событий Windows может появиться запись о переключении режима запуска службы с "Автоматический" на "Отключено". Такое изменение настроек запуска службы может свидетельствовать о ряде возможных неполадок, без дополнительной информации о которых сложно сделать однозначный вывод.

Наличие уязвимостей в активной службе может послужить причиной несанкционированного доступа. В этом случае запуск службы следует либо исключить, либо принять меры по минимизации рисков, например, временно приостановить ее работу. Такая мера может дестабилизировать работу операционной системы, поскольку отключение некоторых служб может вызвать синий экран смерти (BSOD) или сделать запуск системы невозможным.

Кроме того, нарушение правил безопасности может возникнуть, если изменение настроек запуска службы разрешено исключительно узкому кругу администраторов или если служба должна функционировать в неизменном состоянии. Попытка запустить такую службу от имени обычного пользователя может указывать на несанкционированное вмешательство.

При наличии указанных признаков [18] возникает очевидная сложность, которая, скорее всего, будет актуальна при реализации систем SIEM с упрощенным подходом к управлению конфигурацией, политиками безопасности или следованием нормативным документам (в зависимости от задач). Например, при создании правил корреляции позволит, которые будут отслеживать работоспособность антивирусных программ и их присутствие на IT-ресурсах корпоративной сети.

Идентификация отдельных признаков неисправностей при помощи бессигнатурных корреляционных методов может быть непростой задачей. К примеру, графовый метод способен диагностировать проблемы, возникающие при разрыве связи между определенной службой и ее ресурсом [17].

Методы, основанные на сигнатурах, работают путем поиска совпадений, определяемых заранее заданными критериями. Каждый критерий связан с конкретной проблемой, которую в системе мониторинга и реагирования на инциденты (SIEM) называют «инцидентом». Важно отметить, что одно и то же событие может соответствовать сразу нескольким критериям [9].

Регламент может быть оснащен триггером, состоящим из условий, счетчика и плана действий. Счетчики служат для отслеживания количества срабатываний, связанных с конкретным правилом. К примеру, пять неудачных попыток авторизации от одного аккаунта в течение пяти минут могут быть интерпретированы как подозрительная активность. В случае первой неудачной попытки авторизации срабатывает счетчик и триггер, давая счетчику четыре дополнительных шанса на регистрацию неудачных попыток в течение определенного времени. Если в течение этого периода не будет зафиксировано новых неудачных попыток, счетчик сбросится. Но если события последуют с задержкой, например, из-за проблем с соединением с агентом, правило будет вновь активировано, и уже зарегистрированная неудачная попытка станет основой для создания инцидента (рис. 2).

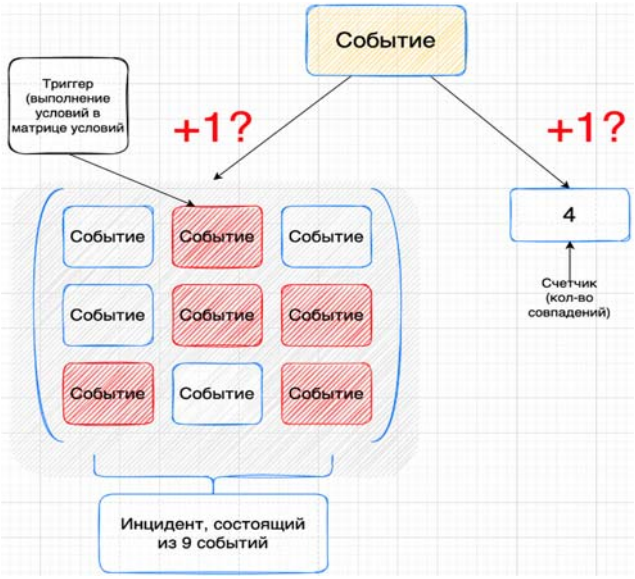

Рис. 2. Сценарий обработки события

Кроме подсчета событий, аналогичного запросу SELECT COUNT в T-SQL, счетчик отслеживает и определенные комбинации событий. В данном контексте триггер выступает как своеобразный датчик, срабатывающий при наступлении определенного события, соответствующего заданным критериям корреляции, которые могут быть заложены как в правилах, так и в алгоритмах. К примеру, сигнал о неудачной попытке запуска неизвестного исполняемого файла формируется на основе двух взаимосвязанных факторов: попытки доступа к файлу и запуска нового процесса. Если одно из обозначенных событий произойдет, триггер активируется и начнет наблюдать за соблюдением второго условия в течение заданного времени. По окончании этого интервала триггер деактивируется, чтобы предотвратить непреднамеренные реакции на инциденты, которые уже не нуждаются в вмешательстве [19].

Существует несколько подходов к определению весов связей. Можно разделить связи на прямые и косвенные, а также классифицировать их по характеру связанных событий: однотипные и разнородные. Для связей между конкретными событиями рассчитываются индивидуальные веса, которые зависят от степени их похожести и специфических весовых коэффициентов. В результате анализа заданного временного периода формируется список пар, где каждая пара включает в себя относительный вес связи и интервал времени. Анализ частотности событий разных типов помогает установить связи между типами событий и их конкретными проявлениями, выявляя таким образом причинно-следственные зависимости.

Эффективный метод корреляции подбирается с учётом конкретной задачи и имеющихся данных, а его оптимальность определяется экспериментальным путем. В ходе эволюционного анализа защищаемой системы выделяются основные компоненты: хосты, серверы, операционные системы и сервисы, а алгоритм корреляции обучается на основе выявленных метрик. Компоненты системы защиты информации выполняют функцию "учителя", отслеживая актуальные угрозы и обобщая их в понятные высокоуровневые отчеты о происшествиях [3].

Современные системы анализа данных безопасности успешно применяют анализ структуры различных событий для повышения эффективности работы. Свойства типов событий служат для упорядочивания и индексирования поступающей информации, что упрощает последующие этапы обработки. Благодаря такому подходу, выявленные в исходных данных связи могут быть использованы для создания более точных запросов, интегрирующих как автоматизированный, так и ручной анализ. Поиск косвенных связей дополнительно обогащается посредством частотного анализа материалов с аналогичным содержанием, что существенно расширяет возможности исследовательской работы [12].

Для обеспечения круглосуточной работы в режиме реального времени, когда требуется полной анализ всей поступающей информации и большого массива входных данных, а также при масштабировании инфраструктуры, актуальны методы параллельной и многопоточной обработки. Анализ корреляции событий безопасности в SIEM-системах выявил необходимость разработки новой стратегии, охватывающей все этапы процесса корреляции и определяющей единую конечную задачу для анализа, что позволит повысить его результативность. Использование новой модели корреляции, основанной на классификации типов событий, открывает новые возможности для решения задач безопасности, таких как надзор за

киберфизическими системами, выявление целеустремленных атак и автоматизированная оценка защищенности инфраструктур, состоящих из "интернет вещей" и других современных технических средств [20].

Важно отметить, что методы корреляции, интегрирующие как традиционные, так и интеллектуальные подходы, должны быть адаптированы к постоянно растущему объему данных и разнообразию угроз. В результате успешной реализации предложенной методологии корреляции безопасности возможно поднять уровень защиты киберфизических систем и IoT-устройств за счет автоматизации анализа событий. Таким образом, необходимо продолжать развитие инструментов и подходов к корреляции, чтобы противостоять новым вызовам в области информационной безопасности.

Заключение

В результате проведенного анализа подчеркивается ключевая роль процесса корреляции данных в системах управления информационной безопасностью (SIEM) для выявления инцидентов и анализа угроз, что становится крайне актуальным в условиях растущих объемов данных и усложняющихся атак в финансовой среде. Методология автоматизированной корреляции информации об угрозах, основанная на многогранном анализе, помогает установить причинно-следственные связи и выявлять аномалии, связанные с небезопасными действиями пользователей.

Исследование выявило, что корреляция в SIEM системах начинается с непрерывного сбора данных из различных источников. Методы корреляции варьируются от правила-ориентированных до статистических и графовых подходов, каждый из которых нацелен на решение определенных задач и требует адаптивности в условиях динамичных угроз.

Анализ показал, что применение интеллектуального анализа данных, такие как нейронные сети и байесовские сети, значительно повышает степень автоматизации и эффективность корреляции, хотя внедрение таких методов требует тщательной предварительной обработки данных и оценки их адекватности. Полученные результаты подчеркивают необходимость разработки новых подходов к корреляции событий, уделяя внимание адаптации систем к постоянно меняющемуся ландшафту угроз.

Литература

1. *Головин С.В.* Обеспечение безопасности информации в современных условиях. М.: Наука, 2021.

2. *Кузнецов А.Г.* Корреляция инцидентов при помощи SIEM-систем // Информационная безопасность. Т. 12, Выпуск 3, 2022. С. 45-58.

3. *Шевелёв И.Б.* Обнаружение кибератак с использованием методов машинного обучения. Издательство "Квартиль", 2020.

4. *Борисов А.В.* Методы корреляции событий безопасности в SIEM-системах. Диссертация на соискание учёной степени кандидата тех. наук, 2019.

5. *Сидоренко Н.А.* Анализ угроз безопасности информации // Вопросы информационной безопасности, 2023. С. 67-80.

6. *Баранов И.В., Лазарев Р.М.* Анализ и управление инцидентами безопасности. Издательство "Финансовая информмация", 2022.

7. *Самохин Е.Ю.* Системы SIEM: общие принципы и подходы // Защита информации, 2021. С. 12-20.

8. *Джонсон К.* Effective Security Information and Event Management: A Definitive Guide. Cybersecurity Press, 2022.

9. *Поляков С.С.* Статистический анализ данных в системе SIEM. Рецензируемая статья, 2023.

10. *Носков В.И.* Методы автоматизации мониторинга безопасности // Научные труды конференции "IT Security", 2023.

11. *Перевозчиков М.И.* Анализ киберугроз: от теории к практике. Издательство "Технологии безопасности", 2020.

12. *Зиновьев А.В.* "Использование нейронных сетей для корреляции событий безопасности // Техническая информатика, 2022.

13. *Фролова А.В.* Интеллектуальный анализ данных в SIEM // Безопасность информационных технологий, 2021.

14. *Шухов И.Н.* Корреляционные модели и методы: подходы к построению // Вестник информационной технологии, 2022.

15. *Хохрина Г.И.* Проектирование систем SIEM: лучший опыт и рекомендации. Издательство "ИнфоТех", 2023.

16. *Малышев П.П., Чернышёв В.Н.* Кибербезопасность: современные вызовы и решения. Издавая по итогам конференции, 2021.

17. *Синицын А.А.* Применение графовых методов в защите информации // Анализ данных, 2023.

18. *Рябов Е.М.* Корреляция событий в области безопасности: проблемы и решения // Информационные системы, 2022.

19. *Дворянинов С.Н.* Тактики реагирования на инциденты безопасности в системах SIEM // Информационная безопасность, 2023.

20. *Отещенко О.В.* Анализ безопасности киберфизических систем и IoT // Научные труды, конференция "Киберфизика и безопасность", 2024.